

Learn by doing

RHCSA

RED HAT ENTERPRISE LINUX 8 (UPDATED)

**TRAINING AND EXAM PREPARATION
GUIDE (EX200)**

SECOND EDITION

**INCLUDES
SHELL SCRIPTING
AND
CONTAINERS**

- Covers Red Hat Enterprise Linux 8
- Covers ALL Latest Official Exam Objectives
(including Shell Scripting and Containers)
- Great for Self-Study and In-Class/Virtual Training
- 23 Chapters
- 108 Real-Life Step-By-Step Exercises and Shell Scripts
- 80 Do-It-Yourself Challenge Labs
- 408 Review Questions & Answers
- 4 Sample RHCSA Exams (4 x 23 tasks per exam)

ASGHAR GHORI

RHCSA® **Red Hat® Enterprise Linux® 8** **(UPDATED)**

Training
and
Exam Preparation
Guide

Exam Code
EX200

Second Edition
November 2020

Asghar Ghori

1246 Heil Quaker Blvd., La Vergne, TN USA 37086
Chapter House, Pitfield, Kiln Farm, Milton Keynes, UK MK11 3LW
Unit A1/A3, 7 Janine Street, Scoresby, Victoria 3179, Australia
www.ingramspark.com

Technical Reviewers: Many of author's students and peers

Editors: FirstEditing.com and Zainab Ghori

Cover Design: Nid n Nad Graphics Printing Inc. (www.nidnnad.ca)

Printers and Distributors: IngramSpark Inc.

Copyright © 2020 Asghar Ghori

All rights reserved.

No portion of this book may be stored in a retrieval system, transmitted, or reproduced in any form, including but not limited to photocopying or other recording, without the express written consent of the author.

Printed in the USA, Canada, UK, France, Germany, Italy, Spain, and Australia.

ISBN: **978-1-7750621-4-1**

ISBN: **978-1-7750621-5-8 (e-book)**

Library of Congress Control Number: **2020922437**

To order in bulk at special quantity discounts for sales promotions or training programs, please contact the author directly at asghar_ghori2002@yahoo.com

The following are registered trademarks in the U.S. and other countries:

Red Hat® is a registered trademark of Red Hat, Inc.

RHCSA® is a registered trademark of Red Hat, Inc.

Linux® is a registered trademark of Linus Torvalds.

Oracle® and VirtualBox® are registered trademarks of Oracle Corporation, Inc.

UNIX® is a registered trademark of The Open Group.

Microsoft® and Windows® are US registered trademarks of Microsoft Corporation.

Docker and the Docker logo are trademarks or registered trademarks of Docker, Inc.

Intel® is the trademark or registered trademark of Intel Corporation or its subsidiaries.

All other trademarks, registered trademarks, or logos used in this book are the property of their respective owners.

The author has made his best efforts to prepare this book. The contents are based on Red Hat® Enterprise Linux® version 8. The author makes no representation or warranties of any kind with regard to the completeness or accuracy of the contents herein and accepts no liability whatsoever including but not limited to

merchantability, fitness for any particular purpose, or any losses or damages of any kind caused or allegedly caused directly or indirectly from this material.

This book is not a replacement for the official RHCSA training courses offered by Red Hat, Inc. However, it may be used to prepare for the Red Hat Certified System Administrator (RHCSA) exam, EX200, based on the latest version of the exam objectives available on Red Hat's training and certification website. Neither author nor publisher warrants that use of this publication will ensure passing the relevant exams or that the information contained herein is endorsed by Red Hat, Inc.

Preface

Red Hat revised the objectives for the RHCSA (RHEL 8) EX200 certification exam on October 1, 2020 and added shell scripting and containers to the list of existing topics. I found this opportunity to update the book to provide a single, definitive resource to self-learners and instructor-led learners. The updates include two new chapters (22 and 23), fixing reported errata, swapping chapters 16 and 17, expanding all four sample exams, adding RHSMP setup procedure, and adding exercises, labs, and shell scripts.

The RHCSA exam is performance-based and presents several tasks that are to be completed on virtual machines within a stipulated time. This book provides necessary coverage from both theoretical and practical standpoints to assist the learners pass the exam. Moreover, this book may be used for in-class and live virtual trainings and as an on-the-job deskside reference.

Keeping in mind the hands-on nature of the exam, I have included a multitude of step-by-step exercises and Do-It-Yourself (DIY) challenge labs throughout the book. [Chapter 01](#) describes how to obtain copies of RHEL 8 and VirtualBox software and build a lab infrastructure to practice the procedures and perform the labs.

I suggest you study the material presented in each chapter thoroughly before proceeding to the relevant hands-on exercise(s). I have provided several review questions with answers at the end of each chapter. Take the quiz and then attempt the DIY challenge labs offered thereafter. I have not furnished solutions to these labs intentionally, as I am confident that the knowledge and skills you will have gained by that time will be sufficient to accomplish the labs on your own; and, in essence, this is what I want you to eventually get at. Once you have read and understood the material, performed

exercises, completed review questions, and accomplished DIY challenge labs, take time to attempt the sample RHCSA exams provided in appendices.

While performing exercises and labs, if a command does not produce the published result, I advise that you check the message the command has generated and consult relevant log files. Minor issues, such as a wrong path or a typing mistake, prevent commands from running. Sometimes, syntax errors in their constructs could result in execution failures. You might have to adjust them to run as expected. RHEL manual pages prove useful in comprehending commands and their syntaxes.

There are four areas I suggest you focus on to develop expertise with RHEL, as well as to get ready for the exam: 1) grasping concepts; 2) mastering implementation procedures, exercises, and labs; 3) learning commands, understanding configuration files, and knowing service processes; and 4) being able to analyze logs, and troubleshoot and resolve issues. An excellent knowledge of commands and key options, and the files they update should also be developed along with what processes handle which services, and so on. This will help you obtain a greater overall understanding of what exactly happens in the background when a command runs. Debugging becomes easier when concepts are clear and working knowledge is solid.

I maintain www.nixeducation.com where I add errata, additional certification information, helpful videos on Linux concepts and administration topics, and links to other useful resources. I encourage you to visit this website.

To conclude, I would like to request for your constructive feedback sent to my personal email asghar_ghori2002@yahoo.com about any grammatical or technical errors or mistakes in the book, as well as any suggestions. Try to be as specific as possible in your description. Improvement is a continuous process, and I believe your feedback will help me continue to deliver quality books.

Good luck in your endeavors.

Asghar Ghori | November 2020 | Toronto, Canada

Acknowledgments

As always, I am grateful to God who enabled me to write this book successfully.

I would like to acknowledge the valuable feedback my students, friends, and colleagues provided on my previous publications on RHCSA, RHCE, Comptia Linux+, and HP-UX. I am thankful for their help in making this book better in all respects.

I recognize the constructive feedback I had received from the readers of my previous publications. I have used their comments toward the improvement of this edition.

I would like to express my special thanks to my wife, daughters, and sons, who endured my mental absence while writing this book. I could not have accomplished this project without their continuous support and encouragement.

Lastly, I would like to offer my very special tributes to my deceased parents and sisters.

Asghar Ghori

About the Author

Asghar Ghori is a seasoned Linux | Cloud | DevOps consultant, trainer, and author. As a consultant, his experience ranges from deployment, support, and administration to solution design and architecture; as a trainer, he has designed, developed, and delivered numerous advanced training programs; and as a published author, he has nine books on UNIX (HP-UX) and Linux (Red Hat Enterprise Linux and CompTIA Linux+) to his credit.

Asghar holds a BS in Engineering. He is RHCE, RHCSA, HPCSA, HPCSE, Oracle SCSA, IBM Certified Specialist for AIX, and CNE with ITIL-F and PMP certifications. He is 4x AWS Certified, 2x Azure Certified, and HashiCorp Certified Terraform Associate (HCTA).

Asghar lives in a small town near Toronto, Ontario, Canada with his wife and children, and can be reached via email
asghar_ghori2002@yahoo.com or LinkedIn.

Other publications of Asghar Ghori are:

1. RHCSA Red Hat Enterprise Linux 8: Training and Exam Preparation Guide (EX200) (ISBN: 978-1775062127) (RHEL version 8), published January 2020
2. CompTIA Linux+/LPIC-1: Training and Exam Preparation Guide (Exam Codes: LX0-103/101-400 and LX0-104/102-400) (ISBN: 978-1775062103), published 2017
3. RHCSA & RHCE Red Hat Enterprise Linux 7: Training and Exam Preparation Guide (EX200 and EX300) (ISBN: 978-1495148200) (RHEL version 7), published 2015
4. Red Hat Certified System Administrator & Engineer: Training Guide and a Quick Deskside Reference (ISBN: 978-

1467549400) (RHEL version 6), published 2012

5. Red Hat Certified Technician & Engineer (RHCT and RHCE) Training Guide and Administrator's Reference (ISBN: 978-1615844302) (RHEL version 5), published 2009
6. HP-UX: HP Certified Systems Administrator, Exam HP0-A01, Training Guide and Administrator's Reference (ISBN: 978-1606436547) (HP-UX 11iv3), published 2008
7. HP Certified Systems Administrator, Exam HP0-095, Training Guide and Administrator's Reference (ISBN: 978-1424342310) (HP-UX 11iv2 and 11iv3), published 2007
8. Certified System Administrator for HP-UX: Study Guide and Administrator's Reference (ISBN: 978-1419645938) (HP-UX 11iv1), published 2006

Conventions Used in this Book

The following typographic and other conventions are used in this book:

Book Antiqua Italic 10 pt. is used in text paragraphs to introduce new terms. For example:

“Red Hat renamed the Red Hat Linux operating system series *Red Hat Enterprise Linux* (RHEL) in 2003.”

Times Roman Italic 10 pt. is used in text paragraphs to highlight names of files, directories, commands, daemons, users, groups, hosts, domains, and URLs. This font also highlights file and directory paths. For example:

“To go directly from */etc* to a subdirectory *dir1* under *user1*’s home directory, create *dir1*, as”

Times New Roman 9 pt. is used to segregate command output, script/file contents, and information expected to be entered in configuration files from the surrounding text. It is also used in tables, index, and side notes.

Times Roman Bold 10 pt. is used to highlight commands and command line arguments that the user is expected to type and execute at the command prompt. For example:

```
[user1@server1 ~]$ ls -lt
```

There are two white spaces between parts of a typed command for the sake of clarity in text.

There are hundreds of screenshots that show commands and output. They are images taken directly from the Linux terminal screen.

All headings and sub-headings are in **California FB** font, and are bolded.

Ctrl+x key sequence implies that you hold down the Ctrl key and then press the other key. Courier New font is used to highlight such combinations. This font is also used to identify keystrokes, such as Enter and Esc.

. Dotted lines represent truncated command output.

The RHCSA 8 Exam and Exam Objectives

The Red Hat Certified System Administrator (RHCSA) certification exam is a performance-based hands-on exam designed for IT professionals. This exam is presented in electronic format on a live server running Red Hat Enterprise Linux 8. This server has two RHEL 8-based virtual machines to accomplish the exam tasks. During the exam, the candidates do not have access to any external resource such as the Internet, printed material, electronic content, and mobile devices except for the manuals and other documentation that is installed on the exam virtual machines. The official exam objectives (69 in total as of November 11, 2020) are available for reference at

<http://www.redhat.com/training/courses/ex200/examobjective>. Visit the URL for up-to-date information. The exam objectives are covered in detail in the chapters throughout this book. An enumerated list of the objectives is presented below along with the chapter number where each objective is discussed.

Understand and Use Essential Tools

1. Access a shell prompt and issue commands with correct syntax (chapter 2)
2. Use input-output redirection (>, >>, |, 2>, etc) (chapter 7)
3. Use grep and regular expressions to analyze text (chapter 7)
4. Access remote systems using ssh (chapters 01 and 19)
5. Log in and switch users in multi-user targets (chapter 6)
6. Archive, compress, unpack, and uncompress files using tar, star, gzip, and bzip2 (chapter 3)
7. Create and edit text files (chapter 3)
8. Create, delete, copy, and move files and directories (chapter 3)
9. Create hard and soft links (chapter 3)
10. List, set, and change standard ugo/rwx permissions (chapter 4)

11. Locate, read, and use system documentation including man, info, and files in /usr/share/doc (chapter 2)

Create Simple Shell Scripts

- 12. Conditionally execute code (use of: if, test, [], etc.) (chapter 22)
- 13. Use Looping constructs (for, etc.) to process file, command line input (chapter 22)
- 14. Process script inputs (\$1, \$2, etc.) (chapter 22)
- 15. Processing output of shell commands within a script (chapter 22)
- 16. Processing shell command exit codes (chapter 22)

Operate Running Systems

- 17. Boot, reboot, and shut down a system normally (chapter 12)
- 18. Boot systems into different targets manually (chapter 12)
- 19. Interrupt the boot process in order to gain access to a system (chapter 11)
- 20. Identify CPU/memory intensive processes and kill processes (chapter 8)
- 21. Adjust process scheduling (chapter 8)
- 22. Manage tuning profiles (chapter 12)
- 23. Locate and interpret system log files and journals (chapter 12)
- 24. Preserve system journals (chapter 12)
- 25. Start, stop, and check the status of network services (chapter 12)
- 26. Securely transfer files between systems (chapter 19)

Configure Local Storage

- 27. List, create, and delete partitions on MBR and GPT disks (chapter 13)
- 28. Create and remove physical volumes (chapter 14)
- 29. Assign physical volumes to volume groups (chapter 14)
- 30. Create and delete logical volumes (chapter 14)
- 31. Configure systems to mount file systems at boot by Universally Unique ID (UUID) or label (chapter 15)
- 32. Add new partitions and logical volumes, and swap to a system non-destructively (chapters 14 and 15)

Create and Configure File Systems

- 33. Create, mount, unmount, and use vfat, ext4, and xfs file systems (chapter 15)
- 34. Mount and unmount network file systems using NFS (chapter 17)
- 35. Extend existing logical volumes (chapters 14 and 15)
- 36. Create and configure set-GID directories for collaboration (chapter 4)
- 37. Configure disk compression (chapter 13)
- 38. Manage layered storage (chapter 14)
- 39. Diagnose and correct file permission problems (chapter 4)

Deploy, Configure, and Maintain Systems

- 40. Schedule tasks using at and cron (chapter 8)
- 41. Start and stop services and configure services to start automatically at boot (chapter 12)
- 42. Configure systems to boot into a specific target automatically (chapter 12)
- 43. Configure time service clients (chapter 18)
- 44. Install and update software packages from Red Hat Network, a remote repository, or from the local file system (chapter 9 and 10)
- 45. Work with package module streams (chapter 10)
- 46. Modify the system bootloader (chapter 11)

Manage Basic Networking

- 47. Configure IPv4 and IPv6 addresses (chapter 16)
- 48. Configure hostname resolution (chapter 18)
- 49. Configure network services to start automatically at boot (chapter 12)
- 50. Restrict network access using firewall-cmd/firewall (chapter 20)

Manage Users and Groups

- 51. Create, delete, and modify local user accounts (chapter 5)
- 52. Change passwords and adjust password aging for local user accounts (chapter 5 and 6)
- 53. Create, delete, and modify local groups and group memberships (chapter 6)
- 54. Configure superuser access (chapter 6)

Manage Security

- 55. Configure firewall settings using firewall-cmd/firewalld (chapter 20)
- 56. Create and use file access control lists (chapter 4)
- 57. Configure key-based authentication for SSH (chapter 19)
- 58. Set enforcing and permissive modes for SELinux (chapter 21)
- 59. List and identify SELinux file and process context (chapter 21)
- 60. Restore default file contexts (chapter 21)
- 61. Use Boolean settings to modify system SELinux settings (chapter 21)
- 62. Diagnose and address routine SELinux policy violations (chapter 21)

Manage Containers

- 63. Find and retrieve container images from a remote registry (chapter 23)
- 64. Inspect container images (chapter 23)
- 65. Perform container management using commands such as podman and skopeo (chapter 23)
- 66. Perform basic container management such as running, starting, stopping, and listing running containers (chapter 23)
- 67. Run a service inside a container (chapter 23)

68. Configure a container to start automatically as a systemd service (chapter 23)
69. Attach persistent storage to a container (chapter 23)

Taking the Exam

1. Save time wherever possible, as time is of the essence during the exam
2. Make certain that any changes you make must survive system reboots
3. Use any text editor you feel comfortable with to modify text configuration files
4. Exam tasks are split into two groups and each group must be performed in its own assigned virtual machine
5. Inform the proctor right away if you encounter any issues with your exam system or the virtual machines
6. The exam is administered with no access to the Internet, electronic devices, or written material
7. Read each exam task fully and understand it thoroughly before attempting it
8. Read all storage tasks carefully and use the lsblk command as explained in the book to identify the right disk to perform each task on.

Exam Fee and Registration Procedure

The fee for the RHCSA exam is US\$400 (plus any applicable taxes), or equivalent in local currencies. To register, visit <http://www.redhat.com/training/courses/ex200/examobjective>, select your location and an exam format, and click Get Started to continue through the registration process. The RHCSA exam lasts for 3 hours.

About this Book

RHCSA Red Hat Enterprise Linux 8 (UPDATED): Training and Exam Preparation Guide, Second Edition provides in-depth coverage of the latest RHCSA EX200 exam objectives that include **Shell Scripting and Containers**. The most definitive guide available on the subject, this book explains concepts, analyzes configuration files, describes command outputs, shows step-by-step procedures (includes screenshots of actual commands executed and outputs they produced), and challenges the readers' comprehension of the concepts and procedures by presenting plenty of additional labs and sample realistic exam tasks to perform on their own.

This book has **23 chapters** that are organized logically, from setting up the lab to the fundamentals of Linux to sophisticated Linux administration topics. The book covers the topics on local RHEL 8 installation; initial interaction with the system; basic Linux commands; compression and archiving; file editing and manipulation; standard and special permissions; file searching and access controls; user monitoring and authentication files; users, groups, and password aging; bash shell features and startup files; processes and task scheduling; basic and advanced software administration techniques; system boot process and bootloader; kernel management and system initialization; logging and system tuning; basic and advanced storage management tools and solutions; local file systems and swap regions; network device and connection configuration; remote file systems and automounting; time synchronization and hostname resolution; the secure shell service; firewall and SELinux controls; and shell scripting and containers.

Each chapter highlights the major topics and relevant exam objectives at the beginning and ends with several review questions &

answers and Do-It-Yourself challenge labs. Throughout the book, figures, tables, screen shots, examples, notes, and exam tips are furnished to support explanation and exam preparation. There are four sample RHCSA exams that are expected to be performed using the knowledge and skills attained from reading the material, following the exercises, and completing the challenge labs. The labs and the sample exams include hints to relevant topics and/or exercises.

This book may be used as a self-learning guide by RHCSA 8 exam aspirants, a resource by instructors and students to follow in physical and virtual training sessions, an on-the-job resource for reference, and an easy-to-understand guide by novice and non-RHEL administrators.

TABLE OF CONTENTS

Preface

Acknowledgments

About the Author

Conventions Used in this Book

The RHCSA 8 Exam and Exam Objectives

About this Book

01. Local Installation

A Quick Look at Linux Development

Linux History in a Nutshell

Linux from Red Hat

Lab Infrastructure for Practice

What is Needed for the Lab?

The RHEL Installer Program

Where Do Installation Logs Go?

Virtual Console Screens

Exercise 1-1: Download and Install VirtualBox

Software, and Create a Virtual Machine

Downloading and Installing VirtualBox

Creating a Virtual Machine

Exercise 1-2: Download and Install RHEL 8

Downloading RHEL 8 ISO Image

Attaching RHEL 8 ISO Image to the Virtual Machine

Launching the Installer

Adding Support for Keyboards and Languages

Configuring Time & Date

Choosing an Installation Source

- Selecting Software to be Installed
- Configuring Installation Destination
- Configuring Network and Hostname
- Beginning Installation
- Setting root Password and Creating a User Account
- Concluding Installation
- Changing Default Boot Order
- Performing Post-Installation Tasks

Logging In and Out at the Graphical Console

- Logging In for the First Time
- Logging Out

Exercise 1-3: Logging In from Windows

- Chapter Summary

- Review Questions

- Answers to Review Questions

- Do-It-Yourself Challenge Labs

- Lab 1-1: Build RHEL8-VM2 (server2)

02. Initial Interaction with the System

- Linux Graphical Environment

- Display/Login Manager

- Desktop Environment

Linux Directory Structure and File Systems

- Top-Level Directories

- File System Categories

- The Root File System (/), Disk-Based

- The Boot File System (/boot), Disk-Based

- The Home Directory (/home)

- The Optional Directory (/opt)

- The UNIX System Resources Directory (/usr)

- The Variable Directory (/var)

- The Temporary Directory (/tmp)

- The Devices File System (/dev), Virtual

- The Procfs File System (/proc), Virtual

The Runtime File System (/run), Virtual
The System File System (/sys), Virtual
Viewing Directory Hierarchy

Basic System Commands

Starting a Remote Terminal Session
Understanding the Command Mechanics
Listing Files and Directories
Printing Working Directory
Navigating Directories
Identifying Terminal Device File
Inspecting System's Uptime and Processor Load
Clearing the Screen
Determining Command Path
Viewing System Information
Viewing CPU Specs

Getting Help

Accessing Manual Pages
Headings in the Manual
Manual Sections
Searching by Keyword
Exposing Short Description
The info and pinfo Commands
Documentation in the /usr/share/doc Directory
Red Hat Enterprise Linux 8 Documentation

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 2-1: Navigate Linux Directory Tree
Lab 2-2: Miscellaneous Tasks
Lab 2-3: Identify System and Kernel Information
Lab 2-4: Use Help

03. Basic File Management

Common File Types

- Regular Files
- Directory Files
- Block and Character Special Device Files
- Symbolic Links

Compression and Archiving

- Using gzip and gunzip
- Using bzip2 and bunzip2
- Differences between gzip and bzip2
- Using tar

Exercise 3-1: Create Compressed Archives

File Editing

- Modes of Operation
- Starting vim
- Inserting text
- Navigating within vim
- Deleting Text
- Undoing and Repeating
- Searching for Text
- Replacing Text
- Copying, Moving, and Pasting Text
- Changing Text
- Saving and Quitting vim

File and Directory Operations

- Creating Files and Directories
- Displaying File Contents
- Counting Words, Lines, and Characters in Text Files
- Copying Files and Directories
- Moving and Renaming Files and Directories
- Removing Files and Directories

File Linking

- Hard Link
- Soft Link
- Differences between Copying and Linking

Exercise 3-2: Create and Manage Hard Links

Exercise 3-3: Create and Manage Soft Links

Chapter Summary

[Review Questions](#)

[Answers to Review Questions](#)

[Do-It-Yourself Challenge Labs](#)

[Lab 3-1: Archive, List, and Restore Files](#)

[Lab 3-2: Practice the vim Editor](#)

[Lab 3-3: File and Directory Operations](#)

04. Advanced File Management

[File and Directory Access Permissions](#)

[Determining Access Permissions](#)

[Permission Classes](#)

[Permission Types](#)

[Permission Modes](#)

[Modifying Access Permission Bits](#)

[Exercise 4-1: Modify Permission Bits Using Symbolic Form](#)

[Exercise 4-2: Modify Permission Bits Using Octal Form](#)

[Default Permissions](#)

[Calculating Default Permissions](#)

[Special File Permissions](#)

[The setuid Bit on Binary Executable Files](#)

[Exercise 4-3: Test the Effect of setuid Bit on Executable Files](#)

[The setgid Bit on Binary Executable Files](#)

[Exercise 4-4: Test the Effect of setgid Bit on Executable Files](#)

[The setgid Bit on Shared Directories](#)

[Exercise 4-5: Set up Shared Directory for Group Collaboration](#)

[The Sticky Bit on Public and Shared Writable Directories](#)

[Exercise 4-6: Test the Effect of Sticky Bit](#)

[File Searching](#)

[Using the find Command](#)

[Using find with -exec and -ok Flags](#)

[Access Control Lists \(ACLs\)](#)

[ACL Management Commands](#)

The getfacl Command
The setfacl Command
The Role of the mask Value
Exercise 4-7: Identify, Apply, and Erase Access ACLs
Default ACLs
Exercise 4-8: Apply, Identify, and Erase Default ACLs

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 4-1: Manipulate File Permissions
Lab 4-2: Configure Group Collaboration and Prevent File Deletion
Lab 4-3: Find Files
Lab 4-4: Find Files Using Different Criteria
Lab 4-5: Apply ACL Settings

05. Basic User Management

User Login Activity and Information

Listing Logged-In Users
Inspecting History of Successful Login Attempts and System Reboots
Viewing History of Failed User Login Attempts
Reporting Recent User Login Attempts
Examining User and Group Information

Local User Authentication Files

The passwd File
The shadow File
The group File
The gshadow File

The useradd and login.defs Configuration Files

User Account Management

The useradd, usermod, and userdel Commands
Exercise 5-1: Create a User Account with Default Attributes

Exercise 5-2: Create a User Account with Custom Values
Exercise 5-3: Modify and Delete a User Account
No-Login (Non-Interactive) User Account
Exercise 5-4: Create a User Account with No-Login Access

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 5-1: Check User Login Attempts
Lab 5-2: Verify User and Group Identity
Lab 5-3: Create Users
Lab 5-4: Create User with Non-Interactive Shell

06. Advanced User Management

Password Aging and its Management

The chage Command
Exercise 6-1: Set and Confirm Password Aging with chage
The passwd Command
Exercise 6-2: Set and Confirm Password Aging with
passwd
The usermod Command
Exercise 6-3: Lock and Unlock a User Account with
usermod and passwd

Linux Groups and their Management

The groupadd, groupmod, and groupdel Commands
Exercise 6-4: Create a Group and Add Members
Exercise 6-5: Modify and Delete a Group Account

Substituting Users and Doing as Superuser

Substituting (or Switching) Users
Doing as Superuser (or Doing as Substitute User)

Owning User and Owning Group

Exercise 6-6: Modify File Owner and Owning Group

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 6-1: Create User and Configure Password Aging

Lab 6-2: Lock and Unlock User

Lab 6-3: Modify Group

Lab 6-4: Configure sudo Access

Lab 6-5: Modify Owning User and Group

07. The Bash Shell

The Bourne-Again Shell

Shell and Environment Variables

Setting and Unsetting Variables

Command and Variable Substitutions

Exercise 7-1: Modify Primary Command Prompt

Input, Output, and Error Redirections

History Substitution

Editing at the Command Line

Tab Completion

Tilde Substitution

Alias Substitution

Metacharacters and Wildcard Characters

Piping Output of One Command as Input to Another

Quoting Mechanisms

Regular Expressions

Running and Controlling Jobs in Foreground and

Background

Shell Startup Files

System-wide Shell Startup Files

Per-user Shell Startup Files

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 7-1: Customize the Command Prompt

Lab 7-2: Redirect the Standard Input, Output, and Error

08. Linux Processes and Task Scheduling

Processes and Priorities

Process States

Viewing and Monitoring Processes with ps

Viewing and Monitoring Processes with top

Listing a Specific Process

Listing Processes by User and Group Ownership

Understanding Process Niceness and Priority

Exercise 8-1: Start Processes at Non-Default Priorities

Exercise 8-2: Alter Process Priorities

Controlling Processes with Signals

Job Scheduling

Controlling User Access

Scheduler Log File

Using at

Exercise 8-3: Submit, View, List, and Erase an at Job

Using crontab

Syntax of User Crontables

Exercise 8-4: Add, List, and Erase a Cron Job

Anacron

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 8-1: Nice and Renice a Process

Lab 8-2: Configure a User Crontab File

09. Basic Package Management

Package Overview

Packages and Packaging

Package Naming

Package Dependency

Package Database

Package Management Tools

Package Management with rpm

The rpm Command

Exercise 9-1: Mount RHEL 8 ISO Persistently

Querying Packages

Installing a Package

Upgrading a Package

Freshening a Package

Overwriting a Package

Removing a Package

Extracting Files from an Installable Package

Validating Package Integrity and Credibility

Viewing GPG Keys

Verifying Package Attributes

Exercise 9-2: Perform Package Management Using rpm

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 9-1: Install and Verify Packages

Lab 9-2: Query and Erase Packages

10. Advanced Package Management

Advanced Package Management Concepts

Package Groups

Application Streams and Modules

BaseOS Repository

AppStream Repository

Benefits of Segregation

Module Streams

Module Profiles

dnf/yum Repository

Software Management with dnf

dnf Configuration File

The dnf Command

Exercise 10-1: Configure Access to Pre-Built Repositories

Individual Package Management

Listing Available and Installed Packages

Installing and Updating Packages

Exhibiting Package Information

Removing Packages

Exercise 10-2: Manipulate Individual Packages

Determining Provider and Searching Package Metadata

Package Group Management

Listing Available and Installed Package Groups

Installing and Updating Package Groups

Removing Package Groups

Exercise 10-3: Manipulate Package Groups

Module Management

Listing Available and Installed Modules

Installing and Updating Modules

Displaying Module Information

Removing Modules

Exercise 10-4: Manipulate Modules

Switching Module Streams

Exercise 10-5: Install a Module from an Alternative Stream

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 10-1: Configure Access to RHEL 8 Repositories

Lab 10-2: Install and Manage Individual Packages

Lab 10-3: Install and Manage Package Groups

Lab 10-4: Install and Manage Modules

Lab 10-5: Switch Module Streams and Install Software

11. Boot Process, GRUB2, and the Linux Kernel

Linux Boot Process

The Firmware Phase (BIOS and UEFI)

The Bootloader Phase

The Kernel Phase

The Initialization Phase

The GRUB2 Bootloader

Interacting with GRUB2

Understanding GRUB2 Configuration Files

Exercise 11-1: Change Default System Boot Timeout

Booting into Specific Targets

Exercise 11-2: Reset the root User Password

The Linux Kernel

Kernel Packages

Analyzing Kernel Version

Understanding Kernel Directory Structure

Installing the Kernel

Exercise 11-3: Download and Install a New Kernel

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 11-1: Enable Verbose System Boot

Lab 11-2: Reset root User Password

Lab 11-3: Install New Kernel

12. System Initialization, Message Logging, and System Tuning

System Initialization and Service Management

Units

Targets

The systemctl Command

Listing and Viewing Units

Managing Service Units

Managing Target Units

System Logging

The Syslog Configuration File

Rotating Log Files

The Boot Log File

The System Log File

Logging Custom Messages

The systemd Journal

Retrieving and Viewing Messages

Preserving Journal Information

Exercise 12-1: Configure Persistent Storage for Journal Information

System Tuning

Tuning Profiles

The tuned-adm Command

Exercise 12-2: Manage Tuning Profiles

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 12-1: Modify Default Boot Target

Lab 12-2: Record Custom Alerts

Lab 12-3: Apply Tuning Profile

13. Basic Storage Partitioning

Storage Management Overview

Master Boot Record (MBR)

GUID Partition Table (GPT)

Disk Partitions

Storage Management Tools

Thin Provisioning

Adding Storage for Practice

Exercise 13-1: Add Required Storage to server2

MBR Storage Management with parted

Exercise 13-2: Create an MBR Partition

Exercise 13-3: Delete an MBR Partition

GPT Storage Management with gdisk

Exercise 13-4: Create a GPT Partition

Exercise 13-5: Delete a GPT Partition

Storage Optimization with Virtual Data Optimizer (VDO)

How VDO Conserves Storage Space

Creating and Managing VDO Volumes

Exercise 13-6: Install Software and Activate VDO

Exercise 13-7: Create a VDO Volume

Exercise 13-8: Delete a VDO Volume

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 13-1: Create and Remove Partitions with parted

Lab 13-2: Create and Remove Partitions with gdisk

Lab 13-3: Create and Delete VDO Volumes

Lab 13-4: Disable and Enable VDO Volume Features

14. Advanced Storage Partitioning

Logical Volume Manager (LVM)

Physical Volume

Volume Group

Physical Extent

Logical Volume

Logical Extent

LVM Operations and Commands

Exercise 14-1: Create a Physical Volume and Volume Group

Exercise 14-2: Create Logical Volumes

Exercise 14-3: Extend a Volume Group and a Logical Volume

Exercise 14-4: Rename, Reduce, Extend, and Remove Logical Volumes

Exercise 14-5: Reduce and Remove a Volume Group

Exercise 14-6: Uninitialize Physical Volumes

Stratis Volume-Managing File System

Stratis Management Operations and Command

Exercise 14-7: Install Software and Activate Stratis

Exercise 14-8: Create and Confirm a Pool and File System

Exercise 14-9: Expand and Rename a Pool and File System

Exercise 14-10: Destroy a File System and Pool

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 14-1: Create Volume Group and Logical Volumes

Lab 14-2: Expand Volume Group and Logical Volume

Lab 14-3: Reduce and Remove Logical Volumes

Lab 14-4: Remove Volume Group and Physical Volumes

Lab 14-5: Create Stratis Pool

Lab 14-6: Expand and Destroy Stratis Pool

15. Local File Systems and Swap

File Systems and File System Types

Extended File Systems

XFS File System

VFAT File System

ISO9660 File System

File System Management

File System Administration Commands

Mounting and Unmounting File Systems

Determining the UUID of a File System

Labeling a File System

Automatically Mounting a File System at Reboots

Monitoring File System Usage

Calculating Disk Usage

Exercise 15-1: Create and Mount Ext4, VFAT, and XFS File Systems in Partitions

Exercise 15-2: Create and Mount XFS File System in VDO Volume

Exercise 15-3: Create and Mount Ext4 and XFS File Systems in LVM Logical Volumes

Exercise 15-4: Resize Ext4 and XFS File Systems in LVM Logical Volumes

Exercise 15-5: Create, Mount, and Expand XFS File System in Stratis Volume

Swap and its Management

Determining Current Swap Usage

Prioritizing Swap Spaces

Swap Administration Commands

Exercise 15-6: Create and Activate Swap in Partition and Logical Volume

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 15-1: Create VFAT, Ext4, and XFS File Systems in Partitions & Mount Persistently

Lab 15-2: Create XFS File System in VDO Volume and Mount Persistently

Lab 15-3: Create Ext4 and XFS File Systems in LVM Volumes and Mount Persistently

Lab 15-4: Extend Ext4 and XFS File Systems in LVM Volumes

Lab 15-5: Create XFS File System in Stratis Volume and Mount Persistently

Lab 15-6: Create Swap in Partition and LVM Volume and Activate Persistently

16. Networking, Network Devices, and Network Connections

Networking Fundamentals

Hostname

Exercise 16-1: Change System Hostname
IPv4 Address
Network Classes
Subnetting
Subnet Mask
Classless Inter-Domain Routing (CIDR) Notation
Protocol
TCP and UDP Protocols
Well-Known Ports
ICMP Protocol
Ethernet Address
IPv6 Address
Major Differences between IPv4 and IPv6

Network Devices and Connections

Consistent Network Device Naming
Understanding Interface Connection Profile
Exercise 16-2: Add Network Devices to server10 and server20
Network Device and Connection Administration Tools
Exercise 16-3: Configure New Network Connection
Manually
The NetworkManager Service
The nmcli Command
Exercise 16-4: Configure New Network Connection Using nmcli
Understanding Hosts Table
Testing Network Connectivity
Exercise 16-5: Update Hosts Table and Test Connectivity

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 16-1: Add New Interface and Configure Connection Profile with nmcli
Lab 16-2: Add New Interface and Configure Connection Profile Manually

17. Network File System

Network File System

Benefits of Using NFS

NFS Versions

NFS Server and Client Configuration

Exercise 17-1: Export Share on NFS Server

Exercise 17-2: Mount Share on NFS Client

Auto File System (AutoFS)

Benefits of Using AutoFS

How AutoFS Works

AutoFS Configuration File

AutoFS Maps

Exercise 17-3: Access NFS Share Using Direct Map

Exercise 17-4: Access NFS Share Using Indirect Map

Automounting User Home Directories

Exercise 17-5: Automount User Home Directories Using Indirect Map

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 17-1: Configure NFS Share and Automount with Direct Map

Lab 17-2: Automount NFS Share with Indirect Map

18. Time Synchronization and Hostname Resolution

Time Synchronization

Time Sources

NTP Roles

Stratum Levels

Chrony Configuration File

Chrony Daemon and Command

Exercise 18-1: Configure NTP Client

Displaying and Setting System Date and Time

DNS and Name Resolution

DNS Name Space and Domains

DNS Roles

Understanding Resolver Configuration File

Performing Name Resolution with dig

Performing Name Resolution with host

Performing Name Resolution with nslookup

Performing Name Resolution with getent

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 18-1: Modify System Date and Time

Lab 18-2: Configure Chrony

19. The Secure Shell Service

The OpenSSH Service

Common Encryption Techniques

Authentication Methods

OpenSSH Protocol Version and Algorithms

OpenSSH Packages

OpenSSH Server Daemon and Client Commands

Server Configuration File

Client Configuration File

System Access and File Transfer

Exercise 19-1: Access RHEL System from Another RHEL System

Exercise 19-2: Generate, Distribute, and Use SSH Keys

Executing Commands Remotely Using ssh

Copying Files Remotely Using scp

Transferring Files Remotely Using sftp

Synchronizing Files Remotely Using rsync

Chapter Summary

[Review Questions](#)

[Answers to Review Questions](#)

[Do-It-Yourself Challenge Labs](#)

[Lab 19-1: Establish Key-Based Authentication](#)

[Lab 19-2: Test the Effect of PermitRootLogin Directive](#)

20. The Linux Firewall

[Firewall Overview](#)

[Overview of firewalld](#)

[firewalld Zones](#)

[Zone Configuration Files](#)

[firewalld Services](#)

[Service Configuration Files](#)

[Firewall Management](#)

[The firewall-cmd Command](#)

[Querying the Operational Status of firewalld](#)

[Exercise 20-1: Add Services and Ports, and Manage Zones](#)

[Exercise 20-2: Remove Services and Ports, and Manage Zones](#)

[Exercise 20-3: Test the Effect of Firewall Rule](#)

[Chapter Summary](#)

[Review Questions](#)

[Answers to Review Questions](#)

[Do-It-Yourself Challenge Labs](#)

[Lab 20-1: Add Service to Firewall](#)

[Lab 20-2: Add Port Range to Firewall](#)

21. Security Enhanced Linux

[Security Enhanced Linux](#)

[Terminology](#)

[SELinux Contexts for Users](#)

[SELinux Contexts for Processes](#)

[SELinux Contexts for Files](#)

Copying, Moving, and Archiving Files with SELinux Contexts

SELinux Contexts for Ports

Domain Transitioning

SELinux Booleans

SELinux Administration

Management Commands

Viewing and Controlling SELinux Operational State

Querying Status

Exercise 21-1: Modify SELinux File Context

Exercise 21-2: Add and Apply File Context

Exercise 21-3: Add and Delete Network Ports

Exercise 21-4: Copy Files with and without Context

Exercise 21-5: View and Toggle SELinux Boolean Values

Monitoring and Analyzing SELinux Violations

Chapter Summary

Review Questions

Answers to Review Questions

Do-It-Yourself Challenge Labs

Lab 21-1: Disable and Enable the SELinux Operating Mode

Lab 21-2: Modify Context on Files

Lab 21-3: Add Network Port to Policy Database

Lab 21-4: Copy Files with and without Context

Lab 21-5: Flip SELinux Booleans

22. Shell Scripting

Shell Scripts

Script01: Displaying System Information

Executing a Script

Debugging a Script

Script02: Using Local Variables

Script03: Using Pre-Defined Environment Variables

Script04: Using Command Substitution

Understanding Shell Parameters

Script05: Using Special and Positional Parameters

Script06: Shifting Command Line Arguments
Logical Constructs
Exit Codes
Test Conditions
The if-then-fi Construct
Script07: The if-then-fi Construct
The if-then-else-fi Construct
Script08: The if-then-else-fi Construct
The if-then-elif-fi Construct
Script09: The if-then-elif-fi Construct (Example 1)
Script10: The if-then-elif-fi Construct (Example 2)

Looping Constructs
Test Conditions
The for Loop
Script11: Print Alphabets Using for Loop
Script12: Create Users Using for Loop

Chapter Summary
Review Questions
Answers to Review Questions

DIY Challenge Labs
Lab 22-1: Write a Script to Create Logical Volumes
Lab 22-2: Write a Script to Create File Systems
Lab 22-3: Write a Script to Configure a New Network Profile

23. Containers

Introduction to Containers
Containers and the Linux Features
Benefits of Using Containers
Container Home: Bare Metal or Virtual Machine
Container Images and Container Registries
Root vs. Rootless Containers

Working with Images and Containers
Exercise 23-1: Install Necessary Container Support
The podman Command

The skopeo Command
The registries.conf File
Viewing Podman Configuration and Version

Image Management

Exercise 23-2: Search, Examine, Download, and Remove an Image

Basic Container Management

Exercise 23-3: Run, Interact with, and Remove a Named Container

Exercise 23-4: Run a Nameless Container and Auto-Remove it After Entry Point Command Execution

Advanced Container Management

Containers and Port Mapping

Exercise 23-5: Configure Port Mapping

Exercise 23-6: Stop, Restart, and Remove a Container

Containers and Environment Variables

Exercise 23-7: Pass and Set Environment Variables

Containers and Persistent Storage

Exercise 23-8: Attach Persistent Storage and Access Data Across Containers

Container State Management with systemd

Exercise 23-9: Configure a Root Container as a systemd Service

Exercise 23-10: Configure a Rootless Container as a systemd Service

Chapter Summary

Review Questions

Answers to Review Questions

DIY Challenge Labs

Lab 23-1: Prepare to Launch Containers

Lab 23-2: Launch a Named Root Container with Port Mapping

Lab 23-3: Launch a Nameless Rootless Container with Two Variables

[Lab 23-4: Launch a Named Rootless Container with Persistent Storage](#)

[Lab 23-5: Launch a Named Rootless Container with Port Mapping, Environment Variables, and Persistent Storage](#)

[Lab 23-6: Control Rootless Container States via systemd](#)

[Lab 23-7: Control Root Container States via systemd](#)

Appendix A: Sample RHCSA Exam 1

Appendix B: Sample RHCSA Exam 2

Appendix C: Sample RHCSA Exam 3

Appendix D: Sample RHCSA Exam 4

Glossary

Index

List of Figures

- Figure 1-1 Lab Setup
- Figure 1-2 VirtualBox Website
- Figure 1-3 VirtualBox Download
- Figure 1-4 VirtualBox Installation 1
- Figure 1-5 VirtualBox Installation 2
- Figure 1-6 VirtualBox Installation 3
- Figure 1-7 VirtualBox Installation 4
- Figure 1-8 VirtualBox Installation 5
- Figure 1-9 VirtualBox Installation 6
- Figure 1-10 Virtual Machine Creation 1
- Figure 1-11 Virtual Machine Creation 2
- Figure 1-12 Virtual Machine Creation 3
- Figure 1-13 Virtual Machine Creation 4
- Figure 1-14 Virtual Machine Creation 5
- Figure 1-15 Virtual Machine Creation 6
- Figure 1-16 Virtual Machine Creation 7
- Figure 1-17 Virtual Machine Creation 8
- Figure 1-18 Red Hat User Account Creation 1
- Figure 1-19 Red Hat User Account Creation 2
- Figure 1-20 Red Hat Developer Login
- Figure 1-21 Attach ISO Image to VM
- Figure 1-22 Boot Menu
- Figure 1-23 Language Selection
- Figure 1-24 Installation Summary | Main
- Figure 1-25 Installation Summary | Time & Date
- Figure 1-26 Installation Summary | Installation Source
- Figure 1-27 Installation Summary | Software Selection
- Figure 1-28 Installation Summary | Installation Destination
- Figure 1-29 Installation Summary | Network & Hostname
- Figure 1-30 Installation Summary | Network & Hostname | Configure
- Figure 1-31 Installation Summary | Network & Hostname
- Figure 1-32 Installation Summary | Begin Installation
- Figure 1-33 Configuration | User Settings
- Figure 1-34 Configuration | User Settings | Create User
- Figure 1-35 Configuration | Finishing Installation
- Figure 1-36 VirtualBox Manager | System | Boot Order

Figure 1-37 VirtualBox Manager | System | Boot Order | Alter
Figure 1-38 Initial Setup
Figure 1-39 Graphical Desktop | Sign-in Screen
Figure 1-40 First Time User Login
Figure 1-41 First Time User Login | Getting Started
Figure 1-42 GNOME Desktop Environment
Figure 1-43 GNOME Desktop Environment| Log Out
Figure 2-1 GNOME Display Manager
Figure 2-2 GNOME Desktop Environment
Figure 2-3 GNOME Desktop Environment | Activities
Figure 2-4 Linux Directory Structure
Figure 2-5 Terminal Session
Figure 2-6 Red Hat's Webpage for RHEL 8 Documentation
Figure 3-1 Hard Link
Figure 3-2 Soft Link
Figure 4-1 Permission Weights
Figure 4-2 find Command Syntax
Figure 5-1 The passwd File
Figure 5-2 The shadow File
Figure 5-3 The group File
Figure 5-4 The gshadow File
Figure 8-1 Process State Transition
Figure 8-2 Syntax of Crontables
Figure 11-1 GRUB2 Menu
Figure 11-2 GRUB2 Kernel Edit
Figure 11-3 GRUB2 Commands
Figure 11-4 Anatomy of a Kernel Version
Figure 13-1 VirtualBox Interface
Figure 13-2 VirtualBox – Add Storage
Figure 13-3 VirtualBox – Adjust Disk Name and Size
Figure 13-4 VirtualBox – 7 New Disks Added
Figure 14-1 LVM Structure
Figure 14-2 Stratis Structure
Figure 16-1 VirtualBox – Add Network Interface
Figure 17-1 NFS Server/Client
Figure 18-1 NTP Stratum Levels
Figure 18-2 Sample DNS Hierarchy
Figure 22-1 Shell Parameters
Figure 22-2 The if-then-fi Construct
Figure 22-3 The if-then-else-fi Construct
Figure 22-4 The if-then-el-if-fi Construct
Figure 22-5 The for Loop
Figure 23-1 Container Home: Bare Metal or Virtual Machine

List of Tables

- Table 1-1 Installation Logs
- Table 1-2 Installation Summary | Software Selection | Base Environments
- Table 2-1 tree Command Options
- Table 2-2 ls Command Options
- Table 2-3 Navigating within Manual Pages
- Table 2-4 Navigating within info Documentation
- Table 3-1 tar Command Options
- Table 3-2 tar with Compression Options
- Table 3-3 Inserting Text
- Table 3-4 Navigating within vim
- Table 3-5 Deleting Text
- Table 3-6 Undoing and Repeating
- Table 3-7 Searching for Text
- Table 3-8 Replacing Text
- Table 3-9 Copying, Moving, and Pasting Text
- Table 3-10 Changing Text
- Table 3-11 Saving and Quitting vim
- Table 3-12 Navigating with less and more
- Table 3-13 wc Command Options
- Table 3-14 Copying vs. Linking
- Table 4-1 Octal Permission Notation
- Table 4-2 setfacl Command Format for Access ACLs
- Table 4-3 setfacl Command Switches
- Table 5-1 login.defs File Directives
- Table 5-2 useradd Command Options
- Table 5-3 usermod Command Options
- Table 6-1 chage Command Options
- Table 6-2 passwd Command Options
- Table 6-3 usermod Command Options for User Lock/Unlock
- Table 6-4 groupadd Command Options
- Table 7-1 Common Predefined Environment Variables
- Table 7-2 Helpful Command Line Editing Shortcuts
- Table 7-3 Predefined Aliases
- Table 7-4 Job Control Commands and Control Sequences
- Table 7-5 System-wide Startup Files
- Table 7-6 Per-user Startup Files

- [Table 8-1 ps Command Output Description](#)
- [Table 8-2 Control Signals](#)
- [Table 8-3 User Access Restrictions to Scheduling Tools](#)
- [Table 8-4 Crontable Syntax Explained](#)
- [Table 9-1 rpm Command Query Options](#)
- [Table 9-2 rpm Command Install/Remove/Verify Options](#)
- [Table 9-3 Red Hat GPG Key Files](#)
- [Table 9-4 Package Verification Codes](#)
- [Table 9-5 File Type Codes](#)
- [Table 10-1 Directive Settings in dnf.conf File](#)
- [Table 10-2 dnf Subcommands for Packages and Repositories](#)
- [Table 10-3 dnf Subcommands for Package Groups and Modules](#)
- [Table 11-1 GRUB2 Default Settings](#)
- [Table 11-2 Kernel Packages](#)
- [Table 12-1 systemd Unit Types](#)
- [Table 12-2 systemd Targets](#)
- [Table 12-3 systemctl Subcommands](#)
- [Table 12-4 Journal Data Storage Options](#)
- [Table 12-5 Tuning Profiles](#)
- [Table 13-1 Common parted Subcommands](#)
- [Table 13-2 vdo Subcommands](#)
- [Table 14-1 Common LVM Operations and Commands](#)
- [Table 14-2 Common Stratis Subcommands](#)
- [Table 15-1 File System Types](#)
- [Table 15-2 File System Management Commands](#)
- [Table 15-3 Common mount Command Options](#)
- [Table 16-1 IPv4 vs IPv6](#)
- [Table 16-2 Network Connection Configuration Directives](#)
- [Table 16-3 Basic Network Management Tools](#)
- [Table 16-4 Network Connection and Device Administration Tools](#)
- [Table 17-1 AutoFS Directives](#)
- [Table 18-1 Chrony Directives](#)
- [Table 18-2 The Resolver Configuration File](#)
- [Table 18-3 Name Service Source and Order Determination](#)
- [Table 19-1 OpenSSH Client Tools](#)
- [Table 19-2 OpenSSH Server Configuration File](#)
- [Table 19-3 OpenSSH Client Configuration File](#)
- [Table 20-1 firewalld Default Zones](#)
- [Table 20-2 Common firewall-cmd Options](#)
- [Table 21-1 SELinux Management Commands](#)
- [Table 22-1 Test Conditions](#)
- [Table 22-2 Arithmetic Operators](#)
- [Table 23-1 Common podman Subcommands](#)

Chapter 01

Local Installation

This chapter describes the following major topics:

- A quick look at Linux and Open Source
- Linux distribution from Red Hat
- Recommended lab setup for RHCSA exam preparation
- Overview of the installer program
- Where are installation messages stored?
- What are virtual console screens?
- Download and install VirtualBox
- Create virtual machine
- Download and install Red Hat Enterprise Linux 8 in virtual machine
- Execute post-installation configuration tasks
- Log in and out at the graphical console
- Log in and out at over the network

RHCSA Objectives:

04. Access remote systems using ssh

This chapter sets up the foundation for learning and practicing the exam objectives for RHCSA

Linux is a free operating system and it has been in existence for almost three decades. Its source code is available to developers, amateurs, and general public for enhancements and customization. Red Hat Inc. modifies a copy of a selected version of Linux source code and introduces features, adds improvements, and fixes bugs. The company packages the updated version as a Linux distribution of their own for commercial purposes. This distribution is thoroughly tested to run smoothly and perform well on a wide range of computer hardware platforms. It is stable, robust, feature-rich, and is ready to host workload of any size.

Red Hat Enterprise Linux may be downloaded for learning, practicing, and preparing for the RHCSA exam. It is available as a single installable image file. A lab environment is necessary to practice the procedures to solidify the understanding of the concepts and tools learned. The install process requires careful planning to identify critical system configuration pieces prior to launching the installer program. Once the operating system is installed, users can log in at the console or over the network.

A Quick Look at Linux Development

Linux is a free computer operating system (OS) that is similar to the UNIX OS in terms of concepts, features, functionality, and stability. It is referred to as a UNIX-like operating system.

Linux powers an extensive range of computer hardware platforms, from laptop and desktop computers to massive mainframes and supercomputers. Linux also runs as the base OS on networking, storage, gaming, smart television, and mobile devices. Numerous vendors, including Red Hat, IBM, Canonical, Oracle, DXC Technology, Novell, and Dell, offer commercial support to Linux users worldwide.

Linux is the main alternative to proprietary UNIX and Windows operating systems because of its functionality, adaptability, portability, and cost-effectiveness. At present, over one hundred different Linux distributions are circulating from various vendors, organizations, non-

profit groups, and individuals, though only a few of them are popular and widely recognized.

Linux is largely used in government agencies, corporate businesses, academic institutions, scientific organizations, as well as in home computers. Linux deployment and usage are constantly on the rise.

Linux History in a Nutshell

In 1984, Richard Stallman, an American software engineer, had a goal to create a completely free UNIX-compatible open source (non-proprietary) operating system. The initiative was called the GNU Project (*GNU's Not Unix*) and by 1991, significant software had been developed. The only critical piece missing was a core software component called *kernel* to drive and control the GNU software and to regulate its communication with the hardware.

Around the same time, Finnish computer science student Linus Torvalds developed a kernel and proclaimed its availability. The new kernel was named *Linux*, and it was gradually integrated with the GNU software to form what is now referred to as *GNU/Linux*, *Linux operating system*, or simply *Linux*.

Linux was released under the GNU General Public License (GPL). Initially written to run on Intel x86-based computers, the first version (0.01) was released in September 1991 with little more than 10,000 lines of code. In 1994, the first major release (1.0.0) was introduced, followed by a series of successive major and minor versions until the version 4.0 in 2015. Development and enhancements continued, and version 4.0 was followed by several stable versions. At the time of this writing, version 4.19, with its millions of lines of code, is the latest stable kernel.

The Linux kernel, and the operating system in general, has been enhanced with contributions from tens of thousands of software programmers, amateurs, and organizations around the world into a large and complex system under GNU GPL, which provides public access to its source code free of charge and with full consent to amend, package, and redistribute.

Linux from Red Hat

Red Hat, Inc., founded in 1993, used the available Linux source code and created one of the first commercial Linux operating system distribution called *Red Hat Linux* (RHL). The company released the first version 1.0 in November 1994. Several versions followed until the last version in the series, Red Hat Linux 9 (later referred to as RHEL 3), based on kernel 2.4.20, was released in March 2003. Red Hat renamed their Red Hat Linux brand as *Red Hat Enterprise Linux* (RHEL) commencing 2003.

RHL was originally assembled and enhanced within the Red Hat company. In 2003, Red Hat sponsored and facilitated the *Fedora Project* and invited the user community to join hands in enhancing and updating the source code. This project served as the test bed for developing and testing new features and enabled Red Hat to include the improved code in successive versions of RHEL.

The Fedora distribution is completely free, while RHEL is commercial. RHEL 4 (based on kernel 2.6.9 and released in February 2005), RHEL 5 (based on kernel 2.6.18 and released in March 2007), RHEL 6 (based on kernel 2.6.32 and released in November 2010), RHEL 7 (based on kernel 3.10 and released in June 2014), and RHEL 8 (based on kernel 4.18 and released in May 2019) have been built using Fedora distributions 3, 6, 12, 13, 19, and 28, respectively.

RHEL 8 has been tested to run on bare-metal computer hardware, virtualized platforms, high-end graphics workstations, IBM Power little endian, IBM System Z, and in the cloud.

Lab Infrastructure for Practice

RHEL 8 is available as a free download from Red Hat for Intel and AMD processor machines. You will need to create a free Red Hat user account in order to download it. The downloaded image file can then be attached to a *Virtual Machine* (VM) as an ISO image, burned to a DVD to support installation on a physical computer, or placed on a remote server for network-based installations via HTTP, FTP, or NFS protocols.



An ISO image is a single file that represents the content of an entire DVD or CD.

Burning the image to a DVD and configuring a server for network-based installation are beyond the scope of this book. This chapter will focus on installing the operating system with an ISO image.

What is Needed for the Lab?

Throughout this book, there will be several discussions about system, network, and security, along with examples on how to implement and administer them. Each chapter will contain a number of exercises that will help you perform certain tasks and execute commands.

You'll need a laptop or a desktop computer with at least a dual-core processor, 8GB of physical memory, and 27GB of free storage space to run two virtual machines with required storage. If you want to use the static IP addresses on your home router, make sure that you keep a map between them, and the ones provided below to avoid any confusion. The computer must have hardware virtualization support enabled in the BIOS to allow for 64-bit OS installation. Here is a snapshot of what is needed and how it will be configured:

Base Operating System:	Windows 10 or MacOS 10.12 or higher
Hypervisor Software:	Oracle VirtualBox (VB) 5.2.24 or higher
Number of VMs:	2
vCPUs in each VM:	1
OS in each VM:	RHEL 8.0 or 8.2 (the RHCSA exam is now delivered on RHEL 8.2)
VM1 (RHEL8-VM1):	server1.example.com with static IP 192.168.0.110, 1024MB memory, 1x10GB virtual disk for OS, and one virtual network device. This VM will be built using the RHEL 8 ISO image. Exercises 1-1 and 1-2 will walk through the process of installation. In Chapter 16, “Networking, Network Devices, and Network Connections”, you will add another virtual network device and will rename this server to server10.example.com .
VM2 (RHEL8-VM2):	server2.example.com with static IP 192.168.0.120, 2048MB memory, 1x10GB virtual disk for OS, and one virtual network device. In Chapter 13, “Basic Storage Partitioning”, you will add 4x250MB data disks for parted and LVM exercises, 1x4GB data disk for VDO exercise and 2x1GB data disks for Stratis exercises that are presented in chapters 13, 14, and 15. You will build this

VM using the RHEL 8 ISO image by referencing the steps outlined in Exercises 1-1 and 1-2. In [Chapter 16](#), “Networking, Network Devices, and Network Connections”, you will add another virtual network device and will rename this server to [server20.example.com](#).

The entire setup for the lab is shown in [Figure 1-1](#).

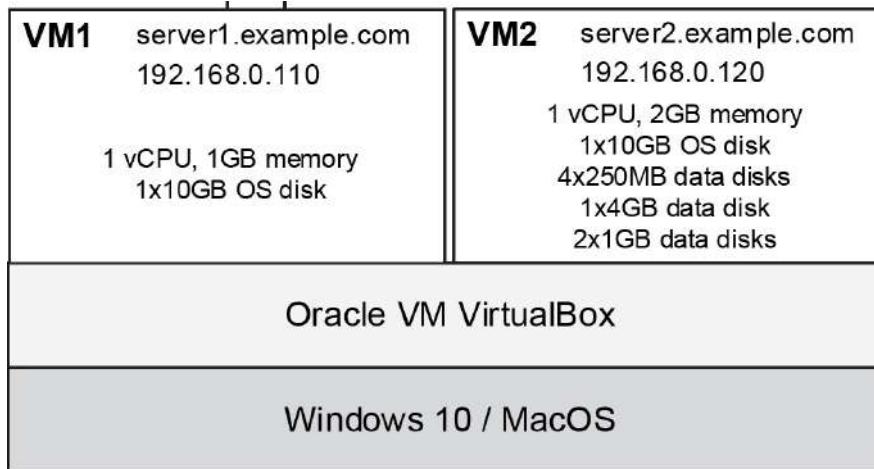


Figure 1-1 Lab Setup

You will install VirtualBox 5.2.24 (or higher) on Windows 10 or MacOS 10.12 (or higher). You may use VMware or other virtualization software as an alternative; however, all of the exercises and examples in this book reference VirtualBox (VB).

The RHEL Installer Program

The RHEL installer program is called *Anaconda*. There are several configuration options on the main screen that require modification before the installation process begins. Some of the questions are compulsory and must be answered appropriately while others are optional and may be skipped for post-installation setup.

The configuration can be done in any sequence that you prefer. You should have the minimum mandatory configuration data handy and be ready to enter it when prompted. Some of the key configuration items are language, keyboard type, time zone, disk partitioning, hostname/IP, software selection, root password, user information, and kdump.

Where Do Installation Logs Go?

There are plentiful log files created and updated as the installation progresses. These files record configuration and status information. You can view their contents after the installation has been completed to check how the installation proceeded. These files are described in [Table 1-1](#).

File	Description
/root/anaconda-ks.cfg	Records the configuration entered
/root/install.log	Lists the packages being installed
/root/install.log.syslog	Stores general messages
/var/log/anaconda/anaconda.log	Contains informational, debug, and other general messages
/var/log/anaconda/ifcfg.log	Captures messages related to network interface and connections
/var/log/anaconda/journal.log	Stores messages generated by many services and components during system installation
/var/log/anaconda/packaging.log	Records messages generated by the yum and rpm commands during software installation
/var/log/anaconda/program.log	Captures messages generated by external programs
/var/log/anaconda/storage.log	Records messages generated by storage modules
/var/log/anaconda/syslog	Records messages related to the kernel
/var/log/anaconda/X.log	Stores X Window System information
/tmp/yum.log	Contains messages related to yum packages

Table 1-1 Installation Logs

Files in the `/var/log/anaconda` directory are actually created and updated in the `/tmp` directory during the installation; however, they are moved over once the installation is complete.

Virtual Console Screens

During the installation, there are six text-based virtual console screens available to monitor the process, view diagnostic messages, and discover and fix any issues encountered. The information displayed on

the console screens is captured in the installation log files ([Table 1-1](#)). You can switch between screens by pressing a combination of keys as described below.

Console 1 (Ctrl+Alt+F1): This is the main screen. Before Anaconda begins, you will select a language to use during installation, and then it will switch the default console to the sixth screen (Console 6).

Console 2 (Ctrl+Alt+F2): The screen displays the shell interface to run commands as the *root* user.

Console 3 (Ctrl+Alt+F3): This screen displays installation messages and stores them in */tmp/anaconda.log* file. This file also captures information on detected hardware, in addition to other data.

Console 4 (Ctrl+Alt+F4): This screen shows storage messages and records them in */tmp/storage.log* file.

Console 5 (Ctrl+Alt+F5): This screen displays program messages and logs them to */tmp/program.log* file.

Console 6 (Ctrl+Alt+F6): This is the default graphical configuration and installation console screen.

Exercise 1-1: Download and Install VirtualBox Software, and Create a Virtual Machine

In this exercise, you will download and install VirtualBox software. You will create a virtual machine to set up the foundation to install RHEL 8 for the next exercise.

EXAM TIP: Downloading and installing VirtualBox software and creating a virtual machine are not part of the exam objectives. These tasks have been included here only to support readers with building their own lab environment for practice.

Downloading and Installing VirtualBox

VirtualBox is available for free download and use. At the time of this writing, the latest version is 6.0; however, you can use any previous

5.x or a future version. Here is a quick guide on how to download and install the current version of VirtualBox on a Windows 10 computer.

1. Go to www.virtualbox.org (Figure 1-2) and click “Download VirtualBox 6.0”.

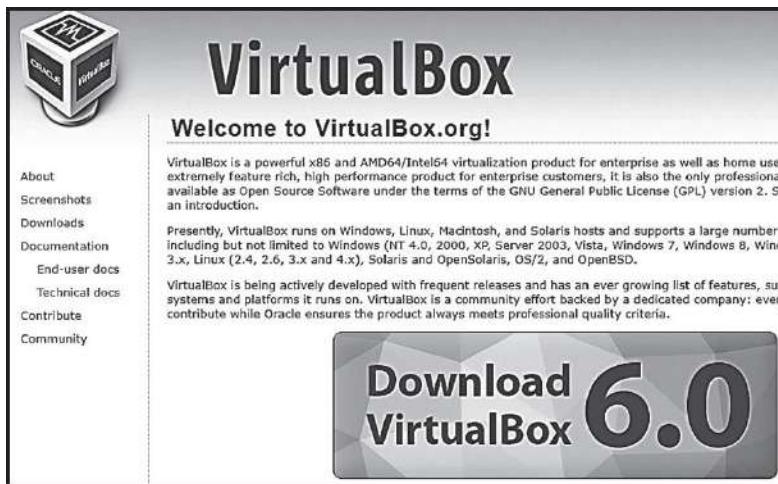


Figure 1-2 VirtualBox Website

2. On the next screen, click on “Windows hosts”. This will start the download to your computer.

A screenshot of the "Download VirtualBox" page. The main title "Download VirtualBox" is at the top. Below it is a sub-instruction: "Here you will find links to VirtualBox binaries and its source code." A section titled "VirtualBox binaries" follows, with a note: "By downloading, you agree to the terms and conditions of the resp...". Another note below states: "If you're looking for the latest VirtualBox 5.2 packages, see VirtualBox 5.2. VirtualBox 5.2 has been discontinued in 6.0. Version 5.2 will remain supported until July 20...". A final section titled "VirtualBox 6.0.10 platform packages" lists download links for various hosts:

- Windows hosts
- OS X hosts
- Linux distributions
- Solaris hosts

Figure 1-3 VirtualBox Download

The software is now available on your computer.

3. Double-click on the VirtualBox binary to start the installation.
Click Next to proceed on the first screen that appears.



Figure 1-4 VirtualBox Installation 1

4. If needed, choose a different location on the disk to install VirtualBox. Click Next to continue.

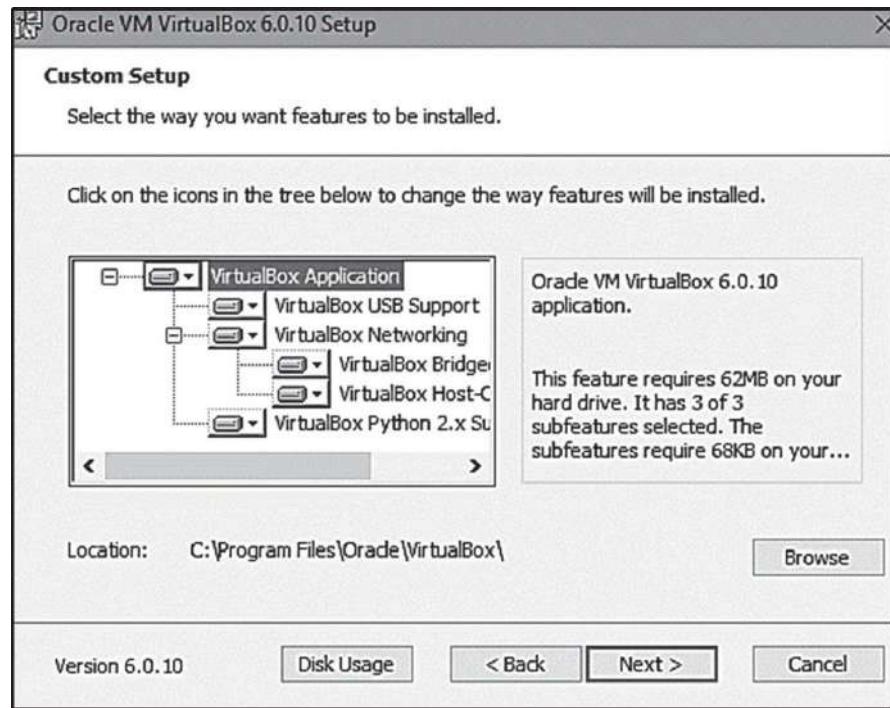


Figure 1-5 VirtualBox Installation 2

5. Untick any of the checked items if required and click Next to continue.

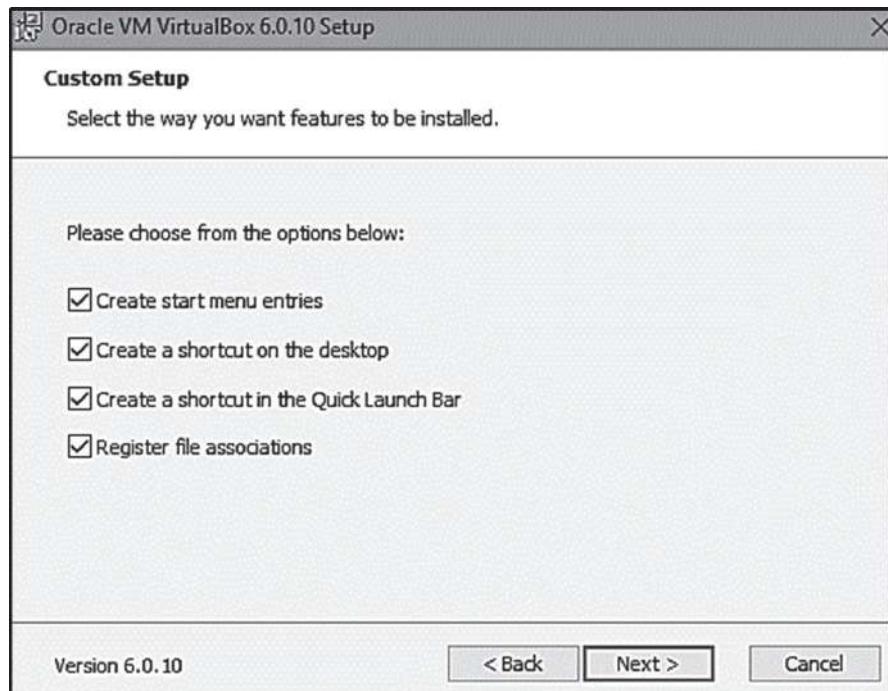


Figure 1-6 VirtualBox Installation 3

6. Accept the warning and continue by pressing Yes.

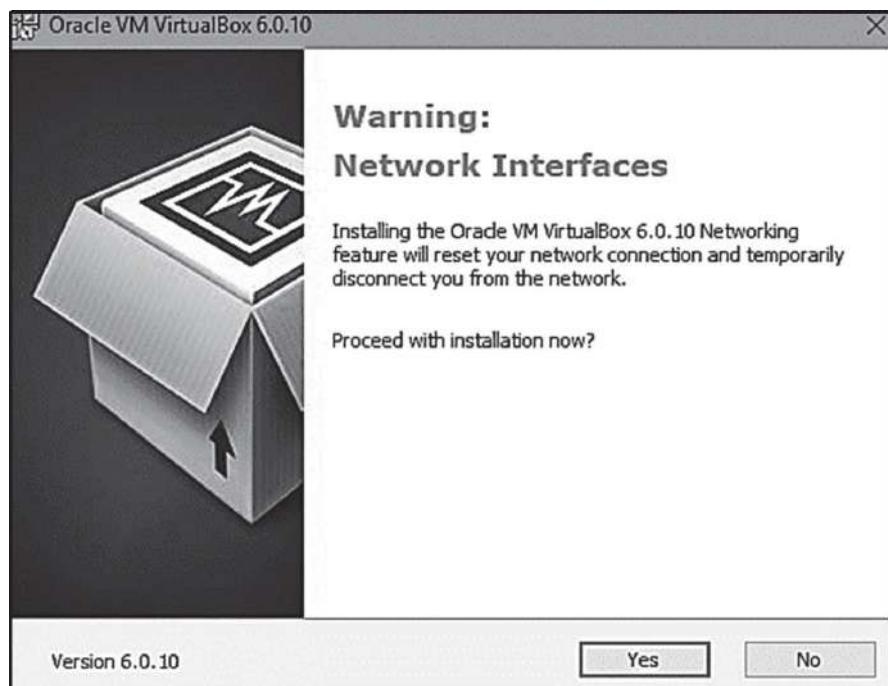


Figure 1-7 VirtualBox Installation 4

7. The setup wizard is now ready to begin the installation. Click Install to proceed.

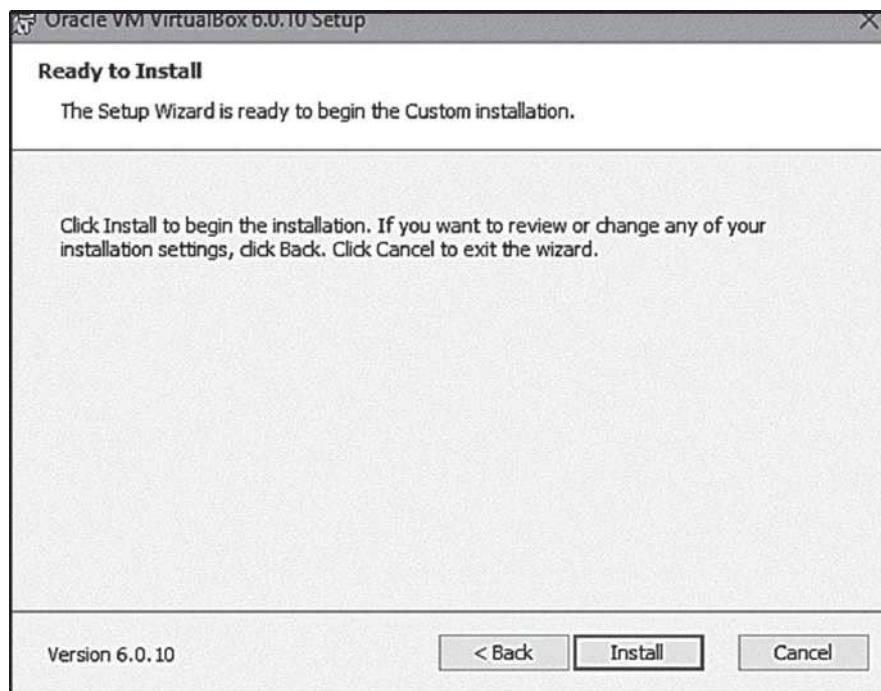


Figure 1-8 VirtualBox Installation 5

8. Click Finish to continue.

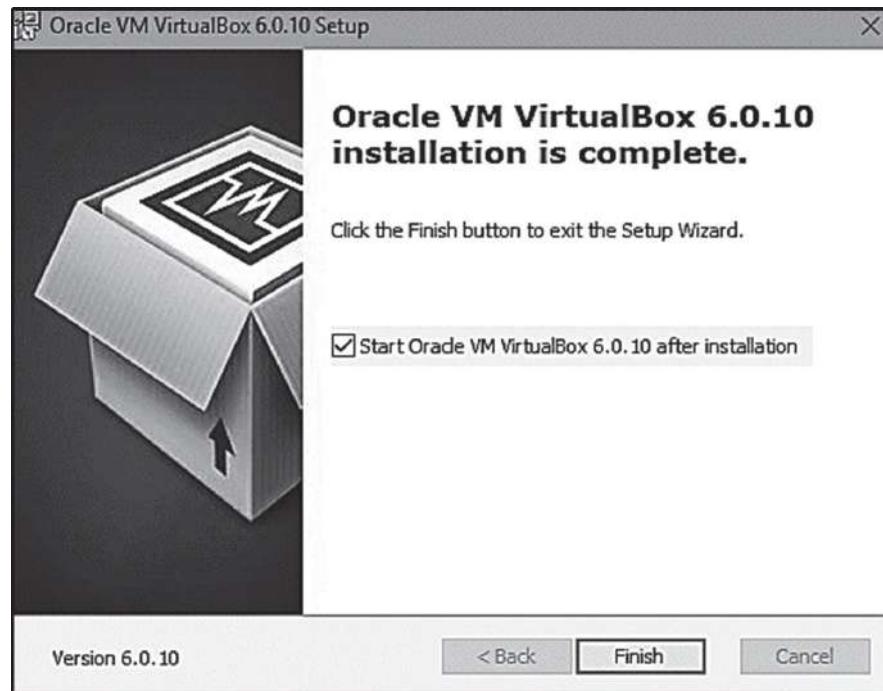


Figure 1-9 VirtualBox Installation 6

This brings the installation of VirtualBox to a successful completion. It will also launch the application.

Creating a Virtual Machine

Use VirtualBox to create the first virtual machine called *RHEL8-VM1* with specifications described earlier in this chapter. Here are the steps for the creation.

9. Launch VirtualBox if it is not already running. The interface looks similar to what is shown in [Figure 1-10](#).

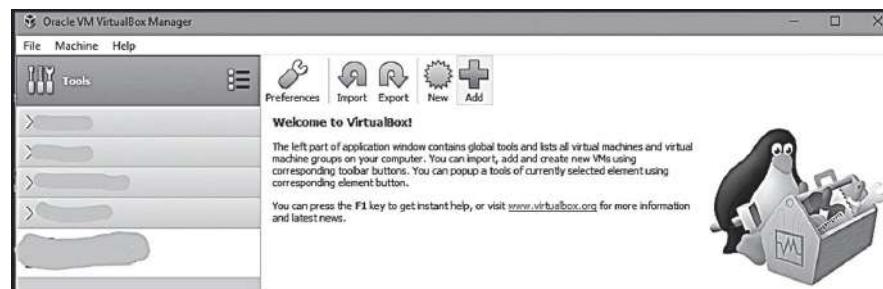


Figure 1-10 Virtual Machine Creation 1

10. Click on New on the top menu bar to start the virtual machine creation wizard (see [Figure 1-11](#)). Enter the name *RHEL8-VM1*, select Linux as the operating system type, and Red Hat (64-bit) as the version. For this demonstration, accept the default location to store the VM files on the C drive. Click Next to continue.

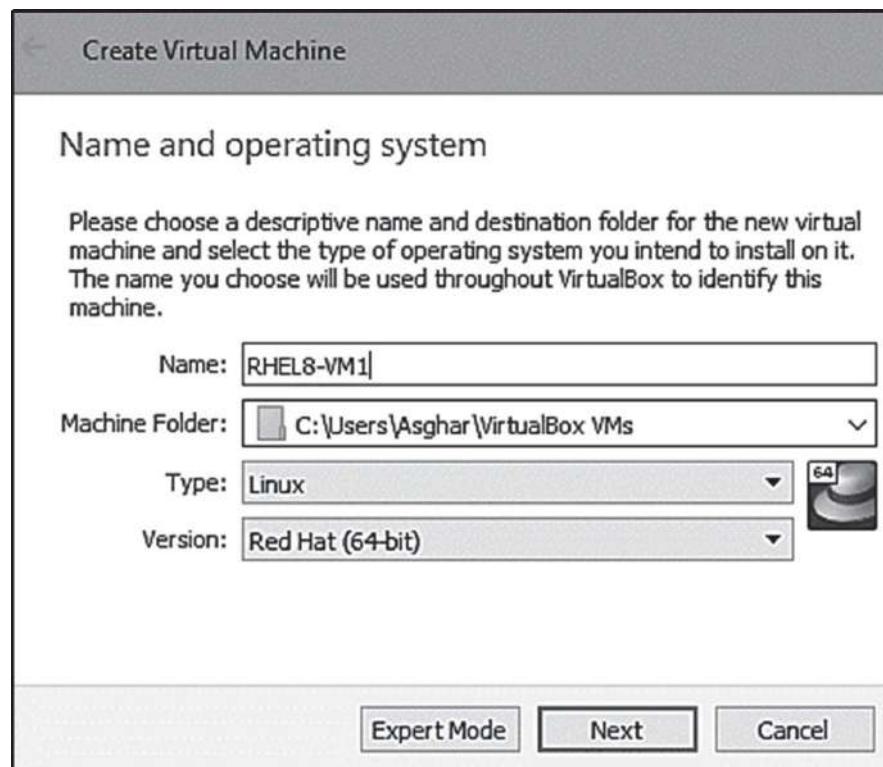


Figure 1-11 Virtual Machine Creation 2

11. In the next window, specify the memory size you want allocated to the VM. Accept the recommended 1GB for this VM and click Next.

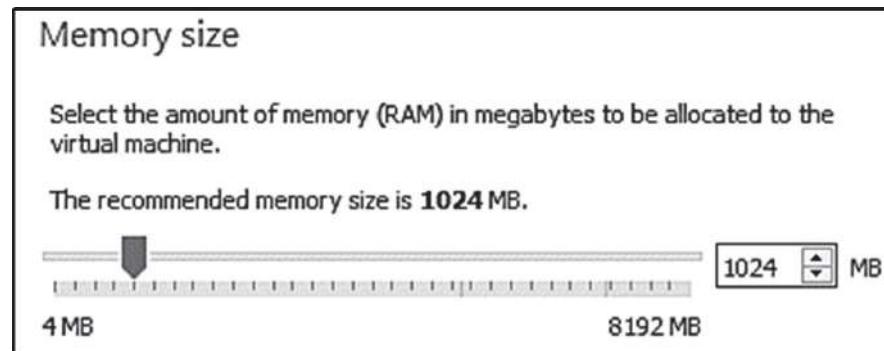


Figure 1-12 Virtual Machine Creation 3

12. The VM will need a hard disk to store the RHEL 8 operating system. For this demonstration, choose the creation of a virtual hard disk option, which is also the default selection. The other two options are irrelevant for this exercise. You will adjust the recommended hard disk size of 8GB in a later step.

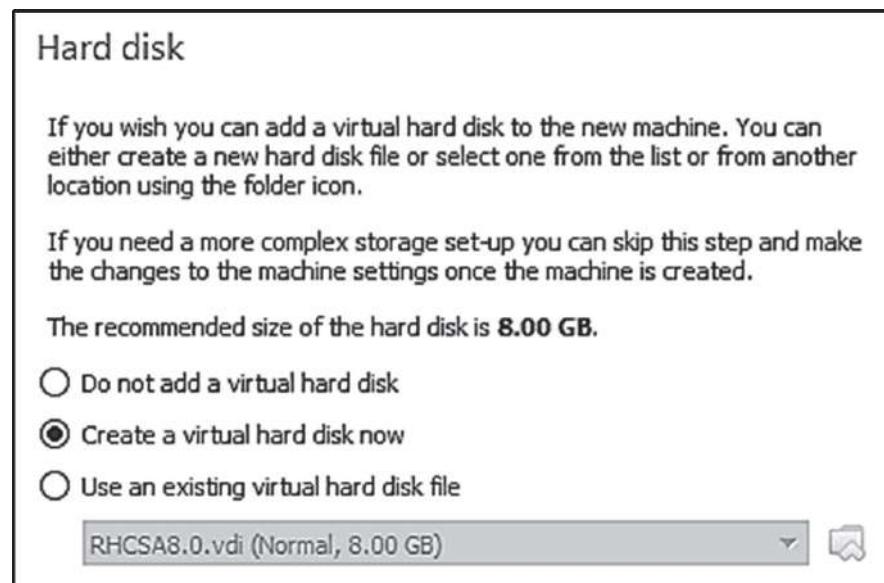


Figure 1-13 Virtual Machine Creation 4

13. The virtual hard disk is in essence a file. VirtualBox supports three different virtual hard disk file types: VDI (*VirtualBox Disk Image*), VHD (*Virtual Hard Disk*), and VMDK (*Virtual Machine Disk*). These file types represent VirtualBox, Microsoft, and

VMware disk image formats. Select the default, VDI, for this demonstration.

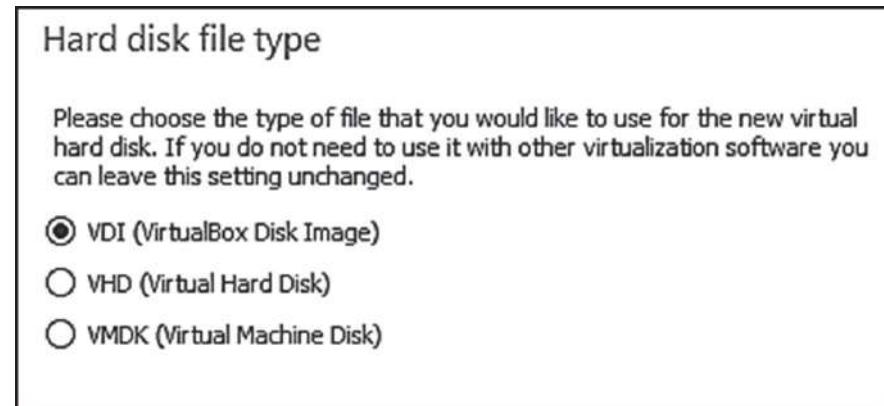


Figure 1-14 Virtual Machine Creation 5

14. There are a couple of methods to allocate storage for the OS. A fixed allocation method reserves the entire specified capacity for the disk right away and takes a while to create the disk. In contrast, the dynamically allocated option only uses the amount of disk space that is needed for the storage without reserving the entire disk capacity. This option is preferred over the former. Click Next to continue with the default option.



Figure 1-15 Virtual Machine Creation 6

15. The RHEL 8 virtual disk image will be stored on a computer disk. The name of the VDI file will match the name of the virtual machine name. You can also specify the size for the disk. Enter 10GB for the OS disk and leave the VDI file storage location to the default.

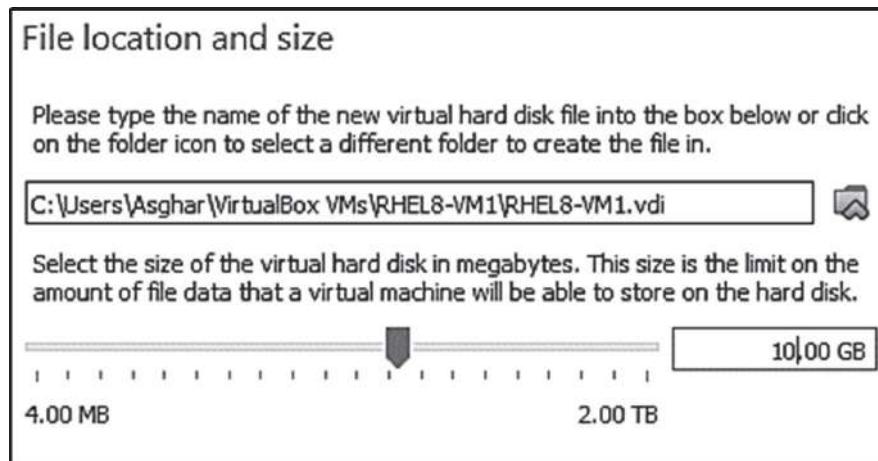


Figure 1-16 Virtual Machine Creation 7

16. Clicking Next on the previous window completes the VM creation process and ends the wizard. VirtualBox will have the VM listed along with its configuration (see [Figure 1-17](#)).

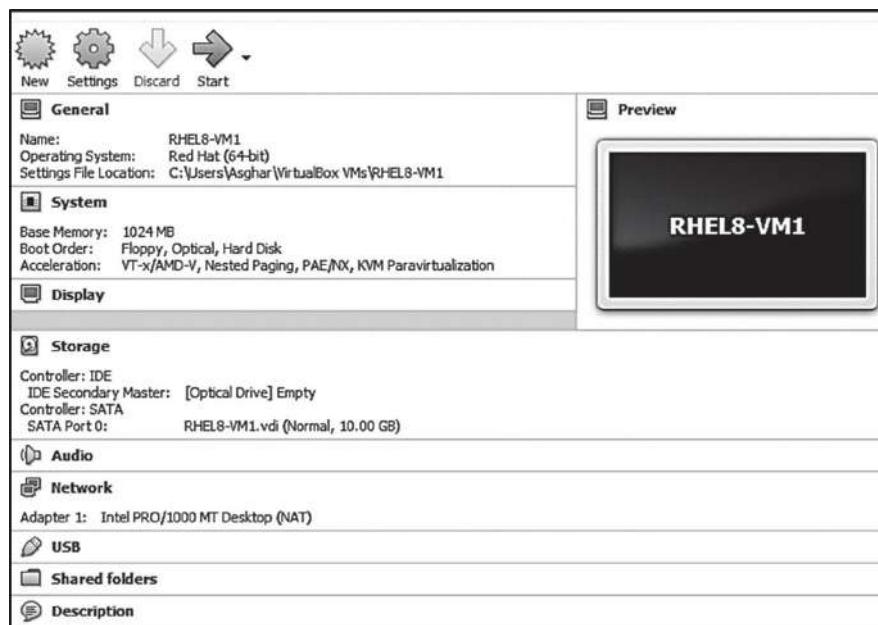


Figure 1-17 Virtual Machine Creation 8

There are a number of other configurable items as depicted in [Figure 1-17](#). You may have to adjust the display controller setting to VMSVGA under Display to view full screen content on the console and attach the network adapter to Bridged Adapter under Network for bi-directional communication with the Windows host and the Internet.

Exercise 1-2: Download and Install RHEL 8

This exercise will build *server1* in *RHEL8-VM1*.

In this exercise, you will download RHEL 8 and install it in *RHEL8-VM1* that you created in [Exercise 1-1](#). You will attach the RHEL 8 ISO image to the VM, name the Linux system [*server1.example.com*](#) and IP 192.168.0.110. Additional configuration will be supplied as the installation advances.

EXAM TIP: Downloading and installing RHEL 8 are beyond the scope of the exam objectives. They have been included here only to support the readers with building their own lab environment for practice.

The user creation, base environments, storage partitioning, network device and connection configuration, time synchronization, and other topics are not explained as part of this exercise; however, they will be discussed in later chapters.

Downloading RHEL 8 ISO Image

RHEL 8 image is available for free download from Red Hat Developer's webpage. You need to create a user account in order to log in and obtain a copy for yourself. Alternatively, you can use your credentials on Facebook, Google, LinkedIn, Twitter, etc. for login. For this demonstration, you will find instructions on how to open a new account and download the software.

1. Visit <https://developers.redhat.com/login> and click “Create one now”.

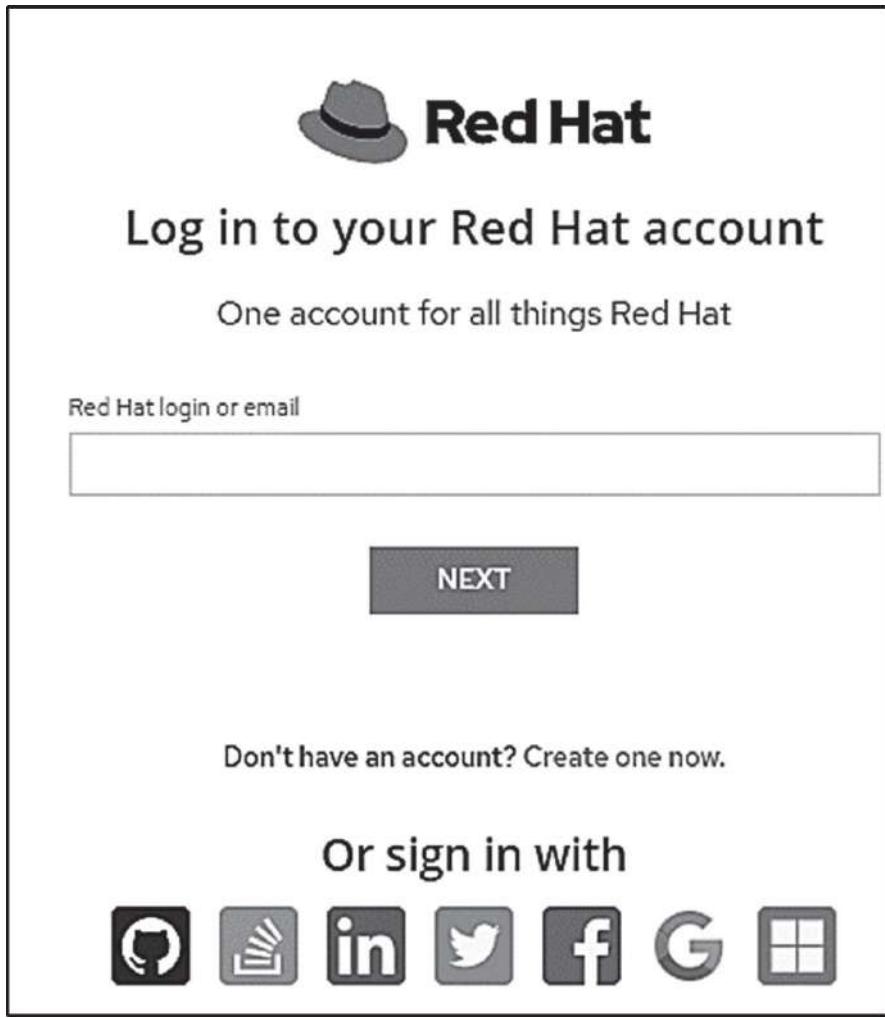


Figure 1-18 Red Hat User Account Creation 1

2. Fill out the form by providing a unique username, email address, and password. Make sure to checkmark the boxes to accept terms and conditions.



Create a Red Hat account

Sign up and use this Red Hat account to access all of Red Hat's applications, communities, support, and more.

* Required fields

Choose your username (Red Hat Login ID) *

User account for this username already exists. Log In

You can use this username (also known as your Red Hat Login ID) to log in to other Red Hat sites. It cannot be changed once created and it must be at least five characters.

Email address *

Choose a password *

 SHOW

Your password must be at least 8 characters long. A strong password combines lowercase letters, uppercase letters, numbers, and symbols.

I have read and agree to all the terms and conditions below (check all boxes).
 * I have read and agree to the Enterprise Agreement.
 * I have read and agree to the Developer Program Terms & Conditions.

I would like to receive the Red Hat Developer Program newsletter.

CREATE MY ACCOUNT

Figure 1-19 Red Hat User Account Creation 2

3. After an account has been created, go back to the login page <https://developers.redhat.com/login> and submit the credentials to log in.
4. Click on Linux at the top of the page under the Red Hat Developer logo.

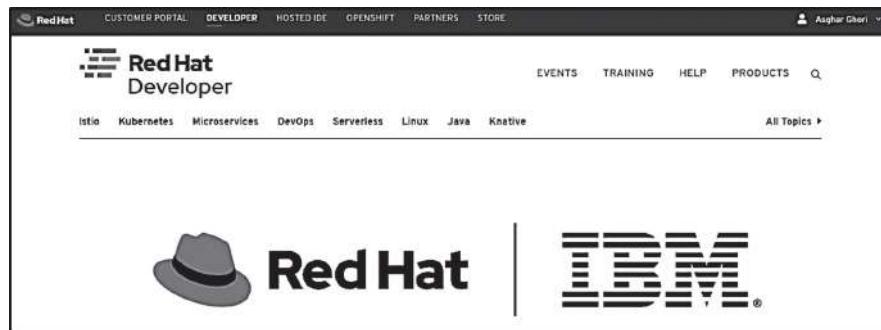


Figure 1-20 Red Hat Developer Login

5. Click DOWNLOAD on the following page.
6. Click DOWNLOAD INTEL ISO on the subsequent page to initiate a download.

The filename of the downloaded image for RHEL version 8.0 will be rhel-8.0-x86_64-dvd.iso and it will be about 6.9GB in size. You can move this file to a disk location on your computer where you want it stored.

Attaching RHEL 8 ISO Image to the Virtual Machine

We now attach the RHEL 8 ISO image to *RHEL8-VM1* in order to boot and install the OS in the VM. Click on “[Optical Drive] Empty” under Storage in VirtualBox for this VM and select Choose Disk Image. Navigate to where you have the ISO image stored. Highlight the image and click Open to attach it to the VM. After the image has been attached, the VirtualBox Storage configuration will look like [Figure 1-21](#).

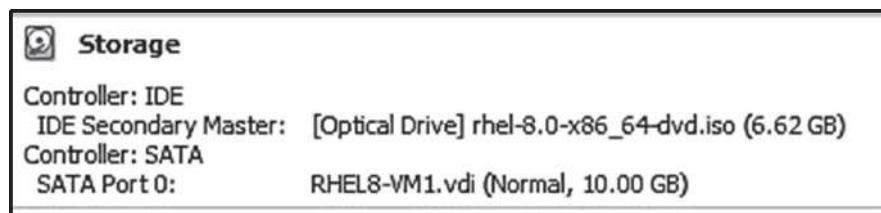


Figure 1-21 Attach ISO Image to VM

Leave the rest of the settings to their default values.

Launching the Installer

7. While the VM is highlighted in VirtualBox, click the Start button at the top to power up the VM.
8. A console screen pops up displaying the boot menu ([Figure 1-22](#)) with three options. Press the Spacebar key to halt the autoboot process.



Figure 1-22 Boot Menu

The first option, “Install Red Hat Enterprise Linux 8.0.0”, is usually used for installing RHEL 8 unless you want the installation media tested for integrity before continuing, in which case you will select the second option. Anaconda waits 60 seconds for you to alter the selection, or it proceeds and autoboots using the second option on the list, which is also the default. The third option, “Troubleshooting”, allows you to address some boot-related issues that might occur during installation.

Use the Up or Down arrow key to select the “Install Red Hat Enterprise Linux 8.0.0” entry and press Enter. The installer is launched in graphical mode.

9. The installer program shows a welcome screen with a long list of supported languages that you could use during the installation. The default is set to English. Click Continue to accept the default and move on.



Figure 1-23 Language Selection



If all the content does not fit on the console screen, try changing the Graphics Controller to VMSVGA under Settings | Display in the VirtualBox Manager for the VM.

10. The “Installation Summary” screen appears next, as shown in [Figure 1-24](#). You have the opportunity to make all necessary configuration changes prior to starting the installation. This screen presents a single interface to configure localization (keyboard, language, date, time, and time zone), software (installation source and software selection), and system (disk selection and partitioning, network and hostname assignments, etc.).



Figure 1-24 Installation Summary | Main



Any items highlighted in red and with a warning sign must be configured before the Begin Installation button at the bottom right of the screen is enabled.

There is no particular sequence to configure these items. If you do not wish to change a non-highlighted item, simply leave it intact and the installation program will apply the default settings for it.

Adding Support for Keyboards and Languages

11. Anaconda presents additional choices for keyboard layouts and languages for use during and after the installation. This should only be done on systems where support for multiple keyboard layouts and languages is required. The default is the US English for both.

Configuring Time & Date

12. Click Time & Date to set the time zone (region and city), date, and time for the system. See [Figure 1-25](#). Click Done in the upper left corner to save the changes and return to the Installation Summary screen.

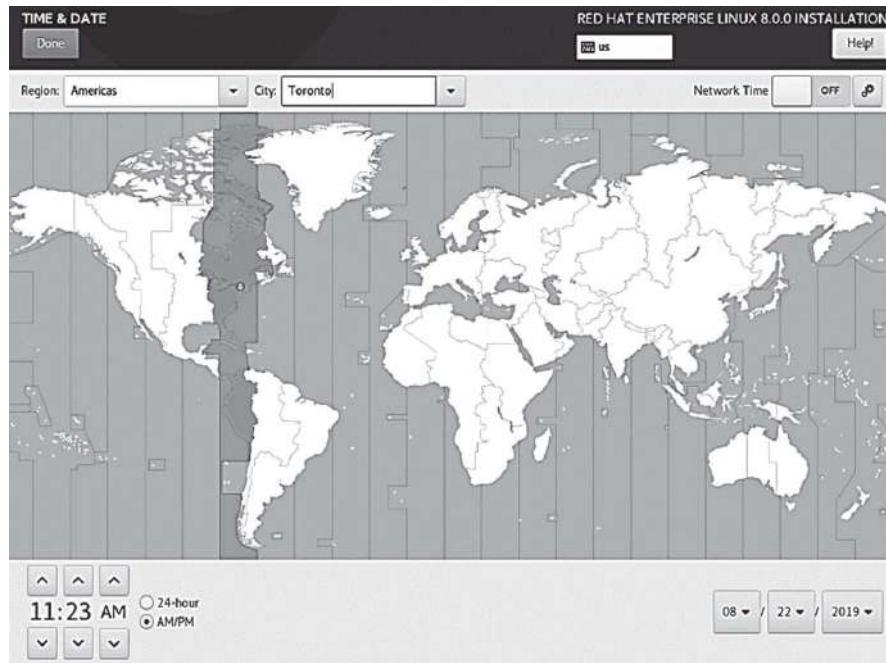


Figure 1-25 Installation Summary | Time & Date

[Figure 1-25](#) reflects two adjustments from the default. The city is changed to Toronto, and the clock format is switched to AM/PM.

Choosing an Installation Source

13. You can set the installation source for RHEL 8. By default, Anaconda chooses the auto-detected local media (DVD, USB flash drive, or ISO image) that was used to start this installation. For this demonstration, leave the installation source to the default. Click Done to return to the Installation Summary page.

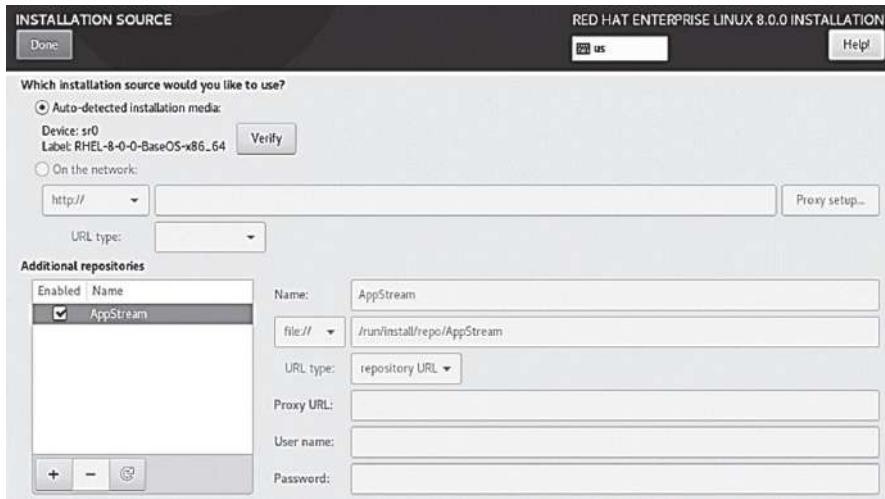


Figure 1-26 Installation Summary | Installation Source

If you have access to a configured network location hosting the installation files, you can choose “On the network” and specify the HTTP, HTTPS, FTP, or NFS protocol, hostname or IP address of the network server, and the path to the files. You can also specify the locations of additional software repositories, besides the default AppStream, if you have access to them.

Selecting Software to be Installed

14. You can choose the base operating environment that you want installed. Base environments are predefined groups of software packages designed for specific use cases. The six base environments are described in [Table 1-2](#).

Base Environment	Description
Server with GUI	Infrastructure server with graphics support
Server	Infrastructure server without graphics support
Minimal Install	Installs a minimum number of packages for basic system use
Workstation	Ideal for desktop and laptop users who require graphical support and with a minimal set of services
Custom Operating System	Gives you a set of basic building blocks for custom installations
Virtualization Host	Infrastructure plus virtualization support to host virtual machines

Table 1-2 Installation Summary | Software Selection | Base Environments

Choosing a base environment in the left pane reveals additional components on the right that may be ticked for installation along with the selected base environment. See [Figure 1-27](#).

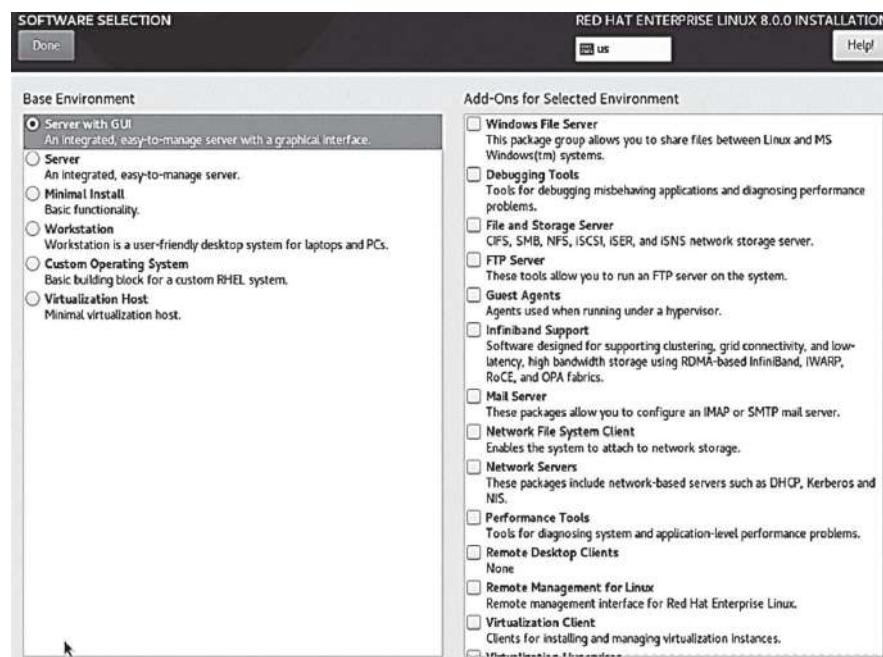


Figure 1-27 Installation Summary | Software Selection

The installer automatically picks and installs prerequisite software components to fulfill dependency requirements for a successful installation. The default base environment is “Server with GUI” for this demonstration. Leave add-ons to the default as well. Click Done to return to the Installation Summary page.

Configuring Installation Destination

15. The Installation Destination allows you to choose an available local or remote disk for partitioning and installing the OS on. Anaconda selects “Automatic partitioning selected” (highlighted in red) on the Installation Summary page ([Figure 1-24](#)), which you can change on Installation Destination ([Figure 1-28](#)). By default, the 10GB virtual disk you assigned to the VM initially is automatically picked up by the installer as the target and it is represented as sda. The “Encrypt my data” checkbox under Encryption encrypts all partitions on the disk. If you choose this option, you will be prompted to enter a passphrase to access the partitions later. The “Full disk summary and bootloader” link at the bottom left allows you to choose a disk to place the bootloader program on. This does not need to be modified on a single disk system. The default and the only bootloader program available in RHEL 8 is called GRUB2, and it is explained at length in [Chapter 11](#), “Boot and Initialization”.



Figure 1-28 Installation Summary | Installation Destination

For this demonstration, stick to the default automatic partitioning scheme. Simply click Done to return to the previous screen. This scheme will create three partitions—*/boot*, */*, and *swap*, and together they will consume the entire selected disk.

Configuring Network and Hostname

16. Assigning appropriate IP and hostname are essential for system functionality in a network environment. Click Network & Hostname on the Installation Summary page and a window similar to the one shown in [Figure 1-29](#) will appear. Anaconda detects all attached network interfaces, but it does not automatically assign them IPs. Also, the default hostname is set to *localhost.localdomain*. You need to modify these assignments so that your system is able to communicate with other systems on the network. Currently, there is one network device assigned to the system, which is represented as *enp0s3*.



The terms “network interface” and “network device” refer to the same network hardware component. These terms are used interchangeably throughout this book. These terms are different from the term “network connection”, which is the software configuration applied to a network interface/device.



The default naming convention for network devices vary based on the underlying virtualization software being used.

Change the hostname to server1.example.com in the Hostname field. For IP assignments, there are a couple of options. You can obtain them automatically from an available DHCP server by simply sliding the ON/OFF switch located in the top right-hand corner. However, for this demonstration, click Configure at the bottom right and enter IP information manually. You also need to ensure that the network connection is set to autostart.

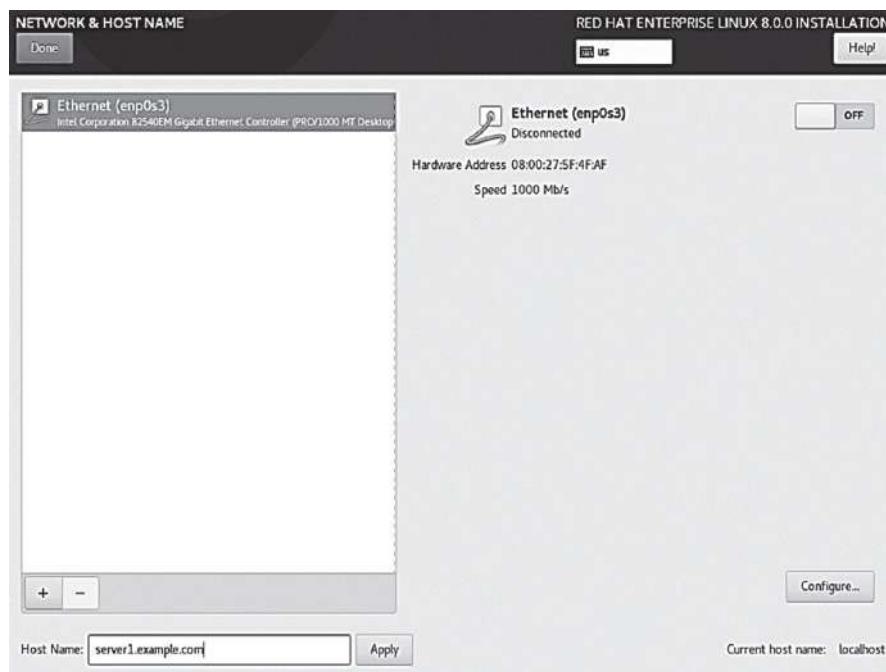


Figure 1-29 Installation Summary | Network & Hostname

There are multiple tabs available on the network connection configuration screen, as depicted in [Figure 1-30](#). Go to IPv4 Settings and choose Manual from the drop-down list against Method. Click Add and enter address 192.168.0.110, netmask 24, and gateway

192.168.0.1. Click Save to save the configuration and return to the Network & Hostname window.

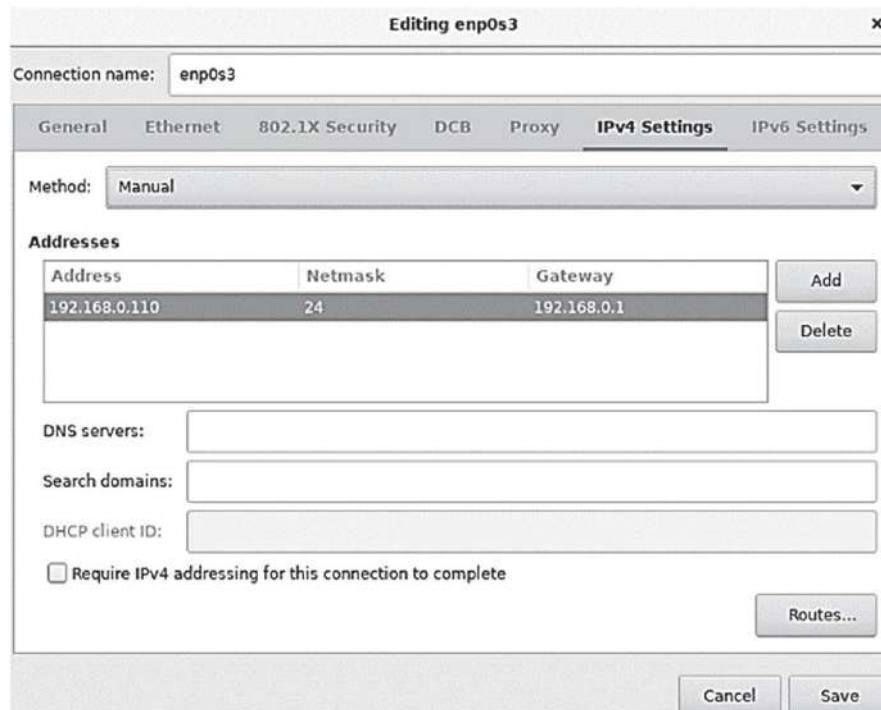


Figure 1-30 Installation Summary | Network & Hostname | Configure

On the Network & Hostname window, slide the ON/OFF switch to the ON position so that the new assignments take effect right away. This will also ensure that the assignments are applied automatically on subsequent system reboots.

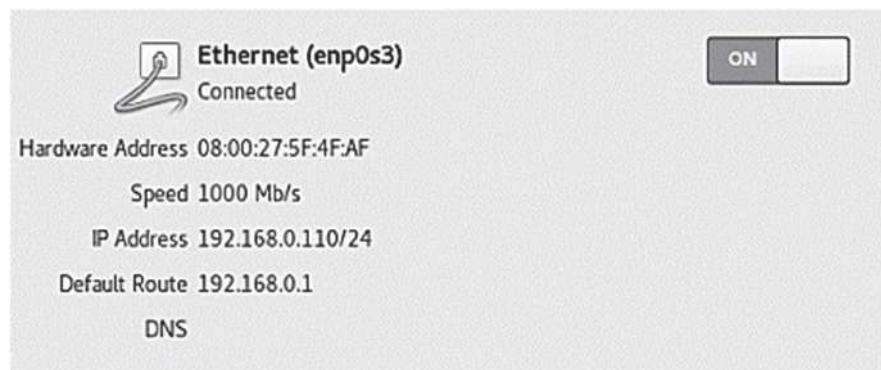


Figure 1-31 Installation Summary | Network & Hostname

Now click Done to return to the Installation Summary page. [Chapter 16](#) “Networking, Network Devices, and Network Connections” discusses configuring hostnames, network interfaces, and network connections in detail.

Beginning Installation

17. You’re now on the Installation Summary page ([Figure 1-32](#)). You still have the opportunity to go back and configure or reconfigure any items you’ve missed. Once you are satisfied, click Begin Installation at the bottom right to initiate the installation based on the configuration entered in the previous steps. Anaconda will now partition the selected disk and install the software. Any data previously stored on the disk will be erased and unrecoverable.



Figure 1-32 Installation Summary | Begin Installation



The Begin Installation button remains inactive until all the items highlighted in red and with a warning sign are configured.

The configuration and software copy will take some time to complete. The progress will depend on the system performance and resources allocated to the VM.

Setting root Password and Creating a User Account

- Once the installation has begun, a new screen, called Configuration, pops up. See [Figure 1-33](#). This is where you can monitor the progress, and it also allows you to assign a password to the *root* user and create a user account.

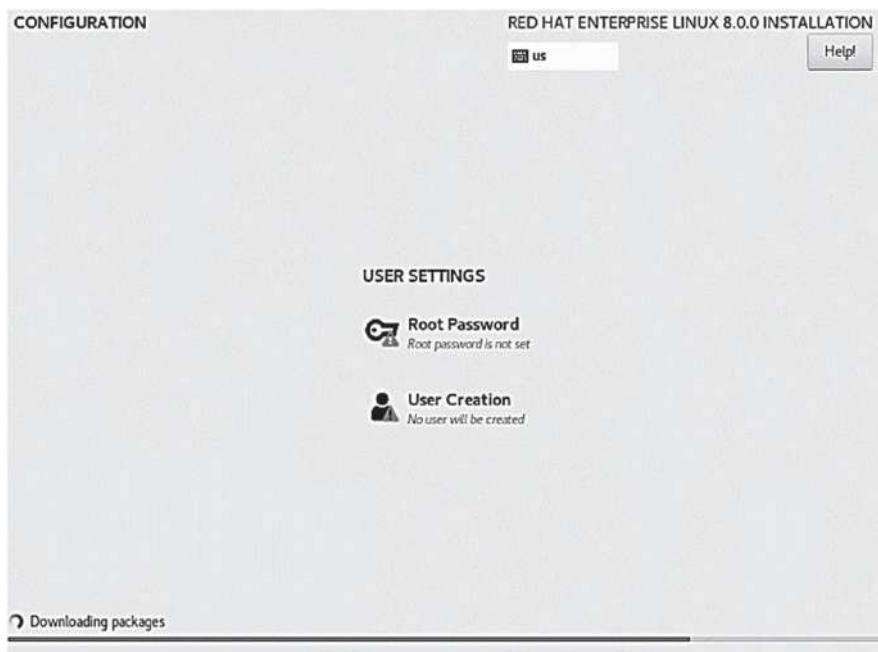


Figure 1-33 Configuration | User Settings

- While the installer continues to run, click Root Password and set a password for the *root* user. Click Done (two clicks if the password entered is too short or simple) to return to the Configuration screen.
- Next, click User Creation to create a user account called *user1* and assign it a password. Leave the “Make this user administrator” option unticked. See [Figure 1-34](#). Click Done (two clicks if the password entered is too short or simple) to return to the Configuration screen.

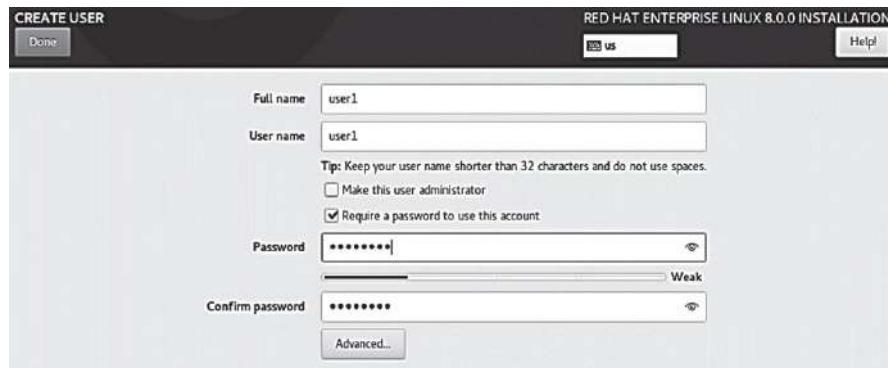


Figure 1-34 Configuration | User Settings | Create User

Anaconda will set the *root* user password and create the user account during the configuration part of the installation.

Concluding Installation

21. When the required setup is complete and all software packages are installed, a Reboot button will appear at the bottom right on the Configuration screen ([Figure 1-35](#)). Click this button to reboot the new system.

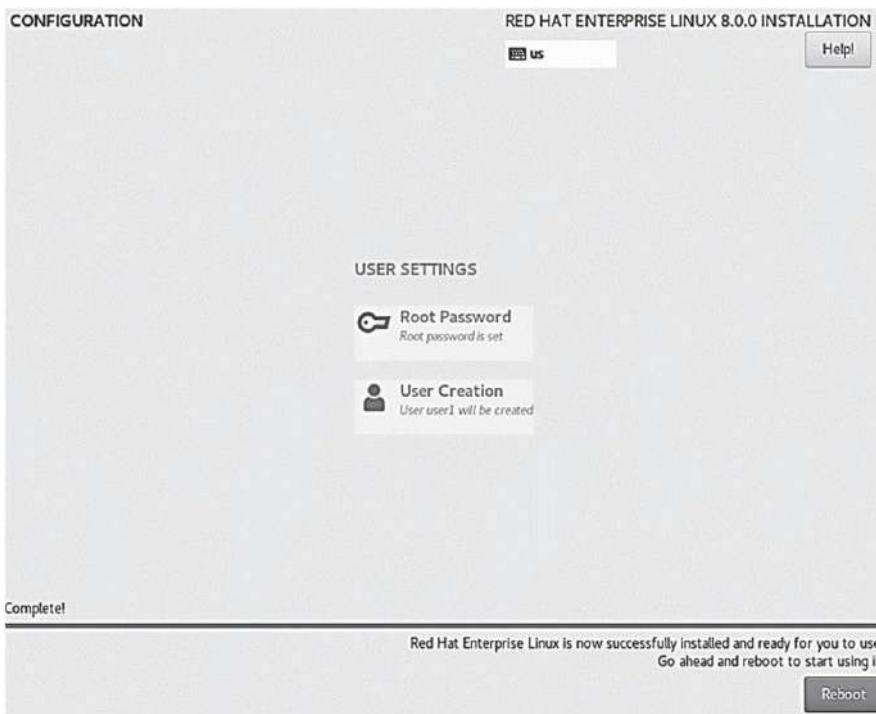


Figure 1-35 Configuration | Finishing Installation

By default, VirtualBox does not automatically change the default boot order. This results in rebooting the VM from the ISO image again and restarting the installation. To avoid this situation, power off the virtual machine from VirtualBox and alter the boot sequence.

Changing Default Boot Order

22. Power off the VM from VirtualBox.
23. The current boot sequence, as shown in [Figure 1-36](#), is set to boot with floppy first and then optical (DVD/CD) followed by hard disk.

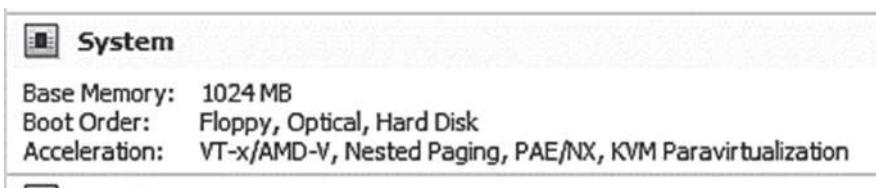


Figure 1-36 VirtualBox Manager | System | Boot Order

24. Change this sequence to hard disk first and optical next. See [Figure 1-37](#). Untick Floppy, as it is not required.

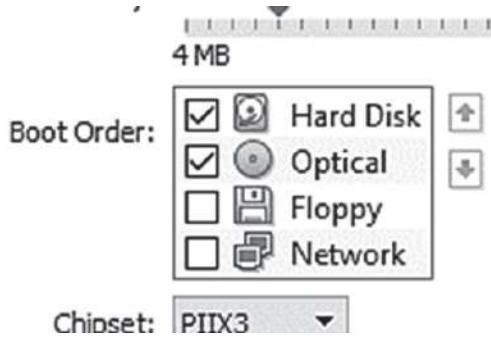


Figure 1-37 VirtualBox Manager | System | Boot Order | Alter

Now power up the VM from the VirtualBox Manager. It will boot the installed OS.

Performing Post-Installation Tasks

25. The system initiates the Initial Setup application upon restart so that you can complete certain post-installation tasks. [Figure 1-38](#) shows the Initial Setup screen with license already accepted.



Figure 1-38 Initial Setup

26. On this screen you must click the License Information icon and accept the license terms and conditions by ticking the box beside "I accept the license agreement". Click Done to return to the Initial Setup screen.
27. The second item on the Initial Setup screen helps you register this system with Red Hat's subscription management service to enable it to receive automatic software updates and perform certain other management tasks. Leave this item intact for this demonstration and click Finish Configuration.

This brings the installation and initial configuration of RHEL 8 to a successful completion.

Logging In and Out at the Graphical Console

Now that the installation is complete, you can log on to the system. You selected the Server with GUI base environment, which includes graphical desktop support to interact with the system. You also entered credentials for a user account, *user1*, during installation. You can now use this account to log in.

Logging In for the First Time

When you sign in with a new user account for the first time, several screens appear in succession to allow you to configure a few basic items for the user. Follow the steps below to go through this process. You will not be prompted again for this configuration upon next logon.

1. On the graphical logon screen, click *user1* and enter the password when prompted.

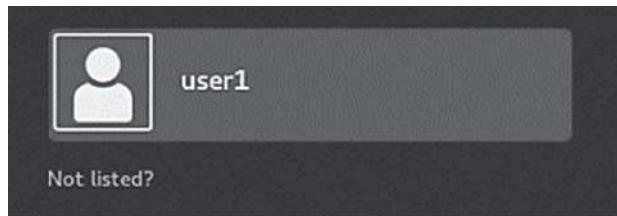


Figure 1-39 Graphical Desktop | Sign-in Screen

The login process continues, and a Welcome screen pops up that shows the language that you had selected at the beginning of the installation. You can change it to a different language if you so desire. Click Next.

2. Add an Input Source to be used on the next screen. The default is the US English keyboard type that you had selected earlier. Click Next.
3. Enable the geographical location of this system to be automatically determined for applications to use. The default is set to on. Click Next.
4. You can connect one of your online user accounts such as Google, Nextcloud, Microsoft, or Facebook in order to access your email, contact, and other information and services. Click Skip for this demonstration.

5. Click “Start using Red Hat Enterprise Linux” on the final “Ready to Go” screen, [Figure 1-40](#).

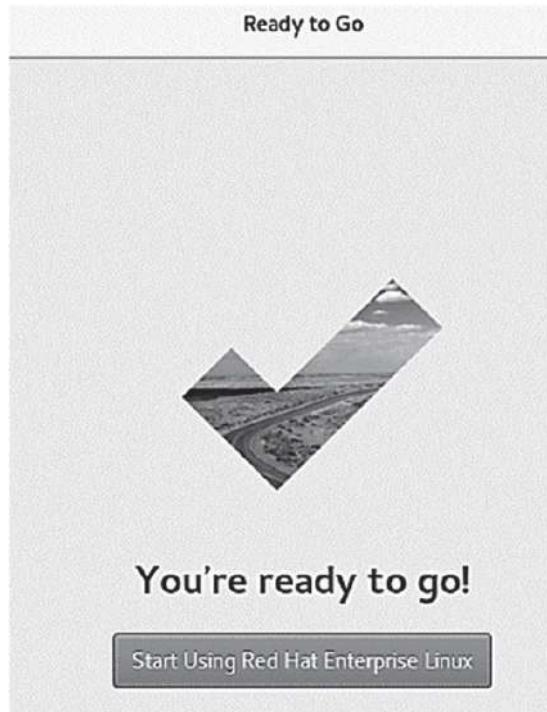


Figure 1-40 First Time User Login

6. A Getting Started help screen pops up ([Figure 1-41](#)). Here, you can watch videos on how to launch applications, switch tasks, and use windows and workspaces in the *Graphical User Interface* (GUI). The help is available in text format as well. You can close this window by clicking the x (exit) button at the top right.

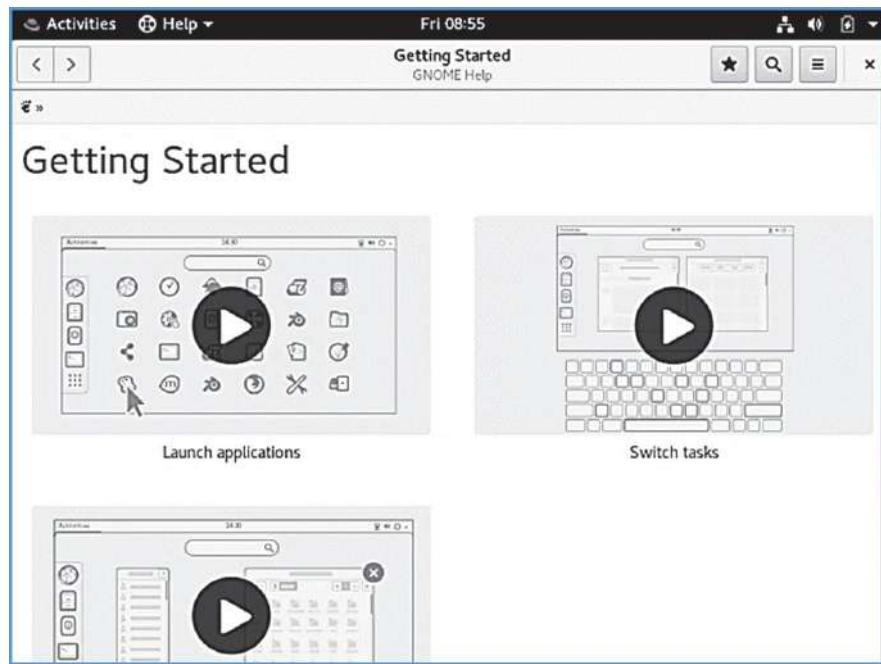


Figure 1-41 First Time User Login | Getting Started

7. The default graphical desktop included in RHEL 8 is the GNOME desktop environment ([Figure 1-42](#)). You should now be able to start using the system as *user1*.



Figure 1-42 GNOME Desktop Environment



GNOME stands for *GNU Network Object Model Environment*. It is the default graphical display manager and desktop environment for users in RHEL 8. [Chapter 02](#) “Initial Interaction with the System” provides more details on this topic.

Logging Out

8. Logging out of the system is easy. Click on the down arrow (top right), expand *user1* (the name of the logged-in user), and click Log Out. See [Figure 1-43](#). The user will be signed out and the main login screen will reappear.



Figure 1-43 GNOME Desktop Environment| Log Out

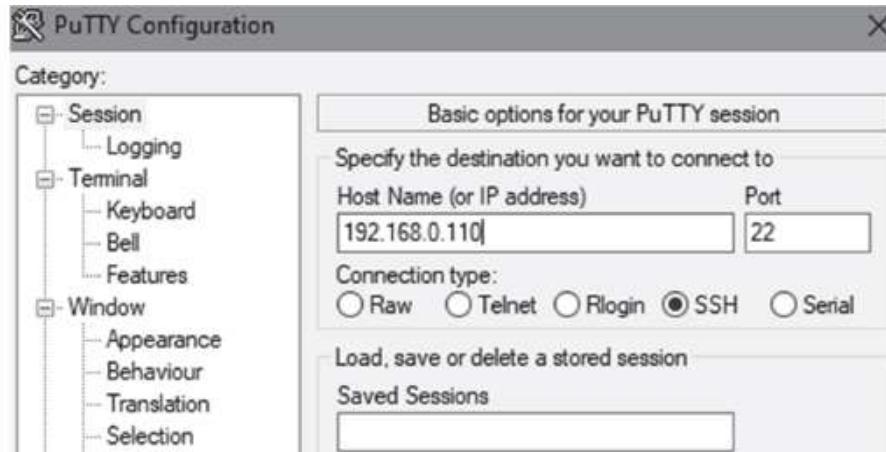
Now, let’s look at how to connect and log in to the system remotely from over the network.

Exercise 1-3: Logging In from Windows

This exercise should be done on the Windows computer hosting the virtual machine for *server1*.

In this exercise, you will use a program called PuTTY to access *server1* using its IP address and as *root*. You will run appropriate commands on the server for validation. You will log off to terminate the session.

1. On Windows desktop, download PuTTY free of charge from the Internet. Launch this program and enter the target host's IP address. Leave the rest of the settings to their defaults.



You may assign a name to this session (typically the hostname is used as the session name) in the Saved Sessions field and click Save to store this information so as to avoid retyping in the future.

2. Click on the “Open” button at the bottom of the screen to try a connection.
3. Enter *root* and password at the “login as” prompt to log in:

```
root@server1:~  
login as: root  
root@192.168.0.110's password:  
Activate the web console with: systemctl enable --now cockpit.socket  
  
Last login: Mon Oct 12 12:38:59 2020 from 192.168.0.146  
[root@server1 ~]#  
[root@server1 ~]#
```

2. Issue the basic Linux commands *whoami*, *hostname*, and *pwd* to confirm that you are logged in as *root* on *server1* and placed in the correct home directory:

```
[root@server1 ~]# whoami  
root  
[root@server1 ~]# hostname  
server1.example.com  
[root@server1 ~]# pwd  
/root  
[root@server1 ~]#
```

3. Run the *logout* or the *exit* command or press the key combination *Ctrl+d* to log off *server1* and terminate the login session:

```
[root@server1 ~]# logout
```

This concludes the exercise. Going forward, you should be doing all the exercises and labs presented in this book in PuTTY (ssh) terminal sessions.

[Chapter 02](#) will explore how to navigate within the GNOME desktop environment, execute basic Linux commands at the command prompt, and obtain necessary help.

Chapter Summary

In this chapter, we started by looking at Linux history and exploring available versions of Linux from Red Hat. We examined various pre-installation items for our lab environment to prepare for a smooth installation in order to practice the exercises and labs presented in this book. We demonstrated downloading the images for VirtualBox Manager software and RHEL 8. We built a virtual machine and

installed RHEL 8 in it. We completed post-installation tasks to conclude the demonstration. Finally, we logged in to the new system at the console and over the network via PuTTY to verify the installation.

Review Questions

1. The minimal Install base environment includes the graphical support. True or False?
2. Can you install RHEL 8 in text mode?
3. You can use the */boot* partition within LVM to boot RHEL. True or False?
4. Which kernel version is the initial release of RHEL 8 based on?
5. Several log files are created and updated in the */tmp* directory during the installation process. Where are these files moved to after the completion of installation?
6. Name the RHEL installer program.
7. How many console screens do you have access to during the installation process?
8. RHEL 8 may be downloaded from Red Hat's developer site. True or False?
9. RHEL 8 cannot be installed over the network. True or False?
10. What is the name of the default graphical user desktop if Server with GUI is installed?

Answers to Review Questions

1. False. Minimal Install base environment does not include graphics support.
2. Yes, RHEL 8 can be installed in text mode.
3. False. */boot* cannot reside within LVM.
4. The initial release of RHEL 8 is based on kernel version 4.18.
5. These files are moved to the */var/log* directory.
6. The name of the RHEL installer program is Anaconda.
7. There are six console screens available to you during the installation process.
8. True. RHEL 8 may be downloaded from developers.redhat.com. You need to open a new account or use an existing before you can download it.

9. False. RHEL 8 can be installed with installation files located on a network server.
10. The default graphical desktop is called GNOME desktop environment.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 1-1: Build RHEL8-VM2 (server2)

Create another virtual machine called *RHEL8-VM2* in VirtualBox, attach the ISO image to it, and install RHEL 8. Use the configuration provided in “What is Needed for the Lab?” and follow the procedures outlined in Exercises 1-1 and 1-2. Use PuTTY with the IP address of the new server to connect to it.

Chapter 02

Initial Interaction with the System

This chapter describes the following major topics:

- Interact with display manager and understand graphical interface
- Overview of Linux directory structure
- Recognize top-level directories
- Understand command construct
- Describe and run basic Linux commands
- Obtain help using multiple native tools and RHEL documentation

RHCSA Objectives:

01. Access a shell prompt and issue commands with correct syntax
11. Locate, read, and use system documentation including man, info, and files in /usr/share/doc

Wayland is an advanced display protocol that sets up the foundation for running graphical applications, which includes system administration tools, user applications, as well as Linux graphical display and desktop manager programs. Working in a graphical environment to interact with the system is convenient for users with limited command line knowledge or specific requirements.

Linux files are organized logically for ease of administration. This file organization is maintained in hundreds of directories located in larger containers (file systems). Red Hat Enterprise Linux follows the File system Hierarchy Standard for file organization, which describes names, locations, and permissions for many file types and directories.

Linux offers a variety of commands for users and system managers. User commands are general purpose that are intended for execution by any user on the system. However, system management commands require elevated privileges of the superuser. Knowledge of these tools is essential for productive usage and efficient administration of the system. This chapter provides an analysis of command components and how to construct a command. Following that, it introduces a few basic user-level commands.

The availability of native help on the system simplifies task execution for Linux users and system administrators alike. This assistance is available on commands and configuration files via locally installed searchable manual pages and documentation for installed packages. In addition, Red Hat documentation website provides a wealth of information on various topics, procedures, and command usage.

Linux Graphical Environment

RHEL allows users to work in both text and graphical environments. Text interface might be cumbersome, but many administrators and programmers prefer to work in a text-mode setting without needing graphics capabilities. Nevertheless, a graphical environment provides easier and convenient interaction with the OS by hiding the challenges that users may otherwise experience when working in text-mode.

Wayland is a client/server display protocol that sets up the foundation for running graphical programs and applications in RHEL 8. It is available alongside the legacy X Window System, which has been around in RHEL for decades. Wayland provides superior graphics capabilities, features, and performance than X. There are two components that are critical to the

functionality of a graphical environment: the *display manager* (a.k.a. *login manager*) and the *desktop environment*. Both are launched following the completion of the groundwork established by Wayland.

Display/Login Manager

A display/login manager handles the presentation of graphical login screen. It allows users to enter credentials to log on to the system. A preconfigured graphical *desktop manager* appears after the credentials are verified. In RHEL 8, the default display manager is called *GNOME Display Manager* (GDM).

[Figure 2-1](#) provides an image of GDM.



Figure 2-1 GNOME Display Manager

The login screen presents a list of all normal user accounts that exist on the system. You can log in as any one of them by selecting the desired account. If you wish to sign in as an unlisted user or the *root* user, click “Not Listed?” and enter the username and password for the desired account. The current system day and time also appear at the top of the login screen.

There are two downward arrowheads at the top right of the login screen. The arrowhead on the left is to enable or disable an accessibility feature. The arrowhead on the right allows you to power off or reboot the system and change the system volume. More controls become available after you have logged in. There are three additional icons at the top right that show the network connectivity, sound level, and battery/power status.

Desktop Environment

Once the credentials are validated for a user, the display/login manager establishes a *Desktop Environment* (DE) to work in. RHEL 8 comes with several graphical desktop software with GNOME desktop environment set as the default. It provides an easy and point-and-click GUI for users to run programs and operating system tools. [Figure 2-2](#) is an image of the default GNOME desktop environment for *root*.



Figure 2-2 GNOME Desktop Environment

If you have worked with Microsoft Windows, you should have no difficulty using this desktop environment. The default screen has an Activities icon at the top left, which allows you to search and access programs. [Figure 2-3](#) depicts a list of application icons when you click on Activities.



Figure 2-3 GNOME Desktop Environment | Activities

These application icons represent Firefox web browser, file manager, software updates, GNOME help, and shell terminal. The icon with nine dots displays all available programs, including Settings. The Settings application includes administrative and user-level controls to view or modify configuration items such as Wi-Fi, Bluetooth, desktop background, notifications, regional settings, privacy, sound, power, screensaver, network, and more.

Linux Directory Structure and File Systems

Linux files are organized logically in a hierarchy for ease of administration and recognition. This organization is maintained in hundreds of directories located in larger containers called *file systems*. Red Hat Enterprise Linux follows the *Filesystem Hierarchy Standard* (FHS) for file organization, which describes names, locations, and permissions for many file types and directories.

Linux file systems contain files and subdirectories. A subdirectory, also referred to as a *child* directory, is located under a *parent* directory. The parent directory is a subdirectory of a higher-level directory. The Linux directory structure is analogous to an inverted tree, where the top of the tree is the root of the directory, tree branches are subdirectories, and leaves are files. The

root of the directory is represented by the forward slash character (/), and this is where the entire directory structure is ultimately connected. The forward slash is also used as a directory separator in a path, such as `/etc/rc.d/init.d/functions`.

In this example, the `etc` subdirectory is located under /, making `root` the parent of `etc` (which is a child), `rc.d` (child) is located under `etc` (parent), `init.d` (child) is located under `rc.d` (parent), and `functions` (leaf) is located under `init.d` (parent) at the bottom.



The term *subdirectory* is used for a directory that has a parent directory.

Each directory has a parent directory and a child directory, with the exception of the root and the lowest level subdirectories. The root directory has no parent, and the lowest level subdirectory has no child.

Top-Level Directories

The key top-level directories under / are shown in [Figure 2-4](#). Some of these directories hold *static* data, while others contain *dynamic* (or *variable*) information. Static data refers to file content that remains unchanged unless modified explicitly. Dynamic or variable data, in contrast, refers to file content that is modified and updated as required by system processes. Static directories normally contain commands, configuration files, library routines, kernel files, device files, etc., and dynamic directories contain log files, status files, temporary files, etc.

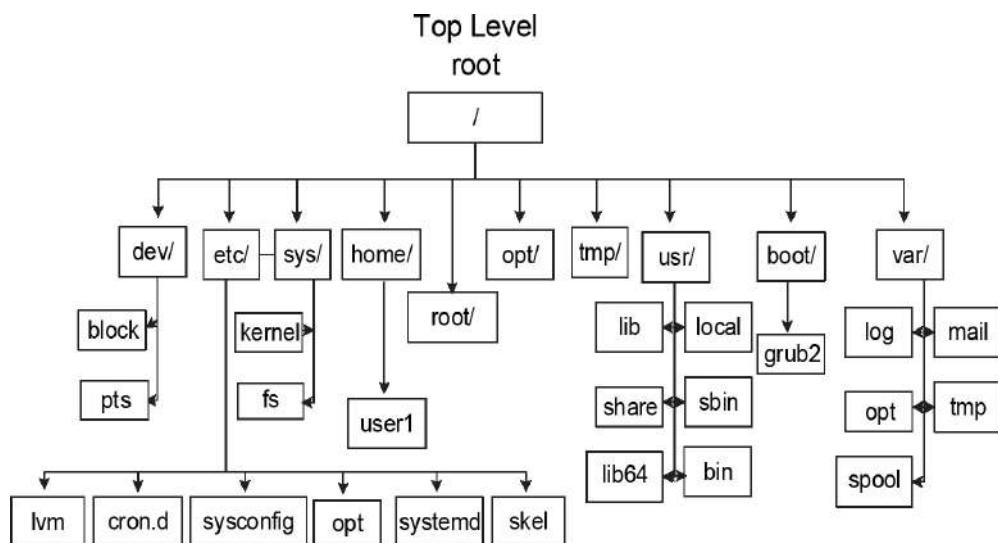


Figure 2-4 Linux Directory Structure

The hierarchical directory structure keeps related information together in a logical fashion. Compare this concept with a file cabinet containing several drawers, with each drawer storing multiple file folders.

File System Categories

There are a variety of file system types supported in RHEL that can be categorized in three basic groups: *disk-based*, *network-based*, and *memory-based*. Disk-based file systems are typically created on physical media such as a hard drive or a USB flash drive. Network-based file systems are essentially disk-based file systems that are shared over the network for remote access. Memory-based file systems are virtual; they are created automatically at system startup and destroyed when the system goes down. The first two types of file systems store information persistently, while any data saved in virtual file systems is lost at system reboots.

During RHEL installation, two disk-based file systems are created when you select the default partitioning. These file systems are referred to as the root and boot file systems. Furthermore, several memory-based file systems are vital to the operation of a RHEL system.

The Root File System (/), Disk-Based

The root directory is the top-level file system in the FHS and contains many higher-level directories that store specific information. Some of the key directories are:

/etc: The *etcetera* (or *extended text configuration*) directory holds system configuration files. Some common subdirectories are *systemd*, *sysconfig*, *lvm*, and *skel*, which comprise configuration files for systemd, most system services, the Logical Volume Manager, and per-user shell startup template files, respectively.

/root: This is the default home directory location for the *root* user.

/mnt: This directory is used to mount a file system temporarily.

The size of the root file system is automatically determined by the installer program based on the available disk space when you select the default partitioning; however, it may be altered if required.

The Boot File System (/boot), Disk-Based

The */boot* file system contains the Linux kernel, boot support files, and boot configuration files. Just like the *root* file system, the size of this file system is also automatically determined by the installer program based on the available

disk space when you select the default partitioning; however, it may be set to a different size during or after the installation if required.

The Home Directory (`/home`)

The `/home` directory is designed to store user *home* directories and other user contents. Each user is assigned a home directory to save personal files, and the user can block access to other users.

The Optional Directory (`/opt`)

This directory can be used to hold additional software that may need to be installed on the system. A subdirectory is created for each installed software.

The UNIX System Resources Directory (`/usr`)

This directory contains most of the system files. Some of the important subdirectories are:

`/usr/bin`: The *binary* directory contains crucial user executable commands.

`/usr/sbin`: Most commands are required at system boot, and those that require the *root* user privileges in order to run are located in this *system binary* directory. In other words, this directory contains crucial system administration commands that are not intended for execution by normal users (although they can still run a few of them). This directory is not included in the default search path for normal users because of the nature of data it holds.

`/usr/lib` and `/usr/lib64`: The *library* directories contain shared library routines required by many commands and programs located in the `/usr/bin` and `/usr/sbin` directories, as well as by the kernel and other applications and programs for their successful installation and operation. The `/usr/lib` directory also stores system initialization and service management programs. The subdirectory `/usr/lib64` contains 64-bit shared library routines.

`/usr/include`: This directory contains header files for C language.

`/usr/local`: This directory serves as a system administrator repository for storing commands and tools downloaded from the web, developed in-house, or obtained elsewhere. These commands and tools are not generally included with the original Linux distribution. In particular, `/usr/local/bin` holds executables, `/usr/local/etc` holds configuration files, and `/usr/local/lib` and `/usr/local/lib64` holds library routines.

/usr/share: This is the directory location for manual pages, documentation, sample templates, configuration files, etc., that may be shared with other Linux platforms.

/usr/src: This directory is used to store source code.

The Variable Directory (**/var**)

The **/var** directory contains data that frequently changes while the system is operational. Files in this directory contain log, status, spool, lock, and other dynamic data. Some common subdirectories under **/var** are:

/var/log: This is the storage for most system log files, such as system logs, boot logs, user logs, failed user logs, installation logs, cron logs, mail logs, etc.

/var/opt: This directory stores log, status, and other variable data files for additional software installed in **/opt**, t.

/var/spool: Directories that hold print jobs, cron jobs, mail messages, and other queued items before being sent out to their intended destinations are located here.

/var/tmp: Large temporary files or temporary files that need to exist for longer periods of time than what is typically allowed in another temporary directory, **/tmp**, are stored here. These files survive system reboots and are automatically deleted if they are not accessed or modified for a period of 30 days.

The Temporary Directory (**/tmp**)

This directory is a repository for temporary files. Many programs create temporary files here during runtime or installation. These files survive system reboots and are automatically removed if they are not accessed or modified for a period of 10 days.

The Devices File System (**/dev**), Virtual

The Devices (*dev file system*) file system is accessible via the **/dev** directory, and it is used to store device nodes for physical hardware and virtual devices. The Linux kernel communicates with these devices through corresponding device nodes located here. These device nodes are automatically created and deleted by the *udevd* service (a Linux service for dynamic device management) as necessary.

There are two types of device files: *character* (or *raw*) device files, and *block* device files. The kernel accesses devices using one of these files or both.

Character devices are accessed serially with streams of bits transferred during kernel and device communication. Examples of such devices are console, serial printers, mice, keyboards, terminals, etc.

Block devices are accessed in a parallel fashion with data exchanged in blocks (parallel) during kernel and device communication. Data on block devices is accessed randomly. Examples of block devices are hard disk drives, optical drives, parallel printers, etc.

The Procfs File System (`/proc`), Virtual

The Procfs (*process file system*) file system is accessible via the `/proc` directory, and it is used to maintain information about the current state of the running kernel. This includes the details for current hardware configuration and status information on CPU, memory, disks, partitioning, file systems, networking, running processes, and so on. This information is stored in a hierarchy of subdirectories that contain thousands of zero-length pseudo files. These files point to relevant data maintained by the kernel in the memory. This virtual directory structure simply provides an easy interface to interact with kernel-maintained information. The Procfs file system is dynamically managed by the system.

The contents in `/proc` are created in memory at system boot time, updated during runtime, and destroyed at system shutdown.

The Runtime File System (`/run`), Virtual

This virtual file system is a repository of data for processes running on the system. One of its subdirectories, `/run/media`, is also used to automatically mount external file systems such as those that are on optical (CD and DVD) and flash USB.

The contents of this file system are automatically deleted at system shutdown.

The System File System (`/sys`), Virtual

Information about hardware devices, drivers, and some kernel features is stored and maintained in the `/sys` file system. This information is used by the kernel to load necessary support for the devices, create device nodes in `/dev`, and configure the devices. This file system is auto-maintained as well.

Viewing Directory Hierarchy

The `tree` command lists a hierarchy of directories and files. There are a number of options with this command that can be specified to include additional information. [Table 2-1](#) describes some common options.

Option	Description
-a	Includes hidden files in the output
-d	Excludes files from the output
-h	Displays file sizes in human-friendly format
-f	Prints the full path for each file
-p	Includes file permissions in the output

Table 2-1 tree Command Options

Let's try to understand the usage of the *tree* command with the help of some examples:

To list only the directories (-d) in the *root* user's home directory (*/root*):

```
[root@server1 ~]# tree -d
.
├── Desktop
├── Documents
├── Downloads
├── Music
├── Pictures
├── Public
├── Templates
└── Videos

8 directories
```

The output above indicates that there are eight directories under */root*.

To list files in the */etc/sysconfig* directory along with their permissions (-p), sizes in human-readable format (-h), and full path (-f):

```
[root@server1 ~]# tree -hpf /etc/sysconfig
/etc/sysconfig
├── [-rw-r--r-- 309] /etc/sysconfig/anaconda
├── [-rw-r--r-- 403] /etc/sysconfig/atd
├── [drwxr-xr-x 43] /etc/sysconfig/cfq
│   ├── [-rw-r--r-- 11] /etc/sysconfig/cfq/avpkt
│   └── [-rw-r--r-- 79] /etc/sysconfig/cfq/cfq-0000.example
├── [-rw-r--r-- 46] /etc/sysconfig/chronyrd
├── [drwxr-xr-x 6] /etc/sysconfig/console
├── [-rw-r--r-- 150] /etc/sysconfig/cpupower
├── [-rw-r--r-- 110] /etc/sysconfig/crond
├── [-rwx----- 417] /etc/sysconfig/ebtables-config
├── [-rw-r--r-- 73] /etc/sysconfig/firewalld
├── [lrwxrwxrwx 15] /etc/sysconfig/grub -> ../default/grub
├── [-rws----- 2.1K] /etc/sysconfig/ip6tables-config
├── [-rw----- 2.1K] /etc/sysconfig/iptables-config
├── [-rw-r--r-- 903] /etc/sysconfig/irqbalance
└── [-rw-r--r-- 1.7K] /etc/sysconfig/kdump

.....
```

The output shows permissions in column 1, sizes in column 2, and full path of the files in column 3.

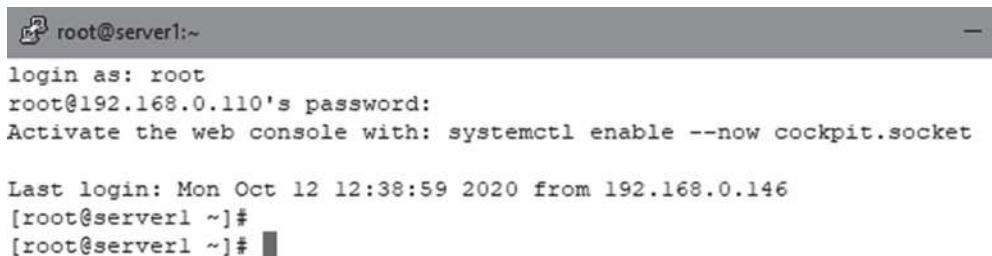
Run **man tree** at the command prompt to view the manual pages of the *tree* command for additional options and their usage.

Basic System Commands

There are hundreds of commands available in RHEL that range from simple to complex in terms of their construct and usage. Commands can be combined to build complex structures for innovative use cases. Some commands offer a few options, while others have as many as 70 or more. This section furnishes an understanding of how commands are formed and demonstrates the use of some of the basic, common Linux commands. You will learn more commands and their advanced usages throughout this book.

Starting a Remote Terminal Session

In order to issue commands and run programs at the command prompt, you need access to a terminal session. Follow the steps identified in [Exercise 1-3](#) to log in to the system as the *root* user. [Figure 2-5](#) illustrates a typical terminal window.

A screenshot of a terminal window. The title bar shows a small icon followed by "root@server1:~". The main area of the terminal displays the following text:

```
root@server1:~  
login as: root  
root@192.168.0.110's password:  
Activate the web console with: systemctl enable --now cockpit.socket  
  
Last login: Mon Oct 12 12:38:59 2020 from 192.168.0.146  
[root@server1 ~]# [root@server1 ~]# █
```

The terminal window has a dark background with light-colored text. The cursor is shown as a small black square with a white border.

Figure 2-5 Terminal Session

The header bar as depicted in [Figure 2-5](#) displays the logged-in username, hostname, and your current directory location.

The “[root@server1 ~]#” is the default representation of the command prompt. It reflects the logged-in username (*root*, *user1*, etc.), hostname of the system (*server1*, *server2*, etc.), and your current directory location, all enclosed within the square brackets []. The command prompt ends with the hash sign (#) for the *root* user or the dollar sign (\$) for normal users (*user1*, etc.) outside of the closing square bracket. Commands are typed at the cursor position and executed by pressing the Enter key.

Understanding the Command Mechanics

To practice the commands provided in this chapter, you can log in as *user1*, run the commands, and observe their outputs. However, as you are learning

Linux system administration, it's important to feel comfortable working as *root* in the beginning. If something breaks, *server1* and *server2* are lab servers so they can be rebuilt.

The basic syntax of a Linux command is:

command option(s) argument(s)

Options (a.k.a. a *switch* or *flag*) are optional. You can specify zero or more options with a command. Arguments, in contrast, may be optional or mandatory depending on the command and its usage. Many commands have preconfigured default options and arguments. You are not required to specify them. Other commands do require at least one option or argument in order to work. An option modifies the behavior of the command. An argument supplies a target on which to perform the command action.

An option may start with a single hyphen character (-la, for instance), and it is referred to as the short-option format. Each individual letter in this depiction represents a separate option (l and a are two options in -la). This is a frequent format throughout this book.

An option may also begin with two hyphen characters (--all, for instance), and it is referred to as the long-option format. All letters in this representation are collectively identified as a single option (-all is one option).

The following examples express some command structures with a description on the right that states the number of options and arguments supplied:

# ls	No option, no explicit argument; the default argument is the current directory name
# ls -l	One option, no explicit argument; the default argument is the current directory name
# ls -al	Two options, no explicit argument; the default argument is the current directory name
# ls --all	One option, no explicit argument; the default argument is the current directory name
# ls -l directory_name	One option, one explicit argument

EXAM TIP: Use online help on the usage of a command if needed. Refer to “Getting Help” later in this chapter for more on how to access and use help.

Now let's take a look at some essential Linux commands and understand their usage.

Listing Files and Directories

One of the most rudimentary commands in Linux is the */s (list)* command. It is used to show the list of files and directories. This command supports a multitude of options, some of which are listed in [Table 2-2](#) along with a short explanation.

Option	Description
-a	Includes hidden files and directories in the output. A file or directory name that begins with the period character (.) is considered hidden.
-l	Displays long listing with detailed file information including the file type, permissions, link count, owner, group, size, date and time of last modification, and name of the file
-ld	Displays long listing of the specified directory but hides its contents
-lh	Displays long listing with file sizes shown in human-friendly format
-lt	Lists all files sorted by date and time with the newest file first
-ltr	Lists all files sorted by date and time with the oldest file first (reverse)
-R	Lists contents of the specified directory and all its subdirectories (recursive listing)

Table 2-2 Is Command Options

A grasp of the usage of this command and the output it produces is important. The following examples will illustrate the impact of options used with the */s* command. Again, log in as the *root* user, if you are not already, to run these examples.

To list files in the current directory with the assumption that you are in the */root* directory:

```
[root@server1 ~]# ls  
anaconda-ks.cfg  Documents  initial-setup-ks.cfg  Pictures  Templates  
Desktop          Downloads  Music                  Public    Videos
```

To list files in the current directory with detailed information:

```
[root@server1 ~]# ls -l
total 8
-rw-----. 1 root root 1447 Aug 22 15:27 anaconda-ks.cfg
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Desktop
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Documents
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Downloads
-rw-r--r--. 1 root root 1602 Aug 23 06:23 initial-setup-ks.cfg
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Music
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Pictures
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Public
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Templates
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Videos
```

The long listing in the output above furnishes a unique piece of information about the file or directory in nine discrete columns:

Column 1: The first character (hyphen or d) divulges the file type, and the next nine characters (rw-rw-r--) indicate permissions.

Column 2: Displays the number of links (links are explained later in this chapter)

Column 3: Shows the owner name

Column 4: Exhibits the owning group name

Column 5: Identifies the file size in bytes. For directories, this number reflects the number of blocks being used by the directory to hold information about its contents.

Columns 6, 7, and 8: Displays the month, day, and time of creation or last modification

Column 9: Indicates the name of the file or directory



As an alternative to **ls -l**, you may use its shortcut **ll** for brevity and convenience unless there is a specific need to use the former.

To show the long listing of only the specified directory without showing its contents:

```
[root@server1 ~]# ls -ld /usr
drwxr-xr-x. 12 root root 144 Aug 22 15:06 /usr
```

To display all files in the current directory with their sizes in human-friendly format:

```
[root@server1 ~]# ls -lh
total 8.0K
-rw-----. 1 root root 1.5K Aug 22 15:27 anaconda-ks.cfg
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Desktop
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Documents
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Downloads
-rw-r--r--. 1 root root 1.6K Aug 23 06:23 initial-setup-ks.cfg
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Music
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Pictures
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Public
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Templates
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Videos
```

To list all files, including the hidden files, in the current directory with detailed information:

```
[root@server1 ~]# ls -la
total 44
dr-xr-x---. 15 root root 4096 Aug 30 23:17 .
dr-xr-xr-x. 17 root root 224 Aug 22 15:06 ..
-rw-----. 1 root root 1447 Aug 22 15:27 anaconda-ks.cfg
-rw-----. 1 root root 1044 Aug 30 23:36 .bash_history
-rw-r--r--. 1 root root 18 Aug 12 2018 .bash_logout
-rw-r--r--. 1 root root 176 Aug 12 2018 .bash_profile
-rw-r--r--. 1 root root 176 Aug 12 2018 .bashrc
drwx-----. 10 root root 230 Aug 26 12:59 .cache
drwx-----. 11 root root 215 Aug 26 12:59 .config
-rw-r--r--. 1 root root 100 Aug 12 2018 .cshrc
drwx-----. 3 root root 25 Aug 22 20:46 .dbus
.......
```

To repeat the previous example with the output sorted by date and time with the newest file first:

```
[root@server1 ~]# ls -lt
total 8
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Videos
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Desktop
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Documents
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Downloads
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Music
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Pictures
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Public
drwxr-xr-x. 2 root root 6 Aug 26 12:56 Templates
-rw-r--r--. 1 root root 1602 Aug 23 06:23 initial-setup-ks.cfg
-rw-----. 1 root root 1447 Aug 22 15:27 anaconda-ks.cfg
```

To list contents of the */etc* directory recursively:

```
[root@server1 ~]# ls -R /etc
/etc:
adjtime           hostname          profile.d
aliases            hosts             protocols
alsa               hosts.allow       pulse
alternatives       hosts.deny       qemu-ga
anacrontab         hp                qemu-kvm
asound.conf        idmapd.conf     radvd.conf
at.deny            init.d           ras
audit              inittab          rc0.d
....
```

Run **man ls** at the command prompt to view the manual pages of the *ls* command with all the options it supports and how to use them.

Printing Working Directory

The *pwd* (*print working directory* or *present working directory*) command displays a user's current location in the directory tree. The following example shows that *root* is currently in the */root* directory:

```
[root@server1 ~]# pwd
/root
```

/root is the home directory for the *root* user. The *pwd* command always returns the absolute path to a file or directory.

Navigating Directories

Files are placed in various directories in Linux, and there are tens of thousands of them. Each file and directory is recognized by a unique path in the directory tree. A *path* (or *pathname*) is like a road map that shows you how to get from one place in the directory hierarchy to another. It uniquely identifies a particular file or directory by its absolute or relative location in the directory structure.

An *absolute path* (a.k.a. a *full path* or a *fully qualified pathname*) points to a file or directory in relation to the top of the directory tree. It always starts with the forward slash (/). You can use the *pwd* command to view your current absolute path in the tree:

```
[root@server1 ~]# pwd
/root
```

The output indicates that */root* is the current location for the *root* user in the directory hierarchy, and the leading / identifies this location as a full path.

A *relative path*, on the other hand, points to a file or directory in relation to your current location. This file path never begins with the forward slash (/). It

may begin with two period characters (..) or with a subdirectory name without a leading /, such as `etc/sysconfig`.

It's easy to navigate in the directory hierarchy if you have a good understanding of the absolute and relative paths and the key difference between them. Let's run a couple of examples using the `cd` (*change directory*) command, which is used to switch between directories, and verify the result with the `pwd` command.

To determine the current location and then go one level up into the parent directory using the relative path:

```
[root@server1 ~]# pwd  
/root  
[root@server1 ~]# cd ..  
[root@server1 /]# pwd  
/
```

The above sequence of commands display the current location (`/root` output of `pwd`), then move one level up (`cd ..` is relative to the current location), and finally verify the new location (`/` output of `pwd`). You may want to use the absolute path (`cd /`) instead of (`cd ..`) to go to the top of the directory tree (parent directory of `/root`).

Now, let's switch to the directory `sysconfig`, which is located under `/etc`. There are two options: the absolute path (`/etc/sysconfig`), or the relative path (`etc/sysconfig`):

```
# cd /etc/sysconfig or      # cd etc/sysconfig
```

To change into the `/usr/bin` directory, for instance, you can run either of the following:

```
# cd /usr/bin      or      # cd ../../usr/bin
```

The above example indicates that the absolute path can be used to switch into a target directory regardless of your current location in the hierarchy. However, a relative path must be entered based on your current location in order to move to a target directory.

At this point, if you want to return to your home directory, you can simply run the `cd` command without inputting a path or by supplying the *tilde* character (~) with it. Either of the two will produce the desired result.

```
# cd      or      # cd ~
```

In the above example, you could also use the absolute path (`cd /root`) or a relative path (`cd ../../root`) instead.

To switch between the current and previous directories, issue the `cd` command with the hyphen character (-). See the following example and observe the output:

```
[root@server1 ~]# pwd  
/root  
[root@server1 ~]# cd -  
/usr/bin  
[root@server1 bin]# cd -  
/root  
[root@server1 ~]# pwd  
/root
```

The example shows the use of the hyphen character (-) with the `cd` command to switch between the current and previous directories.

Identifying Terminal Device File

Linux allocates unique *pseudo* (or *virtual*) numbered device files to represent terminal sessions opened by users on the system. It uses these files to communicate with individual sessions. By default, these files are stored in the `/dev/pts` (*pseudo terminal session*) directory. These files are created by the system when a user opens a new terminal session and they are removed on its closure. The destroyed files are recreated and reused for new terminal sessions.

Linux provides a command called `tty` (*teletype*) to identify your current active terminal session. Here is an example:

```
[root@server1 ~]# tty  
/dev/pts/0
```

The output discloses the filename “0” and its location (`/dev/pts` directory).

Inspecting System’s Uptime and Processor Load

The `uptime` command is used to display the system’s current time, length of time it has been up for, number of users currently logged in, and the average CPU (processing) load over the past 1, 5, and 15 minutes. See the following output:

```
[root@server1 ~]# uptime  
08:22:25 up 1 day, 23:58, 2 users, load average: 0.00, 0.00, 0.00
```

The output shows the current system time (08:22:25), up duration (1 day, 23 hours, and 58 minutes), number of logged-in users (2), and the CPU load averages over the past 1, 5, and 15 minutes (0.00, 0.00, and 0.00), respectively.

The load average numbers correspond to the percentage of CPU load with 0.00 and 1.00 represent no load and full load, and a number greater than 1.00 signifies excess load (over 100%).

Clearing the Screen

The *clear* command clears the terminal screen and places the cursor at the top left of the screen. This command is useful to clear the screen of any distractive content and run new commands on a clean slate.

```
# clear
```

You can also use Ctrl+I for this command.

Determining Command Path

RHEL provides a set of tools that can be used to identify the absolute path of the command that will be executed when you run it without specifying its full path. These tools are the *which*, *whereis*, and *type* commands. The following examples show the full location of the *ls* command:

```
[root@server1 ~]# which ls
/usr/bin/ls
[root@server1 ~]# whereis ls
ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz /usr/share/man/man1p/ls.1p.gz
[root@server1 ~]# type ls
ls is hashed (/usr/bin/ls)
```

As shown above, all three commands responded with an identical path location for the *ls* command, which is */usr/bin/ls*. This implies that there is no need to type the entire path */usr/bin/ls* to run the *ls* command, as the system will automatically determine the location of *ls* based on some predefined settings. Refer to [Chapter 07 “The Bash Shell”](#) for more guidance.

Viewing System Information

There are several elements in the RHEL system that identify various information regarding the operating system, hardware, kernel, storage, networking, and so on. The *uname* command identifies elementary information about the system including its hostname. Without any options, the output of this command is restricted to displaying the operating system name only; however, it reports other details by adding the -a option.

```
[user1@server1 ~]$ uname  
Linux  
[user1@server1 ~]$ uname -a  
Linux server1.example.com 4.18.0-80.el8.x86_64 #1 SMP Wed Mar 13 12:02:46 UTC 20  
19 x86_64 x86_64 x86_64 GNU/Linux
```

The data returned by the second command above is elaborated below:

Linux	Kernel name
server1.example.com	Hostname of the system
4.18.0-80.el8.x86_64	Kernel release
#1 SMP Wed Mar 13 12:02:46 UTC 2019	Date and time of the kernel build
x86_64	Machine hardware name
x86_64	Processor type
x86_64	Hardware platform
GNU/Linux	Operating system name

Try running the *uname* command with the *-s* (kernel name), *-n* (node name), *-r* (kernel release), *-v* (kernel build date), *-m* (hardware name), *-p* (processor type), *-i* (hardware platform), and *-o* (OS name) options separately to view specific information.

Viewing CPU Specs

A CPU has many architectural pieces that can be looked at using the *lscpu* command. These pieces include the CPU architecture, its operating modes, vendor, family, model, speed, cache memory, and whether it supports virtualization. The following example shows the CPU information from *server1*:

```
[root@server1 ~]# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                1
On-line CPU(s) list:  0
Thread(s) per core:   1
Core(s) per socket:   1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 58
Model name:            Intel(R) Core(TM) i7-3610QM CPU @ 2.30GHz
Stepping:               9
CPU MHz:               2294.784
BogoMIPS:              4589.56
Hypervisor vendor:    KVM
Virtualization type:  full
L1d cache:             32K
L1i cache:             32K
L2 cache:               256K
L3 cache:               6144K
NUMA node0 CPU(s):    0
Flags:     fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cm
ov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_
good nopl xtopology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq monitor ssse3
cx16 pcid sse4_1 sse4_2 x2apic popcnt aes xsave avx rdrand hypervisor lahf_lm p
ti fsgsbase flush_l1d
```

The output indicates the architecture of the CPU (x86_64), supported modes of operation (32-bit and 64-bit), sequence number of the CPU on this system (1), threads per core (1), cores per socket (1), number of sockets (1), vendor ID (GenuineIntel), CPU model (58) model name (Intel ...), speed (2294.784 MHz), the amount and levels of cache memory (L1d, L1i, L2, and L3), and other information.

Getting Help

While working on the system, you may require help to obtain information about a command or a configuration file. RHEL offers online help via *manual* pages. Manual pages are online documentation that provides details on commands, configuration files, etc. They are installed under the */usr/share/man* directory when associated software packages are installed.

In addition to the manual pages, *apropos*, *whatis*, *info*, and *pinfo* commands as well as documentation located in the */usr/share/doc* directory are also available on the system.

EXAM TIP: If you need help with a command or configuration file, do not hesitate to use the man pages, refer to the documentation available in the */usr/share/doc* directory, or employ one of the other help tools.

Accessing Manual Pages

Use the *man* command to view manual pages. The following example shows how to check manual pages for the *passwd* command:

```
PASSWD(1)           User utilities          PASSWD(1)

NAME
    passwd - update user's authentication tokens

SYNOPSIS
    passwd [-k] [-l] [-u [-f]] [-d] [-e] [-n mindays] [-x maxdays] [-w
    warndays] [-i inactivedays] [-S] [--stdin] [username]

DESCRIPTION
    The passwd utility is used to update user's authentication token(s).

    This task is achieved through calls to the Linux-PAM and Libuser API.
    Essentially, it initializes itself as a "passwd" service with Linux-PAM
    and utilizes configured password modules to authenticate and then
    update a user's password.

    A simple entry in the global Linux-PAM configuration file for this service would be:

    #
    # passwd service entry that does strength checking of
    # a proposed password before updating it.
    #
    passwd password requisite pam_cracklib.so retry=3
    passwd password required pam_unix.so use_authok
    #

Note, other module types are not required for this application to funco
Manual page passwd(1) line 1 (press h for help or q to quit)
```

The output returns the name of the command, the section of the manual pages it is documented in within the parentheses, and the type (User utilities) of the command on line 1. It then shows a short description (NAME), the command's usage (SYNOPSIS), and a long description (DESCRIPTION), followed by a detailed explanation of each option that the command supports and other relevant data. The highlighted line at the bottom indicates the line number of the manual page. Press h to get help on navigation, press q to quit and return to the command prompt, use the Up and Down arrow keys to scroll up and down, and the PgUp and PgDn keys to scroll one page at a time.

Table 2-3 summarizes the six keys described above and introduces a few more to assist in navigation.

Key	Action
Enter / Down arrow	Moves forward one line
Up arrow	Moves backward one line
f / Spacebar / Page down	Moves forward one page
b / Page up	Moves backward one page
d / u	Moves down / up half a page
g / G	Moves to the beginning / end of the man pages
:f	Displays line number and bytes being viewed
q	Quits the man pages
/pattern	Searches forward for the specified pattern
?pattern	Searches backward for the specified pattern
n / N	Finds the next / previous occurrence of a pattern
h	Gives help on navigational keys

Table 2-3 Navigating within Manual Pages

Run *man* on any of the commands that you have reviewed so far and navigate using the keys provided in [Table 2-3](#) for practice.

Headings in the Manual

Each page in the manual organizes information under several headings. Some common headings are NAME (of the command or file with a short description), SYNOPSIS (syntax summary), DESCRIPTION (an overview of the command or file), OPTIONS (options available for use), EXAMPLES (some examples to explain the usage), FILES (a list of related files), SEE ALSO (reference to other manual pages or topics), BUGS (any reported bugs or issues), and AUTHOR (contributor information). You may find a subset of these headings or additional headings depending on what information is documented for that command or file.

Manual Sections

Depending on the type of information, the manual information is split into nine sections for organization and clarity. For instance, section 1 refers to user commands (see “NAME” for an example of the *passwd* command), section 4 contains special files, section 5 describes file formats for many system

configuration files, and section 8 documents system administration and privileged commands that are designed for execution by the *root* user.

The default behavior of the *man* command is to search through section 1 and each successive section until it finds a match. There are a few commands in Linux that also have a configuration file with an identical name. One instance is the *passwd* command and the *passwd* file. The former is located in the */usr/bin* directory and the latter in the */etc* directory.

When you run **man passwd**, the *man* command scans through the manual pages and the first occurrence it finds is of the *passwd* command that's stored in section 1. If you want to consult the manual pages of the *passwd* configuration file, you would have to specify the section number with the command (**man 5 passwd**) to instruct it to scan that one particular section only.

```
PASSWD(5)          Linux Programmer's Manual          PASSWD(5)

NAME
    passwd - password file

DESCRIPTION
    The /etc/passwd file is a text file that describes user login accounts
    for the system. It should have read permission allowed for all users
    (many utilities, like ls(1) use it to map user IDs to usernames), but
    write access only for the superuser.

    In the good old days there was no great problem with this general read
    permission. Everybody could read the encrypted passwords, but the
    hardware was too slow to crack a well-chosen password, and moreover the
    basic assumption used to be that of a friendly user-community. These
    days many people run some version of the shadow password suite, where
    /etc/passwd has an 'x' character in the password field, and the
    encrypted passwords are in /etc/shadow, which is readable by the superuser
    only.

    If the encrypted password, whether in /etc/passwd or in /etc/shadow, is
    an empty string, login is allowed without even asking for a password.
    Note that this functionality may be intentionally disabled in applications,
    or configurable (for example using the "nullok" or "nonull"
    arguments to pam_unix.so).

    If the encrypted password in /etc/passwd is "*NP*" (without the
    quotes), the shadow record should be obtained from an NIS+ server.

Manual page passwd(5) line 1 (press h for help or q to quit)
```

The header at the top in the above output is an indication that this help is for the *passwd* file and it is documented in section 5.

Searching by Keyword

Sometimes you need to use a command, but you can't recall its name. Linux allows you to perform a keyword search on manual pages using the *man* command with the *-k* (lowercase) flag, or the *apropos* command. These commands search all sections of the manual pages and show a list of all entries matching the specified keyword in their names or descriptions.

Before you can perform keyword searches on a new Linux installation, you'll need to run the *mandb* command in order to build an indexed database of the manual pages. This activity depends on the speed and the number of RHEL packages installed on the system, and it should not take long to perform. Simply type the command at the prompt and press the Enter key as follows:

```
# mandb
Updating index cache for path '/usr/share/man/ja/man7'. Wait...done.
Checking for stray cats under /usr/share/man/ja...
Checking for stray cats under /var/cache/man/ja...
Processing manual pages under /usr/share/man/fr...
Updating index cache for path '/usr/share/man/fr/man5'. Wait...done.
Checking for stray cats under /usr/share/man/fr...
Checking for stray cats under /var/cache/man/fr...
Processing manual pages under /usr/share/man/it...
Updating index cache for path '/usr/share/man/it/man8'. Wait...done.
.....
Checking for stray cats under /usr/local/share/man...
Checking for stray cats under /var/cache/man/local...
107 man subdirectories contained newer manual pages.
7318 manual pages were added.
0 stray cats were added.
0 old database entries were purged.
```

Now you are ready to run keyword lookups. As an example, to find a forgotten XFS administration command, search for a string “xfs” by running either **man -k xfs** or **apropos xfs**. Both will produce an identical result.

```
[root@server1 ~]# man -k xfs
attr (1)           - extended attributes on XFS filesystem objects
filesystems (5)    - Linux filesystem types: ext, ext2, ext3, ext4, hpfs, i...
fs (5)             - Linux filesystem types: ext, ext2, ext3, ext4, hpfs, i...
fsck.xfs (8)       - do nothing, successfully
fsfreeze (8)       - suspend access to a filesystem (Ext3/4, ReiserFS, JFS,...)
mkfs.xfs (8)       - construct an XFS filesystem
xfs (5)            - layout, mount options, and supported file attributes f...
xfs_admin (8)      - change parameters of an XFS filesystem
xfs_bmap (8)       - print block mapping for an XFS file
....
```

Once you have identified the command you were looking for, you can check that command's manual pages for usage.

Some commands also support the use of the --help and -? parameters. These parameters provide a brief list of options and a description without going through the manual pages. For example, to get quick help on the *passwd* command, run either **passwd --help** or **passwd -?**:

```
[root@server1 ~]# passwd --help
Usage: passwd [OPTION...] <accountName>
  -k, --keep-tokens      keep non-expired authentication tokens
  -d, --delete           delete the password for the named account (root
                        only); also removes password lock if any
  -l, --lock              lock the password for the named account (root only)
  -u, --unlock            unlock the password for the named account (root only)
  -e, --expire             expire the password for the named account (root only)
  -f, --force              force operation
....
```



Not all commands support the `--help` and `-?` parameters.

Exposing Short Description

The `whatis` command searches for a short description of the specified command or file in the manual database. It quickly scans through the installed manual pages for the specified string and displays all matching entries. For instance, the following shows outputs of the command when run on `yum.conf` and `passwd` files:

```
[root@server1 ~]# whatis yum.conf
yum.conf (5)          - redirecting to DNF Configuration Reference
[root@server1 ~]# whatis passwd
passwd (5)            - password file
passwd (1)             - update user's authentication tokens
openssl-passwd (1ssl) - compute password hashes
```

The first output indicates that the specified file is a configuration file associated with the `yum` command, and the second output points to three entries for the `passwd` file (one configuration file and two commands).

You may alternatively run `man -f yum.conf` and `man -f passwd` for the exact same results. Both `man -f` and `whatis` produce identical output.

The `info` and `pinfo` Commands

The `info` and `pinfo` commands display command documentation in a document-like format in great detail. Documentation is divided into sections called *nodes*. The header across the top of the screen shows the name of the file being displayed, names of the current, next, and previous nodes, and the name of the node prior to the current node. The operation of the `info` and `pinfo` commands is almost identical. The following example provides a look at the first screen exhibited when either command is executed on the `/s`:

```

Next: dir invocation, Up: Directory listing

10.1 'ls': List directory contents
=====
The 'ls' program lists information about files (of any type, including
directories). Options and file arguments can be intermixed arbitrarily,
as usual.

For non-option command-line arguments that are directories, by
default 'ls' lists the contents of directories, not recursively, and
omitting files with names beginning with '.'. For other non-option
arguments, by default 'ls' lists just the file name. If no non-option
argument is specified, 'ls' operates on the current directory, acting as
if it had been invoked with a single argument of '.'.

```

While viewing help with *info* or *pinfo*, some common keys listed in [Table 2-4](#) will help you navigate efficiently.

Key	Action
Down / Up arrows	Moves forward / backward one line
Spacebar / Del	Moves forward / backward one page
q	Quits the tutorial
[/]	Goes to the previous / next node in the document
t	Goes to the top node of the document
s	Searches forward for the specified string
{ / }	Searches for the previous / next occurrence of the string

Table 2-4 Navigating within info Documentation

Run *info/pinfo* on any of the commands you have reviewed so far and navigate using the keys provided in [Table 2-4](#) for practice.

Documentation in the /usr/share/doc Directory

The */usr/share/doc* directory stores general documentation for installed packages under subdirectories that match their names. For example, the documentation for the *gzip* package includes the following files:

```
[root@server1 ~]# ls -l /usr/share/doc/gzip
total 144
-rw-r--r--. 1 root root    98 Oct  5  2014 AUTHORS
-rw-r--r--. 1 root root 88326 Aug 12  2018 ChangeLog
-rw-r--r--. 1 root root 22553 Jan  7  2018 NEWS
-rw-r--r--. 1 root root   6134 Jan  7  2018 README
-rw-r--r--. 1 root root 13262 Oct  5  2014 THANKS
-rw-r--r--. 1 root root  3467 Jan  7  2018 TODO
```

These text files contain general information about the gzip package. However, the manual pages and the info documentation prove to be more resourceful to Linux users and administrators.

Red Hat Enterprise Linux 8 Documentation

The website at docs.redhat.com hosts documentation for Red Hat's various products including RHEL 8 in HTML, PDF, and EPUB formats (see Figure 2-6).



Figure 2-6 Red Hat's Webpage for RHEL 8 Documentation

This set of documentation includes release notes, as well as guides on planning, installation, administration, security, storage management, virtualization, and so on. For reference, you can download any of these guides for free in a format of your choice.

Chapter Summary

This chapter touched upon a few basic topics. We started by looking at a new display protocol that has replaced the legacy X Window System in RHEL 8. We interacted with the operating system at the graphical console screen and examined various settings.

Next, we reviewed the Linux file system structure standard and significant higher-level subdirectories that consisted of static and variable files, and were grouped logically into lower-level subdirectories.

We analyzed command construct, and learned and run selected user level commands for familiarity. These tools included listing, viewing, and identifying basic information, and navigating in the directory hierarchy.

Finally, we learned how to access online help for commands and configuration files. We saw how to search through manual pages for desired text. Explanations regarding what commands to use were offered for additional help.

Review Questions

1. What is the use of the *apropos* command?
2. Which Linux service creates device nodes at system startup?
3. What is the function of the *pwd* command?
4. Which three formats RHEL 8 documentation is available in at docs.redhat.com?
5. What type of information does section 5 in manual pages contain?
6. Which protocol has replaced X Window System as the default display protocol in RHEL 8?
7. Consider a Linux system with six processor cores. How many times do you have to run the *lscpu* command to pull information for all the cores?
8. Which two commands can you use to display the tutorial of a command?
9. Which three categories can Linux file systems be divided into?
10. Which three commands may be used to identify the full path of a command?
11. Which Linux command displays the hierarchy of a directory?
12. Linux commands may be categorized into two groups: privileged and non-privileged. True or False?
13. What is the name of the default display manager in RHEL 8?
14. What are the -R and -a options used for with the *ls* command?
15. What is the default number of days files in */tmp* are kept before they are automatically deleted if not accessed or modified.
16. Which command can be used to determine the time the kernel was build?
17. RHEL follows the Filesystem Hierarchy Standard. True or False?

Answers to Review Questions

1. The *apropos* command can be used to perform a keyword search in the manual pages.
2. The *udevd* service.
3. The *pwd* command shows the absolute path of the current working directory.

4. The RHEL 8 documentation is available in PDF, EPUB, and HTML formats.
5. Section 5 of the manual pages contain information on configuration files.
6. Wayland is the new default display protocol in RHEL 8.
7. Only one time.
8. You can use the *info* or the *pinfo* command to display the tutorial of a command.
9. Linux file systems may be divided into disk-based, network-based, and memory-based file systems.
10. The *which*, *whereis*, and *type* commands may be used to determine the full path for a specified command.
11. The *tree* command shows the hierarchy of a directory.
12. True.
13. GNOME Display Manager is the default display manager in RHEL 8.
14. The -R option is used for recursive directory listing and the -a option for listing hidden files.
15. Ten days.
16. The *uname* command shows the kernel build time.
17. True.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 2-1: Navigate Linux Directory Tree

As *user1* on *server1*, execute the *pwd* command to check your location in the directory tree. Run the *ls* command with appropriate switches to show files in the current directory including the hidden files. Change directory into */etc* and run *pwd* again to confirm the directory change. Switch back to the directory where you were before, and run *pwd* again to verify. (Hint: Basic System Commands).

Lab 2-2: Miscellaneous Tasks

As *user1* on *server1*, execute the **tty** command to identify the terminal device file. Observe the terminal number reported. Open a couple of more terminal sessions, and run the **tty** command and compare the terminal numbers. Execute the **uptime** command and analyze the system uptime and processor

load information. Use the three commands—**which**, **whereis**, and **type**—and identify the location of the **vgs** command. (Hint: Basic System Commands).

Lab 2-3: Identify System and Kernel Information

As **user1** on **server1**, issue **uname -a**. Analyze the basic information about the system and kernel reported. Run the **lscpu** command and examine the key items relevant to the processor. (Hint: Basic System Commands).

Lab 2-4: Use Help

As **user1** on **server1**, run **man uname** and **man 5 shadow**, and browse various headings and understand what they contain. Try **apropos ext4** and **man -k ext4**, **whatis group**, and observe their outputs. (Hint: Getting Help).

Chapter 03

Basic File Management

This chapter describes the following major topics:

- Common types of file in Linux
- Compress and uncompress files
- Archive and compress files and directories
- Edit files with the vim editor
- Create, list, display, copy, move, rename, and remove files and directories
- Create file and directory links
- Identify differences between copying and linking

RHCSA Objectives:

06. Archive, compress, unpack, and uncompress files using tar, star, gzip, and bzip2
07. Create and edit text files
08. Create, delete, copy, and move files and directories
09. Create hard and soft links

Linux supports different file types that are identified based on the kind of data they store. There are files that save information in plain text or binary format. This file type is very common. There are other files that store device information or simply point to the same data on the disk. A good comprehension of Linux file types is important for both Linux users and administrators.

Compressing and archiving one or more large files or an entire directory hierarchy allows users to conserve disk space or remote copy them at a faster pace. The resulting compressed archive can be easily uncompressed and unarchived whenever and wherever needed. RHEL offers native tools to support both user needs.

Normal and application users and database and system administrators all need to edit text files on a regular basis as part of their job. Linux delivers several text editors for this purpose, including the vim editor, which is popular within the Linux community. A sound, working knowledge of this tool is essential for all these roles.

There are a number of common operations that can be performed on files and directories in addition to viewing their contents. These operations include creating, listing, copying, moving, renaming, and removing both files and directories. Normal users will require higher privilege in order to perform these operations outside of their realm.

There is a tool available in the operating system that helps in linking files and directories for various use cases. An understanding of when to link files or directories versus when to copy them is important.

Common File Types

RHEL supports seven types of files: regular, directory, block special device, character special device, symbolic link, named pipe, and socket. The first two are the most common in Linux. The two types of device files are used by the operating system to communicate with peripheral devices. There are many instances of symbolic links as well. The last two types—named pipes and sockets—are used in inter-process communication.

Linux does not require an extension to a file to identify its type. It provides two elementary commands called *file* and *stat*, in addition to the *ls* command, to ascertain the type of data that a file may contain. This chapter discusses the

first five file types and skips the last two (named pipes and sockets), as they are beyond the scope of this book.

Regular Files

Regular files may contain text or binary data. These files may be shell scripts or commands in the binary form. When you list a directory, all line entries for files in the output that begin with the hyphen character (-) represent regular files. The following truncated output is of the `/root` directory:

```
[root@server1 ~]# ls -l  
total 8  
-rw----- 1 root root 1447 Aug 22 15:27 anaconda-ks.cfg
```

Notice the hyphen in field 1 of column 1 before `rw`. This character indicates that the listed file is a regular file. Now, let's run the `file` and `stat` commands on this file and see what they report:

```
[root@server1 ~]# file anaconda-ks.cfg  
anaconda-ks.cfg: ASCII text  
[root@server1 ~]#  
[root@server1 ~]# stat anaconda-ks.cfg  
  File: anaconda-ks.cfg  
  Size: 1447          Blocks: 8          IO Block: 4096   regular file  
Device: fd00h/64768d  Inode: 10176129      Links: 1  
Access: (0600/-rw-----)  Uid: (    0/    root)    Gid: (    0/    root)  
Context: system_u:object_r:admin_home_t:s0  
Access: 2019-09-02 22:18:40.881000000 -0400  
Modify: 2019-08-22 15:27:48.030000000 -0400  
Change: 2019-08-22 15:27:48.030000000 -0400  
 Birth: -
```

The two commands report the file type differently. The first command returns the specific type of data that the file contains (ASCII text), and the latter simply states that it is a regular file.

Directory Files

Directories are logical containers that hold files and subdirectories. The following `ls` command output shows a few directories from `/usr/bin`:

```
[root@server1 ~]# ls -l /usr  
total 224  
dr-xr-xr-x.  2 root root 40960 Aug 22 15:16 bin  
drwxr-xr-x.  2 root root     6 Aug 12  2018 games  
drwxr-xr-x.  3 root root    24 Aug 22 15:08 include  
dr-xr-xr-x.  37 root root  4096 Aug 22 15:16 lib  
dr-xr-xr-x. 119 root root 65536 Aug 22 15:22 lib64  
drwxr-xr-x.  49 root root 12288 Aug 22 15:16 libexec  
drwxr-xr-x.  12 root root  131 Aug 22 15:06 local
```

The letter “d” at the beginning of each line entry identifies the file as a directory. Try running the **file** and **stat** commands on `/usr` and see what they report.

Block and Character Special Device Files

Each piece of hardware in the system has an associated file in the `/dev` directory that is used by the system to communicate with that device. This type of file is called a *device file*. There are two types of device files: *character* (or *raw*) and *block*. In the example below, the `ls` command in field 1 of column 1 distinguishes between the two with a “c” for character and a “b” for block:

```
[root@server1 ~]# ls -l /dev/console
crw-----. 1 root root 5, 1 Aug 31 08:25 /dev/console
[root@server1 ~]#
[root@server1 ~]# ls -l /dev/sd*
brw-rw----. 1 root disk 8, 0 Aug 31 08:24 /dev/sda
brw-rw----. 1 root disk 8, 1 Aug 31 08:24 /dev/sda1
brw-rw----. 1 root disk 8, 2 Aug 31 08:24 /dev/sda2
```

Use the **file** and **stat** commands on `/dev/console` and `/dev/sda` files for additional verification.

Every hardware device such as a disk, CD/DVD, printer, and terminal has an associated device driver loaded in the kernel. The kernel communicates with hardware devices through their respective device drivers. Each device driver is assigned a unique number called the *major* number, which the kernel uses to recognize its type.

Furthermore, there may be more than one instance of the same device type in the system. In that case, the same driver is used to control all those instances. For example, SATA device driver controls all SATA hard disks and CD/DVD drives. The kernel in this situation allots a *minor* number to each individual device within that device driver category to identify it as a unique device. This scheme applies to disk partitions as well. In short, a major number points to the device driver, and a minor number points to a unique device or partition that the device driver controls.

In the above long listing, columns 5 and 6 depict the major and minor number association for each device instance. Major number 8 represents the block device driver for SATA disks. Similarly (not shown in the above output), major number 253 denotes the driver for device mapper (VDO volumes, LVM logical volumes, etc.), and 11 signifies optical devices. VDO volumes are discussed in [Chapter 13 “Basic Storage Partitioning”](#) and LVM logical volumes are covered in [Chapter 14 “Advanced Storage Partitioning”](#).

Symbolic Links

A *symbolic link* (a.k.a. a *soft link* or a *symlink*) may be considered a shortcut to another file or directory. If you issue `ls -l` on a symbolically linked file or directory, the line entry will begin with the letter “l”, and an arrow will be pointing to the target link. For example:

```
[root@server1 ~]# ls -l /usr/sbin/vigr
lrwxrwxrwx. 1 root root 4 Dec 18 2018 /usr/sbin/vigr -> vigrw
```

Run the **file** and **stat** commands on `/usr/sbin/vigr` for additional confirmation.

Compression and Archiving

Compression tools are used to compress one or more files to conserve space. They may be used with archive commands, such as `tar` or `star`, to create a single compressed archive of hundreds of files and directories. A compressed archive can then be copied to a remote system faster than a non-compressed archive or stored at a backup location. RHEL offers a multitude of compression tools such as `gzip` (`gunzip`) and `bzip2` (`bunzip2`).

The `tar` and `star` commands have the ability to preserve general file attributes such as ownership, owning group, and timestamp as well as extended attributes such as ACLs and SELinux contexts.



[Chapter 04](#) “Advanced File Management” expounds on ACLs, and [Chapter 21](#) “Security Enhanced Linux” on SELinux.

The syntax and usage of `tar` and `star` commands are similar, and most of the common options are identical. We discuss only the `tar` command.

Using `gzip` and `gunzip`

The `gzip/gunzip` compression utility pair has been available in Linux for over two decades. The `gzip` command is used to create a compressed file of each of the specified files and it adds the `.gz` extension to each file for identification. This tool can be used with the `-r` option to compress an entire directory tree, and with the `-l` option to display compression information about a gzipped file. The `-l` option also instructs the command to display the filename that will be given to the file when it is uncompressed.

To compress the file `fstab` located in the `/etc` directory, copy this file in the `root` user’s home directory `/root` using the `cp` command and confirm with `ls`:

```
[root@server1 ~]# pwd  
/root  
[root@server1 ~]# cp /etc/fstab .  
[root@server1 ~]# ls -l fstab  
-rw-r--r--. 1 root root 579 Sep 2 23:01 fstab
```

Now use the *gzip* command to compress this file and *ls* to confirm:

```
[root@server1 ~]# gzip fstab  
[root@server1 ~]# ls -l fstab.gz  
-rw-r--r--. 1 root root 349 Sep 2 23:01 fstab.gz
```

Notice that the original file is compressed, and it now has the *.gz* extension added to it. If you wish to view compression information for the file, run the *gzip* command again with the *-l* option:

```
[root@server1 ~]# gzip -l fstab.gz  
      compressed      uncompressed  ratio   uncompressed_name  
            349                  579    43.9%  fstab
```

To decompress this file, use the *gunzip* command:

```
[root@server1 ~]# gunzip fstab.gz  
[root@server1 ~]# ls -l fstab  
-rw-r--r--. 1 root root 579 Sep 2 23:01 fstab
```

Check the file after the decompression with the *ls* command. It will be the same file with the same size, timestamp, and other attributes.

Using bzip2 and bunzip2

The *bzip2/bunzip2* is another compression pair that has been available in Linux for a long time. The *bzip2* command creates a compressed file of each of the specified files and it adds the *.bz2* extension to each file for identification.

Let's compress the *fstab* file again, but this time with *bzip2* and confirm with *ls*:

```
[root@server1 ~]# bzip2 fstab  
[root@server1 ~]# ls -l fstab.bz2  
-rw-r--r--. 1 root root 386 Sep 2 23:01 fstab.bz2
```

Notice that the original file is compressed, and it now has the *.bz2* extension. To decompress this file, use the *bunzip2* command:

```
[root@server1 ~]# bunzip2 fstab.bz2  
[root@server1 ~]# ls -l fstab  
-rw-r--r--. 1 root root 579 Sep 2 23:01 fstab
```

Check the file after the decompression with the `/s` command. It will be the same file with the same size, timestamp, and other attributes.

Differences between gzip and bzip2

The function of both `gzip` and `bzip2` is the same: to compress and decompress files. However, in terms of the compression and decompression rate, `bzip2` has a better compression ratio (smaller target file size), but it is slower. These differences are evident on fairly large files. On small files, you can use either of the two. Both commands support several identical options.

Using tar

The `tar` (*tape archive*) command is used to create, append, update, list, and extract files or an entire directory tree to and from a single file, which is called a *tarball* or *tarfile*. This command can be instructed to also compress the tarball after it has been created.

`tar` supports a multitude of options such as those described in [Table 3-1](#).

Option	Definition
<code>-c</code>	Creates a tarball.
<code>-f</code>	Specifies a tarball name.
<code>-p</code>	Preserve file permissions. Default for the root user. Specify this option if you create an archive as a normal user.
<code>-r</code>	Appends files to the end of an extant uncompressed tarball.
<code>-t</code>	Lists contents of a tarball.
<code>-u</code>	Appends files to the end of an extant uncompressed tarball provided the specified files being added are newer.
<code>-v</code>	Verbose mode.
<code>-x</code>	Extracts or restores from a tarball.

Table 3-1 tar Command Options

The `-r` and `-u` options do not support adding files to an existing compressed tarball.

A few examples are provided below to elucidate the use of `tar`. Pay special attention to the syntax and options used in each command and observe the

output.

To create a tarball called */tmp/home.tar* of the entire */home* directory, use the *-v* option for verbosity and the *-f* option to specify the name of the archive file with the command. The following is a truncated output of the command:

```
[root@server1 ~]# tar -cvf /tmp/home.tar /home
tar: Removing leading `/' from member names
/home/
/home/user1/
/home/user1/.mozilla/
/home/user1/.mozilla/extensions/
/home/user1/.mozilla/plugins/
/home/user1/.mozilla/firefox/
/home/user1/.mozilla/firefox/3cbv1n5q.default/
```

The resulting tarball will not include the leading forward slash (/) in the file paths as indicated on line 1 of the output even though the full path of */home* is supplied. This is the default behavior of the *tar* command, which gives you the flexibility to restore the files at any location of your choice without having to worry about the full pathnames. Use the *-P* option at the creation time to override this behavior.

To create a tarball called */tmp/files.tar* containing only a select few files (two files in this example) from the */etc* directory:

```
[root@server1 ~]# tar -cvf /tmp/files.tar /etc/passwd /etc/yum.conf
tar: Removing leading `/' from member names
/etc/passwd
tar: Removing leading `/' from hard link targets
/etc/yum.conf
```

To append files located in the */etc/yum.repos.d* directory to the existing tarball */tmp/home.tar*:

```
[root@server1 ~]# tar -rvf /tmp/files.tar /etc/yum.repos.d
tar: Removing leading `/' from member names
/etc/yum.repos.d/
/etc/yum.repos.d/redhat.repo
tar: Removing leading `/' from hard link targets
```

To list what files are included in the *home.tar* tarball:

```
[root@server1 ~]# tar -tvf /tmp/files.tar
-rw-r--r-- root/root      2484 2019-08-26 23:20 etc/passwd
lrwxrwxrwx root/root          0 2019-02-14 07:02 etc/yum.conf -> dnf/dnf.conf
drwxr-xr-x root/root        0 2019-08-22 15:46 etc/yum.repos.d/
-rw-r--r-- root/root       358 2019-08-22 15:46 etc/yum.repos.d/redhat.repo
```

To restore a single file, *etc/yum.conf*, from */tmp/files.tar* under */root* and confirm the output with *ls*:

```
[root@server1 ~]# cd  
[root@server1 ~]# tar -xf /tmp/files.tar etc/yum.conf  
[root@server1 ~]# ls -l etc/yum.conf  
lrwxrwxrwx. 1 root root 12 Feb 14 2019 etc/yum.conf -> dnf/dnf.conf
```

To restore all files from */tmp/files.tar* under */root* and confirm the output with *ls*:

```
[root@server1 ~]# tar -xf /tmp/files.tar  
[root@server1 ~]# ls -l etc  
total 4  
-rw-r--r--. 1 root root 2484 Aug 26 23:20 passwd  
lrwxrwxrwx. 1 root root 12 Feb 14 2019 yum.conf -> dnf/dnf.conf  
drwxr-xr-x. 2 root root 25 Aug 22 15:46 yum.repos.d
```

tar also supports options to directly compress the target file while being archived using the *gzip* or *bzip2* command. These options are described in [Table 3-2](#).

Option	Description
-j	Compresses a tarball with bzip2
-z	Compresses a tarball with gzip

Table 3-2 tar with Compression Options

You will use the options in [Table 3-2](#) to create compressed archives in [Exercise 3-1](#).

EXAM TIP: Archiving and compression are tasks usually done together to produce smaller archive files.

Exercise 3-1: Create Compressed Archives

This exercise should be done on *server1* as *root*.

In this exercise, you will create a tarball called *home.tar.gz* of the */home* directory under */tmp* and compress it with *gzip*. You will create another tarball called *home.tar.bz2* of the */home* directory under */tmp* and compress it with *bzip2*. You will list the content of *home.tar.gz* without uncompressing it and then extract all the files in the current directory. Finally, you will extract the *bzip2*-compressed archive in the */tmp* directory.

1. Create (-c) a *gzip*-compressed (-z) tarball under */tmp* (-f) for */home*:

```
[root@server1 ~]# tar -czf /tmp/home.tar.gz /home
```

2. Create (-c) a *bzip2*-compressed (-j) tarball under */tmp* (-f) for */home*:

```
[root@server1 ~]# tar -cjf /tmp/home.tar.bz2 /home
```

3. List (-t) the content of the *gzip*-compressed archive (-f) without uncompressing it:

```
[root@server1 ~]# tar -tf /tmp/home.tar.gz
home/
home/user1/
home/user1/.mozilla/
home/user1/.mozilla/extensions/
. . . . .
```

4. Extract (-x) files from the *gzip*-compressed tarball (-f) in the current directory:

```
[root@server1 ~]# tar -xf /tmp/home.tar.gz
```

5. Extract (-x) files from the *bzip2*-compressed (-f) tarball under */tmp* (a different directory location than the current directory) (-C):

```
[root@server1 ~]# tar -xf /tmp/home.tar.bz2 -C /tmp
```

Run **ls** */tmp* to view the list of extracted files.

File Editing

The *vim editor* is an interactive, full-screen *visual* text-editing tool that allows you to create and modify text files. This tool is available as a standard editor in all vendor UNIX versions and Linux distributions. It does not require the graphical capability and it is not heavy on compute resources. All text editing within *vim* takes place in a *buffer* (a small chunk of memory used to hold file updates). Changes can either be written to the disk or discarded.

It is essential for you as a system administrator to master the *vim* editor skills. The best way to learn *vim* is to practice by opening or creating a file and run the *vim* commands. See the manual pages of *vim* for details. Alternatively, you can run the *vimtutor* command to view the tutorial.

Modes of Operation

The *vim* editor has three modes of operation: the command mode, the input mode, and the last line mode. The fourth mode is referred to as the visual mode, but it is not discussed in the book.

The *command* mode is the default mode of vim. The *vim* editor places you into this mode when you start it. While in the command mode, you can carry out tasks such as copy, cut, paste, move, remove, replace, change, and search on text, in addition to performing navigational operations. This mode is also known as the *escape* mode because the Esc key is used to enter the mode.

In the *input* mode, anything that is typed on the keyboard is entered into the file as text. Commands cannot be run in this mode. The input mode is also called the *edit* mode or the *insert* mode. You need to press the Esc key to return to the command mode.

While in the command mode, you may carry out advanced editing tasks on text by pressing the colon character (:), which places the cursor at the beginning of the last line of the screen, and hence it is referred to as the *last line* or *extended* mode. This mode is considered a special type of command mode.

Starting vim

The *vim* editor may be started by typing the command *vim* at the command prompt, and it may follow an existing or a new filename as an argument. Without a specified filename, it simply opens an empty screen where you can enter text. You can save the text in a file or discard using commands provided in subsequent subsections.

```
[root@server1 ~]# vim
```

Alternatively, you can supply a filename as an argument. This way, *vim* will open the specified file for editing if the file exists, or it will create a file by that name if it does not exist.

```
[root@server1 ~]# vim <filename>
```

There are options available that you may specify at the time of starting this editing tool. Refer to the manual pages for more information.

Inserting text

Once *vim* is started, there are six commands that can switch into the edit mode. These commands are simple lower and uppercase i, a, and o, and are described in [Table 3-3](#).

Command	Action
i	Inserts text before the current cursor position
I	Inserts text at the beginning of the current line
a	Appends text after the current cursor position
A	Appends text to the end of the current line
o	Opens a new line below the current line
O	Opens a new line above the current line

Table 3-3 Inserting Text

Press the Esc key when you've finished entering text in the edit mode to return to the command mode.

Navigating within vim

Navigation keys are helpful in editing small and large files. They allow you to make rapid moves in the file. There are multiple key sequences available within *vim* to control the cursor movement. Some of the elementary keystrokes are elaborated in [Table 3-4](#).

Command	Action
h	Moves backward one character
j	Moves downward one line
k	Moves upward one line
l	Moves forward one character
w	Moves to the start of the next word
b	Moves backward to the start of the preceding word
e	Moves to the ending character of the next word
\$	Moves to the end of the current line
Enter	Moves to the beginning of the next line
Ctrl+f	Scrolls down to the next page
Ctrl+b	Scrolls up to the previous page

Table 3-4 Navigating within vim

You can precede any of the commands listed in [Table 3-4](#) by a numeral to repeat the command action that many times. For instance, 3h would move the cursor three places to the left, 5Enter would move the cursor five lines below, and 2Ctrl+f would move the cursor two screens down.

In addition, you can use 0 (zero) to move to the beginning of the current line, [[to move to the first line of the file, and]] to move to the last line of the file.

Deleting Text

vim provides several commands to carry out delete operations. Some of the commands are described in [Table 3-5](#).

Command	Action
x	Deletes the character at the cursor position
X	Deletes the character before the cursor location
dw	Deletes the word or part of the word to the right of the cursor location
dd	Deletes the current line
D	Deletes at the cursor position to the end of the current line
:6,12d	Deletes lines 6 through 12

Table 3-5 Deleting Text

You can precede any of the commands listed in [Table 3-5](#), except for the last line mode command, by a numeral to repeat the command action that many times. For instance, 2X would delete two characters before the cursor position, and 3dd would delete the current line and the two lines below it.

Undoing and Repeating

[Table 3-6](#) explicates the commands that undo the last change made and repeat the last command run.

Command	Action
u	Undoes the previous command.
U	Undoes all the changes done on the current line.
:u	Undoes the previous last line mode command.
. (dot)	Repeats the last command run.

Table 3-6 Undoing and Repeating

You can precede any of the commands listed in [Table 3-6](#), except for the U and :u commands, by a numeral to repeat the command action that many times. For instance, 2u would undo the previous two changes, and 2U would undo all the changes done on the current and the previous lines.

Searching for Text

You can perform forward and reverse searches while in the command mode by using the / and ? characters followed by the string to be searched. For instance, in a file with numerous occurrences of the string “profile,” you can run /profile or ?profile for a forward or reverse search.

[Table 3-7](#) summarizes these actions.

Command	Action
/string	Searches forward for a string
?string	Searches backward for a string
n	Finds the next occurrence of a string
N	Finds the previous occurrence of a string

Table 3-7 Searching for Text

For forward searches, repeating “n” takes the cursor to the previous occurrences of the searched string, and repeating “N” moves the cursor to the next occurrences.

The behavior is reversed for backward searches. Repeating “n” takes the cursor to the next occurrences of the searched string, and repeating “N” moves the cursor to the previous occurrences.

Replacing Text

[Table 3-8](#) describes two last line mode commands that are used to perform a search and replace operation.

Command	Action
:%s/old/new	Replaces the first occurrence of <i>old</i> with <i>new</i> in a file. For example, replace the first occurrence of profile with Profile, use :%s/profile/Pr
:%s/old/new/g	Replaces all occurrences of <i>old</i> with <i>new</i> in a file. For example, to replace all the occurrences of profile with Profile in a file, use :%s/profile/Pr

Table 3-8 Replacing Text

If you have used either of these and would like to undo it, use the last line mode command :u.

Copying, Moving, and Pasting Text

vim allows you to copy some text and paste it to the desired location within the file. You can copy (yank) a single character, a single word, or an entire line, and then paste it wherever you need it. The copy function can be performed on multiple characters, words, or lines simultaneously. [Table 3-9](#) describes the copy, move, and paste commands.

Command	Action
yl	Yanks the current letter into buffer
yw	Yanks the current word into buffer
yy	Yanks the current line into buffer
p	Pastes yanked data below the current line
P	Pastes yanked data above the current line
:1,3co6	Copies lines 1 through 3 and pastes them after line 6
:4,6m9	Moves lines 4 through 6 after line 9

Table 3-9 Copying, Moving, and Pasting Text

You can precede any of the commands listed in [Table 3-9](#), except for the last line mode commands, by a numeral to repeat the command action that many times. For instance, 2yw would yank two words, 2yy would yank two lines, and 2p would paste two times.

Changing Text

There are numerous commands available within *vim* to change and modify text as summarized in [Table 3-10](#). Most of these commands switch into the edit mode, so you will have to press the Esc key to return to the command mode.

Command	Action
cl	Changes the letter at the cursor location
cw	Changes the word (or part of the word) at the cursor location to the end word
cc	Changes the entire line
C	Changes text at the cursor position to the end of the line
r	Replaces the character at the cursor location with the character entered following this command
R	Overwrites or replaces the text on the current line
J	Joins the next line with the current line
xp	Switches the position of the character at the cursor position with the character to the right of it
~	Changes the letter case (uppercase to lowercase, and vice versa) at the cursor location

Table 3-10 Changing Text

You can precede any of the commands listed in [Table 3-10](#) by a numeral to repeat the command action that many times. For instance, 2cc would change

the entire current and the next line, and 2r would replace the current character and the next character.

Saving and Quitting vim

When you are done with modifications, you can save or discard them. Use one of the commands listed in [Table 3-11](#) as required.

Command	Action
:w	Writes changes into the file without quitting vim
:w file2	Writes changes into a new file called <i>file2</i> without quitting vim
:w!	Writes changes to the file even if the file owner does not have write permission on the file
:wq	Writes changes to the file and quits vim
:wq!	Writes changes to the file and quits vim even if the file owner does not have write permission on the file
:q	Quits vim if no modifications were made
:q!	Quits vim if modifications were made, but we do not wish to save them

Table 3-11 Saving and Quitting vim

The exclamation mark (!) can be used to override the write protection placed on the file for the owner.

EXAM TIP: You may be required to edit configuration files. Working knowledge of the vim editor is crucial.

File and Directory Operations

This section elaborates on various management operations that can be performed on files and directories. These operations include creating, displaying contents, copying, moving, renaming, and deleting files and directories. These common operations can be performed by normal users who own or have appropriate permissions. The *root* user can accomplish these tasks on any file or directory on the system, regardless of who owns it. In case there's a lack of user permissions, an error message is generated.

Creating Files and Directories

Files can be created in multiple ways using different commands; however, there is only one command to create directories.

Creating Empty Files Using touch

The *touch* command creates an empty file. If the file already exists, it simply updates the timestamp on it to match the current system date and time. Execute the following as *root* in the *root* user's home directory to create *file1* and then run *ls* to verify:

```
[root@server1 ~]# cd  
[root@server1 ~]# touch file1  
[root@server1 ~]# ls -l file1  
-rw-r--r--. 1 root root 0 Sep  5 08:59 file1
```

As expected, column 5 (the size column) in the output is 0, meaning that *file1* is created with zero bytes in size. If you rerun the *touch* command on this file after a minute or so, a new timestamp is placed on it:

```
[root@server1 ~]# touch file1  
[root@server1 ~]# ls -l file1  
-rw-r--r--. 1 root root 0 Sep  5 09:03 file1
```

The *touch* command has a few interesting options. The *-d* and *-t* options set a specific date and time on a file; the *-a* and *-m* options enable you to change only the access or the modification time on a file to the current system time; and the *-r* option sets the modification time on a file to that of a reference file's. Let's use a couple of these options in the examples below:

To set the date on *file1* to September 20, 2019:

```
[root@server1 ~]# touch -d 2019-09-20 file1  
[root@server1 ~]# ls -l file1  
-rw-r--r--. 1 root root 0 Sep 20 2019 file1
```

To change the modification time on *file1* to the current system time:

```
[root@server1 ~]# touch -m file1  
[root@server1 ~]# ls -l file1  
-rw-r--r--. 1 root root 0 Sep  5 09:13 file1
```

Try the rest of the options for practice.

Creating Short Files Using *cat*

The *cat* command allows you to create short text files. The ending angle bracket “>” must be used to redirect the output to the specified file (*catfile1* in this example):

```
[root@server1 ~]# cat > catfile1
```

Nothing is displayed when you execute the above, as the system is waiting for you to input something. Type some text. Press the Enter key to open a new line and continue typing. When you are done, press Ctrl+d to save the text in *catfile1* and return to the command prompt. You can verify the file creation with the *ls* command.

Creating Files Using vim

You can use the *vim* editor to create and modify text files of any size. Refer to the previous section in this chapter on how to use vim.

Making Directories Using mkdir

The *mkdir* command is used to create directories. This command shows an output if you run it with the *-v* option. The following example demonstrates the creation of a directory called *dir1* in the *root* user's home directory:

```
[root@server1 ~]# mkdir dir1 -v
mkdir: created directory 'dir1'
[root@server1 ~]# ls -ld dir1
drwxr-xr-x. 2 root root 6 Sep  5 09:23 dir1
```

You can create a hierarchy of subdirectories by specifying the *-p* (parent) option with *mkdir*. In the following example, *mkdir* is used to create the hierarchy *dir2/perl/perl5*:

```
[root@server1 ~]# mkdir -vp dir2/perl/perl5
mkdir: created directory 'dir2'
mkdir: created directory 'dir2/perl'
mkdir: created directory 'dir2/perl/perl5'
```

Notice the placement of options in the two examples. Many commands in Linux accept either format.

Displaying File Contents

RHEL offers a variety of tools for showing file contents. Directory contents are simply the files and subdirectories that it contains. Use the *ls* command as explained earlier to view directory contents.

For viewing files, you can use the *cat*, *more*, *less*, *head*, and *tail* commands. These tools are explained below.

Using cat

cat displays the contents of a text file. It is typically used to view short files. It shows the entire file on the screen. The following example shows the *.bash_profile* file in the *root* user's home directory with the *cat* command:

```
[root@server1 ~]# cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs
PATH=$PATH:$HOME/bin

export PATH
```

You can add the `-n` option to the `cat` command to view the output in numbered format.

Using tac

`tac` displays the contents of a text file in reverse. In the example below, the `.bash_profile` file in the `root` user's home directory is displayed with the `tac` command:

```
[root@server1 ~]# tac .bash_profile
export PATH

PATH=$PATH:$HOME/bin

# User specific environment and startup programs

fi
    . ~/.bashrc
if [ -f ~/.bashrc ]; then
# Get the aliases and functions

# .bash_profile
```

Compare this output with that of the `cat` command's. Both are reversed.

Using less and more

Both `less` and `more` are text filters that are used for viewing long text files one page at a time, starting at the beginning. The `less` command is more capable than the `more` command. `less` does not need to read the entire file before it starts to display its contents, thus making it faster. The `more` command is limited to forward text searching only, whereas `less` is able to perform both forward and backward searches. Run the `less` and `more` commands one at a time and observe the visual difference in the outputs:

```
[root@server1 ~]# less /usr/bin/znew
[root@server1 ~]# more /usr/bin/znew
```

You can navigate with the keys described in [Table 3-12](#) while viewing the files with either tool.

Key	Purpose
Spacebar / f	Scrolls forward one screen
Enter	Scrolls forward one line
b	Scrolls backward one screen
d	Scrolls forward half a screen
h	Displays help
q	Quits and returns to the command prompt
/string	Searches forward for a string
?string	Searches backward for a string; only applies to the less command
n	Finds the next occurrence of a string
N	Finds the previous occurrence of a string; only applies to less command

Table 3-12 Navigating with less and more

If the `/usr/bin/znew` file is unavailable, use `/etc/profile` instead.

Using head and tail

`head` displays the starting few lines of the specified text file. By default, it returns the first ten lines. See the example below:

```
[root@server1 ~]# head /etc/profile
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.
```

The above output includes three empty lines as well. You can pass a numeral to the command as an argument to limit the number of lines in the output. For example, run the following to view only the top three lines from `/etc/profile`:

```
[root@server1 ~]# head -3 /etc/profile
```

On the other hand, the `tail` command displays the ending ten lines from the specified file by default unless a numeral is passed as an argument to alter the behavior. Issue the following two commands on your terminal to witness the difference:

```
[root@server1 ~]# tail /etc/services
[root@server1 ~]# tail -3 /etc/services
```

The *tail* command is particularly useful when watching a log file while it is being updated. The *-f* (follow) option enables this function. The following example enables us to view the updates to the system log file */var/log/messages* in real time:

```
[root@server1 ~]# tail -f /var/log/messages
```

You may have to wait for some time before you see an update. Press *Ctrl+c* to quit when you are done.

Counting Words, Lines, and Characters in Text Files

The *wc* (*word count*) command displays the number of lines, words, and characters (or bytes) contained in a text file or input supplied. For example, when you run this command on the */etc/profile* file, you will see output similar to the following:

```
[root@server1 ~]# wc /etc/profile
 85  294 2078 /etc/profile
```

Column 1 in the output discloses the number of lines (85) in the file followed by the number of words (294), the number of characters (or bytes) (2078), and the filename (*/etc/profile*).

You can use the options listed in [Table 3-13](#) to restrict the output as desired.

Option	Action
<i>-l</i>	Prints a count of lines
<i>-w</i>	Prints a count of words
<i>-c</i>	Prints a count of bytes
<i>-m</i>	Prints a count of characters

Table 3-13 wc Command Options

The following example displays only the count of characters in */etc/profile*:

```
[root@server1 ~]# wc -m /etc/profile
2078 /etc/profile
```

Try running *wc* with the other options and observe the outcomes.

Copying Files and Directories

The copy operation duplicates a file or directory. RHEL provides the *cp* command for this purpose and it has a variety of options.

Copying Files

The *cp* command copies one or more files within a directory or to another directory. To duplicate a file in the same directory, you must give a different name to the target file. However, if the copy is being made to a different directory, you can use either the same filename or assign a different one. Consider the following examples:

To copy *file1* as *newfile1* within the same directory:

```
[root@server1 ~]# cp file1 newfile1
```

To copy *file1* by the same name to another existing directory *dir1*:

```
[root@server1 ~]# cp file1 dir1
```

By default, the copy operation overwrites the destination file if it exists without presenting a warning. To alter this behavior, use the *-i* (interactive) option to instruct *cp* to prompt for confirmation before overwriting:

```
[root@server1 ~]# cp file1 dir1 -i  
cp: overwrite 'dir1/file1'? y
```

Press Enter after keying in a “y” for yes or an “n” for no to proceed.

By default, you do not need to specify the *-i* option for yes/no confirmation if you attempt to copy a file to overwrite the destination file as *root*. The predefined alias—“alias cp='cp -i’”—in the *.bashrc* file in the *root* user’s home directory takes care of that.

Copying Directories

The *cp* command with the *-r* (recursive) option copies an entire directory tree to another location. In the following example, *dir1* is copied to *dir2* and then the directory contents of *dir2* are listed for validation:

```
[root@server1 ~]# cp -r dir1 dir2  
[root@server1 ~]# ls -l dir2  
total 0  
drwxr-xr-x. 2 root root 19 Sep  5 12:18 dir1  
drwxr-xr-x. 3 root root 19 Sep  5 09:25 perl
```

You may use the *-i* option for overwrite confirmation if the destination already has a matching file or directory.

Try running **ls -l dir2 -R** to view the entire *dir2* hierarchy.

The `cp` command can also use `-p`, which can provide the ability to preserve the attributes (timestamp, permissions, ownership, etc.) of a file or directory being copied. Try running `cp -p file1 /tmp` and then use `/s -l` to compare the attributes for both files.

Moving and Renaming Files and Directories

A file or directory can be moved within the same file system or to another. Within the file system move, an entry is added to the target directory and the source entry is removed, which leaves the actual data intact. On the other hand, a move to a different file system physically moves the file or directory content to the new location and deletes the source.

A rename simply changes the name of a file or directory; data is not touched.

Moving and Renaming Files

The `mv` command is used to move or rename files. The `-i` option can be specified for user confirmation if a file by that name already exists. The following example moves `file1` to `dir1` and prompts for confirmation:

```
[root@server1 ~]# mv -i file1 dir1
mv: overwrite 'dir1/file1'? y
```

By default, you do not need to specify the `-i` option for yes/no confirmation if you attempt to move a file to overwrite the destination file as `root`. The predefined alias—“alias `mv='mv -i'`”—in the `.bashrc` file in the `root` user’s home directory takes care of that.

To rename `newfile1` as `newfile2`:

```
[root@server1 ~]# mv newfile1 newfile2
```

Verify the above operations with `ls -l`.

Moving and Renaming Directories

Use the `mv` command to move a directory and its contents to somewhere else or to change the name of the directory. For example, you can move `dir1` into `dir2` (`dir2` must exist, otherwise it will be a simple rename operation):

```
[root@server1 ~]# mv dir1 dir2
```

To rename `dir2` as `dir20`:

```
[root@server1 ~]# mv dir2 dir20
```

Verify the above operations with `ls -l`.

Removing Files and Directories

The remove operation deletes a file entry from the directory structure and marks its data space as free. For a directory, the remove operation weeds corresponding entries out from the file system structure.

Removing Files

You can remove a file using the *rm* command, which deletes one or more specified files. For example, issue the following command to erase *newfile2*:

```
[root@server1 ~]# rm -i newfile2
rm: remove regular empty file 'newfile2'? y
```

By default, you do not need to specify the *-i* option for yes/no confirmation if you attempt to remove a file as *root*. The predefined alias—"alias rm='rm -i'"—in the *.bashrc* file in the *root* user's home directory takes care of that.

The *rm* command can also be used to erase a file that has a wildcard character, such as an asterisk (*) or a question mark (?), embedded in its name. These characters have special meaning to the shell, and filenames containing them must be prepended with the backslash character (\) to instruct the shell to treat them as regular characters.



A careful use of the *rm* command is particularly important when you have administrative rights on the system.

For example, if a file exists by the name * under the */tmp* directory (use **touch /tmp/*** to create it), you can remove it by executing **rm /tmp/***. If you mistakenly run **rm /tmp/*** instead, all files under */tmp* will be deleted.

Wildcard characters are used in filename globbing and in commands where an action needs to occur on multiple files matching certain criteria. They are discussed in [Chapter 07 “The Bash Shell”](#).

Removing Directories

The *rmdir* and *rm* commands erase directories. The *rmdir* command is used to delete empty directories, while *rm* requires the *-d* option to accomplish the same. In addition, the *-r* or *-R* (recursive) flag with *rm* will remove a directory and all of its contents. Both commands support the *-v* switch for reporting what they are doing. Let's look at a few examples.

To erase an empty directory called *emptydir* (assuming *emptydir* exists), use either of the following:

```
[root@server1 ~]# rmdir emptydir -v      or  
[root@server1 ~]# rm -dv emptydir
```

To remove *dir20* and all its contents recursively, use either *-r* or *-R* with the command:

```
[root@server1 ~]# rm -r dir20
```

The *rm* command supports the *-i* flag for interactive deletions.

EXAM TIP: Manipulating files and directories is one of the most common tasks performed in any operating system. Working knowledge of the tools learned in this section is crucial.

The same rules that apply on filenames with wildcard characters in their names, apply on directory names as well. See the previous topic for details.

File Linking

Each file within a file system has a multitude of attributes assigned to it at the time of its creation. These attributes are collectively referred to as the file's *metadata*, and they change when the file is accessed or modified. A file's metadata includes several pieces of information, such as the file type, size, permissions, owner's name, owning group name, last access/modification times, link count, number of allocated blocks, and pointers to the data storage location. This metadata takes 128 bytes of space for each file. This tiny storage space is referred to as the file's *inode* (*index node*).

An inode is assigned a unique numeric identifier that is used by the kernel for accessing, tracking, and managing the file. In order to access the inode and the data it points to, a filename is assigned to recognize it and access it. This mapping between an inode and a filename is referred to as a *link*. It is important to note that the inode does not store the filename in its metadata; the filename and corresponding inode number mapping is maintained in the directory's metadata where the file resides.

Linking files or directories creates additional instances of them, but all of them eventually point to the same physical data location in the directory tree. Linked files may or may not have identical inode numbers and metadata depending on how they are linked.

There are two ways to create file and directory links in RHEL, and they are referred to as hard links and soft links. Links are created between files or between directories, but not between a file and a directory.

Hard Link

A *hard link* is a mapping between one or more filenames and an inode number, making all hard-linked files indistinguishable from one another. This implies that all hard-linked files will have identical metadata. Changes to the file metadata and content can be made by accessing any of the filenames.

[Figure 3-1](#) shows two filenames—*file10* and *file20*—both sharing the same inode number 10176147. Here, each filename is essentially a hard link pointing to the same inode.

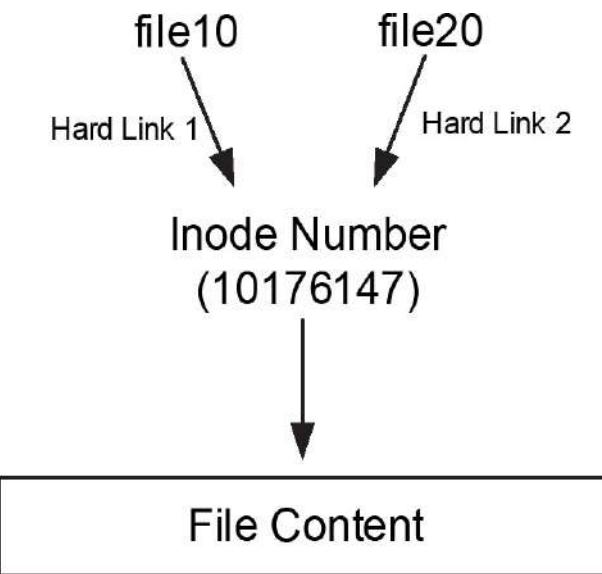


Figure 3-1 Hard Link

A hard link cannot cross a file system boundary, and it cannot be used to link directories because of the restrictions placed within Linux designed to avoid potential issues with some commands.

The following example creates an empty file called *file10* and then uses the *ln* command to create a hard link called *file20* in the same directory:

```
[root@server1 ~]# touch file10
[root@server1 ~]# ln file10 file20
```

After creating the link, run *ls* with the *-li* flags as follows:

```
[root@server1 ~]# ls -li file*
10176147 -rw-r--r--. 2 root root 0 Sep  6 14:29 file10
10176147 -rw-r--r--. 2 root root 0 Sep  6 14:29 file20
```

Look at columns 1 and 3. Column 1 shows the shared inode number (10176147), and column 3 provides a *link count* of the hard links that each file

has (*file10* points to *file20*, and vice versa). If you remove the original file (*file10*), you will still have access to the data through *file20*. Each time you add a hard link to an extant file, the link count will increase by 1. Similarly, if you delete a hard link, the link count will go down by 1. When all the hard links (files) are erased, the link count will set to 0. The increase and decrease in the number of links is reflected on all hard-linked files.

Soft Link

A *soft link* (a.k.a. a *symbolic link* or a *symlink*) makes it possible to associate one file with another. The concept is analogous to that of a shortcut in Microsoft Windows where the actual file is resident somewhere in the directory structure, but there can be one or more shortcuts with different names pointing to it. With a soft link, you can access the file directly via the actual filename as well as any of the shortcuts. Each soft link has a unique inode number that stores the pathname to the file it is linked with. For a symlink, the link count does not increase or decrease, rather each symlinked file receives a new inode number. The pathname can be absolute or relative depending on what was specified at the time of its creation. The size of the soft link is the number of characters in the pathname to the target.

[Figure 3-2](#) shows the file *file10* with a soft link called *softfile10* pointing to it.

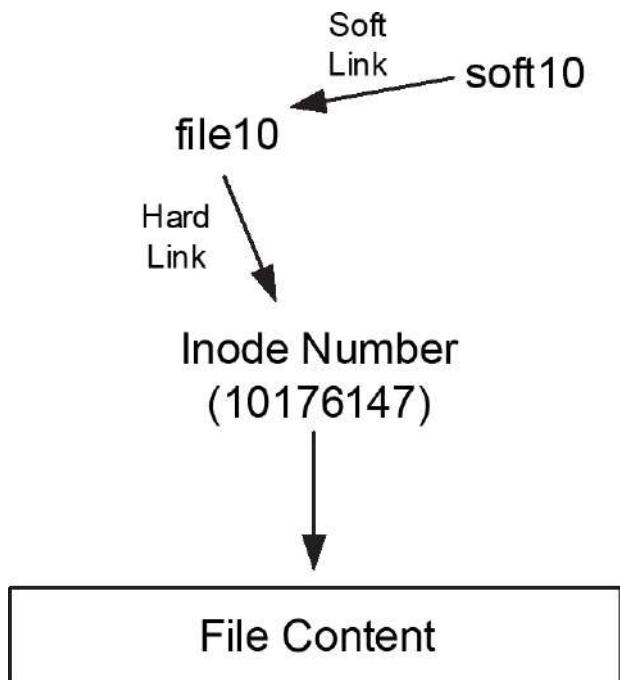


Figure 3-2 Soft Link

A soft link can cross a file system boundary and it can be used to link directories, as it simply uses the pathname of the destination object.

To create a soft link for *file10* as *soft10* in the same directory, use the *ln* command with the *-s* switch:

```
[root@server1 ~]# ln -s file10 soft10
```

After you have created the link, issue *ls -l* and notice the letter “l” as the first character in column 2 of the output. Also notice the arrow that is pointing from the linked file to the original file. Both of these indicate that *soft10* is merely a pointer to *file10*. The *-i* option displays the associated inode numbers in the first column. See the output of **ls -il** below:

```
[root@server1 ~]# ls -il file10 soft10
10176147 -rw-r--r--. 2 root root 0 Sep  6 14:29 file10
10185731 lrwxrwxrwx. 1 root root 6 Sep  6 18:22 soft10 -> file10
```

If you remove the original file (*file10* in this case), the link *soft10* will stay but points to something that does not exist.

RHEL 8 has four soft-linked directories under /. They are:

```
lrwxrwxrwx.  1 root root   7 Aug 12  2018 bin -> usr/bin
lrwxrwxrwx.  1 root root   7 Aug 12  2018 lib -> usr/lib
lrwxrwxrwx.  1 root root   9 Aug 12  2018 lib64 -> usr/lib64
lrwxrwxrwx.  1 root root   8 Aug 12  2018 sbin -> usr/sbin
```

The syntax for creating soft-linked directories is exactly the same as that for soft-linked files.

Differences between Copying and Linking

There are key differences between copying and linking operations. This subsection will discuss when to use copy and when to opt for a soft or hard link. [Table 3-14](#) highlights the main differences between the two:

Copying	Linking
Creates a duplicate of the source file. If either file is modified, the other file will remain intact.	Creates a shortcut that points to the source file. The source can be accessed or modified using either the source file or the link.
Each copied file stores its own data at a unique location.	All linked files point to the same data.
Each copied file has a unique inode number with its unique metadata.	Hard Link: All hard-linked files share the same inode number, and hence the metadata. Symlink: Each symlinked file has a unique inode number, but the inode number stores only the pathname to the source.
If a copy is moved, erased, or renamed, the source file will have no impact, and vice versa.	Hard Link: If the hard link is weeded out, the other file and the data will remain untouched. Symlink: If the source is deleted, the soft link will be broken and become meaningless. If the soft link is removed, the source will have no impact.
Copy is used when the data needs to be edited independent of the other.	Links are used when access to the same source is required from multiple locations.
Permissions on the source and the copy are managed independent of each other.	Permissions are managed on the source file.

Table 3-14 Copying vs. Linking

Keep these differences in mind when you need to decide whether to use copy or a link.

Exercise 3-2: Create and Manage Hard Links

This exercise should be done on *server1* as *root*.

In this exercise, you will create an empty file *hard1* under */tmp* and display its attributes (the inode number, permissions, number of links, owning user, owning group, size, and timestamp). You will create two hard links *hard2* and *hard3* for it, and list the attributes for all three files. You will edit *hard2* and add

some text. You will list the attributes for all three files again, and observe identicalness in all attributes except for the names of files. You will remove *hard1* and *hard3* and list the attributes again for the remaining file. You will notice a decrease in the link count by 2.

1. Create an empty file */tmp/hard1*, and display the long file listing including the inode number:

```
[root@server1 ~]# touch /tmp/hard1
[root@server1 ~]# ls -li /tmp/hard1
103198 -rw-r--r--. 1 root root 0 Jan  8 09:11 /tmp/hard1
```

The file listing indicates the inode number in column 1, followed by permissions (column 2), number of links (column 3), owning user and group (columns 4 and 5), size (column 6), timestamp (columns 7, 8, and 9), and filename (column 10).

2. Create two hard links called *hard2* and *hard3* under */tmp*, and display the long listing:

```
[root@server1 ~]# ln /tmp/hard1 /tmp/hard2
[root@server1 ~]# ln /tmp/hard1 /tmp/hard3
[root@server1 ~]# ls -li /tmp/hard*
103198 -rw-r--r--. 3 root root 0 Jan  8 09:11 /tmp/hard1
103198 -rw-r--r--. 3 root root 0 Jan  8 09:11 /tmp/hard2
103198 -rw-r--r--. 3 root root 0 Jan  8 09:11 /tmp/hard3
```

Observe the file listing. All attributes are identical.

3. Edit file *hard2* and add some random text. Display the long listing for all three files again:

```
[root@server1 ~]# vim /tmp/hard2
[root@server1 ~]# ls -li /tmp/hard*
103198 -rw-r--r--. 3 root root 37 Jan  8 09:21 /tmp/hard1
103198 -rw-r--r--. 3 root root 37 Jan  8 09:21 /tmp/hard2
103198 -rw-r--r--. 3 root root 37 Jan  8 09:21 /tmp/hard3
```

Observe the size and timestamp columns for all three files. They are identical.

4. Erase file *hard1* and *hard3*, and display the long listing for the remaining file:

```
[root@server1 ~]# rm -f /tmp/hard1 /tmp/hard3
[root@server1 ~]# ls -li /tmp/hard*
103198 -rw-r--r--. 1 root root 37 Jan  8 09:21 /tmp/hard2
```

The number of links reduced to 1, all other attributes are the same. You can still access the same data through this last file.

Exercise 3-3: Create and Manage Soft Links

This exercise should be done on `server1` as `root`.

In this exercise, you will create a soft link `soft1` under `/root` pointing to `/tmp/hard2`. You will display the attributes (the inode number, permissions, number of links, owning user, owning group, size, and timestamp) for both files. You will open `soft1` for edit and list the attributes after editing. You will remove `hard2` and then list `soft1`. You will notice that `soft1` becomes invalid, pointing to something that does not exist. Remove `soft1` to complete the exercise.

1. Create soft link `/root/soft1` pointing to `/tmp/hard2`, and display the long file listing for both:

```
[root@server1 ~]# ln -s /tmp/hard2 /root/soft1
[root@server1 ~]# ls -li /tmp/hard2 /root/soft1
9373383 lrwxrwxrwx. 1 root root 10 Jan  8 09:49 /root/soft1 -> /tmp/hard2
103198 -rw-r--r--. 1 root root 37 Jan  8 09:21 /tmp/hard2
```

The file listing indicates the inode number in column 1, followed by permissions (column 2), number of links (column 3), owning user and group (columns 4 and 5), size (column 6), timestamp (columns 7, 8, and 9), and filename (column 10). The soft link file has an “l” prefixed to column 2 and an arrow pointing to the actual file after column 10. Both are indications of a soft link. Notice the file size (10 bytes for the full path `/tmp/hard2`) for `soft1`. Observe similarities and other differences.

2. Edit `soft1` and display the long listing again:

```
[root@server1 ~]# vim /root/soft1
[root@server1 ~]# ls -li /tmp/hard2 /root/soft1
9373383 lrwxrwxrwx. 1 root root 10 Jan  8 09:49 /root/soft1 -> /tmp/hard2
103198 -rw-r--r--. 1 root root 76 Jan  8 09:56 /tmp/hard2
```

The number of bytes for `hard1` and the timestamp reflects the editing. The rest of the attributes are the same.

3. Remove `hard2` and display the long listing:

```
[root@server1 ~]# rm -f /tmp/hard2
[root@server1 ~]# ls -li /tmp/hard2 /root/soft1
ls: cannot access '/tmp/hard2': No such file or directory
9373383 lrwxrwxrwx. 1 root root 10 Jan  8 09:49 /root/soft1 -> /tmp/hard2
```

The actual file, `hard2`, is gone and the soft link is now invalid. You can remove it with `rm -f /root/soft1`.

Chapter Summary

This chapter started with an introduction of common file types that are available in RHEL. A file's type is determined by the type of data it stores. Regular is the most common type of file that stores plain text or binary information. Directories are also very common and there are thousands of them on a typical RHEL system. Other file types include device files and linked files.

We looked at creating and manipulating compressed files and compressed archives. This is a common practice among Linux users for storing old files and transferring a large amount of data to remote systems.

We learned about the vim editor, which is a favorite text file creation and editing tool. We looked at its various modes of operations and switching between them. The basics of vim were discussed, including how to start and insert text, navigate and search for text, copy and paste text, modify and delete text, save edits, and quit with or without saving the changes.

Next, we described file and directory manipulation tools for operations such as creating, listing, displaying, copying, moving, renaming, and removing them. Normal and super users perform these tasks on Linux systems very often.

We examined soft and hard links, and their advantages and limitations. Based on the knowledge gained, we can identify and create the type of link we need for a particular use case.

Finally, we explored the differences between file copying and file linking.

Review Questions

1. Which three Linux utilities can be used to determine a file's type?
2. What is the function of the *tac* command?
3. There are two hard linked files in a directory. How would you identify them?
4. Which *vim* mode allows to execute advanced copy and move functions?
5. A file compressed with *bzip2* can be uncompressed with the *gunzip* command. True or False?
6. Which numeric identifier does the kernel use to determine the uniqueness of a device within a device driver type?
7. What are the two indications in the output of *ls -l* that tells us if the file is a symlink?
8. The *rmdir* command without any switches can be used to erase an entire directory structure. True or False?
9. What would the command *tar pczf output.file /usr/local* do if it is executed by a normal user?

10. The `/s -l` command produces 9 columns in the output by default. True or False?
11. What would the command `wc -c file1` show?
12. What does the kernel use the major number for?
13. The `tail` command can be used to view a file while it is being updated. True or False?
14. Soft linked directories cannot cross file system boundaries, but hard linked directories can. True or False?
15. A file must have the `.exe` extension in order to run. True or False?
16. The `tar` command can be used to archive files with their SELinux contexts. True or False?
17. What would the command `touch file1` do on an existing `file1` file?

Answers to Review Questions

1. The `stat`, `file`, and `/s` commands can be used to determine a file's type.
2. The `tac` command is used to display a text file in reverse.
3. You can identify them by running `/s -li`.
4. The last line mode (extended mode) allows users to copy or move lines.
5. False. The file will have to be uncompressed with either `bzip2` or `bunzip2`.
6. The kernel uses the minor number to identify the uniqueness of a device within a particular device category.
7. A symlink file line entry in the `/s -l` command output begins with the letter `I` and has an arrow pointing to the source file.
8. False. The `rmdir` command is used to erase empty directories.
9. This command will create a gzip'ed tar archive called `output.file` of the `/usr/local` directory with file permissions preserved.
10. True.
11. This command will show the number of bytes in `file1`.
12. The kernel employs the major number to identify the device type.
13. True. You need to include the `-f` switch in the command.
14. False. Soft linked directories can and hard linked directories cannot cross file system boundaries.
15. False.
16. True. The `tar` command has the `--selinux` switch that provides this support.
17. This command will update the access time on `file1`.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without

external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 3-1: Archive, List, and Restore Files

As *user1* on *server1*, execute the *tar* command to create a *gzip*-compressed archive of the */etc* directory. Run the *tar* command again to create a *bzip2*-compressed archive of the */etc* directory. Compare the file sizes of the two archives. Run the *tar* command and uncompress and restore both archives without specifying the compression tool used. (Hint: Compression and Archiving).

Lab 3-2: Practice the vim Editor

As *user1* on *server1*, create a file called *vipractice* in the home directory using *vim*. Type (do not copy and paste) each sentence from Lab 3-1 on a separate line (do not worry about line wrapping). Save the file and quit the editor. Open *vipractice* in *vim* again and reveal line numbering. Copy lines 2 and 3 to the end of the file to make the total number of lines in the file to 6. Move line 3 to make it line 1. Go to the last line and append the contents of the *.bash_profile*. Substitute all occurrences of the string “Profile” with “Pro File”, and all occurrences of the string “profile” with “pro file”. Erase lines 5 to 8. Save the file and quit *vim*. Provide a count of lines, words, and characters in the *vipractice* file using the *wc* command. (Hint: File Editing).

Lab 3-3: File and Directory Operations

As *user1* on *server1*, create one file and one directory in the home directory. List the file and directory and observe the permissions, ownership, and owning group. Try to move the file and the directory to the */var/log* directory and notice what happens. Try again to move them to the */tmp* directory. Duplicate the file with the *cp* command, and then rename the duplicated file using any name. Erase the file and directory created for this lab. (Hint: File and Directory Operations).

Chapter 04

Advanced File Management

This chapter describes the following major topics:

- Understand ugo/rwx access permissions on files and directories
- Know symbolic and octal notations of permission allocation
- Modify permissions for file owner, owning group, and others
- Calculate and set default permissions on new files and directories
- Comprehend and configure special permission bits: setuid, setgid, and sticky
- Use setgid bit for group collaboration
- Apply sticky bit on public and shared writable directories
- Search for files in a variety of different ways
- Grasp and manage extended access controls for named users and named groups
- Set default extended access controls on directories

RHCSA Objectives:

10. List, set, and change standard ugo/rwx permissions

- 36. Create and configure set-GID directories for collaboration
- 39. Diagnose and correct file permission problems
- 56. Create and use file access control lists

Permissions are set on files and directories to prevent access from unauthorized users. Users are grouped into three distinct categories. Each user category is then assigned required permissions. Permissions may be modified using one of two available methods. The user mask may be defined for individual users so new files and directories they create always get preset permissions. Every file in Linux has an owner and a group.

RHEL offers three additional permission bits to control user access to certain executable files and shared directories. A directory with one of these bits can be used for group collaboration. A public or group writable directory may also be configured with one of these bits to prevent file deletion by non-owners.

There is a tool available in RHEL that proves to be very helpful in searching for files at the specified location using a range of options to specify the search criteria. This tool may be set to execute an action on the output files as they are found.

Access Control Lists allows the administrator to enforce extended security attributes on files and directories for specific users or specific user groups. These attributes are on top of the standard Linux access permissions for owning users and owning group members. A directory can have default ACL settings applied to it to allow content sharing among users without having to change permissions on each new file and subdirectory created within it.

File and Directory Access Permissions

Linux is a multi-user operating system that allows hundreds of users the ability to log in and work concurrently. In addition, the OS has hundreds of thousands of files and directories that it must maintain securely to warrant a successful system and application operation from a security standpoint. Given these factors, it is imperative to regulate user access to files and directories and grant them appropriate rights to carry out their designated functions without jeopardizing system security. This control of permissions on files and directories may also be referred to as *user access rights*.

Determining Access Permissions

Access permissions on files and directories allow administrative control over which users (permission classes) can access them and to what level (permission types). File and directory permissions discussed in this section are referred to as *standard ugo/rwx permissions*.

Permission Classes

Users are categorized into three unique classes for maintaining file security through access rights. These classes are *user* (u), *group* (g), and *other* (o, also referred to as *public*). These permission classes represent the owner, the set of users with identical access requirements, and everyone else on the system, respectively. There is another special user class called *all* (a) that represents the three user classes combined.

Permission Types

Permissions control what actions can be performed on a file or directory and by whom. There are three types of permissions bits—read (r), write (w), and execute (x)—and they behave differently for files and directories. For files, the permissions allow viewing and copying (read), modifying (write), and running (execute). And in the case of directories, they allow listing contents with *ls* (read); creating, erasing, and renaming files and subdirectories (write); and enter (with the *cd* command) into it (execute).

If a read, write, or execute permission bit is not desired, the hyphen character (-) is used to represent its absence.

Permission Modes

A permission mode is used to add (+), revoke (-), or assign (=) a permission type to a permission class. You can view the permission settings on files and directories in the long listing of the *ls* command. This information is encapsulated in column 1 of the output, a sample of which is shown below:

- rwx rw- r--

The first character indicates the type of file: - for regular file, d for directory, l for symbolic link, c for character device file, b for block device file, p for named pipe, s for socket, and so on.

The next nine characters—three groups of three characters—show the read (r), write (w), and execute (x) permissions for the three user classes: user (owner), group, and other (public), respectively. The hyphen character (-) represents a permission denial for that level.

Modifying Access Permission Bits

The *chmod* command modifies access rights. It works identically on files and directories. *chmod* can be used by *root* or the file owner, and can modify permissions specified in one of two ways: *symbolic* or *octal*. Symbolic notation

uses a combination of letters (ugo/rwx) and symbols (+, -, =) to add, revoke, or assign permission bits. The octal notation (a.k.a. the *absolute* representation) uses a three-digit numbering system ranging from 0 to 7 to express permissions for the three user classes. Octal values are given in [Table 4-1](#).

Octal Value	Binary Notation	Symbolic Notation	Explanation
0	000	---	No permissions
1	001	--x	Execute permission or
2	010	-w-	Write permission only
3	011	-wx	Write and execute per
4	100	r--	Read permission only
5	101	r-x	Read and execute per
6	110	rw-	Read and write permis
7	111	rwx	Read, write, and execu permissions

Table 4-1 Octal Permission Notation

In [Table 4-1](#), each “1” corresponds to an r, w, or x, and each “0” corresponds to the hyphen character (-) for no permission at that level. [Figure 4-1](#) shows weights associated with each digit position in the 3-digit octal numbering model.

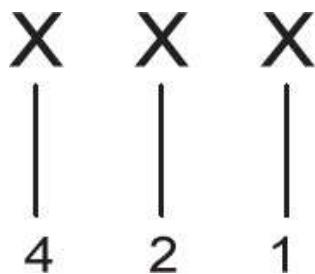


Figure 4-1 Permission Weights

The position to the right is weight 1, the middle position is weight 2, and the left position is weight 4. If we assign a permission of 6, for example, it would correspond to the left and middle positions. Similarly, a permission of 2 would point to the middle position only.

Exercise 4-1: Modify Permission Bits Using Symbolic Form

This exercise should be done on *server1* as *user1*.

For this exercise, presume that a file called *permfile1* exists with read permission for the owner (*user1*), owning group (*user1*), and other, as shown below. If the permissions vary, bring them to the desired state by executing **chmod 444 permfile1** prior to starting the exercise.

```
-r--r--r--. 1 user1 user1 0 Sep  6 20:02 permfile1
```

In this exercise, you will add an execute bit for the owner and a write bit for group and public. You will then revoke the write bit from public and assign read, write, and execute bits to the three user categories at the same time. Finally, you will revoke write from the owning group and write and execute bits from public. The *chmod* command accepts the *-v* switch to display what it has changed. You may alternatively view the long listing after each command execution for verification.

1. Add an execute bit for the owner:

```
[user1@server1 ~]$ chmod u+x permfile1 -v  
mode of 'permfile1' changed from 0444 (r--r--r--) to 0544 (r-xr--r--)
```

2. Add a write bit for group members and public:

```
[user1@server1 ~]$ chmod -v g+w permfile1  
mode of 'permfile1' changed from 0544 (r-xr--r--) to 0566 (r-xrw-rw-)
```

3. Remove the write permission for public:

```
[user1@server1 ~]$ chmod -v o-w permfile1  
mode of 'permfile1' changed from 0566 (r-xrw-rw-) to 0564 (r-xrw-r--)
```

4. Assign read, write, and execute permission bits to all three user categories:

```
[user1@server1 ~]$ chmod a=rwx -v permfile1  
mode of 'permfile1' changed from 0564 (r-xrw-r--) to 0777 (rwxrwxrwx)
```

5. Revoke write bit from the group members and write and execute bits from public:

```
[user1@server1 ~]$ chmod g-w,o-wx permfile1 -v  
mode of 'permfile1' changed from 0777 (rwxrwxrwx) to 0754 (rwxr-xr--)
```

Exercise 4-2: Modify Permission Bits Using Octal Form

This exercise should be done on `server1` as `user1`.

For this exercise, a file called `permfile2` exists with read permission for the owner (`user1`), owning group (`user1`), and other, as shown below. If the permissions vary, bring them to the desired state by executing **chmod 444 permfile2** prior to starting the exercise.

```
-r--r--r--. 1 user1 user1 0 Sep  6 21:12 permfile2
```

In this exercise, you will add an execute bit for the owner and a write permission bit for group and public. You will then revoke the write bit from public and assign read, write, and execute permissions to the three user categories at the same time. The `chmod` command accepts the `-v` flag to display what it has changed. You may alternatively view the long listing after each command execution for verification.

1. Add an execute bit for the owner:

```
[user1@server1 ~]$ chmod -v 544 permfile2
mode of 'permfile2' changed from 0444 (r--r--r--) to 0544 (r-xr--r--)
```

2. Add a write permission bit for group and public:

```
[user1@server1 ~]$ chmod -v 566 permfile2
mode of 'permfile2' changed from 0544 (r-xr--r--) to 0566 (r-xrw-rw-)
```

3. Revoke the write bit for public:

```
[user1@server1 ~]$ chmod -v 564 permfile2
mode of 'permfile2' changed from 0566 (r-xrw-rw-) to 0564 (r-xrw-r--)
```

4. Assign read, write, and execute permission bits to all three user categories:

```
[user1@server1 ~]$ chmod 777 -v permfile2
mode of 'permfile2' changed from 0564 (r-xrw-r--) to 0777 (rwxrwxrwx)
```

Default Permissions

Linux assigns *default permissions* to a file or directory at the time of its creation. Default permissions are calculated based on the *umask* (*user mask*) permission value subtracted from a preset *initial* permissions value.

The umask is a three-digit octal value (also represented in symbolic notation) that refers to read, write, and execute permissions for owner, group, and

public. Its purpose is to set default permissions on new files and directories without touching the permissions on existing files and directories. The default umask value is set to 0022 for the *root* user and 0002 for all normal users. Note that the left-most 0 has no significance. Run the *umask* command without any options and it will display the current umask value in octal notation:

```
[user1@server1 ~]$ umask  
0002
```

Run the command again but with the *-S* option to display the umask in symbolic form:

```
[user1@server1 ~]$ umask -S  
u=rwx,g=rwx,o=rx
```

The predefined initial permission values are 666 (rw-rw-rw-) for files and 777 (rwxrwxrwx) for directories. Even if the umask is set to 000, the new files will always get a maximum of 666 permissions; however, you can add the executable bits explicitly with the *chmod* command if desired.

Calculating Default Permissions

Consider the following example to calculate the default permission values on files for normal users:

Initial Permissions	666
umask	- 002
	(subtract)
Default Permissions	664

This is an indication that every new file will have read and write permissions assigned to the owner and the owning group, and a read-only permission to other.

To calculate the default permission values on directories for normal users:

Initial Permissions	777
umask	- 002
	(subtract)
Default Permissions	775

This indicates that every new directory created will have read, write, and execute permissions assigned to the owner and the owning group, and read and execute permissions to everyone else.

If you want different default permissions set on new files and directories, you will need to modify the umask. You first need to ascertain the desired default

values. For instance, if you want all new files and directories to get 640 and 750 permissions, you can set umask to 027 by running either of the following:

```
[user1@server1 ~]$ umask 027          or  
[user1@server1 ~]$ umask u=rwx,g=rx,o=
```

The new value becomes effective right away, and it will only be applied to files and directories created thereafter. The existing files and directories will remain intact. Now create *tempfile1* and *tempdir1* as *user1* under */home/user1* to test the effect of the new umask:

```
[user1@server1 ~]$ touch tempfile1  
[user1@server1 ~]$ ls -l tempfile1  
-rw-r-----. 1 user1 user1 0 Sep  6 21:42 tempfile1  
[user1@server1 ~]$ mkdir tempdir1  
[user1@server1 ~]$ ls -ld tempdir1  
drwxr-x---. 2 user1 user1 6 Sep  6 21:42 tempdir1
```

The above examples show that the new file and directory were created with different permissions. The file got $(666 - 027 = 640)$ and the directory got $(777 - 027 = 750)$ permissions.

The umask value set at the command line will be lost as soon as you log off. In order to retain the new setting, place it in an appropriate shell startup files discussed in [Chapter 06 “Advanced User Management”](#).

Special File Permissions

Linux offers three types of special permission bits that may be set on binary executable files or directories that respond differently to non-root users for certain operations. These permission bits are *set user identifier* bit (commonly referred to as *setuid* or *suid*), *set group identifier* bit (a.k.a. *setgid* or *sgid*), and *sticky* bit.

The setuid and setgid bits may be defined on binary executable files to provide non-owners and non-group members the ability to run them with the privileges of the owner or the owning group, respectively. The setgid bit may also be set on shared directories for group collaboration. The sticky bit may be set on public directories for inhibiting file erasures by non-owners.



The setuid and sticky bits may be set on directories and files; however, they will have no effect.

The use of the special bits should be regulated and monitored to evade potential security issues to system operation and applications.

The setuid Bit on Binary Executable Files

The setuid flag is set on binary executable files at the file owner level. With this bit set, the file is executed by non-owners with the same privileges as that of the file owner. A common example is the *su* command that is owned by the *root* user. This command has the setuid bit enabled on it by default. See the underlined “s” in the owner’s permission class below:

```
[root@server1 ~]# ls -l /usr/bin/su  
-rwsr-xr-x. 1 root root 52784 Dec 11 2018 /usr/bin/su
```

The *su* (*switch user*) command allows a user to switch to a different user account with the password for the target user. However, the *root* user can switch into any other user account without being prompted for a password. When a normal user executes this command, it will run as if *root* (the owner) is running it and, therefore, the user is able to run it successfully and gets the desired result.

Exercise 4-3: Test the Effect of setuid Bit on Executable Files

This exercise should be done on *server1* as *root* and *user1*.

In this exercise, you will need two terminal windows, one with a *root* session running and another with *user1* on it. As *user1*, you will switch into *root* and observe what happens. As *root*, you will then revoke the setuid bit from the */usr/bin/su* file and retry switching into *root* again. After the completion of the exercise, you will restore the setuid bit on */usr/bin/su*.

1. Log in as *root* and have a terminal window open (let’s call it Terminal 1). Open another terminal (let’s name it Terminal 2) and run the following to switch into *user1*:

```
[root@server1 ~]# su - user1
```

2. On Terminal 2, run the *su* command to switch into *root*:

```
[user1@server1 ~]$ su - root  
Password:  
[root@server1 ~]#
```

The output confirms the switch.

3. On Terminal 1, revoke the setuid bit from */usr/bin/su*:

```
[root@server1 ~]# chmod -v u-s /usr/bin/su  
mode of '/usr/bin/su' changed from 4755 (rwsr-xr-x) to 0755 (rwxr-xr-x)
```

The file is still executable by everyone as indicated by the execute flag; however, it will prevent regular non-owning users from switching accounts, as they have lost that special elevated privilege.

4. On Terminal 2, press Ctrl+d to log off as *root*.
5. On Terminal 2, switch back into *root* and see what happens:

```
[user1@server1 ~]$ su -  
Password:  
su: Authentication failure
```

user1 gets an “authentication failure” message even though they entered the correct password.

6. On Terminal 1, restore the setuid bit on */usr/bin/su*:

```
[root@server1 ~]# chmod -v +4000 /usr/bin/su  
mode of '/usr/bin/su' changed from 0755 (rwxr-xr-x) to 4755 (rwsr-xr-x)
```

With the argument **+4000**, the *chmod* command enables setuid on the specified file without altering any existing underlying permissions.

Alternatively, you can use the symbolic notation as follows:

```
[root@server1 ~]# chmod u+s /usr/bin/su
```



If the file already has the “x” bit set for the group, the long listing will show a lowercase “s”, otherwise it will list it with an uppercase “S”.

The setuid bit has no effect on directories.

The setgid Bit on Binary Executable Files

The setgid attribute is set on binary executable files at the group level. With this bit set, the file is executed by non-owners with the exact same privileges as that of the group members. A common example is the *write* command that is owned by the *root* user with *tty* as the owning group. This command has the setgid bit enabled on it by default. See the “s” in the group’s permission class below:

```
[root@server1 ~]# ls -l /usr/bin/write  
-rwxr-sr-x. 1 root tty 22544 Dec 11 2018 /usr/bin/write
```

The *write* command allows users to write a message on another logged-in user’s terminal. By default, normal users are allowed this special elevated privilege because of the presence of the setgid flag on the file. When a normal user executes this command to write to the terminal of another user, the

command will run as if a member of the *tty* group is running it, and the user is able to execute it successfully.

Exercise 4-4: Test the Effect of setgid Bit on Executable Files

This exercise should be done on *server1* as *root* and *user1*.

In this exercise, you will need two terminal windows, one with a *root* session running and the other with *user1* on it. Both terminal sessions must be opened with ssh. As *user1*, you will produce a list of logged-in users; try to send the *root* user a message and observe what happens. As *root*, you will then revoke the setgid bit from the */usr/bin/write* file and retry sending another message to *root* as *user1*. After the completion of the exercise, you will restore the setgid bit on */usr/bin/write*.

1. Log in as *root* and have a terminal window open (let's call it Terminal 1). Open another terminal (let's name it Terminal 2) and run the following to switch into *user1*:

```
[root@server1 ~]# su - user1
```



The *su* (switch user) command allows a user to switch to a different user account provided the user knows the password of the target user account. However, the *root* user can switch into any other user account without being prompted for a password.

2. On Terminal 2, run the *who* command to list users who are currently logged on:

```
[user1@server1 ~]$ who
root      pts/0          2019-09-06 07:58  (192.168.0.219)
user1     pts/1          2019-09-06 19:36  (192.168.0.219)
```

The output discloses that there are two users—*root* and *user1*—currently signed in.

3. On Terminal 2, execute the *write* command as follows to send a message to *root*:

```
[user1@server1 ~]$ write root
```

4. On Terminal 1, you will see the following message from *user1*:

```
[root@server1 ~]#
Message from user1@server1.example.com on pts/1 at 13:56 ...
```

Any text you type on Terminal 2 will appear on Terminal 1.

5. On Terminal 1, revoke the setgid bit from `/usr/bin/write`:

```
[root@server1 ~]# chmod g-s /usr/bin/write -v  
mode of '/usr/bin/write' changed from 2755 (rwxr-sr-x) to 0755 (rwxr-xr-x)
```

The file is still executable by everyone as indicated by the execute flag; however, it will prevent them from writing to the terminals of other users, as they have lost that special elevated privilege.

6. On Terminal 2, press **Ctrl+c** to terminate the current write session.
7. On Terminal 2, rewrite to *root* and see what happens:

```
[user1@server1 ~]$ write root  
write: effective gid does not match group of /dev/pts/1
```

user1 gets an error as indicated in the above output.

8. On Terminal 1, restore the setgid bit on `/usr/bin/write`:

```
[root@server1 ~]# chmod -v +2000 /usr/bin/write  
mode of '/usr/bin/write' changed from 0755 (rwxr-xr-x) to 2755 (rwxr-sr-x)
```

With the argument `+2000`, the `chmod` command enables setgid on the specified file without altering any existing underlying permissions. Alternatively, you can use the symbolic form as follows:

```
[root@server1 ~]# chmod g+s /usr/bin/write
```



If the file already has the “x” bit set for the group, the long listing will show a lowercase “s”, otherwise it will list it with an uppercase “S”.

The setgid bit has an impact on shared (and public) directories (next subsection).

The setgid Bit on Shared Directories

The setgid bit can also be set on group-shared directories to allow files and subdirectories created underneath to automatically inherit the directory's owning group. This saves group members who are sharing the directory contents from changing the group ID for every new file and subdirectory that they add. The standard behavior for new files and subdirectories is to always receive the creator's group.

Exercise 4-5: Set up Shared Directory for Group Collaboration

This exercise should be done on *server1* as *root* and two test users *user100* and *user200*. Create the user accounts by running **useradd user100** and **useradd user200** (if they don't already exist) as *root*.

In this exercise, you will create a group called *sgrp* with GID 9999, and add *user100* and *user200* to this group as members with shared data needs. You will create a directory called */sdir* with ownership and owning group belonging to *root* and *sgrp*, then set the setgid bit on */sdir* and test. For details on managing users and groups, consult [Chapter 05 “Basic User Management”](#) and [Chapter 06 “Advanced User Management”](#).

1. Add group *sgrp* with GID 9999 with the *groupadd* command:

```
[root@server1 ~]# groupadd -g 9999 sgrp
```

2. Add *user100* and *user200* as members to *sgrp* using the *usermod* command:

```
[root@server1 ~]# usermod -aG sgrp user100  
[root@server1 ~]# usermod -aG sgrp user200
```

3. Create */sdir* directory:

```
[root@server1 ~]# mkdir /sdir
```

4. Set ownership and owning group on */sdir* to *root* and *sgrp*, using the *chown* command:

```
[root@server1 ~]# chown root:sgrp /sdir
```

5. Set the setgid bit on */sdir* using the *chmod* command:

```
[root@server1 ~]# chmod g+s /sdir
```

6. Add write permission to the group members on */sdir* and revoke all permissions from public:

```
[root@server1 ~]# chmod g+w,o-rx /sdir
```

7. Verify the attributes set in the previous three steps using the *ls* command on */sdir*:

```
[root@server1 ~]# ls -ld /sdir  
drwxrws---. 2 root sgrp 6 Sep  7 11:57 /sdir
```

8. Switch or log in as *user100* and change to the */sdir* directory:

```
[root@server1 ~]# su - user100
[user100@server1 ~]$ cd /sdir
[user100@server1 sdir]$
```

9. Create a file and check the owner and owning group on it:

```
[user100@server1 sdir]$ touch file100
[user100@server1 sdir]$ ls -l file100
-rw-rw-r--. 1 user100 sgrp 0 Sep  7 12:07 file100
```

10. Log out as *user100*, and switch or log in as *user200* and change to the */sdir* directory:

```
[user100@server1 sdir]$ exit
logout
[root@server1 ~]# su - user200
[user200@server1 ~]$ cd /sdir
[user200@server1 sdir]$
```

11. Create a file and check the owner and owning group on it:

```
[user200@server1 sdir]$ touch file200
[user200@server1 sdir]$ ls -l file200
-rw-rw-r--. 1 user200 sgrp 0 Sep  7 12:11 file200
```

As shown above, the owning group for each file is the same, *sgrp*, and the group members have identical rights (read and write). Both can modify or delete each other's file. The group members own the files, but the owning group will always be *sgrp* to which they both belong.

EXAM TIP: Some of the steps provided above for setting up a directory for group collaboration may not have to be run in that order.

The Sticky Bit on Public and Shared Writable Directories

The sticky bit is set on public and shared writable directories to protect files and subdirectories owned by normal users from being deleted or moved by other normal users. This attribute is set on the */tmp* and */var/tmp* directories by default as depicted below; however, it can be applied to any writable directory:

```
[root@server1 ~]# ls -l /tmp /var/tmp -d
drwxrwxrwt. 15 root root 4096 Sep  7 12:17 /tmp
drwxrwxrwt.  7 root root 4096 Sep  6 22:03 /var/tmp
```

Notice the underlined letter “t” in other’s permission fields. This indicates the presence of this attribute on the two directories.

Exercise 4-6: Test the Effect of Sticky Bit

This exercise should be done on *server1* as *root* and two test users *user100* and *user200*. Create the user accounts by running **useradd user100** and **useradd user200** (if they don't already exist) as *root*.

In this exercise, you will create a file under */tmp* as *user100* and then try to delete it as *user200*. You will unset the sticky bit on */tmp* and try to erase the file again. After the completion of the exercise, you will restore the sticky bit on */tmp*. For details on managing users and groups, consult [Chapter 05 “Basic User Management”](#) and [Chapter 06 “Advanced User Management”](#).

1. Switch or log in as *user100* and change to the */tmp* directory:

```
[root@server1 ~]# su - user100  
[user100@server1 ~]$ cd /tmp
```

2. Create a file called *stickyfile*:

```
[user100@server1 tmp]$ touch stickyfile
```

3. Log out as *user100* and switch or log in as *user200* and change to the */tmp* directory:

```
[user100@server1 tmp]$ exit  
logout  
[root@server1 ~]# su - user200  
[user200@server1 ~]$ cd /tmp
```

4. Try to erase the file and observe the system reaction:

```
[user200@server1 tmp]$ rm stickyfile  
rm: remove write-protected regular empty file 'stickyfile'? y  
rm: cannot remove 'stickyfile': Operation not permitted
```

It says, “Operation not permitted”. The user cannot remove the file owned by another user.

5. Log out as *user100* and revoke the sticky bit from */tmp* as *root* and confirm:

```
[user100@server1 tmp]$ exit  
logout  
[root@server1 ~]# chmod o-t /tmp  
[root@server1 ~]# ls -ld /tmp  
drwxrwxrwx. 15 root root 4096 Sep 7 13:15 /tmp
```

6. Switch or log back in as *user200* and retry the removal:

```
[root@server1 ~]# su - user200
[user200@server1 ~]$ cd /tmp
[user200@server1 tmp]$ rm stickyfile
rm: remove write-protected regular empty file 'stickyfile'? y
[user200@server1 tmp]$
```

The file is gone. A normal user, *user200*, was able to successfully delete a file, *stickyfile*, that was owned by a different normal user, *user100*, in a public writable directory, */tmp*.

7. Log out as *user200* and restore the sticky bit back on */tmp*:

```
[user200@server1 tmp]$ exit
logout
[root@server1 ~]# chmod -v +1000 /tmp
mode of '/tmp' changed from 0777 (rwxrwxrwx) to 1777 (rwxrwxrwt)
```

With the argument *+1000*, the *chmod* command sets the sticky bit on the specified directory without altering any existing underlying permissions.

Alternatively, you can use the symbolic notation as follows:

```
[root@server1 ~]# chmod o+t /tmp
```



If the directory already has the “x” bit set for public, the long listing will show a lowercase “t”, otherwise it will list it with an uppercase “T”.

The sticky bit can also be set on group writable directories such as */sdir* that you created in the previous exercise.

File Searching

A typical running RHEL system has a few hundred thousand files distributed across several file systems. At times, it is imperative to look for one or more files based on certain criteria. One example would be to find all files owned by employees who left the company over a year ago. Another example would be to search for all the files that have been modified in the past 20 days by a specific user. For such situations, RHEL offers a command called *find*. You supply your search criteria and this command gets you the result. You can also instruct this utility to execute a command on the files as they are found.

Using the *find* Command

The *find* command recursively searches the directory tree, finds files that match the specified criteria, and optionally performs an action on the files as they are discovered. This powerful tool can be tailored to look for files in a

number of ways. The search criteria may include tracking files by name or part of the name, ownership, owning group, permissions, inode number, last access or modification time in days or minutes, size, and file type. [Figure 4-2](#) shows the command syntax.

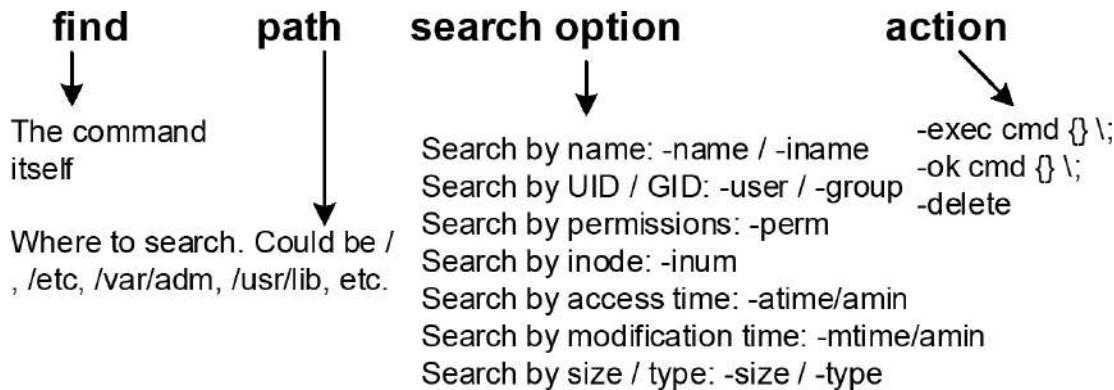


Figure 4-2 find Command Syntax

With *find*, files that match the criteria are located and their full paths are displayed.

To search for a file called *file10* (execute **touch file10** if it does not already exist) by its name (name) in *root*'s home directory, run the *find* command as follows. The period character (.) represents the current directory, which is */root* in this example.

```
[root@server1 ~]# find . -name file10 -print
./file10
```



-print is optional. The *find* command, by default, displays the results on the screen. You do not need to specify this option.

To perform a case-insensitive (-iname) search for files and directories in */dev* that begin with the string “usb” followed by any characters:

```
[root@server1 ~]# find /dev -iname usb*
/dev/bus/usb
/dev/usbmon1
/dev/usbmon0
```

To find files smaller than 1MB (-1M) in size (-size) in the *root* user's home directory (~). You do not need to issue the command from this user's home directory. In fact, you can be anywhere in the directory tree.

```
[root@server1 ~]# find ~ -size -1M
```



The tilde character (~) represents a user's home directory.

To search for files larger than 40MB (+40M) in size (-size) in the /usr directory:

```
[root@server1 ~]# find /usr -size +40M
```

To find files in the entire root file system (/) with ownership (-user) set to user *daemon* and owning group (-group) set to any group other than (-not or ! for negation) *user1*:

```
[root@server1 ~]# find / -user daemon -not -group user1
```

To search for directories (-type) by the name "src" (-name) in /usr at a maximum of two subdirectory levels below (-maxdepth):

```
[root@server1 ~]# find /usr -maxdepth 2 -type d -name src
```

To run the above search but at least three subdirectory levels beneath /usr, substitute -maxdepth 2 with -mindepth 3.

To find files in the /etc directory that were modified (-mtime) more than (the + sign) 2000 days ago:

```
[root@server1 ~]# find /etc -mtime +2000
```

To run the above search for files that were modified exactly 12 days ago, replace "+2000" with "12".

To find files in the /var/log directory that have been modified (-mmin) in the past (the - sign) 100 minutes:

```
[root@server1 ~]# find /var/log -mmin -100
```

To run the above search for files that have been modified exactly 25 minutes ago, replace "-100" with "25".

To search for block device files (-type) in the /dev directory with permissions (-perm) set to exactly 660:

```
[root@server1 ~]# find /dev -type b -perm 660
```

To search for character device files (-type) in the /dev directory with at least (-222) world writable permissions (this example would ignore checking the write and execute permissions):

```
[root@server1 ~]# find /dev -type c -perm -222
```

To find files in the `/etc/systemd` directory that are executable by at least their owner or group members:

```
[root@server1 ~]# find /etc/systemd -perm /110
```

To search for symlinked files (`-type l`) in `/usr` with permissions (`-perm`) set to read and write for the owner and owning group:

```
[root@server1 ~]# find /usr -type l -perm -ug=rw
```

find is a very useful and powerful file-searching tool with numerous other options available to use. Refer to its manual pages, as there are a ton of examples there. Try some of them out for additional practice.

Using `find` with `-exec` and `-ok` Flags

An advanced use of the *find* command is to perform an action on the files as they are found based on any criteria outlined in the previous subsection and in the command's manual pages. The action may include performing basic file management operations such as copying, erasing, renaming, changing ownership, or modifying permissions on each file found. This is done with the `-exec` switch. An equivalent option `-ok` may be used instead, which requires user confirmation before taking an action.

EXAM TIP: The *find* command is very flexible and has a ton of options available to search for files. You should know the use of the `exec` option well.

To search for directories in the entire directory tree (`/`) by the name “core” (`-name`) and list them (`ls -ld`) as they are discovered without prompting for user confirmation (`-exec`):

```
[root@server1 ~]# find / -name core -type d -exec ls -ld {} \;
```



The *find* command replaces `{}` for each filename as it is found. The semicolon character `(;)` marks the termination of the command and it is escaped with the backslash character `(\)`.

In the next example, the *find* command uses the `-ok` switch to prompt for confirmation before it copies each matched file (`-name`) in `/etc/sysconfig` to `/tmp`:

```
[root@server1 ~]# find /etc/sysconfig -name *.conf -ok cp {} \;
```

The destination directory (`/tmp`) is specified between {} and \;. There are many advanced examples provided in the `find` command's manual pages. I suggest to try a few of them.

Access Control Lists (ACLs)

The *Access Control Lists* (ACLs) provide an extended set of permissions that can be applied on files and directories. These permissions are in addition to the standard ugo/rwx permissions and the setuid, setgid, and sticky bit settings. The ACLs define permissions for *named users* and *named groups* using either octal or symbolic representation of permission allocation. The named users may or may not be part of the same group. ACLs are configured and treated the same way on both files and directories.

ACLs are categorized into two groups based on their type and are referred to as *access ACLs* and *default ACLs*. Access ACLs are set on individual files and directories, whereas default ACLs can only be applied at the directory level with files and subdirectories inheriting them automatically. The directory to be applied the default ACLs needs to have the execute bit set at the public level.

ACL Management Commands

RHEL offers two commands—`getfacl` and `setfacl`—to view and manage ACLs on files and directories. The `getfacl` command is used to display ACL settings, and the `setfacl` command can set, modify, substitute, or delete ACL settings.

The `getfacl` Command

The `getfacl` command has several options to see the output as desired; however, it reveals all necessary information without furnishing any flags with it. The example below creates an empty file `acfile1` in `/tmp` and then displays the ACLs on it:

```
[root@server1 ~]# cd /tmp
[root@server1 tmp]# touch acfile1
[root@server1 tmp]# getfacl acfile1
# file: acfile1
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

The output returns the names of the file, owner, and owning group. It then exhibits the existing permissions placed on the file: read and write for the owner, and read-only for everyone else. Notice the pair of colon character (:) in the permission rows. There is a space between them where a named user, a named group, or their corresponding UID or GID is inserted when extended

permissions are set (see [Chapter 05](#) “Basic User Management” for a description of UID and GID). For instance, the ACL setting “user:1000:r--” would imply that the named user with UID 1000, who is neither the file owner nor a member of the owning group, is allowed read-only access to this file. Likewise, the ACL “group:dba:rw-” would give the named group (*dba*) read and write access to the file.

In addition, the file’s long listing is also indicative of the presence of ACLs by exhibiting the plus sign (+) right beside the permissions column. An example column from *ls -l* would be -rw-rw-r--+.

The **setfacl** Command

The *setfacl* command is used to apply, modify, or remove ACL settings. The format for supplying ACL permissions with the command is explained in [Table 4-2](#).

Format	Description
u[ser]:UID:perms	Permissions assigned to a named user (username/UID). The named user must exist in the /etc/passwd file. If this field is left blank, the permission will be applied to the owner of the file or directory, which is equivalent to using the chmod command. See Chapter 05 “Basic User Management” to understand the content and format of the /etc/passwd file.
g[roup]:GID:perms	Permissions assigned to a named group (group/GID). The named group must exist in the /etc/group file. If this field is left blank, the permission will be applied to the owning group of the file or directory, which is equivalent to using the chmod command. See Chapter 05 “Basic User Management” to understand the content and format of the /etc/group file.
o[ther]:perms	Permissions assigned to users that are neither the owner nor part of the owning group.
m[ask]:perms	Maximum permissions for a named user or a named group. If this is set to rw-, for example, then no other user or group will have permissions beyond read and write.

Table 4-2 *setfacl* Command Format for Access ACLs

The *setfacl* command provides a multitude of switches to use depending on what you want to achieve. [Table 4-3](#) describes some of them.

Switch	Description
-b	Removes all access ACLs
-d	Applies to default ACLs
-k	Removes all default ACLs
-m	Sets or modifies ACLs
-n	Prevents an automatic recalculation of the mask
-R	Applies recursively to a directory
-x	Removes an access ACL

Table 4-3 setfacl Command Switches

We will explore some of these options shortly.

EXAM TIP: A good understanding of the usage of the *setfacl* command is critical.

The Role of the mask Value

The value of the ACL mask determines the maximum allowable permissions placed for a named user or group on a file or directory. If it is set to rw, for instance, no named user or group will exceed those permissions. The mask value is displayed on a separate line in the *getfacl* output. Each time ACLs are modified for a file or directory, the mask is recalculated automatically and applied unless an explicit value is input with the *setfacl* command or the -n option is employed to override this behavior. On *aclfile1*, there are currently no ACLs set, as it is a new file. The *getfacl* command with the -c flag displays the output without the header, as shown below:

```
[root@server1 tmp]# getfacl -c aclfile1
user::rw-
group::r--
other::r--
```

If you want to give read and write permissions to a specific user (*user1*) and change the mask to read-only at the same time, the *setfacl* command will allocate the permissions as mentioned; however, the effective permissions for the named user will only be read-only.

```
[root@server1 tmp]# setfacl -m u:user1:rw,m:r aclfile1
[root@server1 tmp]# getfacl -c aclfile1
user::rw-
user:user1:rw-                                #effective:r--
group::r--
mask::r--
other::r--
```

In the example, *user1* will not be able to modify this file even though it appears they have the write permission. The actual permissions for *user1* include both read and write, but they are curtailed to read-only due to the limitation placed by the mask. Now, let's promote the mask value to include the write bit as well:

```
[root@server1 tmp]# setfacl -m m:rw aclfile1
[root@server1 tmp]# getfacl -c aclfile1
user::rw-
user:user1:rw-
group::r--
mask::rw-
other::r--
```

The actual permissions for *user1* are now boosted to include the write bit to reflect the new higher value.

Exercise 4-7: Identify, Apply, and Erase Access ACLs

This exercise should be done on *server1* as *user1*.

In this exercise, you will create a file *acluser1* as *user1* in */tmp* and check to see if there are any ACL settings on the file. You will apply access ACLs on the file for a named user, *user100*, for read and write access. You will observe the change in the mask value. You will then add another named user, *user200*, to the file for full permissions. You will observe the update in the mask value. You will delete the settings for *user200* and then the rest of the access ACLs from the file.

1. Switch or log in as *user1* and create a file *acluser1* in */tmp*:

```
[root@server1 ~]# su - user1
[user1@server1 ~]$ cd /tmp
[user1@server1 tmp]$ touch acluser1
```

2. Use the *ls* and *getfacl* commands and check for the existence of any ACL entries:

The output discloses an absence of ACLs on the file. The owner and group members have read and write permissions and everyone else has the read-only permission.

```
[user1@server1 tmp]$ ls -l acluser1
-rw-rw-r--. 1 user1 user1 0 Sep  8 14:32 acluser1
[user1@server1 tmp]$ getfacl acluser1 -c
user::rw-
group::rw-
other::r--
```

3. Allocate read and write permissions to *user100* with the *setfacl* command using the octal form:

```
[user1@server1 tmp]$ setfacl -m u:user100:6 acluser1
```

4. Run the *ls* command to check if the plus sign (+) has appeared next to the first column, then run the *getfacl* command to check the new access ACLs and the mask:

```
[user1@server1 tmp]$ ls -l acluser1
-rw-rw-r--+ 1 user1 user1 0 Sep  8 14:52 acluser1
[user1@server1 tmp]$ getfacl -c acluser1
user::rw-
user:user100:rw-
group::rw-
mask::rw-
other::r--
```

A row is added for the named user showing read/write permissions. Another row with the mask is also added and it is set to read/write as well. The mask value is auto-calculated based on the current maximum permissions that a named user or group has. This is reflected in the above output.

5. At this point, you can open another terminal session, switch into *user100*, change directory into */tmp*, and open the file *acluser1* in the *vim* editor. You should be able to modify the file and save it.
6. Add *user200* with full rwx permissions to *acluser1* using the symbolic notation and then show the updated ACL settings:

```
[user1@server1 tmp]$ setfacl -m u:user200:rwx acluser1
[user1@server1 tmp]$ getfacl -c acluser1
user::rw-
user:user100:rw-
user:user200:rwx
group::rwx
mask::rwx
other::r--
```

Notice the updated value for the mask, which is increased to rwx to reflect the maximum permission the named user, *user200*, has. If this file were to be an executable command, you would have been able to run it as *user200* based on the assigned ACLs.

7. Delete the ACL entries set for *user200* and validate:

```
[user1@server1 tmp]$ setfacl -x u:user200 acluser1
[user1@server1 tmp]$ getfacl acluser1 -c
user::rw-
user:user100:rw-
group::rw-
mask::rw-
other::r--
```

Notice the reduction in the mask value to *rw-*, which now reflects the new current maximum permissions placed on the named user, *user100*.

8. Delete the rest of the ACLs:

```
[user1@server1 tmp]$ setfacl -b acluser1
```

9. Use the *ls* and *getfacl* commands and confirm for the ACLs removal:

```
[user1@server1 tmp]$ ls -l acluser1
-rw-rw-r--. 1 user1 user1 0 Sep  8 14:52 acluser1
[user1@server1 tmp]$ getfacl acluser1 -c
user::rw-
group::rw-
other::r--
```

This concludes the exercise.

You can also create a group such as *aclgroup1* by running **groupadd -g 8000 aclgroup1** as the *root* user and repeat this exercise by adding this group as a named group along with the two named users (*user100* and *user200*).

Default ACLs

A group collaboration on a shared directory gives members of the group identical access on files and subdirectories in the directory. Access ACLs may be applied to the shared directory to give non-group members certain rights. Furthermore, default ACLs can also be set on the shared directory to ensure new files and subdirectories created under the shared directory always have a consistent set of access rights for group and non-group members. This way the users do not have to adjust permissions on each new file and subdirectory they will create. The inheritance works slightly different for files and subdirectories, as indicated below:

- Files receive the shared directory's default ACLs as their access ACLs
- Subdirectories receive both default ACLs and access ACLs as they are

The default ACLs can be described as the maximum discretionary permissions that can be allocated on a directory. Let's perform the following exercise and see how default ACLs are applied, viewed, and erased.

Exercise 4-8: Apply, Identify, and Erase Default ACLs

This exercise should be done on *server1* as *user1*.

In this exercise, you will create a directory *projects* as *user1* under */tmp*. You will set default ACLs on the directory for named users, *user100* and *user200*, to give them full permissions. You will create a subdirectory *prjdir1* and a file *prjfile1* under *projects* and observe the effects of default ACLs on them. You will delete all the default entries at the end of the exercise.

1. Switch or log in as *user1* and create a directory *projects* in */tmp*:

```
[root@server1 ~]# su - user1
[user1@server1 ~]$ cd /tmp
[user1@server1 tmp]$ mkdir projects
```

2. Use the *getfacl* command for an initial look at the permissions on the directory:

```
[user1@server1 tmp]$ getfacl -c projects
user::rwx
group::rwx
other::r-x
```

The output discloses that the owner and group members have full permissions on the directory and everyone else has read and execute permission.

3. Allocate default read, write, and execute permissions to *user100* and *user200* on the directory. Use both octal and symbolic notations and the *-d* (default) option with the *setfacl* command.

```
[user1@server1 tmp]$ setfacl -dm u:user100:7,u:user200:rwx projects/
[user1@server1 tmp]$ getfacl -c projects/
user::rwx
group::rwx
other::r-x
default:user::rwx
default:user:user100:rwx
default:user:user200:rwx
default:group::rwx
default:mask::rwx
default:other::r-x
```

The named users have the default ACLs. The rest of the default ACL entries are for the directory owner, owning group, and public with the mask reflecting the maximum permissions.

4. Create a subdirectory *prjdir1* under *projects* and observe the ACL inheritance:

```
[user1@server1 projects]$ mkdir prjdir1
[user1@server1 projects]$ getfacl -c prjdir1.
user::rwx
user:user100:rwx
user:user200:rwx
group::rwx
mask::rwx
other::r-x
default:user::rwx
default:user:user100:rwx
default:user:user200:rwx
default:group::rwx
default:mask::rwx
default:other::r-x
```

As stated earlier, the subdirectory *prjdir1* inherited both default and access ACLs. This is reflected in the above output.

5. Create a file *prjfile1* under *projects* and observe the ACL inheritance:

```
[user1@server1 projects]$ touch prjfile1
[user1@server1 projects]$ getfacl -c prjfile1
user::rw-
user:user100:rwx          #effective:rw-
user:user200:rwx          #effective:rw-
group::rwx                #effective:rwx
mask::rw-
other::r--
```

As stated earlier, the file *prjfile1* inherited the parent directory's default ACLs as its access ACLs minus what the mask limits. The maximum effective access for the named users is rw.

6. At this point, you can log in as one of the named users, change directory into */tmp/projects*, and edit *prjfile1* (add some random text). Then change into the *prjdir1* and create file *file100*. You should be able to perform both tasks successfully.

```
[root@server1 ~]# su - user100
[user100@server1 ~]$ cd /tmp/projects
[user100@server1 projects]$ vim prjfile1
[user100@server1 projects]$ ls -l prjfile1
-rw-rw-r--+ 1 user1 user1 9 Sep  9 09:34 prjfile1
[user100@server1 projects]$ cd prjdir1
[user100@server1 prjdir1]$ touch file100
[user100@server1 prjdir1]$ pwd
/tmp/projects/prjdir1
```

7. Delete all the default ACLs from the *projects* directory as *user1* and confirm:

```
[user100@server1 ~]$ exit
logout
[root@server1 ~]# su - user1
[user1@server1 ~]$ cd /tmp
[user1@server1 tmp]$ setfacl -k projects
[user1@server1 tmp]$ getfacl -c projects
user::rwx
group::rwx
other::r-x
```

This concludes the exercise.

You can create a group such as `aclgroup2` by running **groupadd -g 9000 aclgroup2** as the *root* user and repeat this exercise by adding this group as a named group along with the two named users (*user100* and *user200*).

Chapter Summary

This chapter covered four topics: file and directory permissions, special permissions, file searching, and access control.

We learned classes, types, and modes of permissions, looked at octal and symbolic notations of changing permissions, and applied the knowledge to modify user access on files and directories. We examined the concept of default permissions and how they could be employed on parent directories to enable files and subdirectories created underneath to automatically get the desired permissions. We analyzed the role of the `umask` value in determining the new default permissions.

We looked at special permission bits that could be set on executable files to gain privileged access, applied on shared directories for content sharing, and enabled on shared and public writable directories to prevent file deletion by non-owning users.

Next, we explored the criteria and the tool to search for files at the specified directory location. We also employed an extended flag for optional execution of an action on the outcome of the search command.

Finally, we discussed access control on files and directories using ACLs. This feature let named users and named groups to have extended access on given files and directories without the need to modify ownership or owning group on them.

Review Questions

1. The output generated by the `umask` command shows the current user mask in four digits. What is the significance of the left-most digit?
2. What would the command `find /dev -type c -perm 660` do?

3. Default permissions are calculated by subtracting the initial permissions from the umask value. True or False?
4. What would be the effect of the sticky bit on an executable file?
5. Name the permission classes, types, and modes.
6. Default ACLs are meant for directories only. True or False?
7. The default umask for a normal user in bash shell is 0027. True or False?
8. What digit represents the setuid bit in the *chmod* command?
9. What would the command *chmod g-s file1* do?
10. Sticky bit is recommended for every system directory. True or False?
11. What would the command *setfacl -m g:dba:rw file1* do?
12. The setgid bit enables group members to run a command at a higher priority. True or False?
13. What is the equivalent symbolic value for permissions 751?
14. What permissions would the owner of the file get if the *chmod* command is executed with 555?
15. What would the *find / -name core -ok rm {} \;* command do?
16. Which special permission bit is set on a directory for team sharing?

Answers to Review Questions

1. The left-most digit has no significance in the umask value.
2. The *find* command provided will find all character device files under */dev* with exact permissions of 660.
3. False. Default permissions are calculated by subtracting the umask value from the initial permission values.
4. Nothing. The sticky bit is meant for directories only.
5. Permission classes are user, group, and public; permission types are read, write, and execute; and permission modes are add, revoke, and assign.
6. True. Default ACLs are meant for directories only.
7. False. The default umask for bash shell users is 0002.
8. The digit 4 represents the setuid bit.
9. It would erase the setgid bit from *file1*.
10. False.
11. This command will give the users of the *dba* group read and write permissions on *file1*.
12. False.
13. The equivalent for octal 751 is rwxr-x--x.
14. The owner will get read and execute permissions.
15. The *find* command provided will display all files by the name *core* in the entire directory hierarchy and ask for removal confirmation as it finds

them.

16. The setgid bit is set for team sharing.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 4-1: Manipulate File Permissions

As *user1* on *server1*, create file *file11* and directory *dir11* in the home directory. Make a note of the permissions on them. Run the **umask** command to determine the current umask. Change the umask value to 0035 using symbolic notation. Create *file22* and directory *dir22* in the home directory. Observe the permissions on *file22* and *dir22*, and compare them with the permissions on *file11* and *dir11*. Use the **chmod** command and modify the permissions on *file11* to match those on *file22*. Use the **chmod** command and modify the permissions on *dir22* to match those on *dir11*. Do not remove *file11*, *file22*, *dir11*, and *dir22* yet. (Hint: File and Directory Access Permissions).

Lab 4-2: Configure Group Collaboration and Prevent File Deletion

As *root* on *server1*, create directory */sdir*. Create group *sgrp* and add *user1000* and *user2000* (create the users). Set up appropriate ownership (*root*), owning group (*sgrp*), and permissions (rwx for group, --- for public, s for group, and t for public) on the directory to support group collaboration and ensure non-owners cannot delete files. Log on as *user1000* and create a file under */sdir*. Log on as *user2000* and try to edit that file. You should be able to edit the file successfully. As *user2000* try to delete the file. You should not be able to. (Hint: Special File Permissions).

Lab 4-3: Find Files

As *root* on *server1*, execute the *find* command to search for all files in the entire directory structure that have been modified in the last 300 minutes and display their type. Use the *find* command again and search for named pipe and socket files. (Hint: File Searching).

Lab 4-4: Find Files Using Different Criteria

As *root* on *server1*, issue the *find* command to search for regular files under */usr* that were accessed more than 100 days ago, are not bigger than 5MB in size, and are owned by the user *root*. (Hint: File Searching).

Lab 4-5: Apply ACL Settings

As *root* on *server1*, create file *testfile* under */tmp*. Apply ACL settings on the file so that *user2000* gets 7, *user3000* gets 6, and *user4000* gets 4 permissions. Create users. Remove the ACLs for *user2000*, and verify. Erase all remaining ACLs at once, and confirm. (Hint: Access Control Lists).

Chapter 05

Basic User Management

This chapter describes the following major topics:

- Show who is currently logged in
- Review history of successful user login attempts and system reboots
- Report history of failed user log in attempts
- View recent user login attempts
- Examine user and group information
- Understand the content and syntax of local user authentication files
- Analyze user configuration files
- Add, modify, and delete local user accounts with default and custom values
- Set and modify user passwords
- Add user account with nologin access

RHCSA Objectives:

51. Create, delete, and modify local user accounts
52. Change passwords and adjust password aging for local user accounts (only the first part of this objective “change passwords for local user accounts” is covered in this chapter; the second part is in [Chapter 06](#))

User

login activities are monitored and recorded in various files. RHEL offers a set of tools that read the activity data from these files and display the results. In addition to user activities, a history of system reboots is also maintained and it may be viewed with one of the tools. This information may be useful in debugging, testing, or auditing purposes.

In order for an authorized person to gain access to the system, a unique username must be designated and a user account must be created for them. This user is assigned a password and is allowed to change it themselves. User account information is recorded in several files. These files may be edited manually if necessary; however, this practice is discouraged. A good knowledge and grasp of the syntax and the type of information these files store is paramount for Linux administrators. User attributes may be modified later or the account may be removed from the system altogether if not required anymore.

Though service user accounts are added to the system when a corresponding service is installed, there may be situations when they need to be added manually. These accounts do not require login access; their presence is needed to support an installed application.

User Login Activity and Information

On a busy RHEL system, many users sign in and run jobs as themselves, or they switch into the *root* or another user account to run tasks that only those users have the privilege to execute. As an administrator, it is one of your responsibilities to ensure that only authorized users are able to log in to the system. You can keep track of user logins, such as who is currently logged in and their previous and recent successful and unsuccessful login attempts. This information can be of immense help in determining any suspicious login activity. For instance, multiple failed attempts of logging in by an authorized user could be due to a forgotten or lost password, or it might be a result of an unauthorized individual trying to hack in.

There are many log files in RHEL that various services running on the system update automatically and instantly as activities occur. These logs capture user login activities among many others. This section will focus on the log files and tools that are relevant to user logins only. Others will be discussed in later chapters.

Listing Logged-In Users

A list of the users who have successfully signed on to the system with valid credentials can be printed using one of the two basic Linux tools: *who* and *w*. These commands show various pieces of information separated in multiple columns.

The *who* command references the */run/utmp* file and displays the information. Here is a sample from *server1*:

```
[root@server1 ~]# who
root      pts/0          2019-09-06 07:58  (192.168.0.219)
user1    tty2          2019-09-10 12:54  (tty2)
```

Column 1 displays the login name of the user. Column 2 shows the terminal session device filename (pts stands for *pseudo terminal session*, and tty identifies a terminal window on the console). Columns 3 and 4 show the date and time of the user login, and column 5 indicates if the terminal session is graphical (:0), remote (IP address), or textual on the console.

The *w* (*what*) command displays information in a similar format as the *who* command, but it also tells the length of time the user has been idle for (IDLE), along with the CPU time used by all processes including any existing background jobs attached to this terminal (JCPU), the CPU time used by the current process (PCPU), and current activity (WHAT). In the following example, line 1 displays the current system time (12:57:51), the system up duration (4 days, 5 hours, and 3 minutes), number of users currently logged in (2), and the CPU load averages over the past 1, 5, and 15 minutes (0.40, 0.63, and 0.9), respectively. This is exactly what the *uptime* command shows, which was also discussed in [Chapter 02](#) “Initial Interaction with the System”.

```
[root@server1 ~]# w
12:57:51 up 4 days,  5:03,  2 users,  load average: 0.40, 0.63, 0.29
USER     TTY     FROM           LOGIN@   IDLE    JCPU    PCPU WHAT
root     pts/0  192.168.0.219  Fri07    1.00s  1.20s  0.01s w
user1    tty2    tty2          12:54    4days 25.23s  0.35s /usr/libexec/tr
```

The load average numbers represent the percentage of CPU load with 0.00 and 1.00 correspond to no load and full load, and a number greater than 1.00 signifies excess load (over 100%).

Inspecting History of Successful Login Attempts and System Reboots

The *last* command reports the history of successful user login attempts and system reboots by consulting the *wtmp* file located in the */var/log* directory. This file keeps a record of all login and logout activities, including the login

time, duration a user stayed logged in, and tty (where the user session took place). Consider the following two examples.

To list all user login, logout, and system reboot occurrences, issue the *last* command without any arguments:

```
[root@server1 ~]# last
user1  tty2          tty2          Tue Sep 10 12:54  still logged in
user1  pts/1          192.168.0.219   Fri Sep  6 19:36 - 15:21  (19:44)
root   pts/0          192.168.0.219   Fri Sep  6 07:58  still logged in
reboot system boot   4.18.0-80.el8.x8  Fri Sep  6 07:54  still running
root   pts/0          192.168.0.219   Wed Sep  4 22:20 - 20:41  (22:21)
reboot system boot   4.18.0-80.el8.x8  Wed Sep  4 22:20 - 20:42  (22:21)
root   pts/0          192.168.0.219   Wed Sep  4 07:31 - 16:26  (08:55)
root   pts/1          192.168.0.219   Mon Sep  2 13:20 - 13:22  (00:01)
user1  pts/3          192.168.0.219   Sat Aug 31 11:33 - 11:34  (00:00)

.....
reboot  system boot   4.18.0-80.el8.x8 Thu Aug 22 19:55 - 08:57  (13:01)
reboot  system boot   4.18.0-80.el8.x8 Thu Aug 22 15:36 - 08:57  (17:20)

wtmp begins Thu Aug 22 15:36:41 2019
```

The output has information that is distributed across eight to nine columns. Here is the description for each column for user history:

Column 1: Login name of the user

Column 2: Terminal name assigned upon logging in

Column 3: Terminal name or IP address from where the connection was established

Column 4 to 7: Day, month, date, and time when the connection was established

Column 8: Log out time. If the user is still logged on, it will say “still logged in”

Column 9: Duration of the login session

For system reboots, this is what it shows:

Column 1: Action name (reboot)

Column 2: Activity name (system boot)

Column 3: Linux kernel version

Column 4 to 7: Day, month, date, and time when the *reboot* command was issued

Column 8: System restart time

Column 9: Duration the system remained down. If the system is running, it will say “still running”.

The last line in the output indicates the log filename (*wtmp*) being used to record this information and the time when it started to log events.

To list system reboot details only, you can issue the `last` command and specify `reboot` as an argument:

```
[root@server1 ~]# last reboot
reboot    system boot  4.18.0-80.e18.x8 Fri Sep  6 07:54 still running
reboot    system boot  4.18.0-80.e18.x8 Wed Sep  4 22:20 - 20:42 (22:21)
reboot    system boot  4.18.0-80.e18.x8 Sat Aug 31 08:24 - 16:26 (4+08:02)
reboot    system boot  4.18.0-80.e18.x8 Fri Aug 30 20:41 - 07:48 (11:07)
reboot    system boot  4.18.0-80.e18.x8 Mon Aug 26 22:52 - 07:10 (2+08:17)
reboot    system boot  4.18.0-80.e18.x8 Fri Aug 23 14:13 - 15:21 (3+01:07)
```

The output includes the same information that it depicts with the `last` command executed without any argument.

Viewing History of Failed User Login Attempts

The `lastb` command reports the history of unsuccessful user login attempts by reading the `bttmp` file located in the `/var/log` directory. This file keeps a record of all unsuccessful login attempts, including the login name, time, and tty (where the attempt was made). Consider the following example.

To list all unsuccessful login attempts, type the `lastb` command without any arguments. You must be `root` in order to run this command.

```
[root@server1 ~]# lastb
root      ssh:notty      192.168.0.219      Fri Sep  6 07:58 - 07:58 (00:00)
root      ssh:notty      192.168.0.219      Mon Sep  2 13:20 - 13:20 (00:00)

bttmp begins Mon Sep  2 13:20:35 2019
```

The output has information that is presented in nine columns. Here is the description for each column:

Column 1: Name of the user who made the login attempt

Column 2: Name of the protocol used. No tty was assigned as the attempt failed

Column 3: Terminal name or IP address from where the connection attempt was launched

Column 4 to 7: Day, month, date, and time of the attempt

Column 8: Duration the login attempt was tried

Column 9: Duration the login attempt lasted for

The last line in the output discloses the log filename (`bttmp`) being used to record this information and the time when it started to log events.

Reporting Recent User Login Attempts

The `lastlog` command reports the most recent login evidence information for every user account that exists on the system. This information is captured in the `/lastlog` file located in the `/var/log` directory. This file keeps a record of the most recent user login attempts, including the login name, time, and port (or tty). Consider the following example.

```
[root@server1 ~]# lastlog
Username          Port      From           Latest
root              pts/1
bin
daemon
adm
lp
sync
.....
sshd
insights
avahi
tcpdump
user1            tty2
user2            pts/0
user100          pts/0
user200          pts/0
**Never logged in**
Tue Sep 10 12:54:50 -0400 2019
Fri Sep  6 19:38:35 -0400 2019
Mon Sep  9 09:38:41 -0400 2019
Sat Sep  7 13:28:14 -0400 2019
```

The output displays the information across four columns. Here is the description for each column:

Column 1: Login name of the user

Column 2: Terminal name assigned upon logging in

Column 3: Terminal name or IP address from where the session was initiated

Column 4: Timestamp for the latest login or “Never logged in” if the user never signed in

Note that service accounts are used by their respective services, and they are not meant for logging. More information on service accounts is discussed in the next section.

Examining User and Group Information

The `id (identifier)` command displays the calling user's UID (*User IDentifier*), username, GID (*Group IDentifier*), group name, all secondary groups the user is a member of, and SELinux security context. Here is a sample output for the `root` user when this command is executed without an option or argument:

```
[root@server1 ~]# id  
uid=0(root) gid=0(root) groups=0(root) context=unconfined u:unconfined r:unconfined t:s0-s0:c0.c1023
```



Each user and group has a corresponding number (called UID and GID) for identification purposes. These will be discussed in subsequent sections of this chapter. For SELinux, see [Chapter 21](#) “Security Enhanced Linux”.

The *id* command can be executed by a user to view other users’ identification information. The following example shows an instance with the *root* user viewing another user’s id:

```
[root@server1 ~]# id user1  
uid=1000(user1) gid=1000(user1) groups=1000(user1)
```

The *groups* command, in contrast, lists all groups the calling user is a member of:

```
[root@server1 ~]# groups  
root
```

The first group listed is the primary group for the user who executed this command; all other groups are secondary (or supplementary). The *groups* command can also be used to view group membership information for a different user. Try running it as **groups user1** and observe the outcome.

Local User Authentication Files

RHEL supports three fundamental user account types: *root*, *normal*, and *service*. The *root* user (a.k.a. the *superuser* or the *administrator*), has full access to all services and administrative functions on the system. This user is created by default during installation. Normal users have user-level privileges; they cannot perform any administrative functions but can run applications and programs that have been authorized. Service accounts take care of their respective services, which include apache, ftp, mail, and chrony.

User account information for local users is stored in four files that are located in the */etc* directory. These files—*passwd*, *shadow*, *group*, and *gshadow*—are updated when a user or group account is created, modified, or deleted. The same files are referenced to check and validate the credentials for a user at the time of their login attempt, and hence the files are referred to as user authentication files. These files are so critical to the operation of the system that the system creates their automatic backups by default as *passwd-*, *shadow-*, *group-*, and *gshadow-* in the */etc* directory.

Here is the list of the four files and their backups from the */etc* directory:

```
[root@server1 ~]# ls -l /etc/passwd* /etc/group* /etc/shadow* /etc/gshadow*
-rw-r--r--. 1 root root 1024 Sep  7 11:57 /etc/group
-rw-r--r--. 1 root root 1016 Sep  7 11:57 /etc/group-
-----. 1 root root 821 Sep  7 11:57 /etc/gshadow
-----. 1 root root 813 Sep  7 11:57 /etc/gshadow-
-rw-r--r--. 1 root root 2615 Sep  7 11:57 /etc/passwd
-rw-r--r--. 1 root root 2570 Sep  7 11:57 /etc/passwd-
-----. 1 root root 1395 Sep  7 11:57 /etc/shadow
-----. 1 root root 1365 Sep  7 11:57 /etc/shadow-
```

All files are short in size, but they grow bigger as new users are added. Two of the files—*gshadow* and *shadow*—along with their backups have no access permissions for any user, not even for *root*. Let's analyze these files and see what information they store and how.

The *passwd* File

The *passwd* file is a simple plaintext file but it contains vital user login data. Each row in the file holds information for one user account. There are seven colon-separated fields per line entry. A sample row from the file is displayed in Figure 5-1.

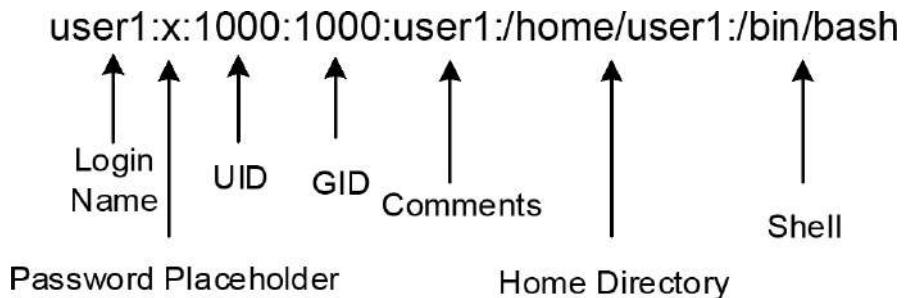


Figure 5-1 The *passwd* File

Here is a description for each field:

Field 1 (Login Name): Contains the login name for signing in. Login names with up to 255 characters, including the underscore (`_`) and hyphen (`-`) characters, are supported. It is not recommended to include special characters and uppercase letters in login names. **Field 2 (Password):** Can contain an “x” (points to the */etc/shadow* file for the actual password), an asterisk * to identify a disabled account, or a hashed password.



A hashed password—a combination of random letters, numbers, and special characters—is an irreversible, unique, and scrambled string of characters to safeguard a clear text password. It is generated as a result of a conversion process of a password using one of the available hashing algorithms. By default, RHEL uses the SHA-512 algorithm for this purpose.



An algorithm is a set of well-defined but complex mathematical instructions used in data encryption and decryption techniques.

Field 3 (UID): Comprises a numeric UID between 0 and approximately 4.2 billion. UID 0 is reserved for the *root* account, UIDs between 1 and 200 are used by Red Hat to statically assign them to core service accounts, UIDs between 201 and 999 are reserved for non-core service accounts, and UIDs 1000 and beyond are employed for normal user accounts. By default, RHEL begins assigning UIDs to new users at 1000.

Field 4 (GID): Holds a GID that corresponds with a group entry in the */etc/group* file. By default, RHEL creates a group for every new user matching their login name and the same GID as their UID. The GID defined in this field represents the user's primary group.

Field 5 (Comments): Called GECOS (*General Electric Comprehensive Operating System* and later changed to GCOS), optionally stores general comments about the user that may include the user's name, phone number, location, or other useful information to help identify the person for whom the account is set up.

Field 6 (Home Directory): Defines the absolute path to the user home directory. A *home* directory is the location where a user is placed after signing in and it is used for personal storage. The default location for user home directories is */home*.

Field 7 (Shell): Consists of the absolute path of the shell file that the user will be using as their primary shell after logging in. The default shell used in RHEL is the Bash shell (*/bin/bash*). Consult [Chapter 07 “The Bash Shell”](#) for details on the Bash shell.

A *head* and *tail* from the *passwd* file for the first and last three lines is shown below:

```
[root@server1 ~]# head -3 /etc/passwd ; tail -3 /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
user1:x:1000:1000:user1:/home/user1:/bin/bash
user100:x:1002:1002::/home/user100:/bin/bash
user200:x:1003:1003::/home/user200:/bin/bash
```

The output indicates the *root* user with UID 0 followed by two service accounts (*bin* and *daemon*). The last three lines display the three user accounts that were created earlier as part of some of the exercises.

Let's verify the permissions and ownership on the *passwd* file:

```
[root@server1 ~]# ls -l /etc/passwd
-rw-r--r--. 1 root root 2574 Sep 11 11:49 /etc/passwd
```

The access permissions on the file are 644 (world-readable and owner-writable), and it is owned by the *root* user.

The shadow File

RHEL has a secure password control mechanism in place that provides an advanced level of password security for local users. This control is referred to as the *shadow password*. With this control mechanism in place, user passwords are hashed and stored in a more secure file */etc/shadow*, but there are certain limits on user passwords in terms of expiration, warning period, etc., that can also be applied on a per-user basis. These limits and other settings are defined in the */etc/login.defs* file, which the shadow password mechanism enforces on user accounts. This is called *password aging*. Unlike the *passwd* file, which is world-readable and owner-writable, the *shadow* file has no access permissions at all. This is done to safeguard the file's content.

With the shadow password mechanism active, a user is initially checked in the *passwd* file for existence and then in the *shadow* file for authenticity.

The *shadow* file contains user authentication and password aging information. Each row in the file corresponds to one entry in the *passwd* file. There are nine colon-separated fields per line entry. A sample row from this file is showcased in [Figure 5-2](#).

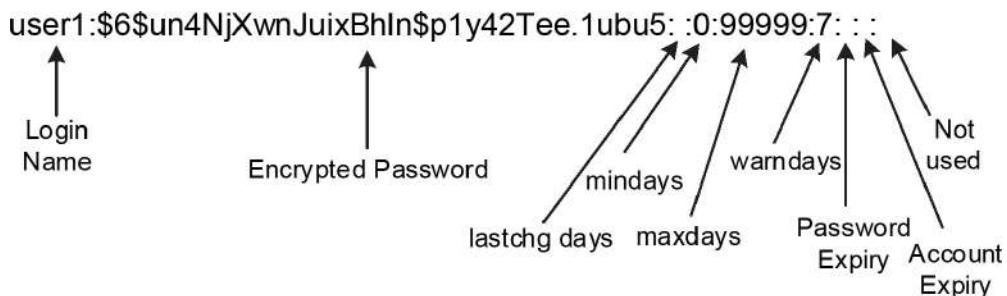


Figure 5-2 The shadow File

Here is a description for each field:

Field 1 (Login Name): Contains the login name as appears in the *passwd* file.

Field 2 (Encrypted Password): Consists of a hashed password. A single exclamation mark (!) at the beginning of this field implies that the user account is locked. If this field is empty, the user will have password-less entry into the system.

Field 3 (Last Change): Sets the number of days (lastchg) since the UNIX epoch, a.k.a. UNIX time (January 01, 1970 00:00:00 UTC) when the password was last modified. An empty field represents the passiveness of

password aging features, and a 0 forces the user to change their password upon next login.

Field 4 (Minimum): Expresses the minimum number of days (mindays) that must elapse before the user is allowed to change their password. This field can be altered using the *chage* command with the -m option or the *passwd* command with the -n option. A 0 or null in this field disables this feature.

Field 5 (Maximum): Defines the maximum number of days (maxdays) of password validity before the user password expires and it must be changed. This field may be altered using the *chage* command with the -M option or the *passwd* command with the -x option. A null value here disables this feature along with other features such as the maximum password age, warning alerts, and the user inactivity period.

Field 6 (Warning): Denotes the number of days (warndays) for which the user gets warnings for changing their password before it actually expires. This field may be altered using the *chage* command with the -W option or the *passwd* command with the -w option. A 0 or null in this field disables this feature.

Field 7 (Password Expiry): Contains the maximum allowable number of days for the user to be able to log in with the expired password. This period is referred to as the inactivity period. This field may be altered using the *chage* command with the -I option or the *passwd* command with the -i option. An empty field disables this feature.

Field 8 (Account Expiry): Determines the number of days since the UNIX time when the user account will expire and no longer be available. This field may be altered using the *chage* command with the -E option. An empty field disables this feature.

Field 9 (Reserved): Reserved for future use.

A *head* and *tail* from the *shadow* file for the first and last three lines is shown below:

```
[root@server1 ~]# head -3 /etc/shadow ; tail -3 /etc/shadow
root:$6$wBS453YXp8510/Gf$xjPHvemqxqXwFgAc3nMyb4tMKB4FQscvHWKvD5boUwaqz3rVoPOZJHBN
PHxAo8DBuz2H80sYFKqhhsF2XoSh.PO::0:99999:7:::
bin:*:17784:0:99999:7:::
daemon:*:17784:0:99999:7:::
user1:$6$q1SIof2tO2DXzTaH$p59YmX12xI8QozHoWWdKNFZxyt..2YMVXCPObTf70k5ufU.9qxTXAD
EfR1PcXcMVD1z9GttbS.peyUreCWXOb1::0:99999:7:::
user100:!:18146:0:99999:7:::
user200:!:18146:0:99999:7:::
```

The output indicates the *root* user with UID 0 followed by two service accounts (*bin* and *daemon*). The last three lines display the three user accounts that were created earlier as part of some of the exercises. Notice that login names are used as a common key between the *shadow* and *passwd* files.

Let's verify the permissions and ownership on the *shadow* file:

```
[root@server1 ~]# ls -l /etc/shadow  
----- 1 root root 1367 Sep 11 11:49 /etc/shadow
```

The access permissions on the file are 000 (no permissions at all), and it is owned by the *root* user. There is a special mechanism in place that is employed in the background to update this file when a user account is added, modified, deleted, or the password changed. This will be discussed in [Chapter 21](#) “Security Enhanced Linux”.

The group File

The *group* file is a simple plaintext file and contains critical group information. Each row in the file stores information for one group entry. Every user on the system must be a member of at least one group, which is referred to as the *User Private Group* (UPG). By default, a group name matches the username it is associated with. Additional groups may be set up, and users with common file access requirements can be added to them. There are four colon-separated fields per line entry. A sample row from the file is exhibited in [Figure 5-3](#).

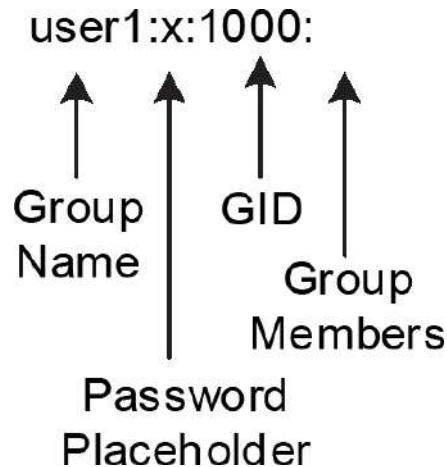


Figure 5-3 The group File

Here is a description for each field:

Field 1 (Group Name): Holds a group name that must begin with a letter. Group names with up to 255 characters, including the underscore (`_`) and hyphen (`-`) characters, are supported. It is not recommended to include special characters and uppercase letters in group names.

Field 2 (Encrypted Password): Can be empty or contain an “x” (points to the */etc/gshadow* file for the actual password), or a hashed group-level

password. You can set a password on a group if you want non-members to be able to change their group identity temporarily using the *newgrp* command. The non-members must enter the correct password in order to do so.

Field 3 (GID): Holds a GID, which is also placed in the GID field of the *passwd* file. By default, groups are created with GIDs starting at 1000 and with the same name as the username. The system allows several users to belong to a single group; it also allows a single user to be a member of multiple groups at the same time.

Field 4 (Group Members): Lists the membership for the group. Note that a user's primary group is always defined in the GID field of the *passwd* file.

A *head* and *tail* from the *group* file for the first and last three lines is shown below:

```
[root@server1 ~]# head -3 /etc/group ; tail -3 /etc/group
root:x:0:
bin:x:1:
daemon:x:2:
sgrp:x:9999:user100,user200
user100:x:1002:
user200:x:1003:
```

The output discloses the *root* user with GID 0 followed by two group service accounts (*bin* and *daemon*). The last three lines showcase the three groups that were created earlier as part of some of the exercises.

Let's verify the permissions and ownership on the *group* file:

```
[root@server1 ~]# ls -l /etc/group
-rw-r--r--. 1 root root 1010 Sep 12 07:48 /etc/group
```

The access permissions on the file are 644 (world-readable and owner-writable), and it is owned by the *root* user.

The *gshadow* File

The shadow password implementation also provides an added layer of protection at the group level. With this mechanism in place, the group passwords are hashed and stored in a more secure file */etc/gshadow*. Unlike the *group* file, which is world-readable and owner-writable, the *gshadow* file has no access permissions at all. This is done to safeguard the file's content.

The *gshadow* file stores hashed group-level passwords. Each row in the file corresponds to one entry in the *group* file. There are four colon-separated fields per line entry. A sample row from this file is exhibited in [Figure 5-4](#).

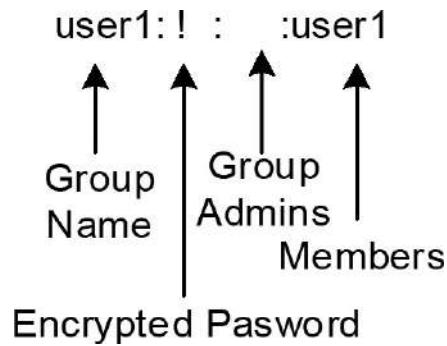


Figure 5-4 The gshadow File

Here is a description for each field:

Field 1 (Group Name): Consists of a group name as appeared in the *group* file.

Field 2 (Encrypted Password): Can contain a hashed password, which may be set with the *gpasswd* command for non-group members to access the group temporarily using the *newgrp* command. A single exclamation mark (!) or a null value in this field allows group members password-less access and restricts non-members from switching into this group.

Field 3 (Group Administrators): Lists usernames of group administrators that are authorized to add or remove members with the *gpasswd* command.

Field 4 (Members): Holds a comma-separated list of members.



The *gpasswd* command is used to add group administrators, add or delete group members, assign or revoke a group-level password, and disable the ability of the *newgrp* command to access a group. This command picks up the default values from the [*/etc/login.defs*](#) file. Additional discussion on this command is beyond the scope; however, you can view the manual pages of this command for details on its usage.

A *head* and *tail* from the *gshadow* file for the first and last three lines is shown below:

```
[root@server1 ~]# head -3 /etc/gshadow ; tail -3 /etc/gshadow
root:::
bin:::
daemon:::
sgrp:!:user100,user200
user100:!:
user200:!:
```

The output indicates the *root* user with GID 0 followed by two group service accounts (*bin* and *daemon*). The last three lines exhibit the three groups that

were created earlier as part of some of the exercises. Notice that group names are used as a common key between the *gshadow* and *group* files.

Let's verify the permissions and ownership on the *gshadow* file:

```
[root@server1 ~]# ls -l /etc/gshadow
----- 1 root root 811 Sep 12 10:54 /etc/gshadow
```

The access permissions on the file are 000 (no permissions at all) and it is owned by the *root* user. There is a special mechanism in place that is employed in the background to update this file when a group account is added, modified, deleted, or the group password changed. This will be discussed further in [Chapter 21](#) "Security Enhanced Linux".

The useradd and login.defs Configuration Files

The *useradd* command (discussed in the next section) picks up the default values from the */etc/default/useradd* and [*/etc/login.defs*](#) files for any options that are not specified at the command line when executing it. Moreover, the *login.defs* file is also consulted by the *usermod*, *userdel*, *chage*, and *passwd* commands (also discussed in the next section) as needed. Both files store several defaults including those that affect the password length and password lifecycle. You can use the *cat* or *less* command to view the *useradd* file content or display the settings with the *useradd* command as follows:

```
[root@server1 ~]# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

There are a multitude of defaults defined with the directives. These include the starting GID (GROUP) provided the USERGROUPS_ENAB directive in the *login.defs* file is set to no, home directory location (HOME), number of inactivity days between password expiry and permanent account disablement (INACTIVE), account expiry date (EXPIRE), login shell (SHELL), skeleton directory location to copy user initialization files from (SKEL), and whether to create mail spool directory (CREATE_MAIL_SPOOL). You will find a description for some of these in the next section.

The other file *login.defs* comprises of additional directives that set several defaults. User and group management commands consult this file to obtain

information that is not supplied at the command line. A *grep* on the file with uncommented and non-empty lines is shown below:

```
[root@server1 ~]# grep -v ^# /etc/login.defs | grep -v ^$  
MAIL_DIR          /var/spool/mail  
PASS_MAX_DAYS    99999  
PASS_MIN_DAYS    0  
PASS_MIN_LEN      5  
PASS_WARN_AGE     7  
UID_MIN           1000  
UID_MAX           60000  
SYS_UID_MIN       201  
SYS_UID_MAX       999  
GID_MIN           1000  
GID_MAX           60000  
SYS_GID_MIN       201  
SYS_GID_MAX       999  
CREATE_HOME        yes  
UMASK             077  
USERGROUPS_ENAB   yes  
ENCRYPT_METHOD    SHA512
```

These directives are elaborated in [Table 5-1](#).

Option	Description
MAIL_DIR	Specifies the mail directory location
PASS_MAX_DAYS, PASS_MIN_DAYS, PASS_MIN_LEN, and PASS_WARN_AGE	Define password aging attributes. See Chapter 10 “Advanced User Management” for details.
UID_MIN, UID_MAX, GID_MIN, and GID_MAX	Identify the ranges of UIDs and GIDs to be allocated to new users and groups
SYS_UID_MIN, SYS_UID_MAX, SYS_GID_MIN, and SYS_GID_MAX	Identify the ranges of UIDs and GIDs to be allocated to new service users and groups
CREATE_HOME	Defines whether to create a home directory
UMASK	Defines the permissions to be set on the user's home directory at creation based on this umask value
USERGROUPS_ENAB	Defines whether to delete a user's group (at the time of user deletion) if it contains no more members
ENCRYPT_METHOD	Specifies the encryption method for user passwords

Table 5-1 `login.defs` File Directives

There are many more directives available that can be added to this file for more control and flexibility. Check the manual pages of the file for details.

User Account Management

Managing user accounts involves creating, assigning passwords to, modifying, and deleting them. RHEL provides a set of tools for performing these operations. These tools are `useradd` to add a new user to the system, `usermod` to modify the attributes of an existing user, and `userdel` to remove a user from the system. In addition, the `passwd` command is available to set or modify a user's password.

The `useradd`, `usermod`, and `userdel` Commands

This set of commands is used to add, modify, and delete a user account from the system. The *useradd* command adds entries to the four user authentication files for each account added to the system. It creates a home directory for the user and copies the default user startup files from the *skeleton* directory */etc/skel* into the user's home directory. It can also be used to update the default settings that are used at the time of new user creation for unspecified settings. The *useradd* command supports a variety of flags; [Table 5-2](#) lists some common options in both short and long versions.

Option	Description
-b (--base-dir)	Defines the absolute path to the base directory for placing user home directories. The default is /home.
-c (--comment)	Describes useful information about the user.
-d (--home-dir)	Defines the absolute path to the user home directory.
-D (--defaults)	Displays the default settings from the /etc/default/useradd file and modifies them.
-e (--expiredate)	Specifies a date on which a user account is automatically disabled. The format for the date specification is YYYY-MM-DD.
-f (--inactive)	Denotes maximum days of inactivity between password expiry and permanent account disablement.
-g (--gid)	Specifies the primary GID. Without this option, a group account matching the username is created with the GID matching the UID.
-G (--groups)	Specifies the membership to supplementary groups.
-k (--skel)	Specifies the location of the skeleton directory (default is /etc/skel), which stores default user files. These files are copied to the user's home directory at the time of account creation. Three hidden bash shell files—.bash_profile, .bashrc, and .bash_logout—are available in this directory by default. You can customize these files or add your own to be used for accounts created thereafter.
-m (--create-home)	Creates a home directory if it does not already exist.
-o (--non-unique)	Creates a user account sharing the UID of an existing user. When two users share a UID, they have identical rights on each other's files. This should be done in specific situations.
-r (--system)	Creates a service account with a UID below 500 and a never-expiring password.
-s (--shell)	Defines the absolute path to the shell file. The default is /bin/bash.
-u (--uid)	Indicates a unique UID. Without this option, the next available UID from the /etc/passwd file is used.
login	Specifies a login name to be assigned to the account.

Table 5-2 useradd Command Options

You can modify the attributes of a user account with the *usermod* command. The syntax of this command is very similar to that of the *useradd*'s, with most switches identical. [Table 5-3](#) describes the options that are specific to *usermod* only, and shows them in both short and long versions. There are two additional flags of interest that are discussed in [Chapter 06](#) “Advanced User Management”.

Option	Description
-a (--append)	Adds a user to one or more supplementary groups
-l (--login)	Specifies a new login name
-m (--move-home)	Creates a home directory and moves the contents over from the old location

Table 5-3 usermod Command Options

The *userdel* command is straightforward. It removes entries for the specified user from the authentication files, and deletes the user’s home directory if the -r flag is also specified. The -f option may be used to force the removal even if the user is still logged in.

Exercise 5-1: Create a User Account with Default Attributes

This exercise should be done on *server1* as *root*.

In this exercise, you will create a user account *user2* using the defaults defined in the *useradd* and *login.defs* files. You will assign this user a password and show the new line entries from all four authentication files.

1. Create *user2* with all the default values:

```
[root@server1 ~]# useradd user2
```

2. Assign this user a password and enter it twice when prompted:

```
[root@server1 ~]# passwd user2
Changing password for user user2.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
```

3. *grep* for *user2*: on the authentication files to examine what the *useradd* command has added:

```
[root@server1 ~]# cd /etc ; grep user2: passwd shadow group gshadow
passwd:user2:x:1004:1004::/home/user2:/bin/bash
shadow:user2:$6$HjGW61CCZRJWifCP$HTJuDdHrpM79.KAbD7bM3DVVCV.Ij9sc6Sgv3MNHqk/azmu
eXniPEMQVDANGEaO6J6FbhvIiNShfgOHff9q9G0:18152:0:99999:7:::
group:user2:x:1004:
gshadow:user2:!:!
```

The command used the next available UID (1004) and GID (1004), and the default settings for the home directory (`/home/user2`) and shell file (`/bin/bash`).

4. Test this new account by logging in as `user2` and then run the `id` and `groups` commands to verify the UID, GID, and group membership information:

```
[root@server1 etc]# su - user2
[user2@server1 ~]$ id
uid=1004(user2) gid=1004(user2) groups=1004(user2) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[user2@server1 ~]$ groups
user2
```

The above outputs confirmed the UID and username, GID and group name, and primary group name.

Exercise 5-2: Create a User Account with Custom Values

This exercise should be done on `server1` as `root`.

In this exercise, you will create an account `user3` with UID 1010, home directory `/usr/user3a`, and shell `/bin/sh`. You will assign this user a password and exhibit the new line entries from all four authentication files.

1. Create `user3` with UID 1010 (-u), home directory `/usr/user3a` (-d), and shell `/bin/sh` (-s):

```
[root@server1 ~]# useradd -u 1010 -d /usr/user3a -s /bin/sh user3
```

2. Assign user1234 as password (passwords assigned in the following way is not recommended; however, it is okay in a lab environment):

```
[root@server1 ~]# echo user1234 | passwd --stdin user3
Changing password for user user3.
passwd: all authentication tokens updated successfully.
```

3. `grep` for `user3`: on the four authentication files to see what was added for this user:

```
[root@server1 ~]# cd /etc ; grep user3: passwd shadow group gshadow
passwd:user3:x:1010:1010::/usr/user3a:/bin/sh
shadow:user3:$6$R3jWTWYymYIeF55h$r6qkL8Tk3vhW8jw1BdvOFuJRJGRhFUT2P5pUpdIFkyR40Wf
cNATNv9RcjYKyuexB1GWyC@/Ivf5uiwHQ/ujPP/:18152:0:99999:7:::
group:user3:x:1010:
gshadow:user3:!:!
```

4. Test this account by switching to or logging in as *user3* and entering user1234 as the password. Run the *id* and *groups* commands for further verification.

Exercise 5-3: Modify and Delete a User Account

This exercise should be done on *server1* as *root*.

In this exercise, you will modify certain attributes for *user2* and then delete it. You will change the login name to *user2new*, UID to 2000, home directory to */home/user2new*, and login shell to */sbin/nologin*. You will display the line entry for *user2new* from the *passwd* file for validation. Finally, you will remove this user and confirm the deletion.

1. Modify the login name for *user2* to *user2new* (-l), UID to 2000 (-u), home directory to */home/user2new* (-m and -d), and login shell to */sbin/nologin* (-s). Notice that the options are specified in a different sequence.

```
[root@server1 ~]# usermod -l user2new -m -d /home/user2new -s /sbin/nologin -u 2000 user2
```

2. Obtain the information for *user2new* from the *passwd* file for confirmation:

```
[root@server1 ~]# grep user2new /etc/passwd
user2new:x:2000:1004::/home/user2new:/sbin/nologin
```

3. Remove *user2new* along with their home and mail spool directories (-r):

```
[root@server1 ~]# userdel -r user2new
```

4. Confirm the user deletion:

```
[root@server1 ~]# grep user2new /etc/passwd
```

Nothing is returned in the output, which means this user does not exist in the *passwd* file any longer. This confirms the removal.

No-Login (Non-Interactive) User Account

The *nologin* shell is a special purpose program that can be employed for user accounts that do not require login access to the system. It is located in the */usr/sbin* (or */sbin*) directory. With this shell assigned, the user is refused with the message, “This account is currently not available.” displayed on the screen. If a custom message is required, you can create a file called *nologin.txt* in the */etc* directory and add the desired text to it. The content of

this file is printed on the screen upon user access denial instead of the default message.

EXAM TIP: If a no-login user is able to log in with their credentials, there is a problem. Use the grep command against the /etc/passwd file to ensure '/sbin/nologin' is there in the shell field for that user.

Typical examples of user accounts that do not require login access are the service accounts such as *ftp*, *apache*, and *sshd*. Let's take a look at the *passwd* file and list such users:

```
[root@server1 ~]# grep nologin /etc/passwd
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
```

This output returns a truncated list of non-interactive user accounts. There are tens of them, and they grow as you install more services to the system.

Exercise 5-4: Create a User Account with No-Login Access

This exercise should be done on *server1* as *root*.

In this exercise, you will create an account *user4* with all the default attributes but with a non-interactive shell. You will assign this user the *nologin* shell to prevent them from signing in. You will display the new line entry from the *passwd* file and test the account.

1. Create *user4* with non-interactive shell file */sbin/nologin*:

```
[root@server1 ~]# useradd -s /sbin/nologin user4
```

2. Assign *user1234* as password:

```
[root@server1 ~]# echo user1234 | passwd --stdin user4
```

3. *grep* for *user4* on the *passwd* file and verify the shell field containing the *nologin* shell:

```
[root@server1 ~]# grep user4 /etc/passwd
user4:x:1011:1011::/home/user4:/sbin/nologin
```

4. Test this account by attempting to log in or switch:

```
[root@server1 ~]# su - user4
This account is currently not available.
```

As mentioned earlier, you may create the */etc/nologin.txt* file and add a custom message to it. The new message will appear on your next login attempt.

Chapter Summary

We started this chapter by discovering various tools that are employed to monitor user login and system reboot activities. This information is stored in several system files and may be used for testing, auditing, or troubleshooting reasons. Most of these tools can be executed by normal users to obtain desired results; however, others may require privileged access of the root user in order to be viewed.

Next, we looked closely at the four local user authentication files: *passwd*, *shadow*, *group*, and *gshadow*. We examined their contents and syntax to comprehend what they store and how. These files are critical to the user login process as well as the functioning of applications and services.

Finally, we explored user management tools and put them into action in exercises to create, modify, and delete user accounts. These tools offer a number of switches to add accounts with default and custom attributes, as well as user accounts that do not require login access.

Review Questions

1. What would the command *useradd -D* do?
2. What does the *lastlog* command do?
3. What would the command *useradd user500* do?
4. The *id* and *groups* commands are useful for listing a user identification. True or False.
5. Name the four local user authentication files.
6. The *passwd* file contains secondary user group information. True or False?
7. What is the first UID assigned to a normal user?
8. Name the three fundamental user account classes in RHEL.
9. Every user in RHEL gets a private group by default. True or False?
10. What does the “x” in the password field in the *passwd* file imply?
11. Name the file that the *who* command consults to display logged-in users.
12. What information does the *lastb* command provide?

13. The *who* command may be used to view logged out users. True or False?
14. What would the *userdel* command do if it is run with the -r option?
15. What is the first GID assigned to a group?
16. What is the name of the default backup file for *shadow*?
17. Which file does the *last* command consult to display reports?
18. UID 999 is reserved for normal users. True or False?
19. Which file is assigned to users to deny them login access?
20. You execute the *lastb* command as a normal user. The output reports an error. What do you need to do to run this command successfully?

Answers to Review Questions

1. This command displays the defaults settings used at the time of user creation or modification.
2. The *lastlog* command provides information about recent user logins.
3. The command provided will add *user500* with all predefined default values.
4. False. Only the *id* command is used for this purpose.
5. The *passwd*, *shadow*, *group*, and *gshadow* files.
6. False. The *passwd* file contains primary user group information.
7. The first UID assigned to a normal user is 1000.
8. The three fundamental user account categories are root, normal, and system.
9. True.
10. The “x” in the password field implies that the hashed password is stored in the *shadow* file.
11. The *who* command consults the */run/utmp* file to list logged-in users.
12. The *lastb* command reports the history of unsuccessful user login attempts.
13. False. The *who* command shows currently logged-in users only.
14. The *userdel* command with -r will delete the specified user along with their home directory.
15. The first GID assigned to a normal user is 1000.
16. The name of the default backup file for *shadow* is *shadow-*.
17. The *last* command consults the */var/log/wtmp* file to display reports.
18. False. UID 999 is reserved for system accounts.
19. The */sbin/nologin* file is assigned to users to deny them login.
20. You need to run the *lastb* command as *root* or with the *root* privilege.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 5-1: Check User Login Attempts

As *root* on *server1*, execute the *last*, *lastb*, and *lastlog* commands, and observe the outputs. Check which users recently logged in and out successfully (*last*) and unsuccessfully (*lastb*). List the timestamps when the system was last rebooted (*last*). Check the last login status for each user (*lastlog*). Use the *vim* editor to record your results. (Hint: User Login Activity and Information).

Lab 5-2: Verify User and Group Identity

As *user1* on *server1*, run the **who** and **w** commands one at a time, and compare the outputs. Execute the **id** and **groups** commands, and compare the outcomes. Examine the extra information that the *id* command shows, but not the *groups* command. (Hint: User Login Activity and Information).

Lab 5-3: Create Users

As *root* on *server1*, create user account *user4100* with UID 4100 and home directory under */usr*. Create another user account *user4200* with default attributes. Assign both users a password. View the contents of the *passwd*, *shadow*, *group*, and *gshadow* files, and observe what has been added for the two new users. (Hint: Local User Authentication Files, and User Account Management).

Lab 5-4: Create User with Non-Interactive Shell

As *root* on *server1*, create user account *user4300* with the disability of logging in. Assign this user a password. Try to log on with this user and see is displayed on the screen. View the content of the *passwd* file, and see what is there that prevents this user from logging in. (Hint: Local User Authentication Files, and User Account Management).

Chapter 06

Advanced User Management

This chapter describes the following major topics:

- Configure password aging attributes on local user accounts
- Lock and unlock user account
- Understand, create, modify, and delete local groups and group memberships
- Switch into another user account
- Configure who can execute which privileged commands
- Identify and manage file owners and owning groups

RHCSA Objectives:

05. Log in and switch users in multi-user targets
52. Change passwords and adjust password aging for local user accounts (only the second part of this objective “adjust password aging for local user accounts” is covered in this chapter; the first part is in [Chapter 05](#))
53. Create, delete, and modify local groups and group memberships
54. Configure superuser access

Password

aging attributes may be set on user accounts for increased control on their logins and passwords. This can be done for an individual user or applied to all users. Password aging information for users is stored in one of the authentication files that was discussed at length in the previous chapter. Individual user accounts may be prevented from logging in to the system by locking their access for a period of time or permanently. This lock may be lifted when required. Setting password aging and locking/unlocking accounts are administrative functions and must be performed by a user with elevated privileges of the root user.

Users are apportioned membership to a single group at the time of their addition to the system. Later, they may be assigned membership to additional groups. Members of the same group possess the same access rights on files and directories. Other users and members of other groups may optionally be given access to those files. Group membership information is stored in user and group authentication files that were examined in the previous chapter as well.

Users may switch into other user accounts, including the root user, provided they know the target user's password. Normal users may be allowed access to privileged commands by defining them appropriately in a configuration file. Each file that exists on the system regardless of its type has an owning user and an owning group. Similarly, every file that a user creates is in the ownership of that user. The ownership may be changed and given to another user by a super user.

Password Aging and its Management

As mentioned, password aging is a secure mechanism to control user passwords in Linux. The key advantages include setting restrictions on password expiry, account disablement, locking and unlocking users, and password change frequency. These controls are applied to all user accounts at the time of their creation and can be set explicitly on a per-user basis later. You can even choose to deactivate it completely for an individual user.

Password aging information is stored in the `/etc/shadow` file (fields 4 to 8) and its default policies in the `/etc/login.defs` configuration file. These files were thoroughly examined in [Chapter 05](#) "Basic User Management". In this section, we explore the aging management tools—`chage` and `passwd`—and look at how to employ them to apply password controls on user accounts, `user100` and `user200`. Alongside `chage` and `passwd`, the `usermod` command can also

be used to implement two aging attributes (user expiry and password expiry); however, this section focuses on this command's ability to lock and unlock user accounts.

The `chage` Command

The `chage` command is used to set or alter password aging parameters on a user account. This command changes various fields in the `shadow` file depending on which option(s) you pass to it. There are plenty of switches available with the command in both short and long formats. [Table 6-1](#) describes most of them.

Option	Description
-d (--lastday)	Specifies an explicit date in the YYYY-MM-DD format, or the number of days since the UNIX epoch when the password was last modified. With -d, the user is forced to change the password at next login. It corresponds to field 3 in the shadow file.
-E (--expiredate)	Sets an explicit date in the YYYY-MM-DD format or the number of days since the UNIX epoch on which the user account is deactivated. This feature can be disabled with -E -1. It corresponds to the eighth field in the shadow file.
-I (--inactive)	Defines the number of days of inactivity after password expiry and before the account is locked. The user may be able to log in during this period with their expired password. This feature can be disabled with -I -1. It corresponds to field 7 in the shadow file.
-l	Lists password aging attributes set on a user account.
-m (--mindays)	Indicates the minimum number of days that must elapse before the password can be changed. A value of 0 allows the user to change their password at any time. It corresponds to field 4 in the shadow file.
-M (--maxdays)	Denotes the maximum number of days of password validity before the user password expires and must be changed. This feature can be disabled with -M -1. It corresponds to field 5 in the shadow file.
-W (--warndays)	Designates the number of days for which the user gets alerts to change their password before it expires. It corresponds to field 6 in the shadow file.

Table 6-1 chage Command Options

You will use most of these flags in [Exercise 6-1](#) as well as later in this chapter.

Exercise 6-1: Set and Confirm Password Aging with chage

This exercise should be done on `server1` as `root`.

In this exercise, you will configure password aging for `user100` using the `chage` command. You will set mindays to 7, maxdays to 28, and warndays to

5, and verify the new settings. You will then rerun the command and set account expiry to January 31, 2020. You will complete the exercise with another confirmation.

1. Set password aging parameters for *user100* to mindays (-m) 7, maxdays (-M) 28, and warndays (-W) 5:

```
[root@server1 ~]# chage -m 7 -M 28 -W 5 user100
```

2. Confirm the new settings:

```
[root@server1 ~]# chage -l user100
Last password change : Sep 13, 2019
Password expires      : Oct 11, 2019
Password inactive     : never
Account expires        : never
Minimum number of days between password change : 7
Maximum number of days between password change : 28
Number of days of warning before password expires : 5
```

The bottom three rows in the output confirm the new settings. The current password expiry (October 11, 2019) reflects the 28-day duration from September 13, 2019.

3. Set the account expiry to January 31, 2020:

```
[root@server1 ~]# chage -E 2020-01-31 user100
```

4. Verify the new account expiry setting:

```
[root@server1 ~]# chage -l user100
Last password change : Sep 13, 2019
Password expires      : Oct 11, 2019
Password inactive     : never
Account expires        : Jan 31, 2020
Minimum number of days between password change : 7
Maximum number of days between password change : 28
Number of days of warning before password expires : 5
```

The middle row reflects the new account expiry for *user100*. Similarly, you can use the -d and -l options with the *chage* command as required.

The *passwd* Command

The common use of the *passwd* command is to set or modify a user's password; however, you can also use this command to modify the password aging attributes and lock or unlock their account. [Table 6-2](#) lists some key options in both short and long formats.

Option	Description
-d (--delete)	Deletes a user password without expiring the account.
-e (--expire)	Forces a user to change their password upon logon.
-i (--inactive)	Defines the number of days of inactivity after password expiry and before the account is locked. It corresponds to field 7 in the shadow file.
-l (--lock)	Locks a user account.
-n (--minimum)	Specifies the number of days that must elapse before the password can be changed. It corresponds to field 4 in the shadow file.
-S (--status)	Displays the status information for a user.
-u (--unlock)	Unlocks a locked user account.
-w (--warning)	Designates the number of days for which the user gets alerts to change their password before it expires. It corresponds to field 6 in the shadow file.
-x (--maximum)	Denotes the maximum number of days of password validity before the user password expires and must be changed. It corresponds to field 5 in the shadow file.

Table 6-2 passwd Command Options

Most of these flags will be used in the next few exercises.

Exercise 6-2: Set and Confirm Password Aging with passwd

This exercise should be done on *server1* as *root*.

In this exercise, you will configure password aging for *user200* using the *passwd* command. You will set mindays to 10, maxdays to 90, and warndays to 14, and verify the new settings.

Next, you will set the number of inactivity days to 5 and ensure that the user is forced to change their password upon next login. You will complete the exercise with another verification.

1. Set password aging attributes for *user200* to mindays (-n) 10, maxdays (-x) 90, and warndays (-w) 14:

```
[root@server1 ~]# passwd -n 10 -x 90 -w 14 user200
```

2. Confirm the new settings:

```
[root@server1 ~]# passwd -S user200
user200 PS 2019-09-14 10 90 14 -1 (Password set, SHA512 crypt.)
```

The output confirms the three new settings (10, 90, and 14).

3. Set the number of inactivity days to 5:

```
[root@server1 ~]# passwd -i5 user200
```

4. Confirm the new setting:

```
[root@server1 ~]# passwd -S user200
user200 PS 2019-09-14 10 90 14 5 (Password set, SHA512 crypt.)
```

The output verifies the new setting (5).

5. Ensure that the user is forced to change their password at next login:

```
[root@server1 ~]# passwd -e user200
```

6. Display the new setting for confirmation:

```
[root@server1 ~]# passwd -S user200
user200 PS 1969-12-31 10 90 14 5 (Password set, SHA512 crypt.)
```

The output shows a date prior to the UNIX time. This user will be prompted to change their password when they attempt to log in the next time.

The usermod Command

The common use of the *usermod* command is to modify a user's attribute, but it can also lock or unlock their account. [Table 6-3](#) explains the two options in both short and long formats.

Option	Description
-L (--lock)	Locks a user account by placing a single exclamation mark (!) at the beginning of the password field before the hashed password string.
-U (--unlock)	Unlocks a user's account by removing the exclamation mark (!) from the beginning of the password field.

Table 6-3 usermod Command Options for User Lock/Unlock

Let's apply these options in the next exercise.

Exercise 6-3: Lock and Unlock a User Account with usermod and passwd

This exercise should be done on *server1* as *root*.

In this exercise, you will disable the ability of *user200* to log in using the *usermod* and *passwd* commands. You will verify the change and then reverse it.

1. Obtain the current password information for *user200* from the *shadow* file:

```
[root@server1 ~]# grep user200 /etc/shadow
user200:$6$gnvX93VT0eWU94s.$6SPEDtB31g1IULSxbbEAKPMw8H8PXnyston.3n1F/kc15Aw8Kt2e
eTo1HiOoouq/OrNBSa9BowirssW3ZzVmp1:0:10:90:14:5::
```

An unlocked user account never has its password field begin with an exclamation mark (!). The above output is indicative of the fact that the account is currently not locked.

2. Lock the account for *user200*:

```
[root@server1 ~]# usermod -L user200          or
[root@server1 ~]# passwd -l user200
```

3. Confirm the change:

```
[root@server1 ~]# grep user200 /etc/shadow
user200:!$6$gnvX93VT0eWU94s.$6SPEDtB31g1IULSxbbEAKPMw8H8PXnyston.3n1F/kc15Aw8Kt2
eTo1HiOoouq/OrNBSa9BowirssW3ZzVmp1:0:10:90:14:5::
```

Notice that an exclamation mark (!) is prepended to the encrypted password, which indicates a locked account.

4. Unlock the account with either of the following:

```
[root@server1 ~]# usermod -U user200          or
[root@server1 ~]# passwd -u user200
```

Verify the reversal with the *grep* command as demonstrated in step 3. You can also try to log in as *user200* for an additional validation.

Linux Groups and their Management

Linux groups are collections of one or more users with identical permission requirements on files and directories. They allow group members to collaborate on files of common interest.

Group information is stored in the `/etc/group` file and the default policies in the `/etc/login.defs` configuration file. Furthermore, the `/etc/gshadow` file stores group administrator information and group-level passwords. These files were thoroughly examined in [Chapter 05](#) “Basic User Management”.

This section explores group management tools—`groupadd`, `groupmod`, and `groupdel`—and looks at how to utilize them to create, alter, and erase groups. Additional group administration operations, such as adding and deleting group administrators, and setting and revoking group-level passwords, are beyond the scope.

The `groupadd`, `groupmod`, and `groupdel` Commands

This set of management commands is used to add, modify, and delete a group from the system. The `groupadd` command adds entries to the `group` and `gshadow` files for each group added to the system. [Table 6-4](#) lists and explains some of the `groupadd` command options in both short and long formats.

Option	Description
<code>-g (--gid)</code>	Specifies the GID to be assigned to the group.
<code>-o (--non-unique)</code>	Creates a group with a matching GID of an existing group. When two groups have an identical GID, members of both groups get identical rights to each other's files. This should only be done in special situations.
<code>-r</code>	Creates a system group with a GID below 1000.
<code>groupname</code>	Specifies a group name.

Table 6-4 `groupadd` Command Options

The `groupadd` command picks up the default values from the `login.defs` file.

You can modify the attributes of a group with the `groupmod` command. The syntax of this command is very similar to the `groupadd` with most options identical. The only flag that is additional with this command is `-n`, which can change the name of an existing group.

The `groupdel` command is straightforward. It removes entries for the specified group from both `group` and `gshadow` files.

Exercise 6-4: Create a Group and Add Members

This exercise should be done on *server1* as *root*.

In this exercise, you will create a group called *linuxadm* with GID 5000 and another group called *dba* sharing the GID 5000. You will add *user1* as a secondary member to group *linuxadm*.

1. Create the group *linuxadm* with GID 5000:

```
[root@server1 ~]# groupadd -g 5000 linuxadm
```

2. Create a group called *dba* with the same GID as that of group *linuxadm*:

```
[root@server1 ~]# groupadd -o -g 5000 dba
```

3. Confirm the creation of both groups:

```
[root@server1 ~]# grep linuxadm /etc/group  
linuxadm:x:5000:  
[root@server1 ~]# grep dba /etc/group  
dba:x:5000:
```

The GID for both groups is identical.

4. Add *user1* as a secondary member of group *dba* using the *usermod* command. The existing membership for the user must remain intact.

```
[root@server1 ~]# usermod -aG dba user1
```

5. Verify the updated group membership information for *user1* by extracting the relevant entry from the *group* file, and running the *id* and *groups* command for *user1*:

```
[root@server1 ~]# grep dba /etc/group  
dba:x:5000:user1  
[root@server1 ~]#  
[root@server1 ~]# id user1  
uid=1000(user1) gid=1000(user1) groups=1000(user1),5000(dba)  
[root@server1 ~]# groups user1  
user1 : user1 dba
```

The output confirms the primary (*user1*) and secondary (*dba*) group memberships for *user1*.

Exercise 6-5: Modify and Delete a Group Account

This exercise should be done on *server1* as *root*.

In this exercise, you will change the *linuxadm* group name to *sysadm* and the GID to 6000. You will modify the primary group for *user200* to *sysadm*. Finally, you will remove the *sysadm* group and verify the actions.

1. Alter the name of *linuxadm* to *sysadm*:

```
[root@server1 ~]# groupmod -n sysadm linuxadm
```

2. Change the GID of *sysadm* to 6000:

```
[root@server1 ~]# groupmod -g 6000 sysadm
```

3. Confirm the above actions:

```
[root@server1 ~]# grep sysadm /etc/group
sysadm:x:6000:user1,user200
[root@server1 ~]# grep linuxadm /etc/group
[root@server1 ~]# _
```

The outputs confirm both actions. The new group name is *sysadm* and the new GID is 6000. The membership remains the same. Also notice that the *linuxadm* group does not exist anymore. The second command returned nothing for it.

4. Delete the *sysadm* group and confirm:

```
[root@server1 ~]# groupdel sysadm
[root@server1 ~]# grep sysadm /etc/group
[root@server1 ~]#
```

The second command returns nothing, which confirms a successful deletion of the group.

Substituting Users and Doing as Superuser

In real world Linux environments, you'll always sign in as a normal user. Normal users have limited rights on the system. Their profiles are set to allow them to accomplish routine jobs efficiently and securely. This is done to secure the overall system functionality and to prevent unexpected issues. There are times though when a normal user needs to assume the identity of a different user to carry out a task as that user or execute a command successfully that requires elevated privileges. Linux offers two fundamental tools to support users in both situations.

Substituting (or Switching) Users

Even though you can log in to the system directly as *root*, it is not a recommended practice. Instead, log in with a normal user account and then switch to the *root* account if necessary. This is safer and ensures system security and protection. In addition to becoming *root*, you can switch into another user account. In either case, you'll need to know the password for the target user in order for a successful switch. The *su* command has the ability to switch into other user accounts.

To switch from *user1* (assuming you are logged in as *user1*) into *root* without executing the startup scripts (startup scripts are explained later in this chapter) for the target user, run the *su* command and enter the *root* user password when prompted:

```
[user1@server1 ~]$ su
```

Press Ctrl+d key combination to return to the *user1* command prompt. Then rerun the *su* command with the hyphen character (-) specified to ensure that startup scripts for the target user are also executed to provide an environment similar to that of a real login:

```
[user1@server1 ~]$ su -
```

To switch into a different user account, such as *user100*, specify the name of the target user with the command. You must enter the password of the target user when prompted.

```
[user1@server1 ~]$ su - user100
```

RHEL offers two tools—*whoami* (*who am i*) and *logname* (*login name*)—that show a user's current identity (after *su*'ing into the target user) and the identity of the user who originally logged in. Let's see what they report after switching into *user100*:

```
[user1@server1 ~]$ su - user100
Password:
[user100@server1 ~]$ whoami
user100
[user100@server1 ~]$ logname
user1
```

The *whoami* command returns the effective (current) username (*user100*), and the *logname* command reports the user's real (original) username (*user1*).

To issue a command as a different user without switching into that user, use the `-c` option with `su`. For example, the `firewall-cmd` command with the `--list-services` option requires superuser privileges. `user1` can use `su` as follows and execute this privileged command to obtain desired results:

```
[user1@server1 ~]$ su -c 'firewall-cmd --list-services'
```

The `root` user can switch into any user account that exists on the system without being prompted for that user's password.

Although the `su` command does provide the flexibility and convenience, switching into the `root` account to execute privileged actions is not recommended. There is a preferred method available in Linux that should be used instead. The following section sheds light on it.

Doing as Superuser (or Doing as Substitute User)

On production Linux servers, there are necessary steps to ensure that users have the ability to carry out their assigned job without hassle. In most cases, this requires privileged access to certain tools and functions, which the `root` user is normally allowed to run.

RHEL provides normal users the ability to run a set of privileged commands or to access non-owning files without the knowledge of the `root` password. This allows the flexibility of assigning a specific command or a set of commands to an individual user or a group of users based on their needs. These users can then precede one of those commands with a utility called `sudo` (*superuser do*, a.k.a. *substitute user do*) at the time of executing that command. The users are prompted to enter their own password, and if correct, the command is executed successfully for them. The `sudo` utility is designed to provide protected access to administrative functions as defined in the `/etc/sudoers` file or files in the drop-in directory `/etc/sudoers.d`. It can also be used to allow a user or a group of users to run scripts and applications owned by a different user.

Any normal user that requires privileged access to administrative commands or non-owning files is defined in the `sudoers` file. This file may be edited with a command called `visudo`, which creates a copy of the file as `sudoers.tmp` and applies the changes there. After the `visudo` session is over, the updated file overwrites the original `sudoers` file and `sudoers.tmp` is deleted. This is done to prevent multiple users editing this file simultaneously.

The syntax for user and group entries in the file is similar to the following example entries for user *user1* and group *dba*:

```
user1      ALL=(ALL)    ALL
%dba      ALL=(ALL)    ALL
```

These entries may be added to the beginning of the file, and they are intended to provide full access to every administrative function to *user1* and members of the *dba* group (group is prefixed by the percentage sign (%)). In other words, *user1* and *dba* group members will have full *root* user authority on the system.

When *user1* or a *dba* group member attempts to access a privileged function, they will be required to enter their own password. For instance:

```
[user1@server1 ~] $ sudo head /etc/sudoers
[sudo] password for user1:
user1  ALL=(ALL)    ALL
%dba  ALL=(ALL)    ALL
## Sudoers allows particular users to run various commands as
## the root user, without needing the root password.
##
## Examples are provided at the bottom of the file for collections
## of related commands, which can then be delegated out to particular
## users or groups.
##
## This file must be edited with the 'visudo' command.
```

If you want *user1* and *dba* group members not to be prompted for their passwords, modify the entries in the *sudoers* file to look like:

```
user1      ALL=(ALL)    NOPASSWD:ALL
%dba      ALL=(ALL)    NOPASSWD:ALL
```

Rather than allowing them full access to the system, you can restrict their access to the functions that they need access to. For example, to limit their access to a single command */usr/bin/cat*, modify the directives as follows:

```
user1          ALL=/usr/bin/cat
%dba          ALL=/usr/bin/cat
```

These users should now be able to use the *cat* command to view the content of the */etc/sudoers* or any other file that requires the *root* privilege. Try **cat /etc/sudoers** as *user1* and then again as **sudo cat /etc/sudoers**. You will see the difference.

Configuring sudo to work the way it has just been explained may result in a cluttered *sudoers* file with too many entries to fulfill diversified needs of users

and groups. A preferred method is to use predefined aliases—`User_Alias` and `Cmnd_Alias`—to configure groups of users and commands, and assign access as required. For instance, you can define a `Cmnd_Alias` called `PKGCMD` containing `yum` and `rpm` package management commands, and a `User_Alias` called `PKGADM` for `user1`, `user100`, and `user200`. These users may or may not belong to the same Linux group.

Next, assign `PKGCMD` to `PKGADM`. This way one rule is set that allows a group of users access to a group of commands. You can add or remove commands and users anytime as needed, and the change will take effect right away. Here is what this configuration will look like:

<code>Cmnd_Alias</code>	<code>PKGCMD = /usr/bin/yum, /usr/bin/rpm</code>
<code>User_Alias</code>	<code>PKGADM = user1, user100, user200</code>
<code>PKGADM</code>	<code>ALL = PKGCMD</code>

Append the above to the bottom of the `sudoers` file and then run the `yum` or `rpm` command preceded by `sudo` as one of the users listed. You will be able to perform software management tasks just like the `root` user.

The `sudo` command logs successful authentication and command data to the `/var/log/secure` file under the name of the actual user executing the command (and not `root`).

The `sudoers` file contains several examples with a brief explanation. It is a good idea to look at those examples for additional understanding.

Now that the normal user `user1` is added to the `sudoers` configuration file, you will be using this user account with `sudo` where appropriate in the examples and exercises in the remainder of the book.

Owning User and Owning Group

In Linux, every file and directory has an owner. By default, the creator assumes the ownership, but this may be altered and allocated to a different user if required.

Similarly, every user is a member of one or more groups. A group is a collection of users with common permission requirements. By default, the owner's group is assigned to a file or directory.

Let's create a file `file1` as `user1` in their home directory and exhibit the file's long listing:

```
[user1@server1 ~]$ touch file1
[user1@server1 ~]$ ls -l file1
-rw-rw-r--. 1 user1 user1 0 Sep 17 15:00 file1
```

The output indicates that the owner of *file1* is *user1* who belongs to group *user1*. If you wish to view the corresponding UID and GID instead, you can specify the *-n* option with the command:

```
[user1@server1 ~]$ ls -ln file1  
-rw-rw-r--. 1 1000 1000 0 Sep 17 15:00 file1
```

Linux provides the *chown* and *chgrp* commands that can alter the ownership and owning group for files and directories; however, you must have the *root* user privilege to make these modifications.

Exercise 6-6: Modify File Owner and Owning Group

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will first create a file *file10* and a directory *dir10* as *user1* under */tmp*, and then change the ownership for *file10* to *user100* and the owning group to *dba* in two separate transactions. Then you'll apply ownership on *file10* to *user200* and owning group to *user100* at the same time. Finally, you will change the two attributes on the directory to *user200:dba* recursively. Make sure to use *sudo* where necessary.

1. Change into the */tmp* directory and create *file10* and *dir10*:

```
[user1@server1 ~]$ cd /tmp  
[user1@server1 tmp]$ touch file10  
[user1@server1 tmp]$ mkdir dir10
```

2. Check and validate that both attributes are set to *user1*:

```
[user1@server1 tmp]$ ls -l file10  
-rw-rw-r--. 1 user1 user1 0 Sep 17 15:15 file10  
[user1@server1 tmp]$ ls -ld dir10  
drwxrwxr-x. 2 user1 user1 6 Sep 17 15:15 dir10
```

3. Set the ownership to *user100* and confirm:

```
[user1@server1 tmp]$ sudo chown user100 file10  
[user1@server1 tmp]$ ls -l file10  
-rw-rw-r--. 1 user100 user1 0 Sep 17 15:15 file10
```

4. Alter the owning group to *dba* and verify:

```
[user1@server1 tmp]$ sudo chgrp dba file10  
[user1@server1 tmp]$ ls -l file10  
-rw-rw-r--. 1 user100 dba 0 Sep 17 15:15 file10
```

5. Change the ownership to *user200* and owning group to *user100* and confirm:

```
[user1@server1 tmp]$ sudo chown user200:user100 file10
[user1@server1 tmp]$ ls -l file10
-rw-rw-r--. 1 user200 user100 0 Sep 17 15:15 file10
```

6. Modify the ownership to *user200* and owning group to *dba* recursively on *dir10* and validate:

```
[user1@server1 tmp]$ sudo chown -R user200:dba dir10
[user1@server1 tmp]$ ls -ld dir10
drwxrwxr-x. 2 user200 dba 6 Sep 17 15:15 dir10
```

You can also use **ls -IR dir10** to view file and directory information under *dir10*; however, it will show nothing as the directory is currently empty.

Chapter Summary

At the outset, we described various aging attributes for user login and password controls that are available to us. We also looked at files that store default attributes that are applied to user accounts at the time of their creation. These defaults may be modified if necessary.

Next, we modified aging attributes for certain user accounts with the help of the tools we learned. We also employed a tool with a pair of flags to deny and restore user access to the system.

We examined group management commands and used them in exercises to create, modify, and delete group accounts, and add users to supplementary groups. The set of the management commands is straightforward and easy to use, but they require execution by a privileged user.

We looked at a couple of tools towards the end of the chapter to switch into other user accounts, including the root user, and to run privileged commands as a normal user. The use of the latter tool is recommended.

Finally, we discussed ownership and owning groups on files and directories, and how they are assigned. We performed exercises to understand who can modify them and how.

Review Questions

1. Which command can be used to display the effective (current) username?
2. What is the recommended location to store custom *sudo* rules?
3. What would the command *passwd -l user10* do?

4. The *chown* command may be used to modify both ownership and group membership on a file. True or False?
5. What would the command *passwd -n 7 -x 15 -w 3 user5* do?
6. Write the two command names for managing all of the password aging attributes.
7. Which file has the default password aging settings defined?
8. What would the command *chage -E 2020-10-22 user10* do?
9. What would the command *chage -I user5* do?
10. Which two commands can be used to lock and unlock a user account?
11. When using *sudo*, log files record activities under the *root* user account. True or False?
12. What is the difference between running the *su* command with and without the hyphen sign?
13. What is the significance of the *-o* option with the *groupadd* and *groupmod* commands?
14. What would the entry *user10 ALL=(ALL) NOPASSWD:ALL* in the *sudoers* file imply?
15. The *chgdp* command may be used to modify both ownership and group membership on a file. True or False?
16. What would the command *chage -d 0 user60* do?

Answers to Review Questions

1. The *whoami* command reports the effective username of the user running this command.
2. The custom sudo rules should be stored in files under the */etc/sudoers.d* directory.
3. This command will lock *user10*.
4. True.
5. The command will set mindays to 7, maxdays to 15, and warndays to 3 for *user5*.
6. The *passwd* and *chage* commands.
7. The */etc/login.defs* file has the defaults for password aging defined.
8. The command provided will set October 22, 2020 as the expiry date for *user10*.
9. The command provided will display password aging attributes for *user5*.
10. The *usermod* and *passwd* commands can be used to lock and unlock a user account.
11. False. The activities are registered under the username who invokes the *sudo* command.
12. With the dash sign the *su* command processes the specified user's startup files, and it won't without this sign.

13. The `-o` option lets the commands share a GID between two or more groups.
14. It will give `user10` password-less access to all privileged commands.
15. False.
16. The command provided will force `user60` to change their password at next login.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 6-1: Create User and Configure Password Aging

As `root` on `server1`, create group `Inxgrp` with GID 6000. Create user `user5000` with UID 5000 and GID 6000. Assign this user a password, and establish password aging attributes so that this user cannot change their password within 4 days after setting it and with a password validity of 30 days. This user should start getting warning messages for changing password 10 days prior to account lock down. This user account needs to expire on the 20th of December, 2021. (Hint1: [Chapter 05](#): Local User Authentication Files, and User Account Management). (Hint2: Password Aging and its Management, and Linux Groups and their Management).

Lab 6-2: Lock and Unlock User

As `root` on `server1`, lock the user account for `user5000` using the `passwd` command, and confirm by examining the change in the `/etc/shadow` file. Try to log in with `user5000` and observe what happens. Use the `usermod` command and unlock this account. Verify the unlocking by checking the entry for the user in the `/etc/shadow` file. (Hint: Password Aging and its Management).

Lab 6-3: Modify Group

As `root` on `server1`, modify the GID from 6000 to 7000 for the `Inxgrp` group. Add users `user1000` and `user2000` as supplementary members (create users if needed). Change the group's name to `dbagrp`, and verify. (Hint: Linux Groups and their Management).

Lab 6-4: Configure sudo Access

As *root* on *server1*, add a rule for *user5000* to the */etc/sudoers* file to allow this user full *root* access on the system. Make sure that this user is not prompted for a password when they use *sudo* to execute a command. Now switch into this user account and try running **sudo vgs**, and see if that works. (Hint: Substituting Users and Doing as Superuser).

Lab 6-5: Modify Owning User and Group

As *user1* on *server1*, create file *f6* and directory *d6* under */tmp*. Change owning user for *f6* to *user90* (create user) using **sudo chown**, and owning group to *dba* with **sudo chgrp**. Change owning user and group on *d6* to *user90:g1* (create group) recursively using **sudo chown**. (Hint: Owing User and Owning Group).

Chapter 07

The Bash Shell

This chapter describes the following major topics:

- Introduction to the bash shell
- Understand, set, and unset shell and environment variables
- Comprehend and use command and variable expansions
- Grasp and utilize input, output, and error redirections
- Know and use history substitution, command line editing, tab completion, tilde substitution, and command aliasing
- Identify and use metacharacters, wildcard characters, pipes, and pipelines
- Discover the use of quoting mechanisms and regular expressions
- Run and control jobs in background and foreground
- Identify and examine shell startup files

RHCSA Objectives:

02. Use input-output redirection (>, >>, |, 2>, etc.)
03. Use grep and regular expressions to analyze text

Shells

allow users to interact with the operating system for execution of their instructions through programs, commands, and applications. RHEL supports a variety of shells of which the bash shell is the most common. It is also the default shell for users in RHEL 8. This shell offers a variety of features that help users and administrators perform their job with ease and flexibility. Some of these features include recalling a command from history and editing it at the command line before running it, sending output and error messages to non-default target locations, creating command shortcuts, sending output of one command as input to the next, assigning the output of a command to a variable, using special characters to match a string of characters, treating a special character as a literal character, and so on.

Regular expressions are text patterns for matching against an input provided in a search operation. Text patterns may include any sequenced or arbitrary characters or character range. RHEL offers a powerful command to work with pattern matching.

At login, plenty of system and user startup scripts are executed to set up the user environment. The system startup scripts may be customized for all users by the Linux administrator and the user startup scripts may be modified by individual users as per their need.

The Bourne-Again Shell

A *shell* is referred to as the command interpreter, and it is the interface between a user and the Linux kernel. The shell accepts instructions (commands) from users (or programs), interprets them, and passes them on to the kernel for processing. The kernel utilizes all hardware and software components required for a successful processing of the instructions. When concluded, it returns the results to the shell, which then exhibits them on the screen. The shell also shows appropriate error messages, if generated. In addition, the shell delivers a customizable environment to users.

A widely used shell by Linux users and administrators is the *bash (bourne-again shell)* shell. Bash is a replacement for the older *Bourne* shell with numerous enhancements and plenty of new features incorporated from other shells. It is the default shell in most popular Linux distributions including RHEL 8 and offers several features such as variable manipulation, variable substitution, command substitution, input and output redirections, history substitution, tab completion, tilde substitution, alias substitution, metacharacters, pattern matching, filename globbing, quoting mechanisms,

conditional execution, flow control, and shell scripting. This section discusses all of these features except for the last three.

The bash shell is identified by the dollar sign (\$) for normal users and the hash sign (#) for the *root* user. The bash shell is resident in the */usr/bin/bash* file.

Shell and Environment Variables

A *variable* is a transient storage for data in memory. It retains information that is used for customizing the shell environment and referenced by many programs to function properly. The shell stores a value in a variable, and one or more white space characters must be enclosed within quotation marks ("").

There are two types of variables: *local* (or *shell*) and *environment*. A local variable is private to the shell in which it is created, and its value cannot be used by programs that are not started in that shell. This introduces the concept of *current shell* and *sub-shell* (or *child shell*). The current shell is where a program is executed, whereas a sub-shell (or child shell) is created within a shell to run a program. The value of a local variable is only available in the current shell.

The value of an environment variable is inherited from the current shell to the sub-shell during the execution of a program. In other words, the value stored in an environment variable is accessible to the program, as well as any sub-programs that it spawns during its lifecycle. Any environment variable set in a sub-shell is lost when the sub-shell terminates.

There are a multitude of predefined environment variables that are set for each user upon logging in. Use the **env** or the **printenv** command to view their values. Run these commands on *server1* as *user1* and observe the output. There should be around 25 of them. Some of the common predefined environment variables are described in [Table 7-1](#).

Variable	Description
DISPLAY	Stores the hostname or IP address for graphical terminal sessions
HISTFILE	Defines the file for storing the history of executed commands
HISTSIZE	Defines the maximum size for the HISTFILE
HOME	Sets the home directory path
LOGNAME	Retains the login name
MAIL	Contains the path to the user mail directory
PATH	Defines a colon-separated list of directories to be searched when executing a command. A correct setting of this variable eliminates the need to specify the absolute path of a command to run it.
PPID	Holds the identifier number for the parent program
PS1	Defines the primary command prompt
PS2	Defines the secondary command prompt
PWD	Stores the current directory location
SHELL	Holds the absolute path to the primary shell file
TERM	Holds the terminal type value
UID	Holds the logged-in user's UID
USER	Retains the name of the logged-in user

Table 7-1 Common Predefined Environment Variables

RHEL provides the `echo` command to view the values stored in variables. For instance, to view the value for the PATH variable, run the `echo` command and ensure to prepend the variable name (PATH) with the dollar sign (\$):

```
[user1@server1 ~]$ echo $PATH
/home/user1/.local/bin:/home/user1/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/
usr/sbin
```

Try running `echo $HOME`, `echo $SHELL`, `echo $TERM`, `echo $PPID`, `echo $PS1`, and `echo $USER` and see what values they store.

Setting and Unsetting Variables

Shell and environment variables may be set or unset at the command prompt or via programs, and their values may be viewed and used as necessary. We define and undefine variables and view their values using built-in shell

commands such as *export*, *unset*, and *echo*. It is recommended to use uppercase letters to name variables so as to circumvent any possible conflicts with a command, program, file, or directory name that exist somewhere on the system. To understand how variables are defined, viewed, made environment, and undefined, a few examples are presented below. We run the commands as *user1*.

To define a local variable called VR1:

```
[user1@server1 ~]$ VR1=RHEL8
```

To view the value stored in VR1:

```
[user1@server1 ~]$ echo $VR1
```

Now type **bash** at the command prompt to enter a sub-shell and then run **echo \$VR1** to check whether the variable is visible in the sub-shell. You will not find it there, as it was not an environment variable. Exit out of the sub-shell by typing **exit**.

To make this variable an environment variable, use the *export* command:

```
[user1@server1 ~]$ export VR1
```

Repeat the previous test by running the **bash** command and then the **echo \$VR1**. You should be able to see the variable in the sub-shell, as it is now an environment variable. Exit out of the sub-shell with the **exit** command.

To undefine this variable and erase it from the shell environment:

```
[user1@server1 ~]$ unset VR1
```

To define a local variable that contains a value with one or more white spaces:

```
[user1@server1 ~]$ VR2="I love RHEL 8"
```

To define and make the variable an environment variable at the same time:

```
[user1@server1 ~]$ export VR3="I love RHEL 8"
```

The **echo** command is restricted to showing the value of a specific variable and the **env** and **printenv** commands display the environment variables only. If you want to view both local and environment variables, use the **set** command. Try running **set** and observe the output.

Command and Variable Substitutions

The primary command prompt ends in the hash sign (#) for the *root* user and in the dollar sign (\$) for normal users. Customizing this prompt to exhibit useful information such as who you are, the system you are logged on to, and your current location in the directory tree is a good practice. By default, this setting is already in place through the PS1 environment variable, which defines the primary command prompt. You can validate this information by looking at the command prompt [user1@server1 ~]\$.

You can also view the value stored in PS1 by issuing **echo \$PS1**. The value is \u@\h \W\\$. The \u translates into the logged-in username, \h represents the hostname of the system, \W shows your current working directory, and \\$ indicates the end of the command prompt.

[Exercise 7-1](#) demonstrates how to employ the command and variable substitution features to customize the primary command prompt for *user1*. You will use the *hostname* command and assign its output to the PS1 variable. This is an example of *command substitution*. Note that the command whose output you want assigned to a variable must be encapsulated within either backticks `hostname` or parentheses \$(hostname).

There are two instances of the *variable substitution* feature employed in [Exercise 7-1](#). The LOGNAME and the PWD environment variables to display the username and to reflect the current directory location.

Exercise 7-1: Modify Primary Command Prompt

This exercise should be done on *server1* as *user1*.

In this exercise, you will customize the primary shell prompt to display the information enclosed within the quotes “< username on hostname in pwd >:” using the variable and command substitution features. You will do this at the command prompt. You will then edit the *~/.bash_profile* file for *user1* and define the new value in there for permanence (see Shell Startup Files later in this chapter to understand the purpose of *.bash_profile*), otherwise, the value will be lost when *user1* closes the terminal window or logs off.

1. Change the value of the variable PS1 to reflect the desired information:

```
[user1@server1 ~]$ export PS1=< $LOGNAME on $(hostname) in \$PWD > "
< user1 on server1.example.com in /home/user1 >
```

The prompt has changed to display the desired information.

2. Edit the *.bash_profile* file for *user1* and define the value exactly as it was run in Step 1.

3. Test by logging off as *user1* and logging back in. The new command prompt will be displayed.

This concludes the exercise.

Input, Output, and Error Redirections

Programs read input from the keyboard and write output to the terminal window where they are initiated. Any errors, if encountered, are printed on the terminal window too. This is the default behavior. The bash handles input, output, and errors as character streams. If you do not want input to come from the keyboard or output and error to go to the terminal screen, the shell gives you the flexibility to redirect input, output, and error messages to allow programs and commands to read input from a non-default source, and forward output and errors to one or more non-default destinations.

EXAM TIP: You may be asked to run a command and redirect its output and/or error messages to a file.

The default (or the standard) locations for the three streams are referred to as *standard input* (or *stdin*), *standard output* (or *stdout*), and *standard error* (or *stderr*). These locations may also be epitomized using the opening angle bracket symbol (<) for *stdin*, and the closing angle bracket symbol (>) for *stdout* and *stderr*. Alternatively, you may use the *file descriptors* (the digits 0, 1, and 2) to represent the three locations.

Redirecting Standard Input

Input redirection instructs a command to read input from an alternative source, such as a file, instead of the keyboard. The opening angle bracket (<) is used for input redirection. For example, run the following to have the *cat* command read the */etc/redhat-release* file and display its content on the standard output (terminal screen):

```
[user1@server1 ~]$ cat < /etc/redhat-release
Red Hat Enterprise Linux release 8.0 (Ootpa)
```

Redirecting Standard Output

Output redirection sends the output generated by a command to an alternative destination, such as a file, instead of to the terminal window. The closing angle bracket (>) is used for this purpose. For instance, the following directs the *ls* command output to a file called *ls.out*. This will overwrite any existing *ls.out* file if there is one; otherwise, a new file will be created.

```
[user1@server1 ~]$ ls > ls.out
```

The above can also be run as **ls 1> ls.out** where the digit “1” represents the standard output location.

If you want to prevent an inadvertent overwriting of the output file, you can enable the shell’s noclobber feature with the **set** command and confirm its activation by re-issuing the above redirection example:

```
[user1@server1 ~]$ set -o noclobber  
[user1@server1 ~]$ ls > ls.out  
-bash: ls.out: cannot overwrite existing file
```

You are denied the action.



You can disable the noclobber option by running **set +o noclobber** at the command prompt.

To direct the **ls** command to append the output to the **ls.out** file instead of overwriting it, use the two closing angle brackets (**>>**):

```
[user1@server1 ~]$ ls >> ls.out
```

Again, the equivalent for the above is **ls 1>> ls.out**.

Redirecting Standard Error

Error redirection forwards any error messages generated to an alternative destination rather than to the terminal window. An alternative destination could be a file. For example, the following directs the **find** command issued as a normal user to search for all occurrences of files by the name **core** in the entire directory tree and sends any error messages produced to **/dev/null** (**/dev/null** is a special file that is used to discard data). This way only the useful output is exhibited on the screen and errors are thrown away.

```
[user1@server1 ~]$ find / -name core -print 2> /dev/null
```

Redirecting both Standard Output and Error

You may redirect both output and error to alternative locations as well. For instance, issue the following to forward them both to a file called **outerr.out**:

```
[user1@server1 ~]$ ls /usr /cdr &> outerr.out
```

This example will produce a listing of the **/usr** directory and save the result in **outerr.out**. At the same time, it will generate an error message complaining about the non-existence of **/cdr**, and it will send it to the same file as well.



Another method to run the above command is by typing **ls /usr/cdr 1> outerr.out 2>&1**, which essentially means to redirect file descriptor 1 to file *outerr.out* as well as to file descriptor 2.

You can exchange &> with &>> in the above example to append the information in the file rather than overwriting it.

History Substitution

History substitution (a.k.a. *command history* or *history expansion*) is a time-saver bash shell feature that keeps a log of all commands or commandsets that you run at the command prompt in chronological order with one command or commandset per line. The history feature is enabled by default; however, you can disable and re-enable it if required. The bash shell stores command history in a file located in the user's home directory and in system memory. You may retrieve the commands from history, modify them at the command prompt, and rerun them.

There are three variables—`HISTFILE`, `HISTSIZE`, and `HISTFILESIZE`—that control the location and history storage. `HISTFILE` defines the name and location of the history file to be used to store command history, and the default is `.bash_history` in the user's home directory. `HISTSIZE` dictates the maximum number of commands to be held in memory for the current session.

`HISTFILESIZE` sets the maximum number of commands allowed for storage in the history file at the beginning of the current session and are written to the `HISTFILE` from memory at the end of the current terminal session. Usually, `HISTSIZE` and `HISTFILESIZE` are set to a common value. These variables and their values can be viewed with the `echo` command. The following shows the settings for *user1*:

```
[user1@server1 ~]$ echo $HISTFILE  
/home/user1/.bash_history  
[user1@server1 ~]$ echo $HISTSIZE  
1000  
[user1@server1 ~]$ echo $HISTFILESIZE  
1000
```

The values of any of these variables may be altered for individual users by editing the `.bashrc` or `.bash_profile` file in the user's home directory. A discussion on these files is provided later in this chapter.

In RHEL, the `history` command displays or reruns previously executed commands. This command gets the history data from the system memory as well as from the `.bash_history` file. By default, it shows all entries. Run this command at the prompt without any options and it will dump everything on the screen:

```
[user1@server1 ~]$ history
```

The *history* command has some useful options. Let's use them and observe the impact on the output.

To display this command and the ten preceding entries:

```
[user1@server1 ~]$ history 10
```

To re-execute a command by its line number (line 15 for example):

```
[user1@server1 ~]$ !15
```

To re-execute the most recent occurrence of a command that started with a particular letter or series of letters (ch for example):

```
[user1@server1 ~]$ !ch
```

To issue the most recent command that contained “grep”:

```
[user1@server1 ~]$ !?grep?
```

To remove entry 24 from history:

```
[user1@server1 ~]$ history -d 24
```

To repeat the last command executed:

```
[user1@server1 ~]$ !!
```

The second exclamation mark (!) in the above example is used to retrieve the last executed command.

You may disable the shell's history expansion feature by issuing **set +o history** at the command prompt and re-enable it with **set -o history**. The **set** command is used in this way to enable or disable a bash shell feature.

Editing at the Command Line

While typing a command with a multitude of options and arguments at the command prompt, you often need to move the cursor backward to add or modify something. For instance, if you are typing a long command as a normal user and then realize the command needs to have *sudo* added at the beginning, move the cursor quickly to the start of the command. This is a shortcut to save time. There are plenty of key combinations for rapid movement on the command line. [Table 7-2](#) lists and explains several of these.

Key Combinations	Action
Ctrl+a / Home	Moves the cursor to the beginning of the command line
Ctrl+e / End	Moves the cursor to the end of the command line
Ctrl+u	Erase the entire line
Ctrl+k	Erase from the cursor to the end of the command line
Alt+f	Moves the cursor to the right one word at a time
Alt+b	Moves the cursor to the left one word at a time
Ctrl+f / Right arrow	Moves the cursor to the right one character at a time
Ctrl+b / Left arrow	Moves the cursor to the left one character at a time

Table 7-2 Helpful Command Line Editing Shortcuts

For normal users and administrators who often work at the command line, these key combinations will be very helpful in saving time.

Tab Completion

Tab completion (a.k.a. *command line completion*) is a bash shell feature whereby typing one or more initial characters of a file, directory, or command name at the command line and then hitting the Tab key twice automatically completes the entire name. In case of multiple possibilities matching the entered characters, it completes up to the point they have in common and prints the rest of the possibilities on the screen. You can then type one or more following characters and press Tab again to further narrow down the possibilities. When the desired name appears, press Enter to accept it and perform the action. One of the major benefits of using this feature is the time saved on typing long file, directory, or command names.

Try this feature out on your Linux terminal window and get yourself familiarized with it.

Tilde Substitution

Tilde substitution (or *tilde expansion*) is performed on words that begin with the tilde character (~). The rules to keep in mind when using the ~ are:

1. If ~ is used as a standalone character, the shell refers to the \$HOME directory of the user running the command. The following example displays the \$HOME directory of user1:

```
[user1@server1 ~]$ echo ~
/home/user1
```

2. If the plus sign (+) follows the ~, the shell refers to the current directory. For example, if *user1* is in the /usr/bin directory and does ~+, the output exhibits the user's current directory location:

```
[user1@server1 bin]$ echo ~+
/usr/bin
```

3. If the hyphen character (-) follows the ~, the shell refers to the previous working directory. For example, if *user1* switches into the /usr/share/man directory from /usr/bin and does ~-, the output displays the user's last working directory:

```
[user1@server1 bin]$ cd /usr/share/man
[user1@server1 man]$ echo ~-
/usr/bin
```

4. If a username has the ~ prepended, the shell refers to the \$HOME directory of that user:

```
[user1@server1 man]$ echo ~user200
/home/user200
```

Tilde substitution can also be used with commands such as *cd* and *ls*. For instance, to *cd* into the home directory of *user1* (or any other user for that matter) from anywhere in the directory structure, specify the login name with the ~:

```
[user1@server1 man]$ cd ~user1
[user1@server1 ~]$ pwd
/home/user1
```

This command is only successful if *user1* has the right permissions to navigate into the target user's home directory. Let's try this example as the *root* user:

```
[root@server1 ~]# cd ~user100
[root@server1 user100]# pwd
/home/user100
```

The above example demonstrates the navigation into the home directory of *user100* as the *root* user. By default, *root* has full rights to *cd* into any users' home directory.

Another example below exhibits how a user can *cd* into one of their subdirectories (such as /home/user1/dir1) directly from anywhere (/etc/systemd/system) in the directory structure (create *dir1* for this test):

```
[user1@server1 ~]$ cd  
[user1@server1 ~]$ mkdir dir1  
[user1@server1 ~]$ cd /etc/systemd/system/  
[user1@server1 system]$ pwd  
/etc/systemd/system  
[user1@server1 system]$ cd ~/dir1  
[user1@server1 dir1]$ pwd  
/home/user1/dir1
```

Try using the `~` with the `ls` command. For example, run `sudo ls -ld ~root/Desktop` as `user1`.

Alias Substitution

Alias substitution (a.k.a. *command aliasing* or *alias*) allows you to define a shortcut for a lengthy and complex command or a set of commands. Defining and using aliases saves time and saves you from typing. The shell executes the corresponding command or commandset when an alias is run.

The bash shell includes several predefined aliases that are set during user login. These aliases may be viewed with the `alias` command. The following shows all aliases that are currently set for `user1`:

```
[user1@server1 ~]$ alias  
alias vi='vim'  
alias which='(alias; declare -f) | /usr/bin/which --tty-only --read-alias --read  
-functions --show-tilde --show-dot'
```

There are more default aliases set for the `root` user. Run `alias` as `root`:

```
[root@server1 ~]# alias  
alias cp='cp -i'  
alias ll='ls -l'  
alias mv='mv -i'  
alias rm='rm -i'  
alias which='(alias; declare -f) | /usr/bin/which --tty-only --read-alias --read  
-functions --show-tilde --show-dot'
```

[Table 7-3](#) describes aliases from the previous two outputs.

Alias	Value	Definition
vi	vim	Runs the vim command instead of the old vi
cp	cp -i	Issues the cp command in interactive mode
ll	ls -l	Shows the ls command output in long format
mv	mv -i	Executes the mv command in interactive mode
rm	rm -i	Runs the rm command in interactive mode
which	(alias; declare -f) /usr/bin/which -- tty-only --read-alias --read-functions --show-tilde --show-dot	Runs the which command with specified options

Table 7-3 Predefined Aliases

In addition to listing the set aliases, the *alias* command can also be used to define new aliases. The opposite function of unsetting an alias is performed with the *unalias* command. Both *alias* and *unalias* are internal shell commands. Let's look at a few examples.

Create an alias "search" to abbreviate the *find* command with several switches and arguments. Enclose the entire command within single quotation marks ('') to ensure white spaces are taken care of. Do not leave any spaces before and after the equal sign (=).

```
[user1@server1 ~]$ alias search='find / -name core -exec ls -l {} \;'
```

Now, when you type the string "search" at the command prompt and press the Enter key, the shell will trade the alias "search" with what is stored in it and will run it. Essentially, you have created a shortcut to that lengthy command.

Sometimes you define an alias by a name that matches the name of some system command or program. In this situation, the shell gives the execution precedence to the alias. This means the shell will run the alias and not the command or the program. For example, the *rm* command deletes a file without giving any warning if run as a normal user. To prevent accidental deletion of files with *rm*, you may create an alias by the same name as the command but with the interactive option added, as shown below:

```
[user1@server1 ~]$ alias rm='rm -i'
```

When you execute *rm* now to remove a file, the shell will run what is stored in the *rm* alias and not the command *rm*. If you wish to run the *rm* command instead, run it by preceding a backslash (\) with it:

```
[user1@server1 ~]$ \rm file1
```

You can use the *unalias* command to unset one or more specified aliases if they are no longer in need. The following will undefine the two aliases that were defined for our examples:

```
[user1@server1 ~]$ unalias search rm
```

Metacharacters and Wildcard Characters

Metacharacters are special characters that possess special meaning to the shell. Some of them are the dollar sign (\$), caret (^), period (.), asterisk (*), question mark (?), pipe (|), angle brackets (< >), curly brackets ({}), square brackets ([]), parentheses (()), plus (+), exclamation mark (!), semicolon (;), and backslash (\) characters. They are used in pattern matching (a.k.a. *filename expansion* or *file globbing*) and regular expressions. This sub-section discusses the metacharacters (* ? [] !) that are used in pattern matching. The *, ?, and [] characters are also referred to as *wildcard characters*.

The * Character

The asterisk (*) matches zero to an unlimited number of characters except for the leading period (.) in a hidden filename. See the following examples on usage.

To list all files in the /etc directory that begin with letters “ma” and followed by any characters:

```
[user1@server1 ~]$ ls /etc/ma*
/etc/machine-id /etc/mailcap /etc/man_db.conf
/etc/magic /etc/makedumpfile.conf.sample
```

To list all hidden files and directories in /home/user1:

```
[user1@server1 ~]$ ls -d .*
. .bash_history .bash_logout .bash_profile .bashrc .lesshst .mozilla
```

To list all files in the /var/log directory that end in “.log”:

```
[user1@server1 ~]$ ls /var/log/*.log
/var/log/boot.log          /var/log/dnf.rpm.log      /var/log/Xorg.0.log
/var/log/dnf.librepo.log   /var/log/hawkey.log     /var/log/Xorg.9.log
/var/log/dnf.log           /var/log/vmware-vmusr.log
```

The * is probably the most common metacharacter that is used in pattern matching.

The ? Character

The question mark (?) matches exactly one character except for the leading period in a hidden filename. See the following example to understand its usage.

To list all files and directories under /var/log with exactly four characters in their names:

```
[user1@server1 ~]$ ls -d /var/log/?????
/var/log/btmp  /var/log/cups  /var/log/sssd
/var/log/cron  /var/log/rhsm  /var/log/wtmp
```

The ? is another metacharacter that is used widely in pattern matching.

The Square Brackets [] and the Exclamation Mark !

The square brackets ([]) can be used to match either a set of characters or a range of characters for a single character position.

For a set of characters specified in this enclosure, the order in which they are listed has no importance. This means the shell will interpret [xyz], [yxz], [xzy], and [zyx] alike during pattern matching. In the following example, two characters are enclosed within the square brackets. The output will include all files and directories that begin with either of the two characters and followed by any number of characters.

```
[user1@server1 ~]$ ls /usr/bin/[yw]*
/usr/bin/w          /usr/bin/whereis          /usr/bin/wwunpack
/usr/bin/wait        /usr/bin/which           /usr/bin/yelp
/usr/bin/wall        /usr/bin/whiptail        /usr/bin/yelp-build
/usr/bin/watch       /usr/bin/who             /usr/bin/yelp-check
/usr/bin/watchgnupg  /usr/bin/whoami          /usr/bin/yelp-new
/usr/bin/wavpack     /usr/bin/wnck-urgency-monitor /usr/bin/yes
/usr/bin/wc          /usr/bin/word-list-compress /usr/bin/ypdomainname
/usr/bin/wdctl       /usr/bin/write          /usr/bin/yum
/usr/bin/wget        /usr/bin/wvgain         /
/usr/bin/whatis      /usr/bin/wvtag          /
```

A range of characters must be specified in a proper sequence such as [a-z] or [0-9]. The following example matches all directory names that begin with any letter between "m" and "o" in the /etc/systemd/system directory:

```
[user1@server1 ~]$ ls -d /etc/systemd/system/[m-o]*
/etc/systemd/system/multi-user.target.wants
/etc/systemd/system/network-online.target.wants
/etc/systemd/system/nfs-blkmap.service.requires
/etc/systemd/system/nfs-idmapd.service.requires
/etc/systemd/system/nfs-mountd.service.requires
/etc/systemd/system/nfs-server.service.requires
```

The shell enables the exclamation mark (!) to inverse the matches. For instance, [!a-d]* would exclude all filenames that begin with any of the first four alphabets. The following example will produce the reverse of what the previous example did:

```
[user1@server1 ~]$ ls -d /etc/systemd/system/[ !m-o]*
/etc/systemd/system/basic.target.wants
/etc/systemd/system/bluetooth.target.wants
/etc/systemd/system/dbus-org.bluez.service
/etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service
/etc/systemd/system/dbus-org.freedesktop.avahi.service
/etc/systemd/system/dbus-org.freedesktop.ModemManager1.service
/etc/systemd/system/dbus-org.freedesktop.NetworkManager.service
/etc/systemd/system/dbus-org.freedesktop.nm-dispatcher.service
```

The output will have a multitude of filenames printed.

Piping Output of One Command as Input to Another

The *pipe*, represented by the vertical bar (|) and normally resides with the backslash (\) on most keyboards, is used to send the output of one command as input to the next. This character is also used to define alternations in regular expressions. You can use the pipe operator as many times in a command as you require.

The /etc directory contains plenty of files, but they all do not fit on one terminal screen when you want to see its long listing. You can use the pipe operator to pipe the output to the less command in order to view the directory listing one screenful at a time:

```
[user1@server1 ~]$ ls -l /etc | less
total 1352
-rw-r--r--. 1 root root      16 Aug 22 15:26 adjtime
-rw-r--r--. 1 root root    1518 Sep 10 2018 aliases
drwxr-xr-x. 3 root root      65 Aug 22 15:16 alsa
drwxr-xr-x. 2 root root    4096 Aug 22 15:22 alternatives
-rw-r--r--. 1 root root     541 Oct  2 2018 anacrontab
....
```

In another example, the *last* command is run and its output is piped to the *nl* command to number each output line:

```
[user1@server1 ~]$ last | nl
 1 user1 pts/0 192.168.0.219 Wed Sep 18 11:05 still logged in
 2 user1 pts/1 192.168.0.219 Wed Sep 18 10:40 - 11:05 (00:24)
 3 user1 pts/0 192.168.0.219 Wed Sep 18 10:06 - 10:40 (00:34)
 4 user1 pts/0 192.168.0.219 Wed Sep 18 10:00 - 10:02 (00:01)
 5 root pts/0 192.168.0.219 Wed Sep 18 10:00 - 10:00 (00:00)
 6 user1 pts/0 192.168.0.219 Tue Sep 17 15:00 - 10:00 (18:59)
 7 user1 pts/1 192.168.0.219 Tue Sep 17 10:03 - 12:51 (02:47)
.......
```

The following example sends the output of *ls* to *grep* for the lines that do not contain the pattern “root”. The new output is further piped for a case-insensitive selection of all lines that exclude the pattern “dec”. The filtered output is numbered, and the final result shows the last four lines on the display.

```
[user1@server1 ~]$ ls -l /proc | grep -v root | grep -iv dec | nl | tail -4
49 dr-xr-xr-x. 9 gdm gdm 0 Sep 14 20:23 960
50 dr-xr-xr-x. 9 gdm gdm 0 Sep 14 20:23 979
51 dr-xr-xr-x. 9 gdm gdm 0 Sep 14 20:23 987
52 dr-xr-xr-x. 9 gdm gdm 0 Sep 14 20:23 995
```

A construct like the above with multiple pipes is referred to as a *pipeline*.

Quoting Mechanisms

As you know, metacharacters have special meaning to the shell. In order to use them as regular characters, the bash shell offers three *quoting mechanisms* to disable their special meaning and allow the shell to treat them as literal characters. These mechanisms are available through the use of the backslash (\), single quotation (“”), and double quotation (“”) characters, and work by prepending a special character to the backslash, or enclosing it within single or double quotation marks.

Prefixing with a Backslash \

The backslash character (\), also referred to as the *escape* character in shell terminology, instructs the shell to mask the meaning of any special character that follows it. For example, if a file exists by the name * and you want to remove it with the *rm* command, you will have to escape the * so that it is treated as a regular character, not as a wildcard character.

```
[user1@server1 ~]$ rm \*
```

In the above example, if you forget to escape the *, the *rm* command will remove all files from the directory.

Enclosing within Single Quotes “”

The single quotation marks (‘’) instructs the shell to mask the meaning of all encapsulated special characters. For example, LOGNAME is a variable, and its value can be viewed with the `echo` command:

```
[user1@server1 ~]$ echo $LOGNAME  
user1
```

If you encapsulate \$LOGNAME within single quotes, the `echo` command will exhibit the entire string as is:

```
[user1@server1 ~]$ echo '$LOGNAME'  
$LOGNAME
```

In the above example, the shell interprets the dollar sign (\$) as a literal character, and that's why the `echo` command displays what is inside the enclosure, including the \$ rather than the value of the variable.

Enclosing within Double Quotes “”

The double quotation marks (“”) commands the shell to mask the meaning of all but the backslash (\), dollar sign (\$), and single quotes (‘’). These three special characters retain their special meaning when they are enclosed within double quotes. Look at the following examples to understand its usage.

```
[user1@server1 ~]$ echo "$SHELL"  
/bin/bash  
[user1@server1 ~]$ echo "\$PWD"  
$PWD  
[user1@server1 ~]$ echo "\"\\""  
\"
```

The above three instances confirm the use of the double quotes as a special character pair, as depicted in the outputs.

Regular Expressions

A *regular expression*, also referred to as a *regexp* or simply *regex*, is a text pattern or an expression that is matched against a string of characters in a file or supplied input in a search operation. The pattern may include a single character, multiple random characters, a range of characters, word, phrase, or an entire sentence. Any pattern containing one or more white spaces must be surrounded by quotation marks.

RHEL provides a powerful tool called `grep` (*global regular expression print* or *get regular expression and print*) to work with pattern matching in regular expressions. This tool searches the contents of one or more text files or input supplied for a match. If the expression is matched, `grep` prints every line

containing that expression on the screen without changing the source content. *grep* has plenty of options and it accepts expressions in various forms.

EXAM TIP: The *grep* command is a handy tool to extract needed information from a file or command output. The extracted information can then be redirected to a file. The sequence of the grep'ed data remains unchanged.

Let's consider the following examples to comprehend the usage of the *grep* command.

To search for the pattern “operator” in the */etc/passwd* file:

```
[user1@server1 ~]$ grep operator /etc/passwd
operator:x:11:0:operator:/root:/sbin/nologin
```

To search for the space-separated pattern “aliases and functions” in the *\$HOME/.bashrc* file:

```
[user1@server1 ~]$ grep 'aliases and functions' .bashrc
# User specific aliases and functions
```

To search for the pattern “nologin” in the *passwd* file and exclude (-v) the lines in the output that contain this pattern. Add the -n switch to show the line numbers associated with the matched lines.

```
[user1@server1 ~]$ grep -nv nologin /etc/passwd
1:root:x:0:0:root:/root:/bin/bash
6:sync:x:5:0:sync:/sbin:/bin/sync
7:shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
8:halt:x:7:0:halt:/sbin:/sbin/halt
45:user1:x:1000:1000:user1:/home/user1:/bin/bash
46:user100:x:1002:1002::/home/user100:/bin/bash
47:user200:x:1003:1003::/home/user200:/bin/bash
48:user3:x:1010:1010::/usr/user3a:/bin/sh
```

To find any duplicate entries for the *root* user in the *passwd* file. Prepend the caret sign (^) to the pattern “root” to mark the beginning of a line.

```
[user1@server1 ~]$ grep ^root /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

To identify all users in the *passwd* file with bash as their primary shell. Append the dollar sign (\$) to the pattern “bash” to mark the end of a line.

```
[user1@server1 ~]$ grep bash$ /etc/passwd
root:x:0:0:root:/root:/bin/bash
user1:x:1000:1000:user1:/home/user1:/bin/bash
user100:x:1002:1002::/home/user100:/bin/bash
user200:x:1003:1003::/home/user200:/bin/bash
```

To show the entire *login.defs* file but exclude all the empty lines:

```
[user1@server1 ~]$ grep -v '^$' /etc/login.defs
```

To perform a case-insensitive search (-i) for all the lines in the */etc/bashrc* file that match the pattern “path.”

```
[user1@server1 ~]$ grep -i path /etc/bashrc
# Need to redefine pathmunge, it gets undefined at the end of /etc/profile
pathmunge () {
    case ":${PATH}:" in
        PATH=$PATH:$1
        PATH=$1:$PATH
    unset -f pathmunge
```

To print all the lines from the */etc/lvm/lvm.conf* file that contain an exact match for a word (-w). You can use the period character (.) in the search string to match a single position. The following example searches for words in the *lvm.conf* file that begin with letters “acce” followed by exactly two characters:

```
[user1@server1 ~]$ grep -w acce.. /etc/lvm/lvm.conf
# Commands also accept this as a prefix on volume group names.
# This is a list of regular expressions used to accept or reject block
# (or any character) and is preceded by 'a' to accept the path, or
# device is rejected. Unmatching path names do not affect the accept
# The minimum IO size for VDO volume to accept, in bytes.
# Allow quicker VG write access during high volume read access.
# When there are competing read-only and read-write access requests for
# be serviced. Without this setting, write access may be stalled by a
# errors on access. Using 'zero' will return success (and zero) on I/O
```

In addition to the caret (^), dollar (\$), and period (.), the asterisk (*), question mark (?), square brackets ([]), and curly brackets ({})) are also used in regular expressions.

To print all the lines from the *ls* command output that include either (-E) the pattern “cron” or “ly”. The pipe | character is used as an OR operator in this example. This is referred to as *alternation*. Regex allows you to add more patterns to this set if desired. For instance, if you search for three patterns, use ‘pattern1|pattern2|pattern3’. The patterns must be enclosed within single or double quotes. Here is what you will issue when you want to look for the patterns “cron” and “ly” in the */etc* directory listing:

```
[user1@server1 ~]$ ls -l /etc | grep -E 'cron|ly'
-rw-r--r--. 1 root root      541 Oct  2  2018 anacrontab
drwxr-xr-x. 2 root root      39 Aug 22 15:14 cron.d
drwxr-xr-x. 2 root root      36 Aug 22 15:14 cron.daily
-rw-r--r--. 1 root root       0 Oct  2  2018 cron.deny
drwxr-xr-x. 2 root root     22 Aug 22 15:09 cron.hourly
drwxr-xr-x. 2 root root      6 Aug 12 2018 cron.monthly
-rw-r--r--. 1 root root     451 Aug 12 2018 crontab
drwxr-xr-x. 2 root root      6 Aug 12 2018 cron.weekly
drwxr-xr-x. 2 root root     28 Aug 22 15:14 plymouth
```

To show all the lines from the `/etc/ssh/sshd_config` file but exclude (-v) the empty lines and the rows that begin with the hash (#) character (commented lines). Using the -e flag multiple times as shown below is equivalent to how “-E ‘pattern1|pattern2’” was used in the above example.

```
[user1@server1 ~]$ sudo grep -ve ^$ -ve ^# /etc/ssh/sshd_config
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
SyslogFacility AUTHPRIV
```

You can combine options and employ diverse regular expression criteria to perform complex matches on input. Issue **man 7 regex** in a terminal window to learn more about regex. Also consult the manual pages of the `grep` command to view available options and other details.

Running and Controlling Jobs in Foreground and Background

By default, any program or command you issue runs in the foreground and ties itself to the terminal window where it is initiated. It can be moved to continue to run in the background so that the associated terminal session is released for performing other tasks. A *job* is a process that is started in the background and controlled by the terminal where it is spawned. Just like any other process that is started on the system, a job is also assigned a PID (*process identifier*) by the kernel and, additionally, a job ID by the shell. Unlike a normal process, a job does not hold the terminal window where it is initiated. This enables you to run other programs from the same terminal window. See [Chapter 08 “Linux Processes and Task Scheduling”](#) for details on processes and PIDs.

The shell allows running multiple jobs simultaneously, including transferring large amounts of data and running application programs in the background. Background jobs can be brought to foreground, returned to the background, suspended, or stopped. The management of multiple jobs within a shell environment is called *job control*.

The shell offers certain commands and control sequences for administering the jobs. See [Table 7-4](#) for their description.

Command	Description
jobs	Shell built-in command to display jobs
bg	Shell built-in command to move a job to the background or restart a job in the background that was suspended with Ctrl+Z
fg	Shell built-in command to move a job to the foreground
Ctrl+Z	Suspends a foreground job and allows the terminal window to be used for other purposes

Table 7-4 Job Control Commands and Control Sequences

To run a job in the background, type the command at the command prompt followed by the ampersand (&) operator.

The examples below start the *top* and *vim* programs in the background. The shell displays their assigned job IDs (enclosed within square brackets) and PIDs. The job IDs allow us to control the jobs, and the PIDs are used by the kernel to manage the processes.

```
[user1@server1 ~]$ top &
[1] 31726
[user1@server1 ~]$ vim testfile1 &
[2] 31727

[1]+  Stopped                  top
```

Issue the *jobs* command with the -l switch to view all the jobs running in the background:

```
[user1@server1 ~]$ jobs -l
[1]- 31726 Stopped (signal)          top
[2]+ 31727 Stopped (tty output)      vim testfile1
```

The plus sign (+) indicates the current background job and the minus sign (-) signifies the previous job. “Stopped” implies that the jobs are currently suspended and can be signaled to continue their execution in the background with the *bg* command or bring them back to the foreground with the *fg* command.

To bring job ID 1 to the foreground and start running it:

```
[user1@server1 ~]$ fg %1
```

To suspend job ID 1, press **^z** followed by **bg %1** to let it run in the background.

To terminate job ID 1, supply its PID (31726) to the *kill* command:

```
[user1@server1 ~]$ kill 31726
```

A message similar to the following appears when a background job ends its execution:

```
[2] + Done vi testfile1 &
```

See [Chapter 08 “Linux Processes and Task Scheduling”](#) for more information on the *kill* command.

Shell Startup Files

Earlier in this chapter, you used local and environment variables and learned how to modify the primary command prompt to add useful information to it. In other words, you modified the default shell environment to suit your needs. You stored the PS1 value in a shell startup file to ensure the changes are always available after you log off and log back in.

Modifications to the default shell environment can be stored in *startup* (or *initialization*) files. These files are sourced by the shell following user authentication at the time of logging in and before the command prompt appears. In addition, aliases, functions, and scripts can be added to these files as well. There are two types of startup files: *system-wide* and *per-user*.

System-wide Shell Startup Files

System-wide startup files set the general environment for all users at the time of their login to the system. These files are located in the /etc directory and are maintained by the Linux administrator. System-wide files can be modified to include general environment settings and customizations.

[Table 7-5](#) lists and describes system-wide startup files for bash shell users.

File	Comments
/etc/bashrc	Defines functions and aliases, sets umask for user accounts with a non-login shell, establishes the command prompt. It may include settings from the shell scripts located in the /etc/profile.d directory.
/etc/profile	Sets common environment variables such as PATH, LOGNAME, MAIL, HOSTNAME, HISTSIZE, and HISTCONTROL for all users, establishes umask for user accounts with a login shell, processes the shell scripts in the /etc/profile.d directory, and so on.
/etc/profile.d	Contains scripts for bash shell users that are executed from the /etc/profile file.

Table 7-5 System-wide Startup Files

Excerpts from the *bashrc* and *profile* files and a list of files in the *profile.d* directory are displayed below:

```
[user1@server1 ~]$ head /etc/bashrc
# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.

[user1@server1 ~]$ head /etc/profile
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.

[user1@server1 ~]$ ls -1 /etc/profile.d/
total 64
-rw-r--r--. 1 root root  664 Aug 13  2018 bash_completion.sh
-rw-r--r--. 1 root root   80 Sep 10  2018 csh.local
-rw-r--r--. 1 root root  294 Feb 14  2019 flatpak.sh
-rw-r--r--. 1 root root 1107 Dec 14  2017 gawk.csh
-rw-r--r--. 1 root root  757 Dec 14  2017 gawk.sh
-rw-r--r--. 1 root root 2291 Sep 10  2018 lang.csh
-rw-r--r--. 1 root root 2276 Sep 10  2018 lang.sh
-rw-r--r--. 1 root root  500 Aug 12  2018 less.csh
```

Any file in the *profile.d* directory can be edited and updated. Alternatively, you can create your own file and define any customization that you want.

Per-user Shell Startup Files

Per-user shell startup files override or modify system default definitions set by the system-wide startup files. These files may be customized by individual users to suit their needs. By default, two such files, in addition to the *.bash_logout* file, are located in the skeleton directory */etc/skel* and are copied into user home directories at the time of user creation.

You may create additional files in your home directories to set more environment variables or shell properties if required.

Table 7-6 lists and describes per-user startup files for bash shell users.

File	Comments
<i>.bashrc</i>	Defines functions and aliases. This file sources global definitions from the <i>/etc/bashrc</i> file.
<i>.bash_profile</i>	Sets environment variables and sources the <i>.bashrc</i> file.
<i>.gnome2/</i>	Directory that holds environment settings when GNOME desktop is started. Only available if GNOME is installed.

Table 7-6 Per-user Startup Files

Excerpts from the *.bashrc* and *.bash_profile* files are exhibited below:

```
[user1@server1 ~]$ cat .bashrc
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# User specific environment
PATH="$HOME/.local/bin:$HOME/bin:$PATH"
export PATH

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

[user1@server1 ~]$ cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs
```

The order in which the system-wide and per-user startup files are executed is important to grasp. The system runs the `/etc/profile` file first, followed by `.bash_profile`, `.bashrc`, and finally the `/etc/bashrc` file.

EXAM TIP: For persistence, per-user settings must be added to an appropriate file.

There is also a per-user file `.bash_logout` in the user's home directory. This file is executed when the user leaves the shell or logs off. This file may be customized as well.

Chapter Summary

In this chapter, we explored the bash shell, which has numerous features that are essential for users and administrators alike. We touched upon a few that are more prevalent. These features included variable settings, command prompt customization using substitution techniques, input/output/error redirections, history expansion, command line editing, filename completion, tilde substitution, and command aliasing.

We continued to examine additional bash shell features and expanded on metacharacters, wildcard characters, pipe symbol, and quoting mechanisms. We demonstrated an example of building a pipeline by incorporating multiple pipe characters between commands.

Finally, we analyzed various system-wide and per-user shell initialization scripts that are executed upon logging in to set a user environment. These

scripts may be customized for all users or individual users to suit specific needs.

Review Questions

1. You want to change the command prompt for yourself. Where (the bash shell file) would you define it so that you get the custom command prompt whenever you log in?
2. What is the other name for the command line completion feature?
3. What is a common use of the pipe symbol?
4. Which directory location stores most, if not all, privileged commands?
5. You have a file called “?” in your home directory along with other one-letter files. What would you run to delete the “?” file only?
6. What would the command `ls /cdr /usr > output` do? Assume `/cdr` does not exist.
7. What is the name of the command that allows a user to define shortcuts to lengthy commands?
8. Which file typically defines the history variables for all users?
9. You have a command running that is tied to your terminal window. You want to get the command prompt back without terminating the running command. Which key combination would you press to return to the command prompt?
10. Name the three quoting mechanisms?
11. What would the command `export VAR1="I passed RHCSA"` do?
12. Name the default filename and location where user command history is stored?
13. What is the primary function of the shell in Linux?
14. What would the command `cd ~user20` do if executed as `user10`?
15. Which command would you use to display all matching lines from a text file?
16. What would the command `ls > ls.out` do?

Answers to Review Questions

1. You can define it in the `~/.bash_profile` file.
2. The other name for the command line completion feature is tab completion.
3. A common use of the pipe character is to receive the output of one command and pass it along to the next command as an input.
4. Privileged commands are stored in `/usr/sbin` directory.
5. You can issue `rm !?` to remove the file.
6. The command provided will redirect `ls /usr` output to the specified file and `ls /cdr` (error) to the terminal.

7. The *alias* command.
8. */etc/profile* is the file where the history variables are defined for all users.
9. The Ctrl+z key combination will suspend the running program and give you access to the command prompt.
10. The three quoting mechanisms are backslash, single quotation mark, and double quotation mark.
11. The command provided will set an environment variable with the quoted string as its value.
12. The user command history is stored in *~/.bash_history* file.
13. The shell acts as an interface between a user and the system.
14. The command provided will allow *user10* to *cd* into *user20*'s home directory provided *user10* has proper access permissions.
15. The *grep* command.
16. The command provided will redirect the *ls* command output to the specified file.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 7-1: Customize the Command Prompt

As *user1* on *server1*, customize the primary shell prompt to display the information enclosed within the quotes “<*user1@server1* in */etc*>:” when this user switches into the */etc* directory. The prompt should always reflect the current directory path. Add this to the appropriate per-user startup file for permanence. (Hint: Command and Variable Substitutions).

Lab 7-2: Redirect the Standard Input, Output, and Error

As *user1* on *server1*, run the *ls* command on */etc*, */dvd*, and */var*. Have the output printed on the screen and the errors forwarded to file */tmp/ioerror*. Check both files after the command execution and analyze the results. (Hint: Input, Output, and Error Redirections).

Chapter 08

Linux Processes and Task Scheduling

This chapter describes the following major topics:

- Identify and display system and user executed processes
- View process states and priorities
- Examine and change process niceness and priority
- Understand signals and their use in controlling processes
- Review job scheduling
- Control who can schedule jobs
- Schedule and manage jobs using at
- Comprehend crontab and understand the syntax of crontables
- Schedule and manage jobs using cron
- Understand anacron and its function

RHCSA Objectives:

20. Identify CPU/memory intensive processes, adjust process priority with renice, and kill processes
21. Adjust process scheduling

40. Schedule tasks using at and cron

A process is any program, command, or application running on the system.

Every process has a unique numeric identifier and it is managed by the kernel through its entire lifespan. It may be viewed, listed, and monitored, and can be launched at a non-default priority based on the requirement and available computing resources. A process may also be re-prioritized while it is running. A process is in one of several states at any given time during its lifecycle. A process may be tied to the terminal window where it is initiated, or it may run on the system as a service.

There are plenty of signals that may be passed to a process to accomplish various actions. These actions include hard killing a process, soft terminating it, running it as a background job, bringing the background job back to the foreground, and forcing it to restart with the same identifier.

Job scheduling allows a user to schedule a command for a one-time or recurring execution in the future. A job is submitted and managed by authorized users only. All executed jobs are logged. Anacron is a service that automatically runs jobs that were missed while the system was down.

Processes and Priorities

A *process* is a unit for provisioning system resources. It is any program, application, or command that runs on the system. A process is created in memory when a program, application, or command is initiated. Processes are organized in a hierarchical fashion. Each process has a *parent process* (a.k.a. a *calling process*) that spawns it. A single parent process may have one or many *child processes* and passes many of its attributes to them at the time of their creation. Each process is assigned an exclusive identification number known as the *Process IDentifier* (PID), which is used by the kernel to manage and control the process through its lifecycle. When a process completes its lifespan or is terminated, this event is reported back to its parent process, and all the resources provisioned to it (cpu cycles, memory, etc.) are then freed and the PID is removed from the system.

Plenty of processes are spawned at system boot, many of which sit in the memory and wait for an event to trigger a request to use their services. These background system processes are called *daemons* and are critical to system operation.

Process States

A process changes its operating state multiple times during its lifecycle. Many factors, such as load on the processor, availability of free memory, priority of

the process, and response from other applications, affect how often a process jumps from one operating state to another. It may be in a non-running condition for a while or waiting for other process to feed it information so that it can continue to run.

There are five basic process states: *running*, *sleeping*, *waiting*, *stopped*, and *zombie*. Each process is in one state at any given time. See [Figure 8-1](#).

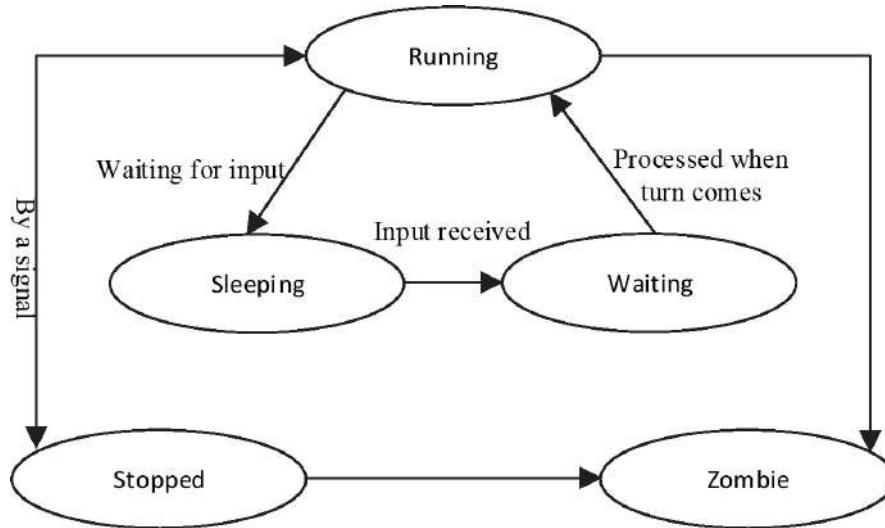


Figure 8-1 Process State Transition

Running: The process is being executed by the system CPU.

Sleeping: The process is waiting for input from a user or another process.

Waiting: The process has received the input it was waiting for and is now ready to run as soon as its turn comes.

Stopped: The process is currently halted and will not run even when its turn comes unless a signal is sent to change its behavior. (Signals are explained later in this chapter.)

Zombie: The process is dead. A zombie process exists in the process table alongside other process entries, but it takes up no resources. Its entry is retained until its parent process permits it to die. A zombie process is also called a *defunct* process.

Viewing and Monitoring Processes with ps

A system may have hundreds or thousands of processes running concurrently depending on the purpose of the system. These processes may be viewed and monitored using various native tools such as *ps* (*process status*) and *top* (*table of processes*). The *ps* command offers plentiful switches that influence its output, whereas *top* is used for real-time viewing and monitoring of processes and system resources.

Without any options or arguments, *ps* lists processes specific to the terminal where this command is issued:

```
[user1@server1 ~]$ ps
  PID TTY      TIME CMD
 1253 pts/0    00:00:00 ps
18359 pts/0    00:00:00 bash
```

The above output returns the elementary information about processes in four columns. These processes are tied to the current terminal window. It exhibits the PID, the terminal (TTY) the process spawned in, the cumulative time (TIME) the system CPU has given to the process, and the name of the actual command or program (CMD) being executed.

Some common options that can be used with the *ps* command to generate detailed reports include -e (every), -f (full-format), -F (extra full-format), and -l (long format). A combination of -e, -F, and -l (**ps -efl**) produces a very thorough process report, however, that much detail may not be needed in most situations. Other common options such as --forest and -x will report the output in tree-like hierarchy and include the daemon processes as well. Check the manual pages of the command for additional options and their usage.

Here are a few sample lines from the beginning and end of the output when *ps* is executed with -e and -f flags on the system.

```
[user1@server1 ~]$ ps -ef
UID      PID  PPID  C STIME TTY      TIME CMD
root      1      0  0 Sep14 ?        00:00:06 /usr/lib/systemd/systemd --switched-root
root      2      0  0 Sep14 ?        00:00:00 [kthreadd]
root      3      2  0 Sep14 ?        00:00:00 [rcu_gp]
root      4      2  0 Sep14 ?        00:00:00 [rcu_par_gp]
root      6      2  0 Sep14 ?        00:00:00 [kworker/0:0H-kblockd]
root      8      2  0 Sep14 ?        00:00:00 [mm_percpu_wq]
root      9      2  0 Sep14 ?        00:00:01 [ksoftirqd/0]
```

This output is disseminated across eight columns showing details about every process running on the system. [Table 8-1](#) describes the content type of each column.

Column	Description
UID	User ID or name of the process owner
PID	Process ID of the process
PPID	Process ID of the parent process
C	CPU utilization for the process
STIME	Process start date or time
TTY	The controlling terminal the process was started on. “Con” represents the system console and “?” represents a daemon process.
TIME	Aggregated execution time for a process
CMD	The command or program name

Table 8-1 ps Command Output Description

The *ps* output above pinpoints several daemon processes running in the background. These processes are not associated with any terminal, which is why there is a ? in the TTY column. Notice the PID and PPID numbers. The smaller the number, the earlier it is started. The process with PID 0 is started first at system boot, followed by the process with PID 1, and so on. Each PID has an associated PPID in column 3. The owner of each process is exposed in the UID column along with the name of the command or program under CMD.

Information for each running process is recorded and maintained in the */proc* file system, which *ps* and many other commands reference to acquire desired data for viewing.

The *ps* command output may be customized to view only the desired columns. For instance, if you want to produce an output with the command name in column 1, PID in column 2, PPID in column 3, and owner name in column 4, run it as follows:

```
[user1@server1 ~]$ ps -o comm,pid,ppid,user
COMMAND      PID  PPID USER
ps          1327 18359 user1
bash        18359 18354 user1
```

Make sure the *-o* option is specified for a user-defined format and there is no white space before or after the column names. You can add or remove columns and switch their positions as needed.

Another switch to look at with the *ps* command is *-C* (command list). This option is used to list only those processes that match the specified command

name. For example, run it to check how many *sshd* processes are currently running on the system:

```
[user1@server1 ~]$ ps -C sshd
  PID TTY      TIME CMD
    877 ?        00:00:00 sshd
  17771 ?        00:00:00 sshd
  17790 ?        00:00:00 sshd
  18101 ?        00:00:00 sshd
  18105 ?        00:00:00 sshd
  18350 ?        00:00:01 sshd
  18354 ?        00:00:02 sshd
```

The output exhibits multiple background *sshd* processes.

Viewing and Monitoring Processes with *top*

The other popular tool for viewing process information is the *top* command. This command displays statistics in real time and continuously, and may be helpful in identifying possible performance issues on the system. A sample of a running *top* session is shown below:

```
[user1@server1 ~]$ top
top - 15:50:11 up 4 days, 19:27,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 153 total,   2 running, 151 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.0 us,  0.3 sy,  0.0 ni, 98.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  821.4 total,     7.6 free,   433.7 used,   300.1 buff/cache
MiB Swap: 1024.0 total, 1014.2 free,     9.8 used.   247.0 avail Mem

PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
1029 gdm      20   0 2381764 170824  81688 S  1.0 20.3 4:56.46 gnome-shell
1387 user1    20   0   63876   4380   3752 R  0.7  0.5  0:00.03 top
18354 user1   20   0 167968   5940   4640 S  0.3  0.7  0:02.08 sshd
  1 root      20   0 244596 13076   8924 S  0.0  1.6  0:06.31 systemd
  2 root      20   0     0     0     0 S  0.0  0.0  0:00.05 kthreadd
  3 root      0 -20     0     0     0 I  0.0  0.0  0:00.00 rcu_gp
  4 root      0 -20     0     0     0 I  0.0  0.0  0:00.00 rcu_par_gp
  6 root      0 -20     0     0     0 I  0.0  0.0  0:00.00 kworker/0:0H-kblock+
  8 root      0 -20     0     0     0 I  0.0  0.0  0:00.00 mm_percpu_wq
  9 root      20   0     0     0     0 S  0.0  0.0  0:01.20 ksoftirqd/0
 10 root     20   0     0     0     0 R  0.0  0.0  0:01.55 rcu_sched
 11 root     rt   0     0     0     0 S  0.0  0.0  0:00.00 migration/0
 12 root     rt   0     0     0     0 S  0.0  0.0  0:00.20 watchdog/0
 13 root     20   0     0     0     0 S  0.0  0.0  0:00.00 cpuhp/0

Press q or Ctrl+c to quit.
```

The *top* output may be divided into two major portions: the summary portion and the tasks portion. The summary area spreads over the first five lines of the output, and it shows the information as follows:

Line 1: Indicates the system uptime, number of users logged in, and system load averages over the period of 1, 5, and 15 minutes. See the description for the *uptime* command output in [Chapter 02 “Initial Interaction with the System”](#).

Line 2: Displays the task (or process) information, which includes the total number of tasks running on the system and how many of them are in running, sleeping, stopped, and zombie states.

Line 3: Shows the processor usage that includes the CPU time in percentage spent in running user and system processes, in idling and waiting, and so on.

Line 4: Depicts memory utilization that includes the total amount of memory allocated to the system, and how much of it is free, in use, and allocated for use in buffering and caching.

Line 5: Exhibits swap (virtual memory) usage that includes the total amount of swap allocated to the system, and how much of it is free and in use. The “avail Mem” shows an estimate of the amount of memory available for starting new processes without using the swap.

The second major portion in the *top* command output showcases the details for each process in 12 columns as described below:

Columns 1 and 2: Pinpoint the process identifier (PID) and owner (USER)

Columns 3 and 4: Display the process priority (PR) and nice value (NI)

Columns 5 and 6: Depict amounts of virtual memory (VIRT) and non-swapped resident memory (RES) in use

Column 7: Shows the amount of shareable memory available to the process (SHR)

Column 8: Represents the process status (S)

Columns 9 and 10: Express the CPU (%CPU) and memory (%MEM) utilization

Column 11: Exhibits the CPU time in hundredths of a second (TIME+)

Column 12: Identifies the process name (COMMAND)

While in *top*, you can press “o” to re-sequence the process list, “f” to add or remove fields, “F” to select the field to sort on, and “h” to obtain help. *top* is highly customizable. See the command’s manual pages for details.

Listing a Specific Process

Though the tools discussed so far provide a lot of information about processes including their PIDs, Linux also offers the *pidof* and *pgrep* commands to list only the PID of a specific process. These commands have a few switches available to modify their behavior; however, their most elementary use is to pass a process name as an argument to view its PID. For instance, to list the PID of the *rsyslogd* daemon, use either of the following:

```
[user1@server1 ~]$ pidof rsyslogd  
1271  
[user1@server1 ~]$ pgrep rsyslogd  
1271
```

Both commands produce an identical result if used without an option.

Listing Processes by User and Group Ownership

A process can be listed by its ownership or owning group. You can use the *ps* command for this purpose. For example, to list all processes owned by *user1*, specify the -U (or -u) option with the command and then the username:

```
[user1@server1 ~]$ ps -U user1  
 PID TTY      TIME CMD  
1733 pts/0    00:00:00 ps  
17777 ?        00:00:00 systemd  
17784 ?        00:00:00 (sd-pam)  
17790 ?        00:00:00 sshd  
18105 ?        00:00:00 sshd  
18354 ?        00:00:03 sshd  
18359 pts/0    00:00:00 bash
```

The command lists the PID, TTY, TIME, and CMD name for all the processes owned by *user1*. You can specify the -G (or -g) option instead and the name of an owning group to print processes associated with that group only:

```
[user1@server1 ~]$ ps -G root  
 PID TTY      TIME CMD  
 1 ?        00:00:06 systemd  
 2 ?        00:00:00 kthreadd  
 3 ?        00:00:00 rcu_gp  
 4 ?        00:00:00 rcu_par_gp  
 6 ?        00:00:00 kworker/0:0H-kblockd
```

The above output reveals all the running processes with *root* as their owning group.

Understanding Process Niceness and Priority

Linux is a multitasking operating system. It runs numerous processes on a single processor core by giving each process a slice of time. The process scheduler on the system performs rapid switching of processes, giving the notion of concurrent execution of multiple processes.

A process is spawned at a certain priority, which is established at initiation based on a numeric value called *niceness* (a.k.a. a *nice* value). There are 40

niceness values, with -20 being the highest or the most favorable to the process, and +19 being the lowest or the least favorable to the process. Most system-started processes run at the default niceness of 0. A higher niceness lowers the execution priority of a process, and a lower niceness increases it. In other words, a process running at a higher priority gets more CPU attention. A child process inherits the niceness of its calling process in its priority calculation. Though programs are normally run at the default niceness, you can choose to initiate them at a different niceness to adjust their priority based on urgency, importance, or system load. As a normal user, you can only make your processes nicer, but the *root* user can raise or lower the niceness of any process.

RHEL provides the *nice* command to launch a program at a non-default priority and the *renice* command to alter the priority of a running program.

The default niceness can be viewed with the *nice* command as follows:

```
[user1@server1 ~]$ nice  
0
```

The *ps* command with the *-l* option along with the *-efl* options can be used to list priority (PRI, column 7) and niceness (NI, column 8) for all processes:

```
[user1@server1 ~]$ ps -efl  
F S UID      PID  PPID  C PRI  NI ADDR SZ WCHAN  STIME TTY          TIME CMD  
4 S root      1     0  80    0 - 61149 -   Sep14 ?          00:00:06 /usr/lib/syst  
1 S root      2     0  80    0 -     0 -   Sep14 ?          00:00:00 [kthreadd]  
1 I root      3     2  60 -20 -     0 -   Sep14 ?          00:00:00 [rcu_gp]  
1 I root      4     2  60 -20 -     0 -   Sep14 ?          00:00:00 [rcu_par_gp]  
1 I root      6     2  60 -20 -     0 -   Sep14 ?          00:00:00 [kworker/0:0H]  
.....
```

The output indicates that the first two processes are running at the default niceness of 0 and the next three at the highest niceness of -20. These values are used by the process scheduler to adjust the execution time of the processes on the CPU. The *ps* command maintains an internal mapping between niceness levels and priorities. A niceness of 0 (NI column) corresponds to priority 80 (PR column), and a niceness of -20 (NI column) maps to priority 60 (PR column).



In contrast to the niceness-priority mapping that the *ps* command uses, the *top* command displays it differently. For a 0-80 ps mapping, the *top* session will report it 0-20. Likewise, *ps*' (-20)-60 will be the same as the *top*'s (-20)-0.

Exercise 8-1: Start Processes at Non-Default Priorities

This exercise should be done on *server1* as *user1* with *sudo* where required.

You will need two terminal sessions to perform this exercise. Let's call them Terminal 1 and Terminal 2.

In this exercise, you will launch the *top* program three times and observe how the *ps* command reports the priority and niceness values. You will execute the program the first time at the default priority, the second time at a lower priority, and the third time at a higher priority. You will control which priority to run the program at by specifying a niceness value with the *nice* command. You will verify the new niceness and priority after each execution of the *top* session.

1. Run the *top* command at the default priority/niceness in Terminal 1:

```
[user1@server1 ~]$ top
```

2. Check the priority and niceness for the *top* command in Terminal 2 using the *ps* command:

```
[user1@server1 ~]$ ps -efl | grep top
0 S user1      2510  2494  0  80    0 - 13911 core_s 18:11 pts/1      00:00:00 top
```

The command reports the values as 80 and 0, which are the defaults.

3. Terminate the *top* session in Terminal 1 by pressing the letter *q* and relaunch it at a lower priority with a nice value of +2:

```
[user1@server1 ~]$ nice -n 2 top
```

4. Check the priority and niceness for the *top* command in Terminal 2 using the *ps* command:

```
[user1@server1 ~]$ ps -efl | grep top
0 S user1      2584  2494  0  82    2 - 13911 core s 18:23 pts/1      00:00:00 top
```

The command reports the new values as 82 and 2.

5. Terminate the *top* session in Terminal 1 by pressing the letter *q* and relaunch it at a higher priority with a nice value of -10. Use *sudo* for *root* privileges.

```
[user1@server1 ~]$ sudo nice -n -10 top
```

6. Check the priority and niceness for the *top* command in Terminal 2 using the *ps* command:

```
[user1@server1 ~]$ ps -efl | grep top
4 S root      2615  2613  0  70 -10 - 13911 -      18:30 pts/1      00:00:00 top
```

As you can see, the process is running at a higher priority (70) with a nice value of -10. Terminate the *top* session by pressing the letter *q*.

This concludes the exercise.

Exercise 8-2: Alter Process Priorities

This exercise should be done on *server1* as *user1* with *sudo* where required.

You will need two terminal sessions to perform this exercise. Let's call them Terminal 1 and Terminal 2.

In this exercise, you will launch the *top* program at the default priority and alter its priority without restarting it. You will set a new priority by specifying a niceness value with the *renice* command. You will verify the new niceness and priority after each execution of the *top* session.

1. Run the *top* command at the default priority/niceness in Terminal 1:

```
[user1@server1 ~]$ top
```

2. Check the priority and niceness for the *top* command in Terminal 2 using the *ps* command:

```
[user1@server1 ~]$ ps -efl | grep top
0 S user1      5759  5651  0  80  0 - 13912 core_s 07:17 pts/0    00:00:00 top
```

The command reports the values as 80 and 0, which are the defaults.

3. While the *top* session is running in Terminal 1, increase its priority by renicing it to -5. Use the command substitution to get the PID of *top*. Prepend the *renice* command by *sudo*.

```
[user1@server1 ~]$ sudo renice -n -5 $(pidof top)
5759 (process ID) old priority 0, new priority -5
```

The output indicates the old (0) and new (-5) priorities for the process.

4. Validate the above change with *ps*. Focus on columns 7 and 8.

```
[user1@server1 ~]$ ps -efl | grep top
0 S user1      5759  5651  0  75  -5 - 13912 core_s 07:17 pts/0    00:00:01 top
```

As you can see, the process is now running at a higher priority (75) with a nice value of -5.

5. Repeat the above but set the process to run at a lower priority by renicing it to 8:

```
[user1@server1 ~]$ sudo renice -n 8 $(pidof top)
5759 (process ID) old priority -5, new priority 8
```

The output indicates the old (-5) and new (8) priorities for the process.

6. Validate the above change with *ps*. Focus on columns 7 and 8.

```
[user1@server1 ~]$ ps -efl | grep top
0 S user1      5759  5651  0  88   8 - 13912 core_s 07:17 pts/0    00:00:02 top
```

The process is reported to now running at a lower priority (88) with niceness 8. This concludes the exercise.

Controlling Processes with Signals

A system may have hundreds or thousands of processes running on it. Sometimes it becomes necessary to alert a process of an event. This is done by sending a control signal to the process. Processes may use signals to alert each other as well. The receiving process halts its execution as soon as it gets the signal and takes an appropriate action as per the instructions enclosed in the signal. The instructions may include terminating the process gracefully, killing it abruptly, or forcing it to re-read its configuration.

There are plentiful signals available for use, but only a few are common. Each signal is associated with a unique numeric identifier, a name, and an action. A list of available signals can be viewed with the *kill* command using the *-l* option:

```
[user1@server1 ~]$ kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5) SIGTRAP
 6) SIGABRT     7) SIGBUS      8) SIGFPE      9) SIGKILL     10) SIGUSR1
 11) SIGSEGV    12) SIGUSR2    13) SIGPIPE     14) SIGALRM    15) SIGTERM
 16) SIGSTKFLT   17) SIGCHLD    18) SIGCONT     19) SIGSTOP     20) SIGTSTP
 21) SIGTTIN    22) SIGTTOU    23) SIGURG      24) SIGXCPU    25) SIGXFSZ
 26) SIGVTALRM   27) SIGPROF    28) SIGWINCH   29) SIGIO      30) SIGPWR
 31) SIGSYS     34) SIGRTMIN   35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
 38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
 43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
 48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
 53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
 58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
 63) SIGRTMAX-1  64) SIGRTMAX
```

The output returns 64 signals available for process-to-process and user-to-process communication. [Table 8-2](#) describes the control signals that are most often used.

Signal Number	Signal Name	Action
1	SIGHUP	Hang up signal causes a process to disconnect from a closed terminal that it was tied to. Also instruct a running daemon to re-read its config without a restart.
2	SIGINT	The ^c (Ctrl+c) signal issued on the controlling terminal to interrupt the execution of a process.
9	SIGKILL	Terminates a process abruptly
15	SIGTERM	Sends a soft termination signal to stop a process orderly fashion. This is the default signal if no signal is specified with the command.
18	SIGCONT	Same as using the bg command to resume a process.
19	SIGSTOP	Same as using Ctrl+z to suspend a job
20	SIGTSTP	Same as using the fg command

Table 8-2 Control Signals

The commands used to pass a signal to a process are *kill* and *pkill*. These commands are usually used to terminate a process. Ordinary users can kill processes that they own, while the *root* user privilege is needed to kill any process on the system.

The *kill* command requires one or more PIDs, and the *pkill* command requires one or more process names to send a signal to. You can specify a non-default signal name or number with either utility.

Let's look at a few examples to understand the usage of these tools.

To pass the soft termination signal to the *crond* daemon, use either of the following:

```
[user1@server1 ~]$ sudo pkill crond
[user1@server1 ~]$ sudo kill $(pidof crond)
```

The *pidof* command in the above example is used to discover the PID of the *crond* process using command substitution and it is then passed to the *kill* command for termination. You may also use the *pgrep* command to determine the PID of a process, as demonstrated in the next example. Use **ps -ef | grep crond** to confirm the termination.

Using the *pkill* or *kill* command without specifying a signal name or number sends the default signal of 15 to the process. This signal may or not terminate

the process. Some processes ignore the soft termination signal as they might be in a waiting state. These processes may be ended forcefully using signal 9 in any of the following ways:

```
[user1@server1 ~]$ sudo pkill -9 crond  
[user1@server1 ~]$ sudo pkill -s SIGKILL crond  
[user1@server1 ~]$ sudo kill -9 $(pgrep crond)
```

You may run the *killall* command to terminate all processes that match a criterion. Here is how you can use this command to kill all *crond* processes (assuming there are many of them running):

```
[user1@server1 ~]$ sudo killall crond
```

There are plenty of options available with the *kill*, *killall*, *pkill*, *pgrep*, and *pidof* commands. Consult respective manual pages for more details.

Job Scheduling

Job scheduling allows a user to submit a command for execution at a specified time in the future. The execution of the command could be one time or periodic based on a pre-determined time schedule. A one-time execution may be scheduled for an activity that needs to be performed at a time of low system usage. One example of such an activity would be the execution of a lengthy shell program. In contrast, a recurring activity could include creating a compressed archive, trimming log files, monitoring the system, running a custom script, or removing unwanted files from the system.

Job scheduling and execution is taken care of by two service daemons: *atd* and *crond*. While *atd* manages the jobs scheduled to run one time in the future, *crond* is responsible for running jobs repetitively at pre-specified times. At startup, this daemon reads the schedules in files located in the */var/spool/cron* and */etc/cron.d* directories, and loads them in the memory for on-time execution. It scans these files at short intervals and updates the in-memory schedules to reflect any modifications. This daemon runs a job at its scheduled time only and does not entertain any missed jobs. In contrast, the *atd* daemon retries a missed job at the same time next day. For any additions or changes, neither daemon needs a restart.

Controlling User Access

By default, all users are allowed to schedule jobs using the *at* and *cron* services. However, this access may be controlled and restricted to specific users only. This can be done by listing users in the *allow* or *deny* file located in

the `/etc` directory for either service. These files are named `at.allow` and `at.deny` for the `at` service, and `cron.allow` and `cron.deny` for the `cron` service.

The syntax for the four files is identical. You only need to list usernames that are to be allowed or denied access to these scheduling tools. Each file takes one username per line. The `root` user is always permitted; it is affected neither by the existence or non-existence of these files, nor by the inclusion or exclusion of its entry in these files.

Table 8-3 shows various combinations and their impact on user access.

<code>at.allow / cron.allow</code>	<code>at.deny / cron.deny</code>	Impact
Exists, and contains user entries	Existence does not matter	All users listed in allow file are permitted
Exists, but is empty	Existence does not matter	No users are permitted
Does not exist	Exists, and contains user entries	All users, other than those in deny files, are permitted
Does not exist	Exists, but is empty	All users are permitted
Does not exist	Does not exist	No users are permitted

Table 8-3 User Access Restrictions to Scheduling Tools

By default, the `deny` files exist and are empty, and the `allow` files are non-existent. This opens up full access to using both tools for all users.

EXAM TIP: One username is entered per line entry in an appropriate allow or deny file.

The following message appears if an unauthorized user attempts to execute `at`:

```
You do not have permission to use at.
```

And the following warning is displayed for unauthorized access attempt to the `cron` service:

```
You (user1) are not allowed to use this program (crontab)
See crontab(1) for more information
```

To generate the denial messages, you need to place entries for *user1* in the *deny* files.

Scheduler Log File

All activities for *atd* and *crond* services are logged to the */var/log/cron* file. Information such as the time of activity, hostname, process name and PID, owner, and a message for each invocation is captured. The file also keeps track of other events for the *crond* service such as the service start time and any delays. A few sample entries from the log file are shown below:

```
[user1@server1 ~]$ sudo cat /var/log/cron
Sep 15 03:29:01 server1 run-parts[8206]: (/etc/cron.daily) finished logrotate
Sep 15 03:29:01 server1 run-parts[8206]: (/etc/cron.daily) starting rhsmcd
Sep 15 03:29:02 server1 run-parts[8206]: (/etc/cron.daily) finished rhsmcd
Sep 15 03:29:02 server1 anacron[7971]: Job `cron.daily' terminated
Sep 15 03:29:02 server1 anacron[7971]: Normal exit (1 job run)
Sep 15 04:01:01 server1 CROND[8501]: (root) CMD (run-parts /etc/cron.hourly)
Sep 15 04:01:01 server1 run-parts[8501]: (/etc/cron.hourly) starting Danacron
Sep 15 04:01:01 server1 run-parts[8501]: (/etc/cron.hourly) finished Danacron
```

The truncated output shows some past entries from the file. You need the *root* user privilege to be able to read the file content.

Using at

The *at* command is used to schedule a one-time execution of a program in the future. All submitted jobs are spooled in the */var/spool/at* directory and executed by the *atd* daemon at the specified time. Each submitted job will have a file created containing the settings for establishing the user's shell environment to ensure a successful execution. This file also includes the name of the command or program to be run. There is no need to restart the daemon after a job submission.

There are multiple ways and formats for expressing the time with *at*. Some examples are:

at 1:15am	(executes the task at the next 1:15 a.m.)
at noon	(executes the task at 12:00 p.m.)
at 23:45	(executes the task at 11:45 p.m.)
at midnight	(executes the task at 12:00 a.m.)
at 17:05 tomorrow	(executes the task at 5:05 p.m. on the next day)
at now + 5 hours	(executes the task 5 hours from now. We can specify minutes, days, or weeks in place of hours)
at 3:00 10/15/20	(executes the task at 3:00 a.m. on October 15, 2020)

2020)



at assumes the current year and today's date if the year and date are not mentioned.

You may supply a filename with the *at* command using the -f option. The command will execute that file at the specified time. For instance, the following will run */home/user1/.bash_profile* file for *user1* 2 hours from now:

```
[user1@server1 ~]$ at -f ~/.bash_profile now + 2 hours
warning: commands will be executed using /bin/sh
job 4 at Fri Sep 20 10:31:00 2019
```

The above will be executed as scheduled and will have an entry placed for it in the log file.

Exercise 8-3: Submit, View, List, and Erase an at Job

This exercise should be done on *server1* as *user1*.

In this exercise, you will submit an at job as *user1* to run the *date* command at 11:30 p.m. on March 31, 2020, and have the output and any error messages generated redirected to the */tmp/date.out* file. You will list the submitted job, exhibit its contents for verification, and then remove the job.

1. Run the *at* command and specify the correct execution time and date for the job. Type the entire command at the first at> prompt and press Enter. Press Ctrl+d at the second at> prompt to complete the job submission and return to the shell prompt.

```
[user1@server1 ~]$ at 11:30pm 3/31/20
warning: commands will be executed using /bin/sh
at> date &> /tmp/date.out
at> <EOT>
job 5 at Tue Mar 31 23:30:00 2020
```

The system assigned job ID 5 to it, and the output also pinpoints the job's execution time.

2. List the job file created in the */var/spool/at* directory:

```
[user1@server1 ~]$ sudo ls -l /var/spool/at/
total 4
-rwx-----. 1 user1 user1 1449 Sep 20 08:39 a0000501934472
```

3. List the spooled job with the *at* command. You may alternatively use **atq** to list it.

```
[user1@server1 ~]$ at -l  
5      Tue Mar 31 23:30:00 2020 a user1
```

4. Display the contents of this file with the *at* command and specify the job ID:

```
[user1@server1 ~]$ at -c 5  
#!/bin/sh  
# atrun uid=1000 gid=1000  
# mail user1 0  
umask 2  
SSH_CONNECTION=192.168.0.219\ 53248\ 192.168.0.110\ 22; export SSH_CONNECTION  
LANG=en_US.UTF-8; export LANG  
HISTCONTROL=ignoredups; export HISTCONTROL  
HOSTNAME=server1.example.com; export HOSTNAME  
OLDPWD=/usr/share/dict; export OLDPWD  
XDG_SESSION_ID=15; export XDG_SESSION_ID  
USER=user1; export USER
```

5. Remove the spooled job with the *at* command by specifying its job ID. You may alternatively run **atrm 5** to delete it.

```
[user1@server1 ~]$ at -d 5
```

This should erase the job file from the */var/spool/at* directory. You can confirm the deletion by running **atq** or **at -l**.

Using crontab

Using the *crontab* command is the other method for scheduling tasks for running in the future. Unlike *atd*, *crond* executes cron jobs on a regular basis if they comply with the format defined in the */etc/crontab* file. Crontables (another name for crontab files) are located in the */var/spool/cron* directory. Each authorized user with a scheduled job has a file matching their login name in this directory.

For example, the crontable for *user1* would be */var/spool/cron/user1*. The other two locations where system crontables can be stored are the */etc/crontab* file and the */etc/cron.d* directory; however, only the *root* user is allowed to create, modify, and delete them. The *crond* daemon scans entries in the files at the three locations to determine job execution schedules. The daemon runs the commands or programs at the specified time and adds a log entry to the */var/log/cron* file for each invocation. There is no need to restart the daemon after submitting a new or modifying an existing cron job.

The *crontab* command is used to edit (-e), list (-l), and remove (-r) crontables. The -u option is also available for users who wish to modify a different user's crontable, provided they are allowed to do so and the other user is listed in the *cron.allow* file. The *root* user can also use the -u flag to alter other users'

crontables even if the affected users are not listed in the *allow* file. By default, crontab files are opened in the *vim* editor when the *crontab* command is issued to edit them.

Syntax of User Crontables

The */etc/crontab* file specifies the syntax that each user cron job must comply with in order for *crond* to interpret and execute it successfully. Based on this structure, each line in a user crontable with an entry for a scheduled job is comprised of six fields. Fields 1 to 5 are for the schedule, field 6 may contain the login name of the executing user, and the rest for the command or program to be executed. See [Figure 8-2](#) for the syntax.

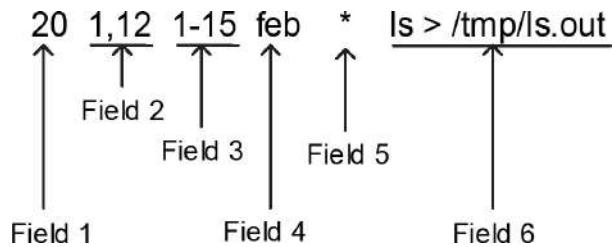


Figure 8-2 Syntax of Crontables

A description of each field is provided in [Table 8-4](#).

Field	Field Content	Description
1	Minute of the hour	Valid values are 0 (the exact hour) to 59. This field can have one specific value as in field 1, multiple comma-separated values as in field 3, a range of values as in field 4, or a mix of values as in field 5.
2	Hour of the day	Valid values are 0 (midnight) to 23. Same usage applies as described for field 1.
3	Day of the month	Valid values are 1 to 31. Same usage applies as described for field 1.
4	Month of the year	Valid values are 1 to 12 or jan to dec. Same usage applies as described for field 1.
5	Day of the week	Valid values are 0 to 7 or sun to sat, where 0 and 7 representing Sunday, 1 representing Monday, and so on. Same usage applies as described for field 1.
6	Command or program to execute	Specifies the full path name of the command or program to be executed, along with options or arguments that it requires.

Table 8-4 Crontable Syntax Explained

Furthermore, step values may be used with * and ranges in the crontables using the forward slash character (/). Step values allow the number of skips for a given value. For example, */2 in the minute field would mean every second minute, */3 would mean every third minute, 0-59/4 would mean every fourth minute, and so on. Step values are also supported in the same format in fields 2 to 5.

EXAM TIP: Make sure you understand and memorize the order of the fields defined in crontables.

Consult the manual pages of the *crontab* configuration file (**man 5 crontab**) for more details on the syntax.

Exercise 8-4: Add, List, and Erase a Cron Job

This exercise should be done on *server1* as *user1* with *sudo* where required.

For this exercise, assume that all users are currently denied access to cron.

In this exercise, you will submit a cron job as *user1* to echo “Hello, this is a cron test.”. You will schedule this command to execute at every fifth minute past the hour between 10:00 a.m. and 11:00 a.m. on the fifth and twentieth of every month. You will have the output redirected to the */tmp/hello.out* file, list the cron entry, and then remove it.

1. Edit the */etc/cron.allow* file and add *user1* to it:

```
[user1@server1 ~]$ sudo vim /etc/cron.allow
user1
```

2. Open the crontable and append the following schedule to it. Save the file when done and exit out of the editor.

```
[user1@server1 ~]$ crontab -e
*/5 10-11 5,20 * * echo "Hello, this is a cron test." > /tmp/hello.out
```

3. Check for the presence of a new file by the name *user1* under the */var/spool/cron* directory:

```
[user1@server1 ~]$ sudo ls -l /var/spool/cron/
total 4
-rw-----. 1 user1 user1 71 Sep 20 10:16 user1
```

4. List the contents of the crontable:

```
[user1@server1 ~]$ crontab -l
*/5 10-11 5,20 * * echo "Hello, this is a cron test." > /tmp/hello.out
```

5. Remove the crontable and confirm the deletion:

```
[user1@server1 ~]$ crontab -r
[user1@server1 ~]$ crontab -l
no crontab for user1
```

Do not run **crontab -r** if you do not wish to remove the crontab file. Instead, edit the file with **crontab -e** and just erase the entry.

Anacron

Anacron is a service that runs after every system reboot. It checks for any cron and at jobs that were scheduled for execution during the time the system was down and were missed as a result. Anacron proves to be useful on laptop, desktop, and similar purpose systems with extended periods of frequent downtimes and are not intended for 24/7 operations. Anacron scans the */etc/cron.hourly/0anacron* file for three factors to learn whether to run missed jobs. The three factors it examines are: (1) the presence of the */var/spool/anacron/cron.daily* file, (2) the elapsed time of 24 hours since it was

last run, and (3) if the system is plugged in to an AC source. If all three conditions are true, Anacron automatically executes the scripts located in the `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly` directories based on the settings and conditions defined in Anacron's own configuration file `/etc/anacrontab`. The default content of the `/etc/anacrontab` file are displayed below. It excludes the commented and empty lines.

```
[user1@server1 ~]$ cat /etc/anacrontab | grep -ve ^# -ve ^$  
SHELL=/bin/sh  
PATH=/sbin:/bin:/usr/sbin:/usr/bin  
MAILTO=root  
RANDOM_DELAY=45  
START_HOURS_RANGE=3-22  
1      5      cron.daily           nice run-parts /etc/cron.daily  
7      25     cron.weekly          nice run-parts /etc/cron.weekly  
@monthly 45    cron.monthly         nice run-parts /etc/cron.monthly
```

This file has five variables defined as depicted at the beginning of the above output. SHELL and PATH set the shell and path to be used for executing the programs. MAILTO defines the login name or an email of the user who is to be sent any output and error messages. RANDOM_DELAY expresses the maximum arbitrary delay in minutes added to the base delay of the jobs as defined in column 2 of the last three lines. START_HOURS_RANGE states the hour duration within which the missed jobs could be run.

The bottom three lines (each line with six columns) define the schedule and the programs to be executed:

Column 1: Denotes the period in days (or @daily, @weekly, @monthly, or @yearly), which Anacron uses to determine how often to run the specified job

Column 2: Specifies the delay in minutes after the system has been booted up for Anacron to wait before executing the job

Column 3: Contains a unique job identifier

Columns 4 to 6: Represent the command to be used to execute the scripts located under the `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly` directories. Here the `run-parts` command is invoked for execution at the default niceness.

For each job, Anacron examines whether the job was already run during the specified period (column 1). It executes it after waiting for the number of minutes (column 2) plus the RANDOM_DELAY value if it wasn't. When all missed jobs have been carried out and there is none pending, Anacron exits.

Anacron may be run manually at the command prompt. To run all the jobs that are scheduled in the `/etc/anacrontab` file but were missed, simply type the **anacron** command and press the Enter key.

Anacron stores job execution dates in files located in the `/var/spool/anacron` directory for each defined entry in the `/etc/anacrontab` file. There are several options available with the `anacron` command. Check the manual pages for further information.

Chapter Summary

This chapter discussed two major topics: process management and job scheduling.

It is vital for users and administrators alike to have a strong grasp on running processes, resources they are consuming, process owners, process execution priorities, etc. They should learn how to list processes in a variety of ways. We looked at the five states a process is in at any given time during its lifecycle. We examined the concepts of niceness and reniceness for increasing or decreasing a process's priority. We analyzed some of the many available signals and looked at how they could be passed to running processes to perform an action on them.

The second and the last topic talked about submitting and managing tasks to run in the future one time or on a recurring basis. We learned about the service daemons that handle the task execution and the control files where we list users who can or cannot submit jobs. We looked at the log file that stores information for all executed jobs. We reviewed the syntax of the crontable and examined a variety of date/time formats for use with both at and cron job submission. We performed two exercises to get a grasp on their usage. Finally, we studied the anacron service that RHEL uses to execute any scheduled jobs that were missed from running due to reasons such as system shutdown.

Review Questions

1. What are the two commands to list the PID of a specific process?
2. By default the `*.allow` files exist. True or False?
3. Where do the scheduling daemons store log information of executed jobs?
4. You must restart the `crond` service after modifying the `/etc/crontab` file. True or False?
5. What are the background service processes normally referred to in Linux?
6. What is the default nice value?
7. Which service runs missed scheduled tasks?
8. The parent process gets the nice value of its child process. True or False?

9. When would the *cron* daemon execute a job that is submitted as *10 * 2-6 6 */home/user1/script1.sh?
10. What is the other command besides *ps* to view running processes?
11. Every process that runs on the system has a unique identifier called UID. True or False?
12. Why would you use the *renice* command?
13. Which user does not have to be explicitly defined in either *.allow or *.deny file to run the at and cron jobs?
14. When would the *at* command execute a job that is submitted as at 01:00 12/12/2020?
15. What are the two commands that you can use to terminate a process?
16. What is the directory location where user crontab files are stored?
17. What would the *nice* command display without any options or arguments?
18. Which command can be used to edit crontables?
19. The default location to send application error messages is the system log file. True or False?
20. What are the five process states?
21. Signal 9 is used for a hard termination of a process. True or False?

Answers to Review Questions

1. The *pidof* and *pgrep* commands.
2. False. By default, the *.deny files exist.
3. The scheduling daemons store log information of executed jobs in the /var/log/cron file.
4. False. The *cron* daemon does not need a restart after a crontable is modified.
5. The background service processes are referred to as daemons.
6. The default nice value is zero.
7. The *anacron* service executes any missed at and cron jobs.
8. False. The child process inherits its parent's niceness.
9. The *cron* daemon will run the script every tenth minute past the hour on the 2nd, 3rd, 4th, 5th, and 6th day of every sixth month.
10. The *top* command.
11. False. It is called the PID.
12. The *renice* command is used to change the niceness of a running process.
13. The *root* user.
14. The *at* command will run it at 1am on December 12, 2020.
15. The *kill* and *pkill* commands.
16. The user crontab files are stored in the /var/spool/cron directory.

17. The *nice* command displays the default nice value when executed without any options.
18. You can use the *crontab* command with the *-e* option to edit crontables.
19. False. The default location is the user screen where the program is initiated.
20. The five process states are running, sleeping, waiting, stopped, and zombie.
21. True.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 8-1: Nice and Renice a Process

As *user1* with *sudo* on *server1*, open two terminal sessions. Run the *top* command in terminal 1. Run the *pgrep* or *ps* command in terminal 2 to determine the PID and the nice value of *top*. Stop *top* on terminal 1 and relaunch at a lower priority (+8). Confirm the new nice value of the process in terminal 2. Issue the *renice* command in terminal 2 and increase the priority of *top* to -10, and validate. (Hint: Processes and Priorities).

Lab 8-2: Configure a User Crontab File

As *user1* on *server1*, run the *tty* and *date* commands to determine the terminal file (assume */dev/pts/1*) and current system time. Create a cron entry to display “Hello World” on the terminal. Schedule **echo “Hello World” > /dev/tty/1** to run 3 minutes from the current system time. As *root*, ensure *user1* can schedule cron jobs. (Hint: Job Scheduling).

Chapter 09

Basic Package Management

This chapter describes the following major topics:

- Overview of Red Hat packages, naming, and management tools
- Package dependency and database
- Query, install, upgrade, freshen, overwrite, and remove packages
- Extract package files from installable package
- Validate package integrity and authenticity
- View GPG keys and verify package attributes
- Manage packages using the rpm command

RHCSA Objectives:

44. Install and update software packages from Red Hat Network, a remote repository, or from the local file system (part of this objective is also covered in Chapter 10)

The Red Hat software management system is known as RPM Package Manager (RPM). RPM also refers to one or more files that are packaged together in a special format and stored in files with the .rpm extension. These rpm files (also called rpms, rpm packages, or packages) are manipulated by the RPM package management system. Each package included in and available for RHEL is in this file format. Packages have meaningful names and contain necessary files, as well as metadata structures such as ownership, permissions, and directory location for each included file. Packages may be downloaded and saved locally or on a network share for quick access, and they may have dependencies over files or other packages. In other words, a package may require the presence of additional files, another package, or a group of packages in order to be installed successfully and operate properly. Once a package has been installed and its metadata information stored in a package database, each attempt to update the package updates its metadata information as well.

RHEL provides a powerful tool for the installation and administration of RPM packages. The rpm command is flexible and it offers multitude of options and subcommands to perform functions such as querying, installing, upgrading, freshening, removing, and decompressing packages, and validating package integrity and authenticity.

Package Overview

RHEL is essentially a set of packages grouped together to create an operating system. They are prepackaged for installation and assembled for various intended use cases. They are built around the Linux kernel and include thousands of packages that are digitally signed, tested, and certified. There are several basic and advanced concepts associated with packages, packaging, and their management that are touched upon in this chapter and [Chapter 10 “Advanced Package Management”](#).

Packages and Packaging

A software package is a group of files organized in a directory structure along with metadata and intelligence that make up a software application. They are available in two types: *binary* (or *installable*) and *source*. Binary packages are installation-ready and are bundled for distribution. They have .rpm extension and contain install scripts, pre- and post-installation scripts, executables, configuration files, library files, dependency information, where to install files, and documentation. The documentation includes detailed instructions on how

to install and uninstall the package, manual pages for the configuration files and commands, and other necessary information pertaining to the installation and usage of the package.

All metadata related to packages is stored at a central location and includes information such as package version, installation location, checksum values, and a list of included files with their attributes. This allows the package management toolset to handle package administration tasks efficiently by referencing this metadata.

The package intelligence is used by the package administration toolset for a successful completion of the package installation process. It may include information on prerequisites, user account setup (if required), and any directories and soft links that need to be created. The intelligence also includes the reverse of this process for uninstallation.

Source packages come with the original unmodified version of the software that may be unpacked, modified as desired, and repackaged in the binary format for installation or redistribution. They are identified with the .src extension.

Package Naming

Red Hat software packages follow a standard naming convention. Typically, there are five parts to a package name: (1) the package name, (2) the package version, (3) the package release (revision or build), (4) the Enterprise Linux the package is created for, and (5) the processor architecture the package is built for. An installable package name always has the .rpm extension; however, this extension is removed from the installed package name.

For example, if the name of an installable package is openssl-1.1.1-8.el8.x86_64.rpm, its installed name would be openssl-1.1.1-8.el8.x86_64. Here is a description of each part of the package name:

openssl: package name

1.1.1: version

8: release

el8: stands for *Enterprise Linux 8* (not all packages have it)

x86_64: processor architecture the package is created for. You may see “noarch” for platform-independent

packages that can be installed on any hardware architecture, or “src” for source code packages.

.rpm: the extension

Package Dependency

An installable package may require the presence of one or more additional packages in order to be installed successfully. Likewise, a software package component may require the functionality provided by one or more packages to exist in order to operate as expected. This is referred to as *package dependency*, where one package depends on one or more other packages for installation or execution. Package dependency information is recorded in each package’s metadata from where it is read by package handling utilities.

Package Database

Metadata for installed packages and package files is stored and maintained in the `/var/lib/rpm` directory. This directory location is referred to as the *package database*, and it is referenced by package manipulation utilities to obtain package name and version data and information about ownerships, permissions, timestamps, and sizes for each and every file that is part of the package. The package database also contains information on dependencies. All this data aids management commands in listing and querying packages, verifying dependencies and file attributes, installing new packages, upgrading and uninstalling existing packages, and carrying out other package handling tasks.

The package database does not update existing package information by simply adding available enhancements. It removes the metadata of the package being replaced and then adds the information of the replacement package. In RHEL 8, it can maintain multiple versions of a single package alongside their metadata.

Package Management Tools

The primary tool for package management on Red Hat Enterprise Linux is called *rpm* (*redhat package manager*). This tool offers abundant options for easy package handling; however, a major caveat is that it does not automatically resolve package dependencies. To overcome this gap, a more innovative tool called *yum* (*yellowdog update, modified*) was introduced, which offered an easier method for package management that can find, get, and install all required dependent packages automatically.

In RHEL 8, a major upcoming enhancement to *yum* has been introduced known as *dnf*. *dnf* is not an official acronym, but some documentation refers to it as *dandified yum*. You may still use the *yum* command; however, it is simply a soft link to *dnf*.

This chapter focuses on the use of the *rpm* command. [Chapter 10 “Advanced Package Management”](#) details the *dnf* command.

Package Management with *rpm*

The *rpm* command handles package management tasks including querying, installing, upgrading, freshening, overwriting, removing, extracting, validating, and verifying packages. As mentioned, this command has a major drawback as it does not have the ability to automatically satisfy package dependencies, which can be frustrating during software installation and upgrade. The *rpm* command works with both installed and installable packages.

The *rpm* Command

Before getting into the details, let’s look at some common *rpm* command options. [Table 9-1](#) describes query options in both short and long option formats. You may use either format.

Query Options	Description
-q (--query)	Queries and displays packages
-qa (--query --all)	Lists all installed packages
-qc (--query --configfiles)	Lists configuration files in a package
-qd (--query --docfiles)	Lists documentation files in a package
-qf (--query --file)	Exhibits what package a file comes from
-qi (--query --info)	Shows installed package information including version, size, installation date, signature, and description
-qip (--query --info --package)	Shows installable package information including version, size, installation date, signature, and description
-ql (--query --list)	Lists all files in a package
-qR (--query --requires)	Lists files and packages a package depends on (requires)
-q --whatprovides	Lists packages that provide the specified package or file
-q --whatrequires	Lists packages that require the specified package or file

Table 9-1 rpm Command Query Options

[Table 9-2](#) describes options related to package installation, removal, and verification. These options are also available in both short and long formats. You may use either format.

Install/Remove Options	Description
-e (--erase)	Removes a package
--force	Installs and replaces a package or file same version
-F (--freshen)	Upgrades an installed package
-h (--hash)	Shows installation progress with hash
-i (--install)	Installs a package
--import	Imports a public key
-K	Validates the signature and package
-U (--upgrade)	Upgrades an installed package or installs if not already installed
-v (--verbose) or -vv	Displays detailed information
-V (--verify)	Verifies the integrity of a package or files

Table 9-2 rpm Command Install/Remove/Verify Options

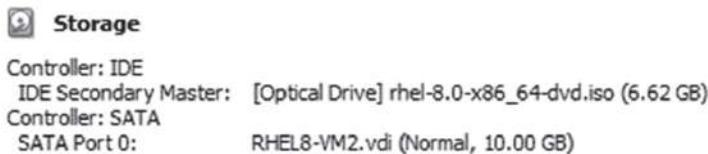
The examples in this chapter employ most of these options in short format to understand how they are used to achieve desired results.

Exercise 9-1: Mount RHEL 8 ISO Persistently

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will attach the RHEL 8 ISO image to the *RHEL8-VM1* in VirtualBox and mount the image on */mnt* directory on *server1* to access and manipulate the software packages in it with the *rpm* command. You will ensure that the image is automatically mounted on each system reboot. You will confirm access to the packages by listing the subdirectories that store them.

1. Go to the VirtualBox VM Manager and make sure that the RHEL 8 image is attached to *RHEL8-VM1* as depicted below:



2. Open the `/etc/fstab` file in the `vim` editor (or another editor of your choice) and add the following line entry at the end of the file to mount the DVD image (`/dev/sr0`) in read-only (ro) mode on the `/mnt` directory.

```
/dev/sr0 /mnt iso9660 ro 0 0
```

.....



sr0 represents the first instance of the optical device and iso9660 is the standard format for optical file systems.

3. Mount the file system as per the configuration defined in the `/etc/fstab` file using the `mount` command with the `-a` (all) option:

```
[user1@server1 ~]$ sudo mount -a  
[user1@server1 ~]$
```

The command should return without displaying anything in the output. This will confirm that the new entry was placed correctly in the file. See [Chapter 15](#) “Local File Systems and Swap” for details on mount and mount point concepts.

4. Verify the mount using the `df` command:

```
[user1@server1 ~]$ df -h | grep mnt  
/dev/sr0           6.7G  6.7G      0 100% /mnt
```

The image and the packages therein can now be accessed via the `/mnt` directory just like any other local directory on the system.

5. List the two directories—`/mnt/BaseOS/Packages` and `/mnt/AppStream/Packages`—that contain all the software packages (directory names are case sensitive):

```
[user1@server1 ~]$ ls -l /mnt/BaseOS/Packages |more  
total 922884  
-r--r--.  868 root root  151876 Dec 14  2018 aajohan-comfortaa-fonts-3.001-2  
.e18.noarch.rpm  
-r--r--.  303 root root   82976 Dec 14  2018 acl-2.2.53-1.e18.x86_64.rpm  
-r--r--.  222 root root 1022540 Dec 14  2018 acpica-tools-20180629-3.e18.x86  
.64.rpm  
-r--r--.  302 root root  104672 Oct 12  2018 adcli-0.8.2-2.e18.x86_64.rpm  
  
.....  
[user1@server1 ~]$ ls -l /mnt/AppStream/Packages |more  
total 5450689  
-r--r--.  165 root root  1977828 Feb  1  2019 389-ds-base-1.4.0.20-7.module+  
e18+2750+1f4079fb.x86_64.rpm  
-r--r--.  165 root root   357236 Feb  1  2019 389-ds-base-devel-1.4.0.20-7.m  
odule+e18+2750+1f4079fb.x86_64.rpm  
-r--r--.  165 root root   469808 Feb  1  2019 389-ds-base-legacy-tools-1.4.0  
.20-7.module+e18+2750+1f4079fb.x86_64.rpm  
.....
```

There are thousands of files in the two directories, each representing a single rpm package.

This concludes the exercise.

Querying Packages

You can query for packages in the package database or at the specified location. The following are some examples.

To query all installed packages:

```
[user1@server1 ~]$ rpm -qa
qemu-kvm-block-gluster-2.12.0-63.module+e18+2833+c7d6d092.x86_64
boost-atomic-1.66.0-6.e18.x86_64
gnome-session-wayland-session-3.28.1-6.e18.x86_64
grub2-tools-2.02-66.e18.x86_64
lohit-gurmukhi-fonts-2.91.2-3.e18.noarch
liberation-fonts-common-2.00.3-4.e18.noarch
....
```

To query whether the specified package is installed:

```
[user1@server1 ~]$ rpm -q perl
package perl is not installed
```

To list all files in a package:

```
[user1@server1 ~]$ rpm -ql iproute
/etc/iproute2
/etc/iproute2/bpf_pinning
/etc/iproute2/ematch_map
/etc/iproute2/group
....
```

To list only the documentation files in a package:

```
[user1@server1 ~]$ rpm -qd audit
/usr/share/doc/audit/ChangeLog
/usr/share/doc/audit/README
/usr/share/doc/audit/auditd.cron
/usr/share/doc/audit/rules/10-base-config.rules
/usr/share/doc/audit/rules/10-no-audit.rules
....
```

To list only the configuration files in a package:

```
[user1@server1 ~]$ rpm -qc cups
/etc/cups/classes.conf
/etc/cups/client.conf
/etc/cups/cups-files.conf
/etc/cups/cupsd.conf
/etc/cups/lpoptions
....
```

To identify which package owns the specified file:

```
[user1@server1 ~]$ rpm -qf /etc/passwd  
setup-2.12.2-1.e18.noarch
```

To display information about an installed package including version, release, installation status, installation date, size, signatures, description, and so on:

```
[user1@server1 ~]$ rpm -qi setup  
Name        : setup  
Version     : 2.12.2  
Release     : 1.e18  
Architecture: noarch  
Install Date: Fri 23 Aug 2019 06:06:31 AM EDT  
Group       : System Environment/Base  
Size        : 725081  
License     : Public Domain  
Signature   : RSA/SHA256, Mon 10 Sep 2018 08:25:55 AM EDT, Key ID 199e2f91fd431d  
51  
Source RPM  : setup-2.12.2-1.e18.src.rpm  
Build Date  : Mon 10 Sep 2018 08:17:59 AM EDT  
Build Host  : s390-001.fedoraproject.org  
Relocations : (not relocatable)  
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>  
Vendor      : Red Hat, Inc.  
URL         : https://pagure.io/setup/  
Summary     : A set of system configuration and setup files  
Description :  
The setup package contains a set of important system configuration and  
setup files, such as passwd, group, and profile.
```

To list all file and package dependencies for a given package:

```
[user1@server1 ~]$ rpm -qR chrony  
/bin/bash  
/bin/sh  
/bin/sh  
/bin/sh  
/bin/sh  
/bin/sh  
config(chrony) = 3.3-3.e18  
libc.so.6() (64bit)  
libc.so.6(GLIBC_2.12) (64bit)  
libc.so.6(GLIBC_2.14) (64bit)  
libc.so.6(GLIBC_2.15) (64bit)  
libc.so.6(GLIBC_2.17) (64bit)  
libc.so.6(GLIBC_2.2.5) (64bit)  
libc.so.6(GLIBC_2.25) (64bit)  
.....
```

To query an installable package for metadata information (version, release, architecture, description, size, signatures, etc.):

```
[user1@server1 ~]$ rpm -qip /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm
warning: /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm: Header V3 RSA/SHA256 S
ignature, key ID fd431d51: NOKEY
Name        : zsh
Version    : 5.5.1
Release    : 6.el8
Architecture: x86_64
Install Date: (not installed)
Group      : Unspecified
Size       : 7270070
License    : MIT
Signature  : RSA/SHA256, Mon 17 Dec 2018 12:38:12 PM EST, Key ID 199e2f91fd431d
51
Source RPM : zsh-5.5.1-6.el8.src.rpm
Build Date : Mon 17 Dec 2018 11:08:47 AM EST
Build Host : x86-vm-05.build.eng.bos.redhat.com
Relocations : (not relocatable)
Packager   : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Vendor    : Red Hat, Inc.
....
```

To determine what packages require the specified package in order to operate properly:

```
[user1@server1 ~]$ rpm -q --whatrequires lvm2
libblockdev-lvm-2.19-7.el8.x86_64
udisks2-lvm2-2.8.0-2.el8.x86_64
libvirt-daemon-storage-logical-4.5.0-23.module+el8+2800+2d311f65.x86_64
vdo-6.2.0-293-10.el8.x86_64
```

The above output lists all the packages that will require the specified package "lvm2" in order to work fully and properly.

Installing a Package

Installing a package creates the necessary directory structure for the package, installs the required files, and runs any post-installation steps. The following command installs a package called *zsh-5.5.1-6.el8.x86_64.rpm* on the system:

```
[user1@server1 ~]$ sudo rpm -ivh /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm
warning: /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm: Header V3 RSA/SHA256 S
ignature, key ID fd431d51: NOKEY
Verifying...          #####[100%]
Preparing...          #####[100%]
Updating / installing...
 1:zsh-5.5.1-6.el8  #####[100%]
```

If this package required the presence of any missing packages, you would see an error message related to failed dependencies. In that case, you would have to first install the missing packages for this package to be loaded successfully.

Upgrading a Package

Upgrading a package upgrades an installed version of the package. In the absence of an existing version, the upgrade simply installs the package.

To upgrade a package called *sushi*, use the **-U** option with the *rpm* command. Notice that the *sushi* package is located in a different directory than the *zsh* package in the previous example.

```
[user1@server1 ~]$ sudo rpm -Uvh /mnt/AppStream/Packages/sushi-3.28.3-1.el8.x86_64.rpm
warning: /mnt/AppStream/Packages/sushi-3.28.3-1.el8.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID fd431d51: NOKEY
Verifying... ###### [100%]
Preparing... ###### [100%]
package sushi-3.28.3-1.el8.x86_64 is already installed
```

The command makes a backup of all the affected configuration files during the upgrade process and adds the extension *.rpmsave* to them. In the above example, the *sushi* package was installed, as it was not already on the system.

Freshening a Package

Freshening a package requires that an older version of the package must already exist on the system.

To freshen the *sushi* package, use the **-F** option:

```
[user1@server1 ~]$ sudo rpm -Fvh /mnt/AppStream/Packages/sushi-3.28.3-1.el8.x86_64.rpm
warning: /mnt/AppStream/Packages/sushi-3.28.3-1.el8.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID fd431d51: NOKEY
```

The above command did nothing because the same package version specified with the command is already installed on the system. It will only work if a newer version of the installed package is available.

Overwriting a Package

Overwriting a package replaces the existing files associated with the package of the same version.

To overwrite the package *zsh-5.5.1-6.el8.x86_64* that was installed earlier, use the **--replacepkgs** option:

```
[user1@server1 ~]$ sudo rpm -ivh --replacepkgs /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm
warning: /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID fd431d51: NOKEY
Verifying... ###### [100%]
Preparing... ###### [100%]
Updating / installing...
 1:zsh-5.5.1-6.el8 ###### [100%]
```

The installation progress indicates that the `zsh` package was replaced successfully. This action is particularly useful when you suspect corruption in one or more installed package files and you want to start fresh.

Removing a Package

Removing a package uninstalls the package and all its associated files and the directory structure.

To remove the package `sushi`, use the `-e` option and specify `-v` for verbosity:

```
[user1@server1 ~]$ sudo rpm sushi -ve
Preparing packages...
sushi-3.28.3-1.el8.x86_64
```

This command performs a dependency inspection to check whether there are any packages that require the existence of the package being weeded out, and fails the removal if it detects a dependency.

Extracting Files from an Installable Package

Files in an installable package can be extracted using the `rpm2cpio` command for reasons such as examining the contents of the package, replacing a corrupted or lost command, or replacing a critical configuration file of an installed package to its original state.

Assuming you have lost the `/etc/chrony.conf` configuration file and want to retrieve it from its installable package and put it back, you'll first need to determine what package this file comes from:

```
[user1@server1 ~]$ rpm -qf /etc/chrony.conf
chrony-3.3-3.el8.x86_64
```

Now use the `rpm2cpio` command to extract (-i) all files from the “`chrony`” package and create (-d) the necessary directory structure during the retrieval. Extract the files in a temporary location such as the `/tmp` directory before you proceed with overwriting the destination under `/etc`.

```
[user1@server1 ~]$ cd /tmp  
[user1@server1 tmp]$ sudo rpm2cpio /mnt/BaseOS/Packages/chrony-3.3-3.el8.x86_64.rpm |  
cpio -imd  
1066 blocks
```

Run the *find* command to locate the *chrony.conf* file:

```
[user1@server1 tmp]$ sudo find . -name chrony.conf  
./etc/chrony.conf
```

The above output shows that the file is in the */tmp/etc* directory. You can copy it to the */etc* directory now, and you're back in business.

Validating Package Integrity and Credibility

Before it is installed, a package may be checked for integrity (completeness and error-free state) and credibility (authenticity) after it has been copied to another location, downloaded from the web, or obtained elsewhere. Use the MD5 checksum for verifying its integrity and the *GNU Privacy Guard* (GnuPG or GPG) public key signature for ensuring the credibility of its developer or publisher. This will ensure an uncorrupted and genuine piece of software.



The commercial version of GPG is referred to as PGP (Pretty Good Privacy).

To check the integrity of a package such as *zsh-5.5.1-6.el8.x86_64.rpm* located in */mnt/BaseOS/Packages*:

```
[user1@server1 ~]$ rpm -K /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm --nosignature  
/mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm: digests OK
```

The OK in the output confirms that the package is free of corruption.

Red Hat signs their products and updates with a GPG key, and includes necessary public keys in the products for verification. For RHEL, the keys are in files on the installation media and are copied to the */etc/pki/rpm-gpg/* directory during the OS installation. Refer to [Table 9-3](#) for a list of files in that directory and a short explanation.

PGP File	Description
RPM-GPG-KEY-redhat-release	Used for packages shipped after November 2009 and their updates
RPM-GPG-KEY-redhat-beta	Used for beta test products shipped before November 2009

Table 9-3 Red Hat GPG Key Files

To check the credibility of a package, import the relevant GPG key and then verify the package. The table above shows that the GPG file for the recent packages is *RPM-GPG-KEY-redhat-release*. In the following example, run the *rpm* command to import the GPG key from this file and verify the signature for the *zsh-5.5.1-6.el8.x86_64.rpm* package using the *-K* option with the command:

```
[user1@server1 ~]$ sudo rpmkeys --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release  
[user1@server1 ~]$ sudo rpmkeys -K /mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm  
/mnt/BaseOS/Packages/zsh-5.5.1-6.el8.x86_64.rpm: digests signatures OK
```

The OK validates the package signature and certifies the authenticity and integrity of the package.

Viewing GPG Keys

The GPG key imported in the previous subsection can be viewed with the *rpm* command. You can list the key and display its details as well. Run the command as follows to list the imported key:

```
[user1@server1 ~]$ rpm -q gpg-pubkey  
gpg-pubkey-fd431d51-4ae0493b  
gpg-pubkey-d4082792-5b32db75
```

The output suggests that there are two GPG public keys currently imported on the system. Let's view the details for the first one:

```
[user1@server1 ~]$ rpm -qi gpg-pubkey-fd431d51-4ae0493b
Name        : gpg-pubkey
Version     : fd431d51
Release     : 4ae0493b
Architecture: (none)
Install Date: Mon 23 Sep 2019 03:23:26 PM EDT
Group       : Public Keys
Size        : 0
License     : pubkey
Signature   : (none)
Source RPM  : (none)
Build Date  : Thu 22 Oct 2009 07:59:55 AM EDT
Build Host  : localhost
Relocations : (not relocatable)
Packager    : Red Hat, Inc. (release key 2) <security@redhat.com>
Summary     : gpg(Red Hat, Inc. (release key 2) <security@redhat.com>)
Description  :
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: rpm-4.14.2 (NSS-3)

mQINBERgSTsBEACh2A4b009t+vzC9VrVtL1AKvUUi90PCjkvR7Xd8DtJxeeM25eF
OHzIG58qDRybwUe89FZprB1ffuUKzdE+HcL3FbNWSS0XVjZIersdXyH3NvnLLF
...
JUfNh3DVRGhg8cMITi2InjiRh7gyFI20ccATY7bBSr79JhuNwe1HuxLrCFpY7V25
OFktl15jZJaMxuQBqYdBgSay2GOU6D1+7VsWufpzd/Abx1/c3oi9ZaJvW22kAgqq
dzdA27UUYjWvx42w9menJwh/0jeQcTecIUd0d0rFcw/clpvqMM1/Q73yzKgKYw==
=zbHE
-----END PGP PUBLIC KEY BLOCK-----
```

The output returns both the metadata and the key data for the specified GPG public key.

Verifying Package Attributes

Verifying the integrity of an installed package compares the attributes of files in the package with the original file attributes saved and stored in the package database at the time of package installation. The verification process uses the `rpm` command with the `-V` option to compare the owner, group, permission mode, size, modification time, digest, and type among other attributes. The command returns to the prompt without exhibiting anything if it detects no changes in the attributes. You can use the `-v` or `-vv` option with the command for increased verbosity.

Run this check on the `at` package:

```
[user1@server1 ~]$ sudo rpm -V at
```

The command returned nothing, which implies that the file attributes are intact. Now change the permissions on one of the files, `/etc/sysconfig/atd`, in this package to 770 from the current value of 644, and then re-execute the verification test:

```
[user1@server1 ~]$ ls -l /etc/sysconfig/atd
-rw-r--r--. 1 root root 403 Aug 12 2018 /etc/sysconfig/atd
[user1@server1 ~]$ sudo chmod -v 770 /etc/sysconfig/atd
mode of '/etc/sysconfig/atd' changed from 0644 (rw-r--r--) to 0770 (rwxrwx---)
[user1@server1 ~]$ sudo rpm -V at
.M..... c /etc/sysconfig/atd
```

The output is indicative of a change in the permission mode on the *atd* file in the *at* package. You may alternatively run the verification check directly on the file by adding the *-f* option to the command and passing the filename as an argument:

```
[user1@server1 ~]$ sudo rpm -Vf /etc/sysconfig/atd
.M..... c /etc/sysconfig/atd
```

The output returns three columns: column 1 contains nine fields, column 2 shows the file type, and column 3 expresses the full path of the file. The command performs a total of nine checks, as illustrated by the codes in column 1 of the output, and displays any changes that have occurred since the package that contains the file was installed. Each of these codes has a meaning. [Table 9-4](#) lists the codes with description as they appear from left to right. The period character (.) appears for an attribute that is not in an altered state.

Code	Description
S	Appears if the file size is different
M	Appears if the (mode) permission or file type is altered
5	Appears if MD5 checksum does not match
D	Appears if the file is a device file and its major or minor number changed
L	Appears if the file is a symlink and its path has altered
U	Appears if the ownership has modified
G	Appears if the group membership has modified
T	Appears if timestamp has changed
P	Appears if capabilities have altered
.	Appears if no modification is detected

Table 9-4 Package Verification Codes

Column 2 in the output above exposes a code that represents the type of file. [Table 9-5](#) lists them.

File Type	Description
c	Configuration file
d	Documentation file
g	Ghost file
l	License file
r	Readme file

Table 9-5 File Type Codes

Based on the information in the tables, the */etc/sysconfig/atd* is a configuration file with a modified permission mode. Reset the attribute to its previous value and rerun the check to ensure the file is back to its original state.

```
[user1@server1 ~]$ sudo chmod -v 644 /etc/sysconfig/atd
mode of '/etc/sysconfig/atd' changed from 0770 (rwxrwx---) to 0644 (rw-r--r--)
[user1@server1 ~]$ sudo rpm -V at
[user1@server1 ~]$
```

The command produced no output, which confirms the integrity of the file as well as the package.

Exercise 9-2: Perform Package Management Using rpm

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will verify the integrity and authenticity of a package called *drawing* located in the */mnt/AppStream/Packages* directory on the installation image and then install it. You will display basic information about the package, show files it contains, list documentation files, verify the package attributes, and erase the package.

1. Run the *ls* command on the */mnt/AppStream/Packages* directory to confirm that the *drawing* package is available:

```
[user1@server1 ~]$ ls -l /mnt/AppStream/Packages/drawing*
-rw-r--r--. 221 root root 273600 Dec 14 2018 /mnt/AppStream/Packages/drawing-9.27.0-9.el8.x86_64.rpm
```

2. Run the *rpm* command and verify the integrity and credibility of the package:

```
[user1@server1 ~]$ rpmkeys -K /mnt/AppStream/Packages/drawing-9.27.0-9.el8.x86_64.rpm
/mnt/AppStream/Packages/drawing-9.27.0-9.el8.x86_64.rpm: digests signatures OK
```

3. Install the package:

```
[user1@server1 ~]$ sudo rpm -ivh /mnt/AppStream/Packages/draw-9.27.0-9.el8.x86_64.rpm
Verifying...
Preparing...
Updating / installing...
 1:draw-9.27.0-9.el8

```

4. Show basic information about the package:

```
[user1@server1 ~]$ rpm -qi draw
Name        : draw
Version     : 9.27.0
Release     : 9.el8
Architecture: x86_64
Install Date: Mon 23 Sep 2019 10:31:43 PM EDT
Group       : Unspecified
Size        : 560675
License     : GPLv2+
Signature   : RSA/SHA256, Fri 14 Dec 2018 04:59:21 PM EST, Key ID 199e2f91fd431d51
Source RPM  : draw-9.27.0-9.el8.src.rpm
Build Date  : Sun 12 Aug 2018 05:19:30 AM EDT
Build Host  : x86-vm-01.build.eng.bos.redhat.com
Relocations : (not relocatable)
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Vendor      : Red Hat, Inc.
URL         : http://cybercom.net/~dc coffin/draw
Summary     : Tool for decoding raw image data from digital cameras
Description :
This package contains draw, a command line tool to decode raw image data
downloaded from digital cameras.
```

5. Show all the files the package contains:

```
[user1@server1 ~]$ rpm -ql draw
/usr/bin/draw
/usr/lib/.build-id
/usr/lib/.build-id/af
/usr/lib/.build-id/af/3c0f26d5d7cbb4d1210c45454d30c3b08d4d16
/usr/share/locale/ca/LC_MESSAGES/draw.mo
/usr/share/locale/cs/LC_MESSAGES/draw.mo
/usr/share/locale/da/LC_MESSAGES/draw.mo
/usr/share/locale/de/LC_MESSAGES/draw.mo
.
.
```

6. List the documentation files the package has:

```
[user1@server1 ~]$ rpm -qd draw
/usr/share/man/ca/man1/draw.1.gz
/usr/share/man/cs/man1/draw.1.gz
/usr/share/man/da/man1/draw.1.gz
/usr/share/man/de/man1/draw.1.gz
/usr/share/man/eo/man1/draw.1.gz
/usr/share/man/es/man1/draw.1.gz
/usr/share/man/fr/man1/draw.1.gz
/usr/share/man/hu/man1/draw.1.gz
/usr/share/man/it/man1/draw.1.gz
.
.
```

7. Verify the attributes of each file in the package. Use verbose mode.

```
[user1@server1 ~]$ rpm -Vv ddraw
..... /usr/bin/ddraw
..... a /usr/lib/.build-id
..... a /usr/lib/.build-id/af
..... a /usr/lib/.build-id/af/3c0f26d5d7chb4d1210c4
..... /usr/share/locale/ca/LC_MESSAGES/draw.mo
..... /usr/share/locale/cs/LC_MESSAGES/draw.mo
..... /usr/share/locale/da/LC_MESSAGES/draw.mo
..... /usr/share/locale/de/LC_MESSAGES/draw.mo
..... /usr/share/locale/eo/LC_MESSAGES/draw.mo
..... /usr/share/locale/es/LC_MESSAGES/draw.mo
..... /usr/share/locale/fr/LC_MESSAGES/draw.mo
```

.....

8. Remove the package:

```
[user1@server1 ~]$ sudo rpm -ve ddraw
Preparing packages...
ddraw-9.27.0-9.el8.x86_64
```

This concludes the exercise.

Chapter Summary

This chapter is the first of the two chapters (second one being [Chapter 10](#)) with coverage on software management. It covered the foundational topics and set the groundwork for more advanced features and functions.

We learned the concepts around packages, packaging, naming convention, dependency, and patch database. We looked at the variety of options available with the *rpm* utility to perform package administration tasks and put many of them into action to demonstrate their usage. Moreover, we employed appropriate options to view and validate package metadata information, GPG keys, and package attributes.

Review Questions

1. What would the *rpm -ql zsh* command do?
2. State the purpose of the *rpm2cpio* command.
3. What is the difference between freshening and upgrading a package?
4. The *rpm* command automatically takes care of package dependencies. True or False?
5. What is the difference between installing and upgrading a package?
6. Package database is located in the */var/lib/rpm* directory. True or False?
7. What would the *rpm -qf /bin/bash* command do?
8. Name the directory where RHEL 8 stores GPG signatures.
9. What would the options *ivh* cause the *rpm* command to do?
10. What would the *rpm -qa* command do?

Answers to Review Questions

1. The command provided will list all the files that are included in the installed *zsh* package.
2. The *rpm2cpio* command is used to extract files from the specified package.
3. Both are used to upgrade an existing package, but freshening requires an older version of the package to exist.
4. False.
5. Installing will install a new package whereas upgrading will upgrade an existing package or install it if it does not already exist.
6. True.
7. The command provided will display information about the */bin/bash* file.
8. RHEL 8 stores GPG signatures in the */etc/pki/rpm-gpg* directory.
9. The *rpm* command will install the specified package and show installation details and hash signs for progress.
10. The command provided will list all installed packages.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 9-1: Install and Verify Packages

As *user1* with *sudo* on *server1*, make sure the RHEL 8 ISO image is attached to the VM and mounted. Use the *rpm* command and install the *zsh* package by specifying its full path. Run the *rpm* command again and perform the following on the *zsh* package: (1) show information, (2) validate integrity, and (3) display attributes. (Hint: Package Management with *rpm*).

Lab 9-2: Query and Erase Packages

As *user1* with *sudo* on *server1*, make sure the RHEL 8 ISO image is attached to the VM and mounted. Use the *rpm* command to perform the following: (1) check whether the *setup* package is installed, (2) display the list of configuration files in the *setup* package, (3) show information for the *zlib-devel* package on the ISO image, (4) reinstall the *zsh* package (*--reinstall -vh*), and (5) remove the *zsh* package. (Hint: Package Management with *rpm*).

Chapter 10

Advanced Package Management

This chapter describes the following major topics:

- Describe package groups
- Understand application streams and modules
- Software repositories and how to access them
- Review module streams and module profiles
- Perform software management operations using dnf
- List, install, update, and delete individual packages, package groups, and modules
- Show package information, determine provider, and search metadata
- Exhibit package group and module information
- Install module from alternative stream

RHCSA Objectives:

44. Install and update software packages from Red Hat Network, a remote repository, or from the local file system (part of this objective is covered in Chapter 09)
45. Work with package module streams

This is the second of the two chapters that discusses software administration in RHEL. While the first chapter ([Chapter 09](#)) expounds upon the basic concepts of rpm package management and demonstrates a basic command to manipulate individual packages, this chapter elaborates on the concepts and handling of package groups, package modules, and package streams. It also presents a coverage on the concept of software repositories and how to configure them.

The dnf command (the yum command) is superior to the rpm command in the sense that it performs automatic dependency checks and marks any identified extra packages for installation. There are numerous options and subcommands available with this advanced tool and it is capable of interacting with software repositories as well.

Advanced Package Management Concepts

We discussed and grasped the basic package management concepts and employed the *rpm* tool in a variety of ways to manipulate individual packages. The focus of this chapter will be on understanding advanced software packaging and distribution techniques, and using a tool that is capable of handling single, discrete packages as well as collections of packages at a time, efficiently.

Package Groups

A *package group* is a collection of correlated packages designed to serve a common purpose. It provides the convenience of querying, installing, and deleting as a single unit rather than dealing with packages individually. There are two types of package groups: *environment groups* and *package groups*. The environment groups available in RHEL 8 are server, server with GUI, minimal install, workstation, virtualization host, and custom operating system. These are listed on the software selection window during RHEL 8 installation. The package groups include container management, smart card support, security tools, system tools, network servers, etc.

Application Streams and Modules

Application Streams is a new concept introduced in RHEL 8. It employs a modular approach to organize multiple versions of a software application

alongside its dependencies to be available for installation from a single repository. A *module* can be thought of as a logical set of application packages that includes everything required to install it, including the executables, libraries, documentation, tools, and utilities as well as dependent components. Modularity gives the flexibility to choose the version of software based on need.

In older RHEL releases, each version of a package would have to come from a separate repository. This has changed in RHEL 8. Now modules of a single application with different versions can be stored and made available for installation from a common repository. The package management tool has also been enhanced to manipulate modules.

RHEL 8 is shipped with two core repositories called *BaseOS* and *Application Stream* (AppStream).

BaseOS Repository

The BaseOS repository includes the core set of RHEL 8 components including the kernel, modules, bootloader, and other foundational software packages. These components lay the foundation to install and run software applications and programs. BaseOS repository components are available in the traditional rpm format.

AppStream Repository

The AppStream repository comes standard with core applications, as well as several add-on applications many of them in the traditional rpm format and some in the new modular format. These add-ons include web server software, development languages, database software, etc. and are shipped to support a variety of use cases and deployments.

Benefits of Segregation

There are two fundamental benefits to a segregation of the BaseOS components from other applications: (1) it separates the application components from the core operating system elements, and (2) it allows publishers to deliver and administrators to apply application updates more frequently. In previous RHEL versions, an OS update would update all installed components including the kernel, service, and application components to the latest versions by default. This could result in an unstable system or a misbehaving application due to an unwanted upgrade of one or more packages. By detaching the base OS components from the applications, either of the two can be updated independent of the other. This provides

enhanced flexibility in tailoring the system components and application workloads without impacting the underlying stability of the system.

Module Streams

A *module stream* is a collection of packages organized by version. Each module can have multiple streams, and each stream receives updates independent of the other streams. A stream can be enabled or disabled. An enabled stream allows the packages it contains to be queried or installed. Only one stream of a specific module can be enabled at a time. Each module has a default stream, which provides the latest or the recommended version. For example, the *perl* module provides two independent streams for the same software, one for version 5.24 and the other for 5.26. Stream 5.26 is the default, as it is a later version. The system will automatically attempt to install it if asked for *perl*.

Module Profiles

A *module profile* is a list of recommended packages organized for purpose-built, convenient deployments to support a variety of use cases such as minimal, development, common, client, server, etc. A profile may also include packages from the BaseOS repository or the dependencies of the stream. Each module stream can have zero, one, or more profiles associated with it with only one of them marked as the *default*.

dnf/yum Repository

A *dnf repository* (*yum repository* or a *repo*) is a digital library for storing software packages. A repository is accessed for package retrieval, query, update, and installation, and it may be free or for a fee. The two repositories—BaseOS and AppStream—come preconfigured with the RHEL 8 ISO image. There are a number of other repositories available on the Internet that are maintained by software publishers such as Red Hat and CentOS. Furthermore, you can build private custom repositories for internal IT use for stocking and delivering software. This may prove to be a good practice for an organization with a large Linux server base, as it manages dependencies automatically and aids in maintaining software consistency across the board. These repositories can also be used to store in-house developed packages.

It is important to obtain software packages from authentic and reliable sources such as Red Hat to prevent potential damage to your system and to circumvent possible software corruption.

There is a process to create repositories and to access preconfigured repositories. Creating repositories is beyond the scope of this book, but there

are two pre-set repositories available on the RHEL 8 image. You will configure access to them via a definition file to support the exercises and lab environment.

A sample repo definition file is shown below with some key directives:

```
[BaseOS_RHEL_8]
name= RHEL 8 base operating system components
baseurl=file:///mnt/BaseOS
enabled=1
gpgcheck=0
```

EXAM TIP: Knowing how to configure a dnf/yum repository using a URL plays an important role in completing some of the RHCSA exam tasks successfully. Use two forward slash characters (//) with the baseurl directive for an FTP, HTTP, or HTTPS source.

The above example shows five lines from a sample repo file. Line 1 defines an exclusive ID within the square brackets. Line 2 is a brief description of the repo with the “name” directive. Line 3 is the location of the repodata directory with the “baseurl” directive. Line 4 shows whether this repository is active. Line 5 shows if packages are to be GPG-checked for authenticity.

Each repository definition file must have a unique ID, a description, and a baseurl directive defined at a minimum; other directives are set as required. The baseurl directive for a local directory path is defined as file:///local_path (the first two forward slash characters represent the URL convention, and the third forward slash is for the absolute path to the destination directory), and for FTP and HTTP(S) sources as ftp://hostname/network_path and http(s)://hostname/network_path, respectively. The network path must include a resolvable hostname or an IP address.

Software Management with dnf

Software for enterprise Linux distributions such as Red Hat and CentOS is available in the rpm format. These distributions offer tools to work with individual packages as well as package groups and modules. The *rpm* command was used in the previous chapter to query, list, install, and erase packages, in addition to a few other tasks that it can perform. This command is limited to managing one package at a time.

A more capable tool available in RHEL for managing a single package, a group of packages, and a module is referred to as *dnf* (or *yum*). This tool has an associated configuration file that can define settings to control its behavior.

dnf Configuration File

The key configuration file for *dnf* is *dnf.conf* that resides in the */etc/dnf* directory. The “main” section in the file sets directives that have a global effect on *dnf* operations. You can define separate sections for each custom repository that you plan to set up on the system. However, the preferred location to store configuration for each custom repository in their own definition files is in the

/etc/yum.repos.d directory, which is the default location created for this purpose. The default content of this configuration file is listed below:

```
[user1@server1 ~]$ cat /etc/dnf/dnf.conf
[main]
gpgcheck=1
installonly_limit=3
clean_requirements_on_remove=True
best=True
```

[Table 10-1](#) explains the above and a few other directives that you may define in the file. The directives in [Table 10-1](#) are listed in an alphabetical order.

Directive	Description
best	Specifies whether to install (or upgrade) the latest available version
clean_requirements_on_remove	Defines whether to remove dependencies during a package removal process if no longer in use
debuglevel	Sets the level between 1 (minimum) and 10 (maximum) at which the debug information is recorded in the logfile. Default is 2. A value of 0 disables this feature.
gpgcheck	Indicates whether to check the GPG signature for package authenticity. A value of 1 (enabled).
installonly_limit	Specifies a count of packages that can be installed concurrently. Default is 3.
keepcache	Defines whether to store the package header cache following a successful installation. Default is 0 (disabled).
logdir	Sets the directory location to store log files. Default is /var/log.
obsoletes	Checks and removes any obsolete dependent packages during installations. Default is 1 (enabled).

Table 10-1 Directive Settings in dnf.conf File

There are a multitude of additional directives available that you may want to set in the main section of this file or in custom repository definition files. Run **man 5 dnf.conf** for details.

The *dnf* Command

The *dnf* package management tool introduced in RHEL 8 is the next upcoming major version of the *yum* package manager that had been around in RHEL for years. You can use either of the two interchangeably in RHEL 8. In fact, *yum* is simply a soft link to the *dnf* utility. The *dnf* command is used in the examples and exercises throughout this book. The *dnf* utility requires the system to have access to a local or remote software repository or to a local installable package file. The *Red Hat Subscription Management* (RHSM) service available in the Red Hat Customer Portal offers access to official Red Hat software repositories. There are other web-based repositories that host

packages that you may want to install and use. Alternatively, you can set up a local, custom repository on your system and add packages of your choice to it.

The primary benefit of using *dnf* over *rpm* is the command's ability to resolve dependencies automatically by identifying and installing any additional required packages for a successful installation of the specified software. With multiple repositories set up, *dnf* extracts the software from wherever it finds it.

dnf may be used to perform abundant software administration tasks. It invokes the *rpm* utility in the background. *dnf* can perform a number of operations on individual packages, package groups, and modules such as listing, querying, installing, and removing them, as well as enabling and disabling specific module streams.

[Table 10-2](#) summarizes the software handling tasks that *dnf* can perform on packages. It also lists two subcommands (clean and repolist) that are specific to repositories. The subcommands in [Table 10-2](#) are sequenced alphabetically. Refer to the manual pages of *dnf* for additional subcommands, operators, options, examples, and other details.

Subcommand	Description
check-update	Checks if updates are available for installed packages
clean	Removes cached data
history	Displays previous dnf activities as recorded in the /var/lib/dnf/history directory
info	Shows details for a package
install	Installs or updates a package
list	Lists installed and available packages
provides	Searches for packages that contain the specified file or feature
reinstall	Reinstalls the exact version of an installed package
remove	Removes a package and its dependencies
repolist	Lists enabled repositories
repoquery	Runs queries on available packages
search	Searches package metadata for the specified string
upgrade	Updates each installed package to the latest version

Table 10-2 dnf Subcommands for Packages and Repositories

[Table 10-3](#) lists and describes *dnf* subcommands that are intended for operations on package groups and modules.

Subcommand	Description
group install	Installs or updates a package group
group info	Returns details for a package group
group list	Lists available package groups
group remove	Removes a package group
module disable	Disables a module along with all the streams it contains
module enable	Enables a module along with all the streams it contains
module install	Installs a module profile including its packages
module info	Shows details for a module
module list	Lists all available module streams along with their profiles and status
module remove	Removes a module profile including its packages
module reset	Resets a module so that it is neither in enable nor in disable state
module update	Updates packages in a module profile

Table 10-3 dnf Subcommands for Package Groups and Modules

You will use most of the subcommands from [Table 10-2](#) and [Table 10-3](#) in the examples and exercises that follow, but first you'll need to create a definition file and configure access to the two repositories available on the RHEL 8 ISO image.

Exercise 10-1: Configure Access to Pre-Built Repositories

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will set up access to the two *dnf* repositories that are available on RHEL 8 image. You've already configured an automatic mounting of RHEL 8 image on */mnt* in [Chapter 09](#) “Basic Package Management”. You will create a definition file for the repositories and confirm.

1. Verify that the image is currently mounted:

```
[user1@server1 ~]$ df -h | grep mnt  
/dev/sr0           6.7G  6.7G      0 100% /mnt
```

2. Create a definition file called */local.repo* in the */etc/yum.repos.d* directory using the *vim* editor and define the following data for both repositories in it:

```
[BaseOS]  
name=BaseOS  
baseurl=file:///mnt/BaseOS  
gpgcheck=0  
  
[AppStream]  
name=AppStream  
baseurl=file:///mnt/AppStream  
gpgcheck=0
```

3. Confirm access to the repositories:

```
[user1@server1 ~]$ sudo dnf repolist  
Updating Subscription Management repositories.  
Unable to read consumer identity  
This system is not registered to Red Hat Subscription Management. You can use su  
bscription-manager to register.  
AppStream                         3.1 MB/s | 3.2 kB     00:00  
BaseOS                            2.7 MB/s | 2.7 kB     00:00  
repo id                           repo name                      status  
AppStream                          AppStream                     4,672  
BaseOS                            BaseOS                       1,658
```

Ignore lines 1-4 in the output that are related to subscription and system registration. Lines 5 and 6 show the rate at which the command read the repo data. Line 7 displays the timestamp of the last metadata check. The last two lines show the repo IDs, repo names, and a count of packages they hold. As depicted, the AppStream repo consists of 4,672 packages, and the BaseOS repo contains 1,658 packages. Both repos are enabled by default and are ready for use.

We have divided software administration into three sections to focus on individual packages, package groups, and modules separately.

Individual Package Management

The *dnf* command can be used to perform a variety of operations on individual packages, just like the *rpm* command. The following subsections elaborate with examples on using this tool to list, install, query, and remove packages.

Listing Available and Installed Packages

Listing the packages that are available for installation from one or more enabled repositories helps you understand what is in the current software inventory and what is needed. Likewise, listing the packages that are already installed on the system enables you to make important decisions as to whether they should be retained, upgraded, downgraded, or erased. The *dnf* command lists available packages as well as installed packages.

To list all packages available for installation from all enabled repos, run a query against the two repositories that we configured earlier, as those are the only ones you currently have access to. The following command will result in a long output.

```
[user1@server1 ~]$ sudo dnf repoquery
Last metadata expiration check: 0:04:03 ago on Tue 24 Sep 2019 11:17:10 PM EDT.
CUnit-0:2.1.3-17.el8.i686
CUnit-0:2.1.3-17.el8.x86_64
GConf2-0:3.2.6-22.el8.i686
GConf2-0:3.2.6-22.el8.x86_64
Judy-0:1.0.5-18.module+el8+2765+cfa4f87b.x86_64
LibRaw-0:0.19.1-1.el8.i686
LibRaw-0:0.19.1-1.el8.x86_64
ModemManager-0:1.8.0-1.el8.x86_64
ModemManager-glib-0:1.8.0-1.el8.i686
....
```

To limit the above to the list of packages that are available only from a specific repo:

```
[user1@server1 ~]$ sudo dnf repoquery --repo "BaseOS"
Last metadata expiration check: 0:06:17 ago on Tue 24 Sep 2019 11:17:10 PM EDT.
ModemManager-0:1.8.0-1.el8.x86_64
ModemManager-glib-0:1.8.0-1.el8.i686
ModemManager-glib-0:1.8.0-1.el8.x86_64
NetworkManager-1:1.14.0-14.el8.x86_64
NetworkManager-adsl-1:1.14.0-14.el8.x86_64
NetworkManager-bluetooth-1:1.14.0-14.el8.x86_64
NetworkManager-config-connectivity-redhat-1:1.14.0-14.el8.noarch
NetworkManager-config-server-1:1.14.0-14.el8.noarch
NetworkManager-dispatcher-routing-rules-1:1.14.0-14.el8.noarch
....
```

You can *grep* for an expression to narrow down your search. For example, to find whether the BaseOS repo includes the *zsh* package, run the following:

```
[user1@server1 ~]$ sudo dnf repoquery --repo BaseOS |grep zsh
Last metadata expiration check: 0:22:59 ago on Tue 24 Sep 2019 04:46:59 PM EDT.
zsh-0:5.5.1-6.el8.x86_64
```

To list all installed packages on the system:

```
[user1@server1 ~]$ sudo dnf list installed
Installed Packages
GConf2.x86_64                  3.2.6-22.el8          @AppStream
ModemManager.x86_64              1.8.0-1.el8           @anaconda
ModemManager-glib.x86_64         1.8.0-1.el8           @anaconda
NetworkManager.x86_64            1:1.14.0-14.el8      @anaconda
NetworkManager-adsl.x86_64       1:1.14.0-14.el8      @anaconda
NetworkManager-bluetooth.x86_64  1:1.14.0-14.el8      @anaconda
NetworkManager-config-server.noarch 1:1.14.0-14.el8      @anaconda
NetworkManager-libnm.x86_64       1:1.14.0-14.el8      @anaconda
NetworkManager-team.x86_64        1:1.14.0-14.el8      @anaconda
....
```

The graphic above shows the output in three columns: package name, package version, and the repo it was installed from. @anaconda means the package was installed at the time of RHEL installation.

To list all installed packages and all packages available for installation from all enabled repositories:

```
[user1@server1 ~]$ sudo dnf list
Installed Packages
GConf2.x86_64                  3.2.6-22.el8          @AppStream
ModemManager.x86_64              1.8.0-1.el8           @anaconda
ModemManager-glib.x86_64         1.8.0-1.el8           @anaconda
NetworkManager.x86_64            1:1.14.0-14.el8      @anaconda
NetworkManager-adsl.x86_64       1:1.14.0-14.el8      @anaconda
NetworkManager-bluetooth.x86_64  1:1.14.0-14.el8      @anaconda
.....
Available Packages
CUnit.i686                      2.1.3-17.el8          AppStream
CUnit.x86_64                     2.1.3-17.el8          AppStream
GConf2.i686                       3.2.6-22.el8          AppStream
Judy.x86_64                      1.0.5-18.module+e18+2765+cfa4f87b  AppStream
LibRaw.i686                       0.19.1-1.el8          AppStream
LibRaw.x86_64                     0.19.1-1.el8          AppStream
.....
```

The @ sign that precedes a repository name in column 3 identifies the package as installed.

To list all packages available from all enabled repositories that should be able to update:

```
[user1@server1 ~]$ sudo dnf list updates
```

To list whether a package (*bc*, for instance) is installed or available for installation from any enabled repository:

```
[user1@server1 ~]$ sudo dnf list bc
```

To list all installed packages whose names begin with the string “gnome” followed by any number of characters:

```
[user1@server1 ~]$ sudo dnf list installed gnome*
Installed Packages
gnome-autoar.x86_64                               0.2.3-1.el8      @AppStream
gnome-bluetooth.x86_64                             1:3.28.2-2.el8   @AppStream
gnome-bluetooth-libs.x86_64                         1:3.28.2-2.el8   @AppStream
gnome-boxes.x86_64                                3.28.5-3.el8    @AppStream
gnome-calculator.x86_64                            3.28.2-1.el8    @AppStream
....
```

To list recently added packages:

```
[user1@server1 ~]$ sudo dnf list recent
```

Refer to the *repoquery* and *list* subsections of the *dnf* command manual pages for more options and examples.

Installing and Updating Packages

Installing a package creates the necessary directory tree for the specified and dependent packages, installs the required files, and runs any post-installation steps. If the package being loaded is already present, the *dnf* command updates it to the latest available version. By default, *dnf* prompts for a yes or no confirmation unless the *-y* flag is entered with the command.

The following attempts to install a package called *ypbind*, but proceeds with an update if it detects the presence of an older version:

```
[user1@server1 ~]$ sudo dnf install ypbind
Dependencies resolved.
=====
 Package           Arch       Version            Repository      Size
 =====
 Installing:
  ypbind           x86_64     3:2.5-2.el8      AppStream      69 k
 Installing dependencies:
  yp-tools         x86_64     4.2.3-1.el8      AppStream      90 k
  nss_nis          x86_64     3.0-8.el8       BaseOS        41 k

Transaction Summary
=====
Install 3 Packages

Total size: 200 k
Installed size: 424 k
Is this ok [y/N]: Y
Downloading Packages:
```

```

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing      : 1/1
Installing   : nss_nis-3.0-8.el8.x86_64 1/3
Running scriptlet: nss_nis-3.0-8.el8.x86_64 1/3
Installing   : ypbind-3:2.5-2.el8.x86_64 2/3
Running scriptlet: ypbind-3:2.5-2.el8.x86_64 2/3
Installing   : yp-tools-4.2.3-1.el8.x86_64 3/3
Running scriptlet: yp-tools-4.2.3-1.el8.x86_64 3/3
Verifying     : yp-tools-4.2.3-1.el8.x86_64 1/3
Verifying     : ypbind-3:2.5-2.el8.x86_64 2/3
Verifying     : nss_nis-3.0-8.el8.x86_64 3/3
Installed products updated.

Installed:
  ypbind-3:2.5-2.el8.x86_64    yp-tools-4.2.3-1.el8.x86_64    nss_nis-3.0-8.el8.x86_64

Complete!

```

The above *dnf* command example resolved dependencies and showed a list of the packages that it would install. It exhibited the size of the packages and the amount of disk space that the installation would consume. It downloaded the packages after confirmation to proceed and installed them. It completed the installation after every package was verified. A list of the installed packages was displayed at the bottom of the output.

To install or update a package called *drawing* located locally at */mnt/AppStream/Packages*:

```
[user1@server1 ~]$ sudo dnf localinstall /mnt/AppStream/Packages/drawing-9.27.0-9.el8.x86_64.rpm
```

To update an installed package (*autofs*, for example) to the latest available version. Note that *dnf* will fail if the specified package is not already installed.

```
[user1@server1 ~]$ sudo dnf update autofs
Package autofs available, but not installed.
No match for argument: autofs
Error: No packages marked for upgrade.
```

To update all installed packages to the latest available versions:

```
[user1@server1 ~]$ sudo dnf -y update
```

Refer to the *install* and *update* subsections of the *dnf* command manual pages for more options and examples.

Exhibiting Package Information

Displaying information for a package shows its name, architecture it is built for, version, release, size, whether it is installed or available for installation, repo name it was installed or is available from, short and long descriptions, license, and so on. This information can be viewed by supplying the *info* subcommand to *dnf*.

To view information about a package called *autofs*:

```
[user1@server1 ~]$ dnf info autofs
Available Packages
Name        : autofs
Epoch       : 1
Version     : 5.1.4
Release     : 29.el8
Arch        : x86_64
Size        : 755 k
Source      : autofs-5.1.4-29.el8.src.rpm
Repo        : BaseOS
Summary     : A tool for automatically mounting and unmounting filesystems
License     : GPLv2+
Description : autofs is a daemon which automatically mounts filesystems when you use
              : them, and unmounts them later when you are not using them. This can
              : include network filesystems, CD-ROMs, floppies, and so forth.
```

The output shows that *autofs* is not currently installed, but it is available for installation from the BaseOS repo. The *info* subcommand automatically determines whether the specified package is installed or not.

Refer to the *info* subsection of the *dnf* command manual pages for more options available for viewing package information.

Removing Packages

Removing a package uninstalls it and removes all associated files and directory structure. It also erases any dependencies as part of the deletion process. By default, *dnf* prompts for a yes or no confirmation unless the *-y* flag is specified at the command line.

To remove a package called *ypbind*:

```
[user1@server1 ~]$ sudo dnf remove ypbnd
Dependencies resolved.
=====
Package           Arch      Version       Repository      Size
=====
Removing:
  ypbnd           x86_64    3:2.5-2.el8   @AppStream     100 k
Removing unused dependencies:
  nss_nis          x86_64    3.0-8.el8    @BaseOS        88 k
  yp-tools         x86_64    4.2.3-1.el8   @AppStream     236 k

Transaction Summary
=====
Remove 3 Packages

Freed space: 424 k
Is this ok [y/N]: y
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing :                                         1/1
  Running scriptlet: yp-tools-4.2.3-1.el8.x86_64      1/1
  Erasing   : yp-tools-4.2.3-1.el8.x86_64             1/3
  Running scriptlet: ypbnd-3:2.5-2.el8.x86_64        2/3
  Erasing   : ypbnd-3:2.5-2.el8.x86_64              2/3
  Running scriptlet: ypbnd-3:2.5-2.el8.x86_64        2/3
  Erasing   : nss_nis-3.0-8.el8.x86_64              3/3

  Running scriptlet: nss_nis-3.0-8.el8.x86_64        3/3
  Verifying  : nss_nis-3.0-8.el8.x86_64              1/3
  Verifying  : yp-tools-4.2.3-1.el8.x86_64            2/3
  Verifying  : ypbnd-3:2.5-2.el8.x86_64              3/3
Installed products updated.

Removed:
  ypbnd-3:2.5-2.el8.x86_64    nss_nis-3.0-8.el8.x86_64    yp-tools-4.2.3-1.el8.x86_64

Complete!
```

The above output resolved dependencies and showed a list of the packages that it would remove. It displayed the amount of disk space that their removal would free up. After confirmation to proceed, it erased the identified packages and verified their removal. A list of the removed packages was exhibited at the bottom of the output.

Refer to the *remove* subsection of the *dnf* command manual pages for more options and examples available for removing packages.

Exercise 10-2: Manipulate Individual Packages

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will perform management operations on a package called *cifs-utils*. You will determine if this package is already installed and if it is available for installation. You will display its information before installing it. You will install the package and exhibit its information. Finally, you will erase the package along with its dependencies and confirm the removal.

1. Check whether the *cifs-utils* package is already installed:

```
[user1@server1 ~]$ dnf list installed |grep cifs-utils  
[user1@server1 ~]$
```

The command returned to the prompt without any output, which implies that the specified package is not installed.

2. Determine if the *cifs-utils* package is available for installation:

```
[user1@server1 ~]$ dnf repoquery cifs-utils  
cifs-utils-0:6.8-2.el8.x86_64
```

The package is available for installation. The output shows its version as well.

3. Display detailed information about the package:

```
[user1@server1 ~]$ dnf info cifs-utils
```

```
Available Packages  
Name        : cifs-utils  
Version     : 6.8  
Release    : 2.el8  
Arch       : x86_64  
Size        : 93 k  
Source      : cifs-utils-6.8-2.el8.src.rpm  
Repo        : BaseOS  
Summary     : Utilities for mounting and managing CIFS mounts  
URL         : http://linux-cifs.samba.org/cifs-utils/  
License     : GPLv3  
Description : The SMB/CIFS protocol is a standard file sharing protocol widely  
              : deployed on Microsoft Windows machines. This package contains  
              : tools for mounting shares on Linux using the SMB/CIFS protocol.  
              : The tools in this package work in conjunction with support in the  
              : kernel to allow one to mount a SMB/CIFS share onto a client and  
              : use it as if it were a standard Linux file system.
```

The output is indicative of the fact that the package is available for installation from the BaseOS repo.

4. Install the package:

```
[user1@server1 ~]$ sudo dnf install -y cifs-utils
Dependencies resolved.
=====
 Package           Arch      Version       Repository      Size
=====
 Installing:
  cifs-utils       x86_64    6.8-2.el8   BaseOS        93 k

Transaction Summary
=====
 Install 1 Package

Total size: 93 k
Installed size: 189 k
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing          :                                         1/1
  Installing         : cifs-utils-6.8-2.el8.x86_64          1/1
  Running scriptlet: cifs-utils-6.8-2.el8.x86_64          1/1
  Verifying          : cifs-utils-6.8-2.el8.x86_64          1/1
Installed products updated.

Installed:
  cifs-utils-6.8-2.el8.x86_64

Complete!
```

5. Display the package information again:

```
[user1@server1 ~]$ dnf info cifs-utils
```

```
Installed Packages
Name        : cifs-utils
Version     : 6.8
Release     : 2.el8
Arch        : x86_64
Size        : 189 k
Source      : cifs-utils-6.8-2.el8.src.rpm
Repo        : @System
From repo   : BaseOS
```

The output reveals that the package is now installed.

6. Remove the package:

```
[user1@server1 ~]$ sudo dnf remove -y cifs-utils
Dependencies resolved.
=====
 Package           Arch      Version       Repository      Size
 =====
 Removing:
 cifs-utils       x86_64    6.8-2.el8     @BaseOS        189 k

 Transaction Summary
 =====
 Remove 1 Package

 Freed space: 189 k
 Running transaction check
 Transaction check succeeded.
 Running transaction test
 Transaction test succeeded.
 Running transaction
   Preparing :                                         1/1
   Running scriptlet: cifs-utils-6.8-2.el8.x86_64      1/1
   Erasing   : cifs-utils-6.8-2.el8.x86_64             1/1
   Running scriptlet: cifs-utils-6.8-2.el8.x86_64      1/1
   Verifying  : cifs-utils-6.8-2.el8.x86_64             1/1
 Installed products updated.

 Removed:
 cifs-utils-6.8-2.el8.x86_64

 Complete!
```

7. Confirm the removal:

```
[user1@server1 ~]$ dnf list installed | grep cifs-utils
[user1@server1 ~]$
```

The no output above confirms that the package is not loaded on the system.

Determining Provider and Searching Package Metadata

Determining package contents includes search operations on installed and available packages. For instance, you can determine what package a specific file belongs to or which package comprises a certain string. The following examples show how to carry out these tasks.

To search for packages that contain a specific file such as `/etc/passwd`, use the `provides` or the `whatprovides` subcommand with `dnf`.

```
[user1@server1 ~]$ dnf provides /etc/passwd
setup-2.12.2-1.el8.noarch : A set of system configuration and setup files
Repo      : @System
Matched from:
Filename  : /etc/passwd

setup-2.12.2-1.el8.noarch : A set of system configuration and setup files
Repo      : BaseOS
Matched from:
Filename  : /etc/passwd
```

The output returns two instances of the file. The first one indicates that the `passwd` file is part of a package called `setup`, which was installed during RHEL installation, and the second instance states that the `setup` package is part of the BaseOS repository.

With the `provides (whatprovides)` subcommand, you can also use a wildcard character for filename expansion. For example, the following command will list all packages that contain filenames beginning with “system-config” followed by any number of characters:

```
[user1@server1 ~]$ dnf whatprovides /usr/bin/system-config*
abrt-gui-2.10.9-10.el8.x86_64 : abrt's gui
Repo      : AppStream
Matched from:
Filename  : /usr/bin/system-config-abrt

policycoreutils-gui-2.8-16.1.el8.noarch : SELinux configuration GUI
Repo      : AppStream
Matched from:
Filename  : /usr/bin/system-config-selinux
```

To search for all the packages that match the specified string in their name or summary:

```
[user1@server1 ~]$ dnf search system-config
=====
Name Matched: system-config =====
system-config-printer-libs.noarch : Libraries and shared code for printer administration
                                  : tool
system-config-printer-udev.x86_64 : Rules for udev for automatic configuration of USB
                                  : printers
system-config-printer-libs.noarch : Libraries and shared code for printer administration
                                  : tool
system-config-printer-udev.x86_64 : Rules for udev for automatic configuration of USB
                                  : printers
=====
Summary Matched: system-config =====
cups-pk-helper.x86_64 : A helper that makes system-config-printer use PolicyKit
cups-pk-helper.x86_64 : A helper that makes system-config-printer use PolicyKit
```

The above outcome depicts six matches, four in package names and two in summaries.

Package Group Management

The `dnf` command can be used to perform ample operations on package groups by specifying the `group` subcommand with it. The following subsections elaborate with examples on using this tool to list, install, query, and remove groups of packages.

Listing Available and Installed Package Groups

The *group list* subcommand can be used with *dnf* to list the package groups available for installation from either or both repos, as well as to list the package groups that are already installed on the system.

To list all available and installed package groups from all repositories:

```
[user1@server1 ~]$ dnf group list
Available Environment Groups:
  Server
  Minimal Install
  Workstation
  Virtualization Host
  Custom Operating System
Installed Environment Groups:
  Server with GUI
Installed Groups:
  Container Management
  Headless Management
Available Groups:
  .NET Core Development
  RPM Development Tools
  Smart Card Support
  Development Tools
  Graphical Administration Tools
  Legacy UNIX Compatibility
  Network Servers
  Scientific Support
  Security Tools
  System Tools
```

The output reveals two categories of package groups: an *environment group* and a *group*. An environment group is a larger collection of RHEL packages that provides all necessary software to build the operating system foundation for a desired purpose. The Server with GUI environment group was selected at the time of installing *server1*, which installed a multitude of packages as well as the GNOME desktop. See [Table 1-2](#) in [Chapter 01](#) “Local Installation” for a description of other environment groups.

A group, on the other hand, is a small bunch of RHEL packages that serve a common purpose. It also saves time on the deployment of individual and dependent packages. The above output shows two installed and several available package groups.

To display the number of installed and available package groups:

```
[user1@server1 ~]$ sudo dnf group summary
Installed Groups: 2
Available Groups: 10
```

To list all installed and available package groups including those that are hidden:

```
[user1@server1 ~]$ sudo dnf group list hidden
Available Environment Groups:
  Server
  Minimal Install
  Workstation
  Virtualization Host
  Custom Operating System
Installed Environment Groups:
  Server with GUI
Installed Groups:
  Container Management
  Guest Desktop Agents
  Internet Browser
  Multimedia
  Printing Client
  Core
  Fonts
  GNOME
....
```

Try *group list* with --installed and --available options to narrow down the output list.

To list all packages that a specific package group such as *Base* contains:

```
[user1@server1 ~]$ sudo dnf group info Base
Group: Base
Description: The standard installation of Red Hat Enterprise Linux.
Mandatory Packages:
  acl
  at
  attr
  bc
  cpio
  crontabs
  cyrus-sasl-plain
  dbus
  ed
  file
  iptstate
  irqbalance
  kpatch
  logrotate
  lsof
....
```

You may use the -v option with the *group info* subcommand for more information. Refer to the *group list* and *group info* subsections of the *dnf* command manual pages for more details.

Installing and Updating Package Groups

Installing a package group creates the necessary directory structure for all the packages included in the group and all dependent packages, installs the required files, and runs any post-installation steps. If the package group is

being loaded or part of it is already present, the command attempts to update all the packages included in the group to the latest available versions. By default, *dnf* prompts for a yes or no confirmation unless the *-y* flag is entered with the command.

The following example attempts to install a package group called *smart card support*, but proceeds with an update if it detects the presence of an older version. You may enclose the group name within either single or double quotation marks.

```
[user1@server1 ~]$ sudo dnf -y groupinstall "smart card support"
Dependencies resolved.
=====
 Package           Arch      Version       Repository      Size
=====
Installing group/module packages:
  esc              x86_64    1.1.2-7.el8   AppStream      175 k
Installing dependencies:
  opensc           x86_64    0.19.0-4.el8  BaseOS        1.2 M
  pcsc-lite        x86_64    1.8.23-3.el8  BaseOS        108 k
  pcsc-lite-ccid   x86_64    1.4.29-3.el8  BaseOS        315 k
  pcsc-lite-libs   x86_64    1.8.23-3.el8  BaseOS        44 k
Installing Groups:
  Smart Card Support

Transaction Summary
=====
Install  5 Packages

Total size: 1.8 M
Installed size: 5.6 M

.....
Installed products updated.

Installed:
  esc-1.1.2-7.el8.x86_64          opensc-0.19.0-4.el8.x86_64
  pcsc-lite-1.8.23-3.el8.x86_64    pcsc-lite-ccid-1.4.29-3.el8.x86_64
  pcsc-lite-libs-1.8.23-3.el8.x86_64

Complete!
```

The output discloses that the command installed five packages located across the two repositories to complete the installation of the package group.

To update the *smart card support* package group to the latest version:

```
[user1@server1 ~]$ sudo dnf groupupdate "smart card support" -y
```

The above command will update all the packages within the package group to their latest versions if it has access to them. Refer to the *group install* and *group update* subsections of the *dnf* command manual pages for more details.

Removing Package Groups

Removing a package group uninstalls all the included packages and deletes all associated files and directory structure. It also erases any dependencies as part of the deletion process. By default, *dnf* prompts for a yes or no confirmation unless the *-y* flag is specified at the command line.

To erase the *smart card support* package group that was installed:

```
[user1@server1 ~]$ sudo dnf -y groupremove 'smart card support'
Dependencies resolved.
=====
Package           Arch    Version      Repository      Size
=====
Removing:
  esc              x86_64  1.1.2-7.el8   @AppStream    507 k
Removing unused dependencies:
  opensc          x86_64  0.19.0-4.el8  @BaseOS       3.6 M
  pcsc-lite       x86_64  1.8.23-3.el8  @BaseOS      224 k
  pcsc-lite-ccid  x86_64  1.4.29-3.el8  @BaseOS      1.4 M
  pcsc-lite-libs  x86_64  1.8.23-3.el8  @BaseOS      53 k
Removing Groups:
  Smart Card Support

Transaction Summary
=====
Remove 5 Packages

Freed space: 5.8 M
Running transaction check
Transaction check succeeded.

.....
Installed products updated.

Removed:
  esc-1.1.2-7.el8.x86_64          opensc-0.19.0-4.el8.x86_64
  pcsc-lite-1.8.23-3.el8.x86_64    pcsc-lite-ccid-1.4.29-3.el8.x86_64
  pcsc-lite-libs-1.8.23-3.el8.x86_64

Complete!
```

The above output resolved dependencies and showed a list of the packages that it would remove. It displayed the amount of disk space that their removal would free up. After confirmation to proceed, it erased the identified packages and verified their removal. A list of the removed packages was exposed at the bottom of the output.

Refer to the *remove* subsection of the *dnf* command manual pages for more details.

Exercise 10-3: Manipulate Package Groups

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will perform management operations on a package group called *system tools*. You will determine if this group is already installed and if it is available for installation. You will list the packages it contains and install it.

Finally, you will remove the group along with its dependencies and confirm the removal.

1. Check whether the *system tools* package group is already installed:

```
[user1@server1 ~]$ dnf group list installed
Installed Environment Groups:
  Server with GUI
Installed Groups:
  Container Management
  Headless Management
```

The output does not show the *system tools* group installed.

2. Determine if the *system tools* group is available for installation:

```
[user1@server1 ~]$ dnf group list available
Available Environment Groups:
  Server
  Minimal Install
  Workstation
  Virtualization Host
  Custom Operating System
Available Groups:
  .NET Core Development
  RPM Development Tools
  Smart Card Support
  Development Tools
  Graphical Administration Tools
  Legacy UNIX Compatibility
  Network Servers
  Scientific Support
  Security Tools
  System Tools
```

The group name is exhibited at the bottom of the list under the available groups.

3. Display the list of packages this group contains:

```
[user1@server1 ~]$ dnf group info 'system tools'
Group: System Tools
Description: This group is a collection of various tools for the system, such as the client for connecting to SMB shares and tools to monitor network traffic.
Default Packages:
  NetworkManager-libreswan
  chrony
  cifs-utils
  libreswan
  nmap
  openldap-clients
  samba-client
  setserial
  tigervnc
  tmux
  xdelta
  zsh
Optional Packages:
  PackageKit-command-not-found
  aide
.....
```

The output returns a long list of packages that are included in this group. All of the packages will be installed as part of the group installation.

4. Install the group:

```
[user1@server1 ~]$ sudo dnf group install 'system tools'
Dependencies resolved.
=====
 Package           Arch      Version       Repository     Size
=====
Installing group/module packages:
 NetworkManager-libreswan   x86_64    1.2.10-3.el8   AppStream    120 k
 libreswan            x86_64    3.27-9.el8   AppStream    1.3 M
 nmap                x86_64    2:7.70-4.el8   AppStream    5.8 M
 tigervnc             x86_64    1.9.0-9.el8   AppStream    276 k
 cifs-utils           x86_64    6.8-2.el8    BaseOS      93 k
 openldap-clients     x86_64    2.4.46-9.el8   BaseOS      202 k
 samba-client          x86_64    4.9.1-8.el8   BaseOS      636 k
 setserial             x86_64    2.17-45.el8   BaseOS      32 k
 tmux                 x86_64    2.7-1.el8    BaseOS      317 k
 xdelta               x86_64    3.1.0-4.el8   BaseOS      96 k
Installing dependencies:
 fltk                 x86_64    1.3.4-5.el8   AppStream    575 k
 ldns                 x86_64    1.7.0-20.el8  AppStream    165 k
 mesa-libGLU            x86_64    9.0.0-15.el8  AppStream    185 k
 nss-tools             x86_64    3.41.0-5.el8  AppStream    567 k
 tigervnc-icons        noarch   1.9.0-9.el8   AppStream    46 k
Installing Groups:
 System Tools

Transaction Summary
=====
Install 15 Packages

Total size: 10 M
Installed size: 39 M

.
.
.
Installed products updated.

Installed:
 NetworkManager-libreswan-1.2.10-3.el8.x86_64
 libreswan-3.27-9.el8.x86_64
 nmap-2:7.70-4.el8.x86_64
 tigervnc-1.9.0-9.el8.x86_64
 cifs-utils-6.8-2.el8.x86_64
 openldap-clients-2.4.46-9.el8.x86_64
 samba-client-4.9.1-8.el8.x86_64
 setserial-2.17-45.el8.x86_64
 tmux-2.7-1.el8.x86_64
 xdelta-3.1.0-4.el8.x86_64
 fltk-1.3.4-5.el8.x86_64
 ldns-1.7.0-20.el8.x86_64
 mesa-libGLU-9.0.0-15.el8.x86_64
 nss-tools-3.41.0-5.el8.x86_64
 tigervnc-icons-1.9.0-9.el8.noarch

Complete!
```

5. Remove the group:

```
[user1@server1 ~]$ sudo dnf group remove 'system tools' -y
```

```

Dependencies resolved.
=====
Package           Arch    Version      Repository   Size
=====
Removing:
NetworkManager-libreswan   x86_64  1.2.10-3.el8   @AppStream  426 k
cifs-utils          x86_64  6.8-2.el8    @BaseOS     189 k
libreswan           x86_64  3.27-9.el8   @AppStream  4.4 M
nmap                x86_64  2:7.70-4.el8  @AppStream  24 M
openldap-clients    x86_64  2.4.46-9.el8  @BaseOS     644 k
samba-client        x86_64  4.9.1-8.el8  @BaseOS     2.1 M
setserial           x86_64  2.17-45.el8  @BaseOS     37 k
tigervnc            x86_64  1.9.0-9.el8  @AppStream  858 k
tmux                x86_64  2.7-1.el8    @BaseOS     770 k
xdelta              x86_64  3.1.0-4.el8  @BaseOS     183 k
Removing unused dependencies:
fltk                x86_64  1.3.4-5.el8  @AppStream  1.6 M
ldns               x86_64  1.7.0-20.el8 @AppStream  428 k
mesa-libGLU          x86_64  9.0.0-15.el8 @AppStream  477 k
nss-tools           x86_64  3.41.0-5.el8 @AppStream  3.6 M
tigervnc-icons      noarch  1.9.0-9.el8  @AppStream  33 k
Removing Groups:
System Tools

Transaction Summary
=====
Remove 15 Packages

Freed space: 39 M

.....
Installed products updated.

Removed:
NetworkManager-libreswan-1.2.10-3.el8.x86_64  cifs-utils-6.8-2.el8.x86_64
libreswan-3.27-9.el8.x86_64                  nmap-2:7.70-4.el8.x86_64
openldap-clients-2.4.46-9.el8.x86_64        samba-client-4.9.1-8.el8.x86_64
setserial-2.17-45.el8.x86_64                 tigervnc-1.9.0-9.el8.x86_64
tmux-2.7-1.el8.x86_64                       xdelta-3.1.0-4.el8.x86_64
fltk-1.3.4-5.el8.x86_64                     ldns-1.7.0-20.el8.x86_64
mesa-libGLU-9.0.0-15.el8.x86_64             nss-tools-3.41.0-5.el8.x86_64
tigervnc-icons-1.9.0-9.el8.noarch         

Complete!

```

6. Confirm the removal:

```

[user1@server1 ~]$ dnf group list installed
Installed Environment Groups:
  Server with GUI
Installed Groups:
  Container Management
  Headless Management

```

The package group is not listed in the above output, which confirms its removal.

Module Management

Modules are a new concept introduced in RHEL 8, and the *yum/dnf* command has also been modified to handle them. The command can be used to

perform a number of operations on modules by specifying the *module* subcommand with it. The following subsections elaborate with examples on using this tool to list, enable, install, query, remove, and disable modules.

Listing Available and Installed Modules

The *dnf* command can list the modules available for installation from either or both repos, as well as the modules that are already installed on the system.

To list all modules along with their stream, profile, and summary information available from all configured repos:

```
[user1@server1 ~]$ dnf module list
Last metadata expiration check: 0:14:50 ago on Tue 24 Sep 2019 11:17:09 PM EDT.
AppStream
Name      Stream    Profiles      Summary
389-ds    1.4        common [d]  389 Directory Server (base)
ant       1.10 [d]   common [d]  Java build tool
container-tools 1.0        common [d]  Common tools and dependencies for container runtimes
container-tools rhe18 [d][e] common [d]  Common tools and dependencies for container runtimes
freeradius 3.0 [d]   server [d]  High-performance and highly configurable free RADIUS server
.....
Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

You may limit the output to a list of modules available from a specific repo such as AppStream by adding *--repo AppStream* to the above. The letters *d*, *e*, *x*, and *i* under the Stream and Profiles columns and at the last line of the output indicate their status as default (*d*), enabled (*e*), disabled (*x*), or installed (*i*).

To list all the streams for a specific module such as *perl* and display their status:

```
[user1@server1 ~]$ dnf module list perl
AppStream
Name  Stream    Profiles      Summary
perl  5.24     common [d], minimal  Practical Extraction and Report Language
perl  5.26 [d][e] common [d], minimal  Practical Extraction and Report Language
Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The above output reveals that there are two streams (5.24 and 5.26) available for the *perl* module from the AppStream repo, and the current default and enabled version is the latter of the two.

To modify the above and list only the specified stream (5.24) for the module *perl*:

```
[user1@server1 ~]$ dnf module list perl:5.24
AppStream
Name      Stream   Profiles          Summary
perl      5.24    common [d], minimal  Practical Extraction and Report Language

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

To list all enabled module streams:

```
[user1@server1 ~]$ dnf module list --enabled
AppStream
Name      Stream   Profiles          Summary
container-tools  rhel8 [d][e] common [d Common tools and dependencies for container runtimes
                           ] [i]
llvm-toolset     rhel8 [d][e] common [d LLVM
                           ]
perl           5.26 [d][e] common [d Practical Extraction and Report Language
                           ] [i], minimal
perl-DBD-SQLite 1.58 [d][e] common [d SQLite DBI driver
                           ]
perl-DBI        1.641 [d][e] common [d A database access API for Perl
                           ]
python36        3.6 [d][e] common [d Python programming language, version 3
                           ], build .6
satellite-5-client 1.0 [d][e] common [d Red Hat Satellite 5 client packages
                           ], gui
virt            rhel [d][e] common [d Virtualization module
                           ]

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

Similarly, you can use the `--installed` and `--disabled` options with **dnf module list** to output only the installed or the disabled streams.

Refer to the *module list* subsection of the *dnf* command manual pages for more details.

Installing and Updating Modules

Installing a module creates the necessary directory tree for all the packages included in the module and all dependent packages, installs the required files for the selected profile, and runs any post-installation steps. If the module being loaded or a part of it is already present, the command attempts to update all the packages included in the profile to the latest available versions. By default, *dnf* prompts for a yes or no confirmation unless the `-y` flag is entered with the command.

To install the *perl* module using its default stream and default profile:

```
[user1@server1 ~]$ sudo dnf -y module install perl
Dependencies resolved.
=====
 Package           Arch   Version      Repository  Size
=====
Installing group/module packages:
 perl              x86_64  4:5.26.3-416.el8    AppStream  72 k
Installing dependencies:
 dwz               x86_64  0.12-9.el8     AppStream  109 k
 efi-srpm-macros noarch  3-2.el8      AppStream  22 k
 ghc-srpm-macros noarch  1.4.2-7.el8    AppStream  9.4 k
 go-srpm-macros  noarch  2-16.el8     AppStream  14 k
 ocaml-srpm-macros noarch  5-4.el8     AppStream  9.5 k
 openblas-srpm-macros noarch  2-2.el8     AppStream  8.0 k
 perl-Algorithm-Diff noarch  1.1903-9.el8  AppStream  52 k
 perl-Archive-Tar  noarch  2.30-1.el8    AppStream  79 k
 perl-Archive-Zip  noarch  1.60-3.el8    AppStream  108 k
.....
redhat-rpm-config-116-1.el8.noarch
rust-srpm-macros-5-2.el8.noarch
systemtap-sdt-devel-4.0-7.el8.x86_64
make-1:4.2.1-9.el8.x86_64
python3-pyparsing-2.1.10-7.el8.noarch

Complete!
```

The default stream and the default profile are the same, which is 5.26, and the above command installed it.

To update a module called *squid* to the latest version:

```
[user1@server1 ~]$ sudo dnf module update squid -y
```

To install the profile “common” with stream “rhel8” for the *container-tools* module (run **dnf module list** to view available *container-tools* modules):

```
[user1@server1 ~]$ sudo dnf module install container-tools:rhel8/common
Dependencies resolved.
=====
 Package  Arch  Version      Repository  Size
=====
Installing group/module packages:
 oci-umount
 x86_64  2:2.3.4-2.git87f9237.module+e18+2769+577ad176  AppStream  38 k
 skopeo  x86_64 1:0.1.32-3.git1715c90.module+e18+2769+577ad176 AppStream 4.8 M
Installing module profiles:
 container-tools/common

Transaction Summary
=====
Install 2 Packages

Total size: 4.8 M
Installed size: 19 M
Is this ok [y/N]: y
Downloading Packages:
.....
```

The syntax of entering module, stream, and profile together (module:stream/profile) is illustrated on the above command line.

Displaying Module Information

Exhibiting information about a module shows its name, stream, version, list of profiles, default profile, repo name it was installed or is available from, summary, description, and artifacts. This information can be viewed by supplying *module info* with *dnf*. Let's take a look at the following examples.

To list all profiles available for the module *perl*:

```
[user1@server1 ~]$ dnf module info --profile perl

Name      : perl:5.24:820190207164249:ee766497:x86_64
common    : perl-core
minimal   : perl

Name      : perl:5.26:820181219174508:9edba152:x86_64
common    : perl
minimal   : perl-interpreter
```

To limit the output to a particular stream such as 5.26:

```
[user1@server1 ~]$ dnf module info --profile perl:5.26
Name      : perl:5.26:820181219174508:9edba152:x86_64
common    : perl
minimal   : perl-interpreter
```

To display details for a specific module stream (a non-default stream), specify the module stream with the module name:

```
[user1@server1 ~]$ dnf module info --profile perl:5.24
Name      : perl:5.24:820190207164249:ee766497:x86_64
common    : perl-core
minimal   : perl
```

Refer to the *module info* subsection of the *dnf* command manual pages for more details.

Removing Modules

Removing a module will uninstall all the included packages and delete all associated files and directory structure. It also erases any dependencies as part of the deletion process. By default, *dnf* prompts for a yes or no confirmation unless the *-y* flag is specified at the command line.

To remove the *container-tools* module with “rhel8” stream:

```
[user1@server1 ~]$ sudo dnf module remove container-tools:rhel8
Dependencies resolved.
=====
 Package Arch Version Repository Size
 =====
 Removing:
 oci-umount
 x86_64 2:2.3.4-2.git87f9237.module+el8+2769+577ad176 @AppStream 67 k
 skopeo x86_64 1:0.1.32-3.git1715c90.module+el8+2769+577ad176 @AppStream 19 M
 Removing module profiles:
 container-tools/common

Transaction Summary
=====
 Remove 2 Packages

Freed space: 19 M
Is this ok [y/N]: y
.....
Removed:
oci-umount-2:2.3.4-2.git87f9237.module+el8+2769+577ad176.x86_64
skopeo-1:0.1.32-3.git1715c90.module+el8+2769+577ad176.x86_64

Complete!
```

The above output resolved dependencies and showed a list of the packages that it would remove. It displayed the amount of disk space that their removal would free up. After confirmation to proceed, it erased the identified packages and verified their removal. A list of the removed packages was exposed at the bottom of the output.

Refer to the *module remove* subsection of the *dnf* command manual pages for more details.

Exercise 10-4: Manipulate Modules

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will perform management operations on a module called *postgresql*. You will determine if this module is already installed and if it is available for installation. You will show its information and install the default profile for stream “10”. Finally, you will remove the module profile along with any dependencies and confirm the removal.

1. Check whether the *postgresql* module is already installed:

```
[user1@server1 ~]$ dnf module list postgresql
AppStream
Name      Stream Profiles          Summary
postgresql 10 [d]  client, server [d] PostgreSQL server and client module
postgresql 9.6    client, server [d] PostgreSQL server and client module

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The *postgresql* module is not currently installed as there is no indication (i). The default stream is “10” and the default profile is “server”. The module with all its streams and profiles is available from the AppStream repository.

2. Display detailed information about the default stream of the module:

```
[user1@server1 ~]$ dnf module info postgresql:10
```

```
Name           : postgresql
Stream         : 10 [d][a]
Version        : 820190104140132
Context        : 9edbal52
Profiles       : client, server [d]
Default profiles: server
Repo           : AppStream
Summary         : PostgreSQL server and client module
Description     : PostgreSQL is an advanced Object-Relational database management system (DBMS). The postgresql-server package contains the programs needed to create and run a PostgreSQL server, which will in turn allow you to create and maintain PostgreSQL databases. The base postgresql package contains the client programs that you'll need to access a PostgreSQL DBMS server.
Artifacts       : postgresql-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-contrib-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-docs-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-plperl-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-plpython3-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-pltcl-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-server-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-server-devel-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-static-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-test-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-test-rpm-macros-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-upgrade-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
                  : postgresql-upgrade-devel-0:10.6-1.module+e18+2469+5ecd5aae.x86_64
x86_64          :
Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled, [a]ctive]
```

3. Install the module with default profile for stream “10”:

```
[user1@server1 ~]$ sudo dnf module install --profile postgresql:10 -y
Dependencies resolved.
=====
 Package      Arch    Version           Repository  Size
=====
 Installing group/module packages:
  postgresql-server  x86_64  10.6-1.module+e18+2469+5ecd5aae  AppStream  5.1 M
 Installing dependencies:
  libpq        x86_64  10.5-1.e18          AppStream  188 k
  postgresql   x86_64  10.6-1.module+e18+2469+5ecd5aae  AppStream  1.5 M
 Installing module profiles:
  postgresql/server
 Enabling module streams:
  postgresql          10
 .
 .
 .
 Installed:
  postgresql-server-10.6-1.module+e18+2469+5ecd5aae.x86_64
  libpq-10.5-1.e18.x86_64
  postgresql-10.6-1.module+e18+2469+5ecd5aae.x86_64

Complete!
```

4. Display the module information again:

```
[user1@server1 ~]$ dnf module info postgresql:10
Name          : postgresql
Stream        : 10 [d][e][a]
Version       : 820190104140132
Context       : 9edbal52
Profiles      : client, server [d] [i]
Default profiles : server
```

Line 2 from the bottom signifies that the default (d) profile “server” is now installed (i).

5. Erase the module profile for the stream:

```
[user1@server1 ~]$ sudo dnf module remove -y postgresql:10
Dependencies resolved.
=====
 Package           Arch    Version            Repository      Size
=====
 Removing:
  postgresql-server x86_64  10.6-1.module+el8+2469+5ecd5aae  @AppStream   20 M
 Removing unused dependencies:
  libpq             x86_64  10.5-1.el8                  @AppStream   702 k
  postgresql        x86_64  10.6-1.module+el8+2469+5ecd5aae  @AppStream   5.6 M
 Removing module profiles:
  postgresql/server

Transaction Summary
=====
 Remove  3 Packages

.....
Removed:
  postgresql-server-10.6-1.module+el8+2469+5ecd5aae.x86_64
  libpq-10.5-1.el8.x86_64
  postgresql-10.6-1.module+el8+2469+5ecd5aae.x86_64

Complete!
```

6. Confirm the removal:

```
[user1@server1 ~]$ dnf module info postgresql:10
Name          : postgresql
Stream        : 10 [d][e][a]
Version       : 820190104140132
Context       : 9edba152
Profiles      : client, server [d]
Default profiles : server
```

Line 2 from the bottom indicates the absence of (i) from the “server” profile, which means the profile is no longer installed.

Switching Module Streams

Switching module streams is typically performed to upgrade or downgrade the version of an installed module. The correct process for either operation is to uninstall the existing version provided by a stream alongside any dependencies that it has, switch to the other stream, and install the desired version. By default, installing a module from a stream automatically enables the stream if it was previously disabled, otherwise you can manually enable or disable it with the `dnf` command. You can have only one stream of a given module enabled at a time. Attempting to enable another one for the same module automatically disables the current enabled stream.

Earlier in this section, you executed `dnf module list` and `dnf module info` against multiple modules to list and view information about them. One of the properties they expose is the enable/disable status of the module stream.

[Exercise 10-5](#) will show you how to switch from one stream to another of a module and install a different version from there.

Exercise 10-5: Install a Module from an Alternative Stream

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will downgrade a module to a lower version. You will remove the stream *perl* 5.26 and confirm its removal. You will manually enable the stream *perl* 5.24 and confirm its new status. You will install the new version of the module and display its information.

1. Check the current state of all *perl* streams:

```
[user1@server1 ~]$ dnf module list perl
AppStream
Name Stream      Profiles          Summary
perl 5.24        common [d], minimal  Practical Extraction and Report Language
perl 5.26 [d][e]  common [d] [i], minima Practical Extraction and Report Language
1

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The above output reveals several interesting facts about the *perl* module and its available and installed streams and profiles. In columns 1 and 2, *perl* 5.26 is the default (d) stream and it is currently enabled (e). In column 3, the default (d) profile for this stream is “common” and it is currently installed (i). The other stream for this module is 5.24, which is also available for installation from the AppStream repository.

2. Let's remove *perl* 5.26:

```
[user1@server1 ~]$ sudo dnf module remove perl -y
Dependencies resolved.
=====
 Package           Arch   Version            Repository Size
 =====
 Removing:
 perl              x86_64  4:5.26.3-416.e18    @AppStream   0
 Removing unused dependencies:
 dwz               x86_64  0.12-9.e18       @AppStream  226 k
 efi-srpm-macros noarch  3-2.e18       @AppStream   38 k
 ....
```

```

Removing module profiles:
perl/common

Transaction Summary
=====
Remove 112 Packages

Freed space: 27 M
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing : 1/1
Erasing   : perl-4:5.26.3-416.el8.x86_64 1/112
.....

```

The module profile for perl stream 5.26 was removed successfully.

3. Confirm the removal:

```

[user1@server1 ~]$ dnf module list perl
AppStream
Name Stream Profiles Summary
perl 5.24 common [d], minimal Practical Extraction and Report Language
perl 5.26 [d][e] common [d], minimal Practical Extraction and Report Language

```

The install (i) indication in column 3 is now gone, which confirms the profile deletion.

4. Reset the module so that neither stream is enabled or disabled. This will remove the enabled (e) indication from perl 5.26.

```

[user1@server1 ~]$ sudo dnf module reset perl
Dependencies resolved.
=====
Package      Arch      Version      Repository      Size
=====
Resetting module streams:
perl          5.26

Transaction Summary
=====

Is this ok [y/N]: y
Complete!

```

5. Install the non-default profile “minimal” for perl stream 5.24. This will auto-enable the stream. The --allowrasing option will instruct the command to remove installed packages for dependency resolution.

```
[user1@server1 ~]$ sudo dnf module install perl:5.24/minimal --allowrasing
```

```
Modular dependency problems with Defaults:
```

```
Problem: conflicting requests
- module freeradius:3.0:820190131191847:fbe42456-0.x86_64 requires module(perl
:5.26), but none of the providers can be installed
- module perl:5.26:820181219174508:9edbal52-0.x86_64 conflicts with module(perl
:5.24) provided by perl:5.24:820190207164249:ee766497-0.x86_64
- module perl:5.24:820190207164249:ee766497-0.x86_64 conflicts with module(perl
:5.26) provided by perl:5.26:820181219174508:9edbal52-0.x86_64
Dependencies resolved.
=====
 Package          Arch    Version           Repository   Size
 =====
Upgrading:
perl-DBD-SQLite  x86_64  1.58-1.module+el8+2519+e351b2a7      AppStream  186 k
perl-DBI          x86_64  1.641-2.module+el8+2701+78cee6b5      AppStream  739 k
perl-Digest       noarch  1.17-395.module+el8+2464+d274aed1     AppStream   27 k
.....
Installing group/module packages:
perl              x86_64  4:5.24.4-403.module+el8+2770+c759b41a AppStream  6.1 M
Removing dependent packages:
perl-IO-Socket-SSL
noarch            2.060-2.el8                                     @AppStream 580 k
perl-Mozilla-CA  noarch  20160104-7.el8                      @AppStream 5.6 k
perl-Net-SSLeay   x86_64  1.85-6.el8                      @AppStream 1.2 M
Downgrading:
perl-Carp          noarch  1.40-366.module+el8+2464+d274aed1     AppStream   30 k
perl-Data-Dumper  x86_64  2.161-4.module+el8+2464+d274aed1     AppStream   57 k
perl-Encode        x86_64  4:2.88-6.module+el8+2464+d274aed1     AppStream  1.5 M
.....
Installing module profiles:
perl/minimal
Enabling module streams:
perl              5.24
Transaction Summary
=====
Install      1 Package
Upgrade     26 Packages
Remove      3 Packages
Downgrade   16 Packages

Total size: 12 M
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing      :                                                 1/1
Downgrading   : perl-libs-4:5.24.4-403.module+el8+2770+c759b41a.x8 1/88
Upgrading     : perl-Exporter-5.72-1000.module+el8+2464+d274aed1.n 2/88
.....
```

The above graphics are excerpts from a long output that the *dnf* command produced. The sequence of activities that occurred are: (1) dependency resolution, (2) a list of packages to be upgraded, installed, removed, and downgraded, along with their versions and sizes, (3) the module and profile to be installed, (4) the stream (5.24) to be enabled, (5) a summary of the entire

transaction with a count of the packages to be manipulated, and (6) execution of the entire transaction.

6. Now that the downgrade is complete and a lower version of the software is installed, you can check the status of the module:

```
[user1@server1 ~]$ dnf module list perl
AppStream
Name Stream Profiles Summary
perl 5.24 [e] common [d], minimal [i] Practical Extraction and Report Language
perl 5.26 [d] common [d], minimal Practical Extraction and Report Language
```

The profile “minimal” is installed (i under Profiles) and the stream 5.24 is enabled (e under Stream).

Chapter Summary

This chapter is the second of the two chapters (the first one being [Chapter 09](#)) with coverage on software management. This chapter covers advanced topics: yum repositories, package groups, and package modules.

We looked at the concept of package repositories and demonstrated setting up access to a local repo. Employing repositories for package installation and automatic dependency selection make software installation much easier than to use the rpm command.

We learned about package groups and the benefits of using them in contrast to handling individual software packages. We explored package modules, streams, and profiles, and how they are organized and manipulated.

Finally, we examined the dnf command and discovered its benefits over the standard rpm command. dnf offers a variety of options and subcommands that we employed in this chapter to demonstrate installing, listing, querying, updating, and deleting individual packages, package groups, package streams, and package profiles. Other management operations such as exhibiting package data, ascertaining provider, searching metadata for information, and switching module streams were also covered.

Review Questions

1. A module profile is a list of recommended packages organized for purpose-built deployments. True or False?
2. What would the *dnf group info Base* command do?
3. What would you run to list all available profiles for the module postgresql?
4. Which *dnf* subcommand can be used to list all available repositories?

5. What is the use of the `-y` option with the *dnf group install* and *dnf module remove* commands?
6. What is the main advantage of using *dnf* over *rpm*?
7. What would be the command to install *perl 5.26*?
8. What *dnf* subcommand would we use to check available updates for installed packages?
9. What is the concept of module in RHEL 8?
10. What must be the extension of a yum repository file?
11. A module stream is a group of packages organized by version. True or False?
12. What would the *dnf list zsh* command do?
13. What would the *dnf list installed *gnome** command do?
14. How many package names can be specified at a time with the *dnf install* command?
15. Name the two default yum repositories that contain all the packages for RHEL 8?
16. We can update all the packages within a package group using the *groupupdate* subcommand with *dnf*. True or False?
17. What would you run to reset a module so that it is neither enabled nor disabled?
18. What would the *dnf info zsh* command do?

Answers to Review Questions

1. True.
2. The command provided will list all packages in the specified package group.
3. To view *postgresql* module profiles, you would run *dnf module info --profile postgresql*.
4. The *repolist* subcommand.
5. *dnf* will not prompt for user confirmation if the `-y` option is used with it.
6. *dnf* resolves and installs dependent packages automatically.
7. It would be *dnf module install perl:5.26*.
8. The *check-update* subcommand.
9. A module is a collection of packages, including the dependent packages, that are required to install an application.
10. The extension of a yum repository configuration file must be `.repo` in order to be recognized as a valid repo file.
11. True.
12. The command provided will display installed and available *zsh* package.
13. The command provided will display all installed packages that contain *gnome* in their names.

14. There is no limit.
15. The two yum repositories the BaseOS and AppStream.
16. True.
17. You would run *dnf module reset* to reset a module.
18. The command provided will display the header information for the *zsh* package.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 10-1: Configure Access to RHEL 8 Repositories

As *user1* with *sudo* on *server1*, make sure the RHEL 8 ISO image is attached to the VM and mounted. Create a definition file under */etc/yum.repos.d*, and define two blocks (one for BaseOS and another for AppStream). Verify the configuration with **dnf repolist**. You should see numbers in thousands under the Status column for both repositories. (Hint1: [Chapter 09](#): Package Management with rpm). (Hint2: Software Management with dnf).

Lab 10-2: Install and Manage Individual Packages

As *user1* with *sudo* on *server1* and using the *dnf* command, list all installed and available packages separately. Show which package contains the */etc/group* file. Install the package *httpd*. Review */var/log/yum.log* for confirmation. Perform the following on the *httpd* package: (1) show information, (2) list dependencies, and (3) remove it. (Hint: Individual Package Management).

Lab 10-3: Install and Manage Package Groups

As *user1* with *sudo* on *server1* and using the *dnf* command, list all installed and available package groups separately. Install package groups *Security Tools* and *Scientific Support*. Review */var/log/yum.log* for confirmation. Show

the packages included in the *Scientific Support* package group, and delete this group. (Hint: Package Group Management).

Lab 10-4: Install and Manage Modules

As *user1* with *sudo* on *server1* and using the *dnf* command, list all modules. Identify which modules, streams and profiles are installed, default, disabled, and enabled from the output. Install the default stream of the development profile for module *php*, and verify. Remove the module. (Hint: Module Management).

Lab 10-5: Switch Module Streams and Install Software

As *user1* with *sudo* on *server1* and using the *dnf* command, list *postgresql* module. This will display the streams and profiles, and their status. Reset both streams, enable the stream for the older version, and install its client profile. (Hint: Module Management).

Chapter 11

Boot Process, GRUB2, and the Linux Kernel

This chapter describes the following major topics:

- Linux boot process: firmware, bootloader, kernel, and initialization
- Understand and interact with GRUB2 to boot into different targets
- Modify GRUB2 configuration
- Boot system into specific targets
- Reset lost or forgotten root user password
- Linux kernel, packages, version anatomy, and key directories
- Download and install a newer kernel version

RHCSA Objectives:

19. Interrupt the boot process in order to gain access to a system
46. Modify the system bootloader

RHEL goes through multiple phases during the boot process. It starts selective services during its transition from one phase into another. It presents the administrator an opportunity to interact with a preboot program to boot the system into a non-default target, pass an option to the kernel, or reset the lost or forgotten root user password. It launches a number of services during its transition to the default or specified target.

The kernel controls everything on the system. It controls the system hardware, enforces security and access controls, and runs, schedules, and manages processes and service daemons. The kernel is comprised of several modules. A new kernel must be installed or an existing kernel must be upgraded when the need arises from an application or functionality standpoint.

Linux Boot Process

RHEL goes through a *boot* process after the system has been powered up or restarted. The boot process lasts until all enabled services are started. A login prompt will appear on the screen, which allows users to log in to the system. The boot process is automatic, but you may need to interact with it to take a non-default action, such as booting an alternative kernel, booting into a non-default operational state, repairing the system, recovering from an unbootable state, and so on. The boot process on an x86 computer may be split into four major phases: (1) the firmware phase, (2) the bootloader phase, (3) the kernel phase, and (4) the initialization phase. The system accomplishes these phases one after the other while performing and attempting to complete the tasks identified in each phase.

The Firmware Phase (BIOS and UEFI)

The *firmware* is the BIOS (*Basic Input/Output System*) or the UEFI (*Unified Extensible Firmware Interface*) code that is stored in flash memory on the x86-based system board. It runs the *Power-On-Self-Test* (POST) to detect, test, and initialize the system hardware components. While doing so, it installs appropriate drivers for the video hardware and exhibit system messages on the screen. The firmware scans the available storage devices to locate a boot device, starting with a 512-byte image that contains 446 bytes of the bootloader program, 64 bytes for the partition table, and the last two bytes with the boot signature. This 512-byte tiny area is referred to as the *Master Boot Record* (MBR) and it is located on the first sector of the boot disk. As

soon as it discovers a usable boot device, it loads the bootloader into memory and passes control over to it.

The BIOS is a small memory chip in the computer that stores system date and time, list and sequence of boot devices, I/O configuration, etc. This configuration is customizable. Depending on the computer hardware, you need to press a key to enter the BIOS setup or display a menu to choose a source to boot the system. The computer goes through the hardware initialization phase that involves detecting and diagnosing peripheral devices. It runs the POST on the devices as it finds them, installs drivers for the graphics card and the attached monitor, and begins exhibiting system messages on the video hardware. It discovers a usable boot device, loads the bootloader program into memory, and passes control over to it. Boot devices on most computers support booting from optical and USB flash devices, hard drives, network, and other media.

The UEFI is a new 32/64-bit architecture-independent specification that computer manufacturers have widely adopted in their latest hardware offerings replacing BIOS. This mechanism delivers enhanced boot and runtime services, and superior features such as speed over the legacy 16-bit BIOS. It has its own device drivers, is able to mount and read extended file systems, includes UEFI-compliant application tools, and supports one or more bootloader programs. It comes with a boot manager that allows you to choose an alternative boot source. Most computer manufacturers have customized the features for their hardware platform. You may find varying menu interfaces among other differences.

The Bootloader Phase

Once the firmware phase is over and a boot device is detected, the system loads a piece of software called *bootloader* that is located in the boot sector of the boot device. RHEL uses GRUB2 (*G*rand *U*nified *B*ootloader) version 2 as the bootloader program. GRUB2 supports both BIOS and UEFI firmware.

The primary job of the bootloader program is to spot the Linux kernel code in the `/boot` file system, decompress it, load it into memory based on the configuration defined in the `/boot/grub2/grub.cfg` file, and transfer control over to it to further the boot process. For UEFI-based systems, GRUB2 looks for the EFI system partition `/boot/efi` instead, and runs the kernel based on the configuration defined in the `/boot/efi/EFI/redhat/grub.efi` file. The next section details the interaction with the bootloader.

The Kernel Phase

The *kernel* is the central program of the operating system, providing access to hardware and system services. After getting control from the bootloader, the kernel extracts the *initial RAM disk* (initrd) file system image found in the /boot file system into memory, decompresses it, and mounts it as read-only on /sysroot to serve as the temporary root file system. The kernel loads necessary modules from the initrd image to allow access to the physical disks and the partitions and file systems therein. It also loads any required drivers to support the boot process. Later, it unmounts the initrd image and mounts the actual physical root file system on / in read/write mode.

At this point, the necessary foundation has been built for the boot process to carry on and to start loading the enabled services. The kernel executes the *systemd* process with PID 1 and passes the control over to it.

The Initialization Phase

This is the fourth and the last phase in the boot process. *systemd* takes control from the kernel and continues the boot process. It is the default system initialization scheme used in RHEL 8. It starts all enabled userspace system and network services and brings the system up to the preset boot target.



A boot target is an operational level that is achieved after a series of services have been started to get to that state. More on targets later in this chapter.

The system boot process is considered complete when all enabled services are operational for the boot target and users are able to log in to the system. A detailed discussion on *systemd* is available in [Chapter 12 “System Initialization, Message Logging, and System Tuning”](#).

The GRUB2 Bootloader

After the firmware phase has concluded, the bootloader presents a menu with a list of bootable kernels available on the system and waits for a predefined amount of time before it times out and boots the default kernel. You may want to interact with GRUB2 before the autoboot times out to boot with a non-default kernel, boot to a different target, or customize the kernel boot string.

Pressing a key before the timeout expires allows you to interrupt the autoboot process and interact with GRUB2. If you wish to boot the system using the default boot device with all the configured default settings, do not press any key, as shown in [Figure 11-1](#), and let the system go through the autoboot process.

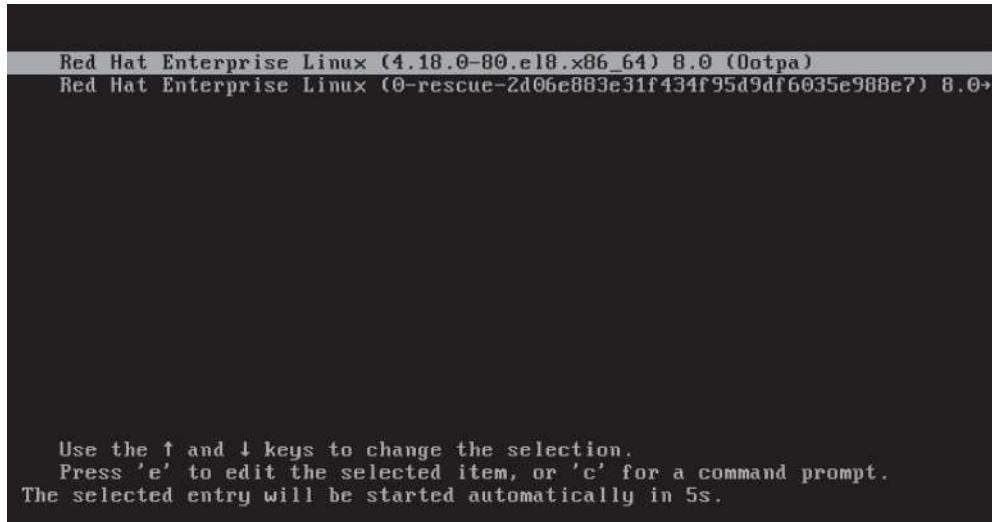


Figure 11-1 GRUB2 Menu

The line at the bottom in [Figure 11-1](#) shows the autoboot countdown in seconds. The default value is 5 seconds. If you press no keys within the 5 seconds, the highlighted kernel will boot automatically.

Interacting with GRUB2

The GRUB2 main menu shows a list of bootable kernels at the top. You can change the selection using the Up or Down arrow key. It lets you edit a selected kernel menu entry by pressing an **e** or go to the **grub>** command prompt by pressing a **c**.

In the edit mode, GRUB2 loads the configuration for the selected kernel entry from the `/boot/grub2/grub.cfg` file in an editor, enabling you to make a desired modification before booting the system. For instance, you can boot the system into a less capable operating target by adding “rescue”, “emergency”, or “3” to the end of the line that begins with the keyword “linux”, as depicted in [Figure 11-2](#). Press **Ctrl+x** when done to boot. Remember that this is a one-time temporary change and it won’t touch the `grub.cfg` file.

```

load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-80.el8.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/sw\ap rhgb quiet
initrd ($root)/initramfs-4.18.0-80.el8.x86_64.img $tuned_initrd

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.

```

Figure 11-2 GRUB2 Kernel Edit

If you do not wish to boot the system at this time, you can press ESC to discard the changes and return to the main menu.

The grub> command prompt appears when you press Ctrl+c while in the edit window or a c from the main menu. The command mode provides you with the opportunity to execute debugging, recovery, and many other tasks. You can view available commands by pressing the TAB key. See [Figure 11-3](#).

```

grub>
Possible commands are:

. [ acpi all_functional_test authenticate background_color background_image
backtrace badram blocklist bls_import blscfg boot break
btrfs-get-default-subvol btrfs-info btrfs-list-subvols btrfs-mount-subvol cat
cbmemc chainloader clear cmosclean cmosdump cmosset cmostest cmp configfile
continue coreboot_boottime cpuid crc cryptomount cutmem date distrust drivemap
dump echo efemu_loadcore efemu_prepare efemu_unload eval exit export
extract_entries_configfile extract_entries_source
extract_legacy_entries_configfile extract_legacy_entries_source
extract_sylinux_entries_configfile extract_sylinux_entries_source false file
freedos functional_test gdbstub gdbstub_break gdbstub_stop gettext gptsync halt
hashsum hdparm hello help hexdump hexdump_random inb initrd initrd16 inl insmod
inw keymap keystatus kfreebsd kfreebsd_loadenv kfreebsd_module
kfreebsd_module_elf knetbsd knetbsd_module knetbsd_module_elf kopenbsd
kopenbsd_ramdisk legacy_check_password legacy_configfile legacy_initrd
legacy_initrd_nounzip legacy_kernel legacy_password legacy_source linux linux16
list_env list_trusted load_env loadfont loopback ls lsacpi lsapm lscoreboot
lsfonts lsmap lsmod lspci macppcbless mactelbless md5sum menuentry module
module2 multiboot multiboot2 nativedisk net_add_addr net_add_dns net_add_route
net_bootp net_bootpb net_del_addr net_del_dns net_del_route net_get_dhcp_option
net_ipv6_autoconf net_ls_addr net_ls_cards net_ls_dns net_ls_routes
--MORE--

```

Figure 11-3 GRUB2 Commands

There are over one hundred commands available to perform a variety of tasks at the GRUB2 level.

Understanding GRUB2 Configuration Files

The GRUB2 configuration file, *grub.cfg*, is located in the */boot/grub2* directory. This file is referenced at boot time. This file is generated automatically when a new kernel is installed or upgraded, so it is not advisable to modify it directly, as your changes will be overwritten. The primary source file that is used to regenerate *grub.cfg* is called *grub*, and it is located in the */etc/default* directory. This file defines the directives that govern how GRUB2 should behave at boot time. Any changes made to the *grub* file will only take effect after the *grub2-mkconfig* utility has been executed.

Let's analyze the two files to understand their syntax and contents.

The */etc/default/grub* File

The *grub* file defines the directives that control the behavior of GRUB2 at boot time. Any changes in this file must be followed by the execution of the *grub2-mkconfig* command in order to be reflected in *grub.cfg*.

Here is an enumerated list of the default settings from the *grub* file, followed by an explanation in [Table 11-1](#):

```
[user1@server1 ~]# nl /etc/default/grub
 1 GRUB_TIMEOUT=5
 2 GRUB_DISTRIBUTOR="$(sed 's, release .*\,,g' /etc/system-release)"
 3 GRUB_DEFAULT=saved
 4 GRUB_DISABLE_SUBMENU=true
 5 GRUB_TERMINAL_OUTPUT="console"
 6 GRUB_CMDLINE_LINUX="crashkernel=auto resume=/dev/mapper/rhel-swap rd.lvm
 .lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet"
 7 GRUB_DISABLE_RECOVERY="true"
 8 GRUB_ENABLE_BLSCFG=true
```

Directive	Description
GRUB_TIMEOUT	Defines the wait time, in seconds, before booting off the default kernel. Default is 5.
GRUB DISTRIBUTOR	Sets the name of the Linux distributor
GRUB_DEFAULT	Boots the selected option from the system boot
GRUB_DISABLE_SUBMENU	Enables/disables the appearance of GRUB2 submenu
GRUB_TERMINAL_OUTPUT	Sets the default terminal
GRUB_CMDLINE_LINUX	Specifies the command line options to the kernel at boot time
GRUB_DISABLE_RECOVERY	Lists/hides system recovery entries in GRUB2 menu
GRUB_ENABLE_BLSCFG	Defines whether to use the new bootstrap specification to manage bootloader configuration

Table 11-1 GRUB2 Default Settings

Generally, you do not need to make any changes to this file, as the default settings are good enough for normal system operation.

The *grub.cfg* File

The *grub.cfg* is the main GRUB2 configuration file that supplies boot-time configuration information. This file is located in the */boot/grub2* directory on BIOS-based systems and in the */boot/efi/EFI/redhat* directory on UEFI-based systems. This file can be recreated manually with the *grub2-mkconfig* utility, or it is automatically regenerated when a new kernel is installed or upgraded. In either case, this file will lose any previous manual changes made to it.

During the recreation process, the *grub2-mkconfig* command also uses the settings defined in helper scripts located in the */etc/grub.d* directory. There are plenty of files located here, as shown below:

```
[user1@server1 ~]# sudo ls -l /etc/grub.d
total 84
-rwxr-xr-x. 1 root root 8958 Dec 19 2018 00_header
-rwxr-xr-x. 1 root root 1043 Jul 4 2018 00_tuned
-rwxr-xr-x. 1 root root 1240 Dec 19 2018 01_menu_auto_hide
-rwxr-xr-x. 1 root root 232 Dec 19 2018 01_users
-rwxr-xr-x. 1 root root 13406 Dec 19 2018 10_linux
-rwxr-xr-x. 1 root root 11696 Dec 19 2018 20_linux_xen
-rwxr-xr-x. 1 root root 2559 Dec 19 2018 20_ppc_terminfo
-rwxr-xr-x. 1 root root 10670 Dec 19 2018 30_os-prober
-rwxr-xr-x. 1 root root 1412 Dec 19 2018 30_uefi-firmware
-rwxr-xr-x. 1 root root 214 Dec 19 2018 40_custom
-rwxr-xr-x. 1 root root 216 Dec 19 2018 41_custom
-rw-r--r--. 1 root root 483 Dec 19 2018 README
```

The first script, *00_header*, sets the GRUB2 environment; the *10_linux* script searches for all installed kernels on the same disk partition; the *30_os-prober* searches for the presence of other operating systems; and *40_custom* and *41_custom* are to introduce any customization. An example would be to add custom entries to the boot menu.

The *grub.cfg* file also sources the *grubenv* file located in the */boot/grub2* directory for kernel options and other settings. Here is what this file contains on *server1*:

```
[root@server1 grub2]# cat grubenv
# GRUB Environment Block
saved_entry=0
kernelopts=root=/dev/mapper/rhel-root ro crashkernel=auto resume=/dev/mapper/rhe
l-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet
boot_success=1
boot_ineterminate=0
#####
#####
```

If a new kernel is installed, the existing kernel entries remain intact. All bootable kernels are listed in the GRUB2 menu, and any of the kernel entries can be selected to boot.

Exercise 11-1: Change Default System Boot Timeout

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will change the default system boot timeout value to 8 seconds persistently, and validate.

1. Edit the */etc/default/grub* file and change the setting as follows:

```
GRUB_TIMEOUT=8
```

2. Execute the *grub2-mkconfig* command to reproduce *grub.cfg*:

```
[user1@server1 ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
done
```

3. Restart the system with *sudo reboot* and confirm the new timeout value when GRUB2 menu appears.

Booting into Specific Targets

RHEL boots into graphical target state by default if the Server with GUI software selection is made during installation. It can also be directed to boot into non-default but less capable operating targets from the GRUB2 menu. Additionally, in situations when it becomes mandatory to boot the system into an administrative state for implementing a function that cannot be otherwise performed in other target states or for system recovery, RHEL offers emergency and rescue boot targets. These special target levels can be launched from the GRUB2 interface by selecting a kernel, pressing **e** to enter the edit mode, and appending the desired target name to the line that begins with the keyword “linux”.

EXAM TIP: You must know how to boot a RHEL 8 system into a specific target from the GRUB2 menu to modify the fstab file or reset an unknown root user password.

Here is how you would append “emergency” to the kernel line entry:

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-80.el8.x86_64 root=/dev/mapper/rhel-root ro crashkernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet emergency_
initrd ($root)/initramfs-4.18.0-80.el8.x86_64.img $tuned_initrd
```

Press **Ctrl+x** after making the modification to boot the system into the supplied target. You will be required to enter the *root* user password to log on. Run *reboot* after you are done to reboot the system.

```
You are in emergency mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue): _
```

Similarly, you can append “rescue” (or simply “1”, “s”, or “single”) to the “linux” line and press **Ctrl+x** to boot into the rescue target.

Exercise 11-2: Reset the root User Password

This exercise should be done on *server1*.

For this exercise, assume that the *root* user password is lost or forgotten, and it needs to be reset.

In this exercise, you will terminate the boot process at an early stage to be placed in a special debug shell in order to reset the *root* password.

1. Reboot or reset *server1*, and interact with GRUB2 by pressing a key before the autoboot times out. Highlight the default kernel entry in the GRUB2 menu and press **e** to enter the edit mode. Scroll down to the line entry that begins with the keyword “linux” and press the End key to go to the end of that line:

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-80.e18.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/sw\ 
ap rhgb quiet
initrd ($root)/initramfs-4.18.0-80.e18.x86_64.img $tuned_initrd
```

2. Modify this kernel string and append “rd.break” to the end of the line. It should look like:

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-80.e18.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/sw\ 
ap rhgb quiet rd.break
initrd ($root)/initramfs-4.18.0-80.e18.x86_64.img $tuned_initrd
```

3. Press **Ctrl+x** when done to boot to the special shell. The system mounts the root file system read-only on the */sysroot* directory. Make */sysroot* appear as mounted on */* using the *chroot* command:

```
switch_root:/# chroot /sysroot
sh-4.4# pwd
/
sh-4.4#
```

4. Remount the root file system in read/write mode for the *passwd* command to be able to modify the *shadow* file with a new password:

```
sh-4.4# mount -o remount,rw /
```

5. Enter a new password for *root* by invoking the *passwd* command:

```
sh-4.4# passwd
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
```

6. Create a hidden file called *.autorelabel* to instruct the operating system to run SELinux relabeling on all files, including the *shadow* file that was updated with the new *root* password, on the next reboot:

```
sh-4.4# touch .autorelabel
```

7. Issue the *exit* command to quit the chroot shell and then the *reboot* command to restart the system and boot it to the default target.

```
sh-4.4# exit
exit
switch_root:/# reboot_
```

Depending on the storage speed, the system might take a few minutes to complete the relabeling and return to a fully functional state.

The Linux Kernel

The *kernel* is the core of the Linux system. It manages hardware, enforces security, regulates access to the system, as well as handles processes, services, and application workloads. It is a collection of software components called *modules* that work in tandem to provide a stable and controlled platform. Modules are *device drivers* that control hardware devices—the processor, memory, storage, controller cards, and peripheral equipment, and interact with software subsystems, such as storage partitioning, file systems, networking, and virtualization.

Some of these modules are static to the kernel and are integral to system functionality, while others are loaded dynamically as needed, making the kernel speedier and more efficient in terms of overall performance and less vulnerable to crashes. RHEL 8.0 and RHEL 8.2 are shipped with kernel version 4.18.0 (4.18.0-80 and 4.18.0-193 to be specific) for the 64-bit Intel/AMD processor architecture computers with single, multi-core, and multi-processor configurations. On a Linux system, **uname -m** reveals the architecture of the system.

The default kernel installed during the installation is usually adequate for most system needs; however, it requires a rebuild when a new functionality is added or removed. The new functionality may be introduced by installing a

new kernel, upgrading an existing one, installing a new hardware device, or changing a critical system component. Likewise, an existing functionality that is no longer needed may be removed to make the overall footprint of the kernel smaller, resulting in improved performance and reduced memory utilization.

To control the behavior of the modules, and the kernel in general, a variety of tunable parameters are set that define a baseline for kernel functionality. Some of these parameters must be tuned to allow certain applications and database software to be installed smoothly and operate properly.

RHEL allows you to generate and store several custom kernels with varied configuration and required modules, but only one of them can be active at a time. A different kernel may be loaded by interacting with GRUB2.

Kernel Packages

Just like any other software that comes in the rpm format for RHEL, the software comprising the kernel is no different. There is a set of core kernel packages that must be installed on the system at a minimum to make it work. Additional packages providing supplementary kernel support are also available. [Table 11-2](#) lists and describes the core and some add-on kernel packages.

Kernel Package	Description
kernel	Contains no files, but ensures other kernel packages are accurately installed
kernel-core	Includes a minimal number of modules to provide core functionality
kernel-devel	Includes support for building kernel modules
kernel-modules	Contains modules for common hardware devices
kernel-modules-extra	Contains modules for not-so-common hardware devices
kernel-headers	Includes files to support the interface between the kernel and userspace libraries and programs
kernel-tools	Includes tools to manipulate the kernel
kernel-tools-libs	Includes the libraries to support the kernel tools

Table 11-2 Kernel Packages

Moreover, packages containing the source code for RHEL 8 are also available for those who wish to customize and recompile the code for their precise

needs.

Currently, the following kernel packages are installed on *server1*:

```
[user1@server1 ~]$ dnf list installed kernel*
Not root, Subscription Management repositories not updated
Installed Packages
kernel.x86_64                      4.18.0-80.e18          @anaconda
kernel-core.x86_64                    4.18.0-80.e18          @anaconda
kernel-headers.x86_64                 4.18.0-80.e18          @BaseOS
kernel-modules.x86_64                  4.18.0-80.e18          @anaconda
kernel-tools.x86_64                   4.18.0-80.e18          @anaconda
kernel-tools-libs.x86_64               4.18.0-80.e18          @anaconda
```

The output returns six kernel packages that were loaded during the OS installation.

Analyzing Kernel Version

Sometimes you need to ascertain the version of the kernel running on the system to check for compatibility with an application or database that you need to deploy. RHEL has a basic tool available to extract the version. The *uname* command with the *-r* switch depicts this information, as follows:

```
[user1@server1 ~]$ uname -r
4.18.0-80.e18.x86_64
```

The output shows the current kernel version in use is 4.18.0-80.el8.x86_64. An anatomy of the version is illustrated in [Figure 11-4](#) and explained subsequently.

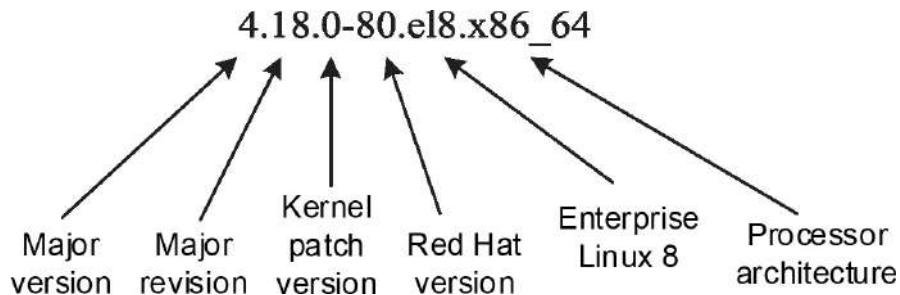


Figure 11-4 Anatomy of a Kernel Version

From left to right:

- ✓ (4) major version of the Linux kernel. It changes when significant alterations, enhancements, and updates to the previous major version are made.
- ✓ (18) major revision of the 4th major version
- ✓ (0) represents patched version of 4.18 with minor bug and security hole fixes, minor enhancements, and so on.

- ✓ (80) Red Hat customized version of 4.18.0
- ✓ (el8) Enterprise Linux this kernel is built for
- ✓ (x86_64) architecture this kernel is built for

A further analysis designates that 4.18.0 holds the general Linux kernel version information, the numbers and letters (80.el8) represent the Red Hat specific information, and x86_64 identifies the hardware architecture type.

Understanding Kernel Directory Structure

Kernel and its support files are stored at different locations in the directory hierarchy, of which three locations—*/boot*, */proc*, and */usr/lib/modules*—are noteworthy.

The */boot* Location

/boot is essentially a file system that is created at system installation. It houses the Linux kernel, GRUB2 configuration, and other kernel and boot support files. A long listing produces the following output for this file system:

```
[user1@server1 ~]$ ls -l /boot
total 126992
-rw-r--r--. 1 root root 180942 Mar 13 2019 config-4.18.0-80.el8.x86_64
drwxr-xr-x. 3 root root 17 Aug 23 06:06 efi
drwx-----. 4 root root 83 Oct 6 20:00 grub2
-rw-----. 1 root root 65698603 Aug 23 06:22 initramfs-0-rescue-2d06e883e31f434
f95d9df6035e988e7.img
-rw-----. 1 root root 26614276 Aug 23 06:27 initramfs-4.18.0-80.el8.x86_64.img
-rw-----. 1 root root 18030660 Oct 10 15:48 initramfs-4.18.0-80.el8.x86_64kdump
.p.img
drwxr-xr-x. 3 root root 21 Aug 23 06:15 loader
-rw-----. 1 root root 3751920 Mar 13 2019 System.map-4.18.0-80.el8.x86_64
-rwrxr-xr-x. 1 root root 7872864 Aug 23 06:20 vmlinuz-0-rescue-2d06e883e31f434f9
5d9df6035e988e7
-rwxr-xr-x. 1 root root 7872864 Mar 13 2019 vmlinuz-4.18.0-80.el8.x86_64
```

The output displays several files, four of which are for the kernel and two for its rescue version. The kernel files are *vmlinuz*, *initramfs*, *config*, and *System.map*, and they all have the current kernel version appended to their names. The *vmlinuz* is the main kernel file with *initramfs*, *config*, and *System.map* storing the main kernel's boot image, configuration, and mapping, respectively.

The files for the rescue version have the string “rescue” embedded within their names, as indicated in the above output.

The *efi* and *grub2* subdirectories under */boot* hold bootloader information specific to firmware type used on the system: UEFI or BIOS. For *server1*, *grub2* contains GRUB2 information as shown below:

```
[user1@server1 ~]$ sudo ls -l /boot/grub2
total 28
-rw-r--r--. 1 root root 64 Aug 23 06:26 device.map
drwxr-xr-x. 2 root root 25 Aug 23 06:26 fonts
-rw-r--r--. 1 root root 5032 Oct 6 20:00 grub.cfg
-rw-r--r--. 1 root root 1024 Oct 11 10:09 grubenv
drwxr-xr-x. 2 root root 8192 Aug 23 06:26 i386-pc
```

The files *grub.cfg* and *grubenv* contain critical data with the former holding bootable kernel information and the latter stores the environment information that the kernel uses.

The subdirectory *loader* under */boot* is the storage location for configuration of the running and rescue kernels. The configuration is stored in files under the *entries* subdirectory, as shown below:

```
[user1@server1 ~]$ sudo ls -l /boot/loader/entries/
total 8
-rw-r--r--. 1 root root 408 Aug 23 06:22 2d06e883e31f434f95d9df6035e988e7-0-rescue.conf
-rw-r--r--. 1 root root 331 Aug 23 06:22 2d06e883e31f434f95d9df6035e988e7-4.18.0-80.el8.x86_64.conf
```

The files are named using the machine id of the system as stored in the */etc/machine-id* file and the kernel version they are for. The content of the kernel file is presented below:

```
[user1@server1 ~]$ sudo cat /boot/loader/entries/2d06e883e31f434f95d9df6035e988e7-4.18.0-80.el8.x86_64.conf
title Red Hat Enterprise Linux (4.18.0-80.el8.x86_64) 8.0 (Octpa)
version 4.18.0-80.el8.x86_64
linux /vmlinuz-4.18.0-80.el8.x86_64
initrd /initramfs-4.18.0-80.el8.x86_64.img $tuned_initrd
options $kernelopts $tuned_params
id rhel-20190313123447-4.18.0-80.el8.x86_64
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

The “title” is displayed on the bootloader screen where the countdown to autoboot a selected kernel entry runs. Other than the kernel version, and the default kernel and boot image filenames, the environment variables “*kernelopts*” and “*tuned_params*” supply various values to the booting kernel to control its behavior.

The */proc* Location

/proc is a virtual, memory-based file system. Its contents are created and updated in memory at system boot and during runtime, and they are destroyed at system shutdown. It maintains information about the current state of the kernel, which includes hardware configuration and status information about processor, memory, storage, file systems, swap, processes, network

interfaces and connections, routing, etc. This data is kept in tens of thousands of zero-byte files organized in a hierarchy of hundreds of subdirectories. A long directory listing of */proc* is provided below:

```
[user1@server1 ~]$ ls -l /proc
total 0
dr-xr-xr-x.  9 root          root          0 Oct 11 10:06 1
dr-xr-xr-x.  9 root          root          0 Oct 11 10:06 10
dr-xr-xr-x.  9 avahi         avahi         0 Oct 11 10:07 1010
dr-xr-xr-x.  9 root          root          0 Oct 11 10:06 1012
dr-xr-xr-x.  9 root          root          0 Oct 11 10:06 1016
....
```

Some subdirectory names are numerical and contain information about a specific process, and the process ID matches the subdirectory name. Within each subdirectory are other files and subdirectories containing a plethora of information, such as memory segments for processes and configuration data for system components. You can view the configuration of a particular item using any text file viewing tool. The following show selections from the *cpuinfo* and *meminfo* files that hold processor and memory information:

```
[user1@server1 ~]$ cat /proc/cpuinfo
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 58
model name    : Intel(R) Core(TM) i7-3610QM CPU @ 2.30GHz
stepping       : 9
microcode     : 0x19
cpu MHz       : 2294.786
cache size    : 6144 KB
physical id   : 0
siblings       : 1
core id        : 0
cpu cores     : 1
apicid         : 0
initial apicid: 0
fpu            : yes
fpu_exception  : yes
.....
[user1@server1 ~]$ cat /proc/meminfo
MemTotal:      841124 kB
MemFree:       167684 kB
MemAvailable:  466216 kB
Buffers:        3488 kB
Cached:        395344 kB
SwapCached:     0 kB
Active:        304880 kB
Inactive:     200116 kB
.....
```

The data stored under */proc* is referenced by many system utilities, including *top*, *ps*, *uname*, *free*, *uptime* and *w*, to display information.

The `/usr/lib/modules` Location

This directory holds information about kernel modules. Underneath it are subdirectories specific to the kernels installed on the system. For example, the long listing of `/usr/lib/modules` below shows that there is only one kernel installed. The name of the subdirectory corresponds with the installed kernel version.

```
[user1@server1 ~]$ ls -l /usr/lib/modules
total 4
drwxr-xr-x. 6 root root 4096 Oct  3 23:22 4.18.0-80.e18.x86_64
```

And this subdirectory contains:

```
[user1@server1 ~]$ ls -l /usr/lib/modules/4.18.0-80.e18.x86_64/
total 15100
-rw-r--r--. 1 root root    303 Mar 13  2019 bls.conf
lrwxrwxrwx. 1 root root     37 Mar 13  2019 build -> /usr/src/kernels/4.18.0-8
0.e18.x86_64
-rw-r--r--. 1 root root 180942 Mar 13  2019 config
drwxr-xr-x. 12 root root   128 Aug 23 06:09 kernel
-rw-r--r--. 1 root root  840950 Oct  3 23:22 modules.alias
-rw-r--r--. 1 root root  806340 Oct  3 23:22 modules.alias.bin
-rw-r--r--. 1 root root    481 Mar 13  2019 modules.block
-rw-r--r--. 1 root root   7443 Mar 13  2019 modules.builtin
....
```

There are several files and a few subdirectories here; they hold module-specific information for the kernel version.

One of the key subdirectories is `/usr/lib/modules/4.18.0-80.e18.x86_64/kernel/drivers`, which stores modules for a variety of hardware and software components in various subdirectories, as shown in the listing below:

```
[user1@server1 ~]$ ls -l /usr/lib/modules/4.18.0-80.e18.x86_64/kernel/drivers/
total 48
drwxr-xr-x. 5 root root 209 Aug 23 06:09 acpi
drwxr-xr-x. 2 root root 169 Aug 23 06:09 ata
drwxr-xr-x. 2 root root  24 Aug 23 06:10 bcma
drwxr-xr-x. 3 root root 163 Aug 23 06:09 block
drwxr-xr-x. 2 root root 272 Aug 23 06:10 bluetooth
drwxr-xr-x. 2 root root  25 Aug 23 06:09 cdrom
drwxr-xr-x. 5 root root 181 Aug 23 06:09 char
drwxr-xr-x. 2 root root 143 Aug 23 06:09 cpufreq
drwxr-xr-x. 7 root root 121 Aug 23 06:09 crypto
drwxr-xr-x. 2 root root  52 Aug 23 06:09 dax
drwxr-xr-x. 2 root root  23 Aug 23 06:09 dca
drwxr-xr-x. 4 root root  48 Aug 23 06:09 dma
drwxr-xr-x. 2 root root 4096 Aug 23 06:09 edac
drwxr-xr-x. 2 root root 113 Aug 23 06:10 firewire
drwxr-xr-x. 2 root root  89 Aug 23 06:10 firmware
....
```

Additional modules may be installed on the system to support more components.

Installing the Kernel

Installing kernel packages requires extra care because it could leave your system in an unbootable or undesirable state. It is advisable to have the bootable medium handy prior to starting the kernel install process. By default, the *dnf* command adds a new kernel to the system, leaving the existing kernel(s) intact. It does not replace or overwrite existing kernel files.

EXAM TIP: Always install a new version of the kernel instead of upgrading it. The upgrade process removes any existing kernel and replaces it with a new one. In case of a post-installation issue, you will not be able to revert to the old working kernel.

A newer version of the kernel is typically required if an application needs to be deployed on the system that requires a different kernel to operate. When deficiencies or bugs are identified in the existing kernel, it can hamper the kernel's smooth operation. In either case, the new kernel addresses existing issues as well as adds bug fixes, security updates, new features, and improved support for hardware devices.

The *dnf* command is the preferred tool to install a kernel, as it resolves and installs any required dependencies automatically. The *rpm* command may alternatively be used; however, you must install any reported dependencies manually.

The kernel packages for RHEL 8 are available to subscribers for download on Red Hat's Customer Portal. You need a user account in order to log in and download.

Exercise 11-3: Download and Install a New Kernel

This exercise should be done on *server1* as *user1* with *sudo* where required.

In this exercise, you will download the latest available kernel packages from the Red Hat Customer Portal and install them using the *dnf* command. You will ensure that the existing kernel and its configuration remain intact.



As an alternative (preferred) to downloading kernel packages individually and then installing them, you can follow the instructions provided in [Exercise 23-1 in Chapter 23](#) "Containers" to register *server1* with RHSMS and run **sudo dnf install kernel** to install the latest kernel and all the dependencies collectively.

1. Check the version of the running kernel:

```
[user1@server1 ~]$ uname -r  
4.18.0-80.el8.x86_64
```

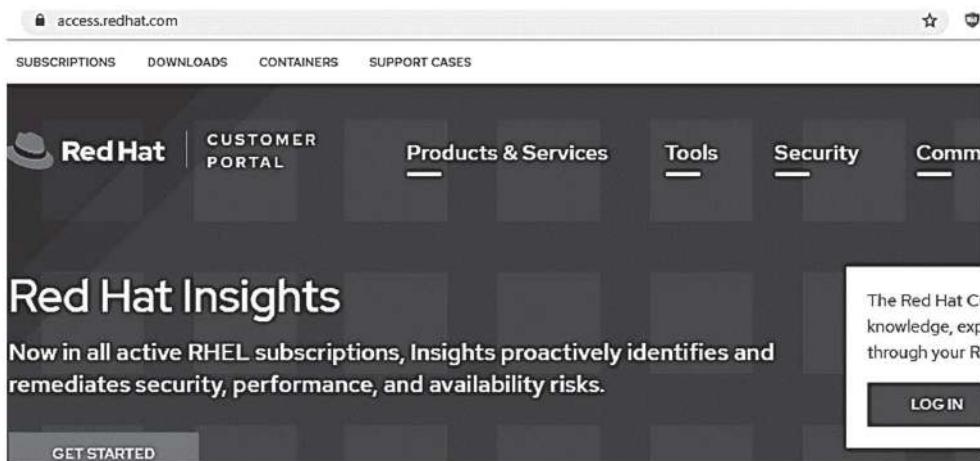
The output discloses the current active kernel version as 4.18.0-80.el8.x86_64.

2. List the kernel packages currently installed:

```
[user1@server1 ~]$ rpm -qa |grep kernel  
kernel-modules-4.18.0-80.el8.x86_64  
kernel-core-4.18.0-80.el8.x86_64  
kernel-headers-4.18.0-80.el8.x86_64  
kernel-4.18.0-80.el8.x86_64  
kernel-tools-4.18.0-80.el8.x86_64  
kernel-tools-libs-4.18.0-80.el8.x86_64
```

There are six kernel packages on the system as reported.

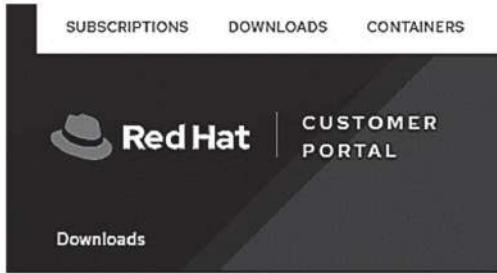
3. Access the Red Hat Customer Portal webpage at access.redhat.com and click LOG IN to sign in to the portal using the credentials you used/created in [Chapter 01 “Local Installation”](#) for the developer account:



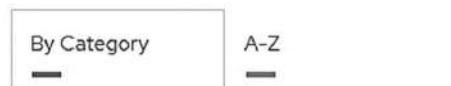
4. Click DOWNLOADS at the top:



5. Click “Red Hat Enterprise Linux 8” under “By Category”:



Product Downloads



INFRASTRUCTURE MANAGEMENT

Product

Red Hat Enterprise Linux 8 Versions 7 and bel

6. Click Packages and enter “kernel” in the Search bar to narrow the list of available packages:

Download Red Hat Enterprise Linux

Product Variant: Version: Architecture:

Red Hat Enterprise Linux for x86_64	8	x86_64
-------------------------------------	---	--------

About Red Hat Enterprise Linux for x86_64

Only Red Hat Enterprise Linux provides an intelligent OS that is the consistent foundation for the enterprise hybrid cloud. Delivering any application on any footprint at any time giving you Control. Confidence. Freedom.

Product Resources

- Get Started
- Documentation
- Red Hat Enterprise Linux Life Cycle

Get Help

- Contact Support
- Create installation media

Product Software Modules Packages Source Errata

Packages for Red Hat Enterprise Linux for x86_64 (v. 8 for x86_64)

Show 25 entries	Search: kernel	
Package	Summary	
Red Hat Enterprise Linux 8 for x86_64 - BaseOS (RPMs)		
audit	User space tools for 2.6 kernel auditing	Download Latest
biktrace	Utilities for performing block layer IO tracing in the Linux kernel	Download Latest

7. Click “Download Latest” against the packages *kernel*, *kernel-core*, *kernel-headers*, *kernel-modules*, *kernel-tools*, and *kernel-tools-libs* to

download them. These are the newer packages for the version you currently have on the system.

8. Once downloaded, move the packages to the `/tmp` directory using the `mv` command.
9. List the packages after moving them:

```
[user1@server1 ~]$ ls /tmp/kernel*
/tmp/kernel-4.18.0-80.11.2.el8_0.x86_64.rpm      /tmp/kernel-modules-4.18.0-80.11.2.el8_0.x86_64.rpm
/tmp/kernel-core-4.18.0-80.11.2.el8_0.x86_64.rpm   /tmp/kernel-tools-4.18.0-80.11.2.el8_0.x86_64.rpm
/tmp/kernel-headers-4.18.0-80.11.2.el8_0.x86_64.rpm /tmp/kernel-tools-libs-4.18.0-80.11.2.el8_0.x86_64.rpm
```

10. Install all the six packages at once using the `dnf` command:

```
[user1@server1 ~]$ sudo dnf install /tmp/kernel* -y
Dependencies resolved.
=====
Package          Arch    Version           Repository      Size
=====
Installing:
kernel          x86_64  4.18.0-80.11.2.el8_0  @commandline   424 k
kernel-core      x86_64  4.18.0-80.11.2.el8_0  @commandline   24 M
kernel-modules   x86_64  4.18.0-80.11.2.el8_0  @commandline   20 M
Upgrading:
kernel-headers   x86_64  4.18.0-80.11.2.el8_0  @commandline   1.6 M
kernel-tools     x86_64  4.18.0-80.11.2.el8_0  @commandline   574 k
kernel-tools-libs x86_64  4.18.0-80.11.2.el8_0  @commandline   433 k
Transaction Summary
=====
Install  3 Packages
Upgrade  3 Packages

Total size: 47 M
.....
Installed products updated.

Upgraded:
kernel-headers-4.18.0-80.11.2.el8_0.x86_64        kernel-tools-4.18.0-80.11.2.el8_0.x86_64
kernel-tools-libs-4.18.0-80.11.2.el8_0.x86_64

Installed:
kernel-4.18.0-80.11.2.el8_0.x86_64                kernel-core-4.18.0-80.11.2.el8_0.x86_64
kernel-modules-4.18.0-80.11.2.el8_0.x86_64

Complete!
```

11. Confirm the installation alongside the previous version:

```
[user1@server1 ~]$ sudo dnf list installed kernel*
```

Installed Packages		
Kernel	Version	Repository
kernel.x86_64	4.18.0-80.el8	@anaconda
kernel.x86_64	4.18.0-80.11.2.el8_0	@commandline
kernel-core.x86_64	4.18.0-80.el8	@anaconda
kernel-core.x86_64	4.18.0-80.11.2.el8_0	@commandline
kernel-headers.x86_64	4.18.0-80.11.2.el8_0	@commandline
kernel-modules.x86_64	4.18.0-80.el8	@anaconda
kernel-modules.x86_64	4.18.0-80.11.2.el8_0	@commandline
kernel-tools.x86_64	4.18.0-80.11.2.el8_0	@commandline
kernel-tools-libs.x86_64	4.18.0-80.11.2.el8_0	@commandline

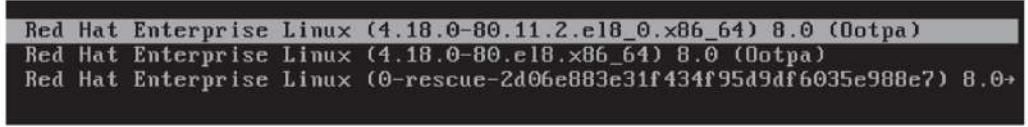
The output indicates that packages for a higher kernel version 4.18.0-80.11.2.el8 have been installed. It also shows the presence of the previous

kernel packages.

12. The `/boot/grub2/grubenv` file now has the directive “`saved_entry`” set to the new kernel, which implies that this new kernel will boot up on the next system restart:

```
[user1@server1 ~]$ sudo cat /boot/grub2/grubenv
# GRUB Environment Block
saved_entry=2d06e883e31f434f95d9df6035e988e7-4.18.0-80.11.2.e18_0.x86_64
kernelopts=root=/dev/mapper/rhel-root ro crashkernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root r
d.lvm.lv=rhel/swap rhgb quiet
boot_success=1
boot_ineterminate=0
```

13. Reboot the system. You will see the new kernel entry in the GRUB2 boot list at the top. The system will autoboot this new default kernel.



```
Red Hat Enterprise Linux (4.18.0-80.11.2.e18_0.x86_64) 8.0 (Ootpa)
Red Hat Enterprise Linux (4.18.0-80.e18.x86_64) 8.0 (Ootpa)
Red Hat Enterprise Linux (0-rescue-2d06e883e31f434f95d9df6035e988e7) 8.0->
```

14. Run the `uname -r` command once the system has been booted up to confirm the loading of the new kernel:

```
[user1@server1 ~]$ uname -r
4.18.0-80.11.2.e18_0.x86_64
```

The new active kernel is the one that was just installed.

15. You can also view the contents of the `version` and `cmdline` files under `/proc` to verify the active kernel:

```
[user1@server1 ~]$ cat /proc/version
Linux version 4.18.0-80.11.2.e18_0.x86_64 (mockbuild@x86-vm-07.build.eng.bos.redhat.com) (gcc version 8.2.1
20180905 (Red Hat 8.2.1-3) (GCC)) #1 SMP Sun Sep 15 11:24:21 UTC 2019
[user1@server1 ~]$
[user1@server1 ~]$ cat /proc/cmdline
BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-80.11.2.e18_0.x86_64 root=/dev/mapper/rhel-root ro crashkernel=auto :
resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet
```

This concludes the process of downloading and installing a newer kernel version on RHEL 8.

Chapter Summary

This chapter presented a high-level look at the Linux boot process, highlighting the four phases the system passes through and the complexities involved. We reviewed firmware and looked at preboot administration tasks, which included interacting with the bootloader program, booting into specific targets, and an analysis of the bootloader configuration files. We performed a critical exercise that focused on resetting a forgotten or lost root user password.

The next major topic discussed the Linux kernel, its key components and management. We explored various packages that made up the core kernel software, performed an anatomy of its versioning, and examined key directories and file systems that are employed to hold kernel-specific information. Before the chapter was concluded, we downloaded and installed a new kernel without impacting the old one.

Review Questions

1. UEFI is replacing BIOS in newer computers. True or False?
2. You have changed the timeout value in the *grub* configuration file located in the */etc/default* directory. Which command would you run now to ensure the change takes effect on next system reboots?
3. You want to view the parameters passed to the kernel at boot time. Which virtual file would you look at?
4. Name the location of the *grub.efd* file in the UEFI-based systems.
5. What is the name of the default bootloader program in RHEL 8?
6. The *systemd* command may be used to rebuild a new kernel. True or False?
7. What does the *chroot* command do?
8. At what stage should you interrupt the boot sequence to boot the system into a non-default target?
9. Which two files would you view to obtain processor and memory information?
10. By default, GRUB2 is stored in the MBR on a BIOS-based system. True or False?
11. Which file stores the location of the boot partition on the BIOS systems?
12. You have lost the *root* user password and you need to reset it. What would you add to the default kernel boot string to break the boot process at an early stage?
13. You have installed a newer version of the kernel. What would you now have to do to make the new kernel the default boot kernel?
14. What is the system initialization and service management scheme called?

Answers to Review Questions

1. True.
2. You will need to run the *grub2-mkconfig* command.
3. The *cmdline* file in the */proc* file system.
4. The *grub.efd* file is located in the */boot/efi/EFI/redhat* directory.
5. GRUB2.
6. False.

7. The *chroot* command changes the specified directory path to /.
8. At the GRUB2 stage.
9. The *cpuinfo* and *meminfo* files in the */proc* file system.
10. True.
11. The *grub.cfg* file stores the location information of the boot partition.
12. You would add *rd.break* to the kernel boot string.
13. Nothing. The install process takes care of that.
14. It is called *systemd*.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 11-1: Enable Verbose System Boot

As *user1* with *sudo* on *server1*, remove “quiet” from the end of the value of the variable *GRUB_CMDLINE_LINUX* in the */etc/default/grub* file and run *grub2-mkconfig* to apply the update. Reboot the system and observe that the system now displays verbose information during the boot process. (Hint: The GRUB2 Bootloader and [Exercise 11-1](#)).

Lab 11-2: Reset root User Password

As *root* on *server2*, reset the *root* user password by booting the system into emergency mode with SELinux disabled. Try to log in with *root* and enter the new password after the reboot. (Hint: The GRUB2 Bootloader).

Lab 11-3: Install New Kernel

As *user1* with *sudo* on *server1*, check the current version of the kernel using the *uname* or *rpm* command. Download a higher version from the Red Hat Customer Portal or [rpmfind.net](#), and install it. Reboot the system and ensure the new kernel is listed on the bootloader menu. (Hint: The Linux Kernel).

Chapter 12

System Initialization, Message Logging, and System Tuning

This chapter describes the following major topics:

- Understand systemd, units, and targets
- Analyze service and target unit configuration files
- List and view status of running units
- Manage service units and target units
- Display and configure default system boot target
- Switch into non-default targets
- Analyze system log configuration file
- Examine log file rotation settings
- Review boot and system log files
- Record custom messages in system log file
- Describe systemd journal service
- Retrieve and scrutinize messages from journal
- Store journal information persistently
- Know system tuning and apply tuning profile

RHCSA Objectives:

17. Boot, reboot, and shut down a system normally
18. Boot systems into different targets manually
25. Start, stop, and check the status of network services

41. Start and stop services and configure services to start automatically at boot
42. Configure systems to boot into a specific target automatically
49. Configure network services to start automatically at boot
22. Manage tuning profiles
23. Locate and interpret system log files and journals
24. Preserve system journals

Systemd is the default system initialization and service management scheme. It boots the system into one of several predefined targets and it is used to handle operational states of services. It employs the concepts of units and targets for initialization, service administration, and state changes. A good grasp of what unit and target configuration files store is key to understanding how systemd operates.

Alerts and messages generated by system services and user activities are forwarded to predefined locations for storage. These alerts and messages include those that are produced during system boot time. The log data may be analyzed for debugging or auditing purposes. Log files grow over time and need to be rotated periodically to prevent the file system space from filling up. There are configuration files that define the default and custom locations to direct the log messages to and to configure rotation settings. The system log file records custom messages sent to it. systemd includes a service for viewing and managing system logs in addition to the traditional logging service. This service maintains a log of runtime activities for faster retrieval and can be configured to store the information permanently.

System tuning service is employed to monitor connected devices and to tweak their parameters to improve performance or conserve power. A recommended tuning profile may be identified and activated for optimal performance and power saving.

System Initialization and Service Management

systemd (short for *system daemon*) is the system initialization and service management mechanism. It has fast-tracked system initialization and state transitioning by introducing features such as parallel processing of startup scripts, improved handling of service dependencies, and on-demand activation of services. Moreover, it supports snapshotting of system states, tracks processes using control groups, and automatically maintains mount points. *systemd* is the first process with PID 1 that spawns at boot and it is the last process that terminates at shutdown.



systemd spawns several processes during a service startup. It places the processes in a private hierarchy composed of *control groups* (or *cgroups* for short) to organize processes for the purposes of monitoring and controlling system resources such as processor, memory, network bandwidth, and disk I/O. This includes limiting, isolating, and

prioritizing their usage of resources. This way resources can be distributed among users, databases, and applications based on need and priority, resulting in overall improved system performance.

In order to benefit from parallelism, systemd initiates distinct services concurrently, taking advantage of multiple CPU cores and other compute resources. To that end, systemd creates sockets for all enabled services that support socket-based activation instantaneously at the very beginning of the initialization process. It passes them on to service daemon processes as they attempt to start in parallel. This approach lets systemd handle inter-service order dependencies and allows services to start without any delays. With systemd, dependent daemons need not be running; they only need the correct socket to be available. systemd creates sockets first, starts daemons next, and caches any client requests to daemons that have not yet started in the socket buffer. It fills the pending client requests when the daemons they were awaiting come online.



Socket is a communication method that allows a single process to talk to another process on the same or remote system.

During the operational state, systemd maintains the sockets and uses them to reconnect other daemons and services that were interacting with an old instance of a daemon before that daemon was terminated or restarted.

Likewise, services that use activation based on D-Bus (*Desktop Bus*) are started when a client application attempts to communicate with them for the first time. Additional methods used by systemd for activation are device-based and path-based, with the former starting the service when a specific hardware type such as USB is plugged in, and the latter starting the service when a particular file or directory alters its state.



D-Bus is another communication method that allows multiple services running in parallel on a system to talk to one another on the same or remote system.

With on-demand activation, systemd defers the startup of services—Bluetooth and printing—until they are actually needed during the boot process or during runtime. Together, parallelization and on-demand activation save time and compute resources, and contribute to expediting the boot process considerably.

Another major benefit of parallelism witnessed at system boot is when systemd uses the autofs service to temporarily mount the configured file systems. During the boot process, the file systems are checked that may result in unnecessary delays. With autofs, the file systems are temporarily

mounted on their normal mount points, and as soon as the checks on the file systems are finished, systemd remounts them using their standard devices. Parallelism in file system mounts does not affect the root and virtual file systems.

Units

Units are systemd objects used for organizing boot and maintenance tasks, such as hardware initialization, socket creation, file system mounts, and service startups. Unit configuration is stored in their respective configuration files, which are auto-generated from other configurations, created dynamically from the system state, produced at runtime, or user-developed. Units are in one of several operational states, including active, inactive, in the process of being activated or deactivated, and failed. Units can be enabled or disabled. An enabled unit can be started to an active state; a disabled unit cannot be started.

Units have a name and a type, and they are encoded in files with names in the form `unitname.type`. Some examples are `tmp.mount`, `sshd.service`, `syslog.socket`, and `umount.target`. There are two types of unit configuration files: (1) system unit files that are distributed with installed packages and located in the `/usr/lib/systemd/system` directory, and (2) user unit files that are user-defined and stored in the `/etc/systemd/user` directory (run `ls -l` on both directories to list their contents). This information can be vetted with the `pkg-config` command:

```
[user1@server1 ~]$ pkg-config systemd --variable=systemdsystemunitdir  
/usr/lib/systemd/system  
[user1@server1 ~]$ pkg-config systemd --variable=systemduserconfdir  
/etc/systemd/user
```

Furthermore, there are additional system units that are created at runtime and destroyed when they are no longer needed. They are located in the `/run/systemd/system` directory. These runtime unit files take precedence over the system unit files, and the user unit files take priority over the runtime files.



Unit configuration files are a direct replacement of the initialization scripts found in the `/etc/rc.d/init.d` directory in older RHEL releases.

systemd includes 11 unit types, which are described in [Table 12-1](#).

Unit Type	Description
Automount	Offers automount capabilities for on-demand mounting systems
Device	Exposes kernel devices in systemd and may be used to implement device-based activation
Mount	Controls when and how to mount or unmount file systems
Path	Activates a service when monitored files or directories are accessed
Scope	Manages foreign processes instead of starting them
Service	Starts, stops, restarts, or reloads service daemons and processes they are made up of
Slice	May be used to group units, which manage system processes in a tree-like structure for resource management
Socket	Encapsulates local inter-process communication or network sockets for use by matching service units
Swap	Encapsulates swap partitions
Target	Defines logical grouping of units
Timer	Useful for triggering activation of other units based on time

Table 12-1 systemd Unit Types

Unit files contain common and specific configuration elements. Common elements fall under the [Unit] and [Install] sections, and comprise the description, documentation location, dependency information, conflict information, and other options that are independent of the type of unit. The unit-specific configuration data is located under the unit type section: [Service] for the service unit type, [Socket] for the socket unit type, and so forth. A sample unit file for `sshd.service` is shown below from the `/usr/lib/systemd/system` directory:

```
[user1@server1 ~]$ cat /usr/lib/systemd/system/sshd.service
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.target
Wants=sshd-keygen.target

[Service]
Type=notify
EnvironmentFile=/etc/crypto-policies/back-ends/opensslserver.config
EnvironmentFile=/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D $OPTIONS ${CRYPTO_POLICY}
ExecReload=/bin/kill -HUP ${MAINPID}
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

Units can have dependency relationships among themselves based on a sequence (ordering) or a requirement. A sequence outlines one or more actions that need to be taken before or after the activation of a unit (the Before and After directives). A requirement specifies what must already be running (the Requires directive) or not running (the Conflicts directive) in order for the successful launch of a unit. For instance, the *graphical.target* unit file tells us that the system must already be operating in the multi-user mode and not in rescue mode in order for it to boot successfully into the graphical mode. Another option, Wants, may be used instead of Requires in the [Unit] or [Install] section so that the unit is not forced to fail activation if a required unit fails to start. Run **man systemd.unit** for details on systemd unit files.

There are a few other types of dependencies that you may see in other unit configuration files. systemd generally sets and maintains inter-service dependencies automatically; however, this can be done manually if needed.

Targets

Targets are simply logical collections of units. They are a special systemd unit type with the .target file extension. They share the directory locations with other unit configuration files. Targets are used to execute a series of units. This is typically true for booting the system to a desired operational run level with all the required services up and running. Some targets inherit services from other targets and add their own to them. systemd includes several predefined targets that are described in [Table 12-2](#).

Target	Description
halt	Shuts down and halts the system
poweroff	Shuts down and powers off the system
shutdown	Shuts down the system
rescue	Single-user target for running administrative and recovery functions. All local file systems are mounted. Some basic services are started, but networking remains disabled.
emergency	Runs an emergency shell. The root file system is mounted in read-only mode; other file systems are not mounted. Networking and other services remain disabled.
multi-user	Multi-user target with full network support, but without GUI
graphical	Multi-user target with full network support and GUI
reboot	Shuts down and reboots the system
default	A special soft link that points to the default system boot (multi-user.target or graphical.target)
hibernate	Puts the system into hibernation by saving the running state to the hard disk and powering it off. When powered up, the system restores from its saved state rather than booting up.

Table 12-2 systemd Targets

Target unit files contain all information under the [Unit] section, and it comprises the description, documentation location, and dependency and conflict information. A sample file for the *graphical.target* target is shown below from the */usr/lib/systemd/system* directory:

```
[user1@server1 ~]$ cat /usr/lib/systemd/system/graphical.target
[Unit]
Description=Graphical Interface
Documentation=man:systemd.special(7)
Requires=multi-user.target
Wants=display-manager.service
Conflicts=rescue.service rescue.target
After=multi-user.target rescue.service rescue.target display-manager.service
AllowIsolate=yes
```

The file shows four dependencies: Requires, Wants, Conflicts, and After. It suggests that the system must have already accomplished the *rescue.service*, *rescue.target*, *multi-user.target*, and *display-manager.service* levels in order to be declared running in the *graphical* target. Run **man systemd.target** for details on *systemd* targets.

The `systemctl` Command

systemd comes with a set of management tools for querying and controlling operations. The primary tool for interaction in this command suite is `systemctl`. The `systemctl` command performs administrative functions and supports plentiful subcommands and flags. [Table 12-3](#) lists and describes some common operations.

Subcommand	Description
daemon-reload	Re-reads and reloads all unit configuration files and recreates the entire user dependency tree.
enable (disable)	Activates (deactivates) a unit for automatic system boot
get-default (set-default)	Shows (sets) the default boot target
get-property (set-property)	Returns (sets) the value of a property
is-active	Checks whether a unit is running
is-enabled	Displays whether a unit is set to automatic system boot
is-failed	Checks whether a unit is in the failed state
isolate	Changes the running state of a system
kill	Terminates all processes for a unit
list-dependencies	Lists dependency tree for a unit
list-sockets	Lists units of type socket
list-unit-files	Lists installed unit files
list-units	Lists known units. This is the default when <code>systemctl</code> is executed without arguments.
mask (unmask)	Prohibits (permits) auto and manual activation of a unit to avoid potential race conditions
reload	Forces a running unit to re-read its configuration file. This action does not change the PID of the running unit.
restart	Stops a running unit and restarts it
show	Shows unit properties
start (stop)	Starts (stops) a unit
status	Presents the unit status information

Table 12-3 systemctl Subcommands

You will use a majority of these subcommands with `systemctl` going forward. Refer to the manual pages of the command for more details.

Listing and Viewing Units

The *systemctl* command is used to view and manage all types of units. The following examples demonstrate common operations pertaining to viewing and querying units.

To list all units that are currently loaded in memory along with their status and description, run the *systemctl* command without any options or subcommands. A long output is generated. The graphic below shows a few starting and concluding lines followed by a brief explanation.

```
[user1@server1 ~]$ systemctl
 _UNIT LOAD ACTIVE SUB DESCRIPTION
 proc-sys-fs-binfmt_misc.automount loaded active waiting Arbitrary Executable File
 sys-devices-pci0000:00-0000:00:01.1-ata2-host1-target1:0:0:0:0-block-sr0.device loaded active
 sys-devices-pci0000:00-0000:00:03.0-net-enp0s3.device loaded active plugged 82540EM Gigabit E
 sys-devices-pci0000:00-0000:00:05.0-sound-card0.device loaded active plugged 82801AA AC'97 Au
 sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0:2:0:0:0-block-sda-sdal.device loaded
 sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0:2:0:0:0-block-sda-sda2.device loaded
 sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0:2:0:0:0-block-sda-sda2.device loaded active
 sys-devices-platform-serial18250-tty-ttys0.device loaded active plugged /sys/devices/platform/
 sys-devices-platform-serial18250-tty-ttys1.device loaded active plugged /sys/devices/platform/
 sys-devices-platform-serial18250-tty-ttys2.device loaded active plugged /sys/devices/platform/
 sys-devices-platform-serial18250-tty-ttys3.device loaded active plugged /sys/devices/platform/
 sys-devices-virtual-block-dm\x2d0.device loaded active plugged /sys/devices/virtual/bloc
 sys-devices-virtual-block-dm\x2d1.device loaded active plugged /sys/devices/virtual/bloc
 sys-devices-virtual-net-virbr0.device loaded active plugged /sys/devices/virtual/net/
 sys-devices-virtual-net-virbr0\x2dnic.device loaded active plugged /sys/devices/virtual/net/
 sys-module-configs.device loaded active plugged /sys/module/configfs
 sys-subsystem-net-devices-enp0s3.device loaded active plugged 82540EM Gigabit Ethernet
 sys-subsystem-net-devices-virbr0.device loaded active plugged /sys/subsystem/net/device
 sys-subsystem-net-devices-virbr0\x2dnic.device loaded active plugged /sys/subsystem/net/devic
 .mount loaded active mounted Root Mount
 boot.mount loaded active mounted /boot
 dev-hugepages.mount loaded active mounted Huge Pages File System
 dev-mqueue.mount loaded active mounted POSIX Message Queue File
 mnt.mount loaded active mounted /mnt
 .....
 LOAD = Reflects whether the unit definition was properly loaded.
 ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
 SUB = The low-level unit activation state, values depend on unit type.

159 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
lines 138-167 (END)
```

Here is a breakdown of the graphic above: the UNIT column shows the name of the unit and its location in the tree, the LOAD column reflects whether the unit configuration file was properly loaded (other possibilities are not found, bad setting, error, and masked), the ACTIVE column returns the high-level activation state (other possible states are active, reloading, inactive, failed, activating, and deactivating), the SUB column depicts the low-level unit activation state (reports unit-specific information), and the DESCRIPTION column illustrates the unit's content and functionality.

By default, the *systemctl* command lists only the active units. You can use the --all option to include the inactive units too.

To list all (--all) active and inactive units of type (-t) socket:

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
avahi-daemon.socket	loaded	active	running	Avahi mDNS/DNS-SD Stack Activation Socket
cups.socket	loaded	active	running	CUPS Scheduler
dbus.socket	loaded	active	running	D-Bus System Message Bus Socket
dm-event.socket	loaded	active	listening	Device-mapper event daemon FIFOs
iscsid.socket	loaded	active	listening	Open-iSCSI iscsid Socket
iscsiuiuo.socket	loaded	active	listening	Open-iSCSI iscsiuiuo Socket
lvm2-lvmpolld.socket	loaded	active	listening	LVM2 poll daemon socket
multipathd.socket	loaded	active	listening	multipathd control socket
rpcbind.socket	loaded	active	running	RPCbind Server Activation Socket
spice-vdagentd.socket	loaded	active	listening	Activation socket for spice guest agent
sssd-kcm.socket	loaded	active	listening	SSSD Kerberos Cache Manager responder socket
syslog.socket	loaded	inactive	dead	Syslog Socket

To list all units of type socket (column 2) currently loaded in memory and the service they activate (column 3), sorted by the listening address (column 1):

LISTEN	UNIT	ACTIVATES
/run/avahi-daemon/socket	avahi-daemon.socket	avahi-daemon.service
/run/dbus/system_bus_socket	dbus.socket	dbus.service
/run/dmeventd-client	dm-event.socket	dm-event.service
/run/dmeventd-server	dm-event.socket	dm-event.service
/run/initctl	systemd-initctl.socket	systemd-initctl.service
/run/lvm/lvmpolld.socket	lvm2-lvmpolld.socket	lvm2-lvmpolld.service

To list all unit files (column 1) installed on the system and their current state (column 2). This will generate a long list of units in the output. The following only shows a selection.

UNIT FILE	STATE
proc-sys-fs-binfmt misc.autounmount	static
-.mount	generated
boot.mount	generated
dev-hugepages.mount	static
dev-mqueue.mount	static
mnt.mount	generated
proc-fs-nfsd.mount	static

To list all units that failed (--failed) to start at the last system boot:

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
* vboxadd-service.service	loaded	failed	failed	vboxadd-service.service
* vboxadd.service	loaded	failed	failed	vboxadd.service

LOAD = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB = The low-level unit activation state, values depend on unit type.

2 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.

To list the hierarchy of all dependencies (required and wanted units) for the current default target:

```
[user1@server1 ~]$ systemctl list-dependencies
default.target
• └─accounts-daemon.service
• └─gdm.service
• └─rtkit-daemon.service
• └─systemd-update-utmp-runlevel.service
• └─udisks2.service
• └─multi-user.target
•   └─atd.service
• └─auditd.service
```

To list the hierarchy of all dependencies (required and wanted units) for a specific unit such as `atd.service`:

```
[user1@server1 ~]$ systemctl list-dependencies atd.service
atd.service
• └─system.slice
•   └─sysinit.target
•     └─dev-hugepages.mount
•     └─dev-mqueue.mount
•     └─dracut-shutdown.service
•     └─import-state.service
• └─iscsi.service
```

There are other listing subcommands and additional flags available that can be used to produce a variety of reports.

Managing Service Units

The `systemctl` command offers several subcommands to manage service units, including starting, stopping, restarting, and checking their status. These and other management operations are summarized in [Table 12-3](#). The following examples demonstrate their use on a service unit called `atd`.

To check the current operational status and other details for the `atd` service:

```
[user1@server1 ~]$ systemctl status atd
● atd.service - Job spooling tools
  Loaded: loaded (/usr/lib/systemd/system/atd.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2019-10-09 07:42:00 EDT; 1 day 4h ago
    Main PID: 942 (atd)
       Tasks: 1 (limit: 5064)
      Memory: 496.0K
        CGroup: /system.slice/atd.service
                 └─942 /usr/sbin/atd -f
```

The above output reveals a lot of information about the `atd` service. On line 1, it shows the service description (read from the `/usr/lib/systemd/system/atd.service` file). Line 2 illustrates the load status, which reveals the current load status of the unit configuration file in memory. Other possibilities for "Loaded" include "error" (if there was a problem loading the file), "not-found" (if no file associated with this unit was found), "bad-setting" (if a key setting was missing), and "masked" (if the unit configuration

file is masked). Line 2 also tells us whether the service is set (enable or disable) for autostart at system boot.

Line 3 exhibits the current activation status and the time the service was started. An activation status designates the current state of the service. Possible states include:

- Active (running):** The service is running with one or more processes
- Active (exited):** Completed a one-time configuration
- Active (waiting):** Running but waiting for an event
- Inactive:** Not running
- Activating:** In the process of being activated
- Deactivating:** In the process of being deactivated
- Failed:** If the service crashed or could not be started

The output also depicts the PID of the service process and other information.

To disable the *atd* service from autostarting at the next system reboot:

```
[user1@server1 ~]$ sudo systemctl disable atd
Removed /etc/systemd/system/multi-user.target.wants/atd.service.
```

To re-enable *atd* to autostart at the next system reboot:

```
[user1@server1 ~]$ sudo systemctl enable atd
Created symlink /etc/systemd/system/multi-user.target.wants/atd.service → /usr/lib/systemd/system/atd.service.
```

To check whether *atd* is set to autostart at the next system reboot:

```
[user1@server1 ~]$ systemctl is-enabled atd
enabled
```

To check whether the *atd* service is running:

```
[user1@server1 ~]$ systemctl is-active atd
active
```

To stop and restart *atd*, run either of the following:

```
[user1@server1 ~]$ sudo systemctl stop atd ; sudo systemctl start atd
[user1@server1 ~]$ sudo systemctl restart atd
```

To show the details of the *atd* service:

```
[user1@server1 ~]$ systemctl show atd
Type=simple
Restart=no
NotifyAccess=none
RestartUSec=100ms
....
```

To prohibit *atd* from being enabled or disabled:

```
[user1@server1 ~]# sudo systemctl mask atd  
Created symlink /etc/systemd/system/atd.service → /dev/null.
```

Try disabling or enabling *atd* and observe the effect of the previous command:

```
[user1@server1 ~]# sudo systemctl disable atd  
Unit /etc/systemd/system/atd.service is masked, ignoring.  
[user1@server1 ~]# sudo systemctl enable atd  
Failed to enable unit: Unit file /etc/systemd/system/atd.service is masked.
```

Reverse the effect of the *mask* subcommand and try disable and enable operations:

```
[user1@server1 ~]# sudo systemctl unmask atd  
Removed /etc/systemd/system/atd.service.  
[user1@server1 ~]#  
[user1@server1 ~]# sudo systemctl disable atd  
Removed /etc/systemd/system/multi-user.target.wants/atd.service.  
[user1@server1 ~]# sudo systemctl enable atd  
Created symlink /etc/systemd/system/multi-user.target.wants/atd.service → /usr/lib/systemd/system/  
atd.service.
```

Notice that the *unmask* subcommand has removed the restriction that was placed on the *atd* service.

Managing Target Units

The *systemctl* command is also used to manage the target units. It can be used to view or change the default boot target, switch from one running target into another, and so on. These operations are briefed in [Table 12-3](#). Let's look at some examples.

To view what units of type (-t) target are currently loaded and active:

```
[user1@server1 ~]$ systemctl -t target
UNIT           LOAD   ACTIVE SUB   DESCRIPTION
basic.target    loaded  active  active Basic System
cryptsetup.target loaded  active  active Local Encrypted Volumes
getty.target    loaded  active  active Login Prompts
graphical.target loaded  active  active Graphical Interface
local-fs-pre.target loaded  active  active Local File Systems (Pre)
local-fs.target  loaded  active  active Local File Systems
multi-user.target loaded  active  active Multi-User System
network-online.target loaded  active  active Network is Online
network-pre.target loaded  active  active Network (Pre)
network.target  loaded  active  active Network
nfs-client.target loaded  active  active NFS client services
nss-user-lookup.target loaded  active  active User and Group Name Lookups
paths.target    loaded  active  active Paths
remote-fs-pre.target loaded  active  active Remote File Systems (Pre)
remote-fs.target  loaded  active  active Remote File Systems
rpc_pipefs.target loaded  active  active rpc_pipefs.target
rpcbind.target  loaded  active  active RPC Port Mapper
slices.target   loaded  active  active Slices
sockets.target  loaded  active  active Sockets
sound.target    loaded  active  active Sound Card
sshd-keygen.target loaded  active  active sshd-keygen.target
swap.target     loaded  active  active Swap
sysinit.target  loaded  active  active System Initialization
timers.target   loaded  active  active Timers

LOAD  = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB   = The low-level unit activation state, values depend on unit type.

24 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

For each target unit, the above output returns the target unit's name, load state, high-level and low-level activation states, and a short description. Add the `--all` option to the above to see all loaded targets in either active or inactive state.

Viewing and Setting Default Boot Target

The `systemctl` command is used to view the current default boot target and to set it. Let's use the `get-default` and `set-default` subcommands with `systemctl` to perform these operations.

To check the current default boot target:

```
[user1@server1 ~]$ systemctl get-default
graphical.target
```

EXAM TIP: You may have to modify the default boot target persistently.

To change the current default boot target from `graphical.target` to `multi-user.target`:

```
[user1@server1 ~]$ sudo systemctl set-default multi-user
Removed /etc/systemd/system/default.target.
Created symlink /etc/systemd/system/default.target → /usr/lib/systemd/system/multi-user.target.
```

The command simply removes the existing symlink (*default.target*) pointing to the old boot target and replaces it with the new target file path.

Execute **sudo systemctl set-default graphical** to revert the default boot target to graphical.

Switching into Specific Targets

The *systemctl* command can be used to transition the running system from one target state into another. There are a variety of potential targets available to switch into as listed in [Table 12-2](#); however, only a few of them—graphical, multi-user, reboot, shutdown—are typically used. The rescue and emergency targets are for troubleshooting and system recovery purposes, poweroff and halt are similar to shutdown, and hibernate is suitable for mobile devices. Consider the following examples that demonstrate switching targets.

The current default target on *server1* is graphical. To switch into multi-user, use the *isolate* subcommand with *systemctl*:

```
[user1@server1 ~]$ sudo systemctl isolate multi-user
```

This should stop the graphical service on the system and display the text-based console login screen, as shown below:

```
Red Hat Enterprise Linux 8.0 (Ootpa)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

Activate the web console with: systemctl enable --now cockpit.socket

server1 login:
```

Type in a username such as *user1* and enter the password to log in:

```
server1 login: user1
Password:
Last login: Wed Oct  9 21:36:56 from 192.168.0.13
[user1@server1 ~]$
```

To return to the graphical target:

```
[user1@server1 ~]$ sudo systemctl isolate graphical
```

The graphical login screen should appear shortly and you should be able to log back in.

To shut down the system and power it off, use the following or simply run the *poweroff* command:

```
[user1@server1 ~]$ sudo systemctl poweroff
```

To shut down and reboot the system, use the following or simply run the *reboot* command:

```
[user1@server1 ~]$ sudo systemctl reboot
```

The *halt*, *poweroff*, and *reboot* commands are mere symbolic links to the *systemctl* command, as the following long listing suggests:

```
[user1@server1 ~]$ ls -l /usr/sbin/halt /usr/sbin/poweroff /usr/sbin/reboot
lrwxrwxrwx. 1 root root 16 Feb 26 2019 /usr/sbin/halt -> ../bin/systemctl
lrwxrwxrwx. 1 root root 16 Feb 26 2019 /usr/sbin/poweroff -> ../bin/systemctl
lrwxrwxrwx. 1 root root 16 Feb 26 2019 /usr/sbin/reboot -> ../bin/systemctl
```

The *halt*, *poweroff*, and *reboot* commands are available in RHEL 8 for compatibility reasons only. It is recommended to use the *systemctl* command instead when switching system states.

The three commands, without any arguments, perform the same action that the *shutdown* command would with the “-H now”, “-P now”, and “-r now” arguments, respectively. In addition, it also broadcasts a warning message to all logged-in users, blocks new user login attempts, waits for the specified amount of time for users to save their work and log off, stops the services, and eventually shut the system down to the specified target state.

System Logging

System logging (*syslog* for short) is one of the most rudimentary elements of an operating system. Its purpose is to capture messages generated by the kernel, daemons, commands, user activities, applications, and other events, and forwarded them to various log files, which store them for security auditing, service malfunctioning, system troubleshooting, or informational purposes.

The daemon that is responsible for system logging is called *rsyslogd* (*rocket-fast system for log processing*). This service daemon is multi-threaded, with support for enhanced filtering, encryption-protected message relaying, and a variety of configuration options. The *rsyslogd* daemon reads its configuration file */etc/rsyslog.conf* and the configuration files located in the */etc/rsyslog.d* directory at startup. The default depository for most system log files is the */var/log* directory. Other services such as audit, Apache, and GNOME desktop manager have subdirectories under */var/log* for storing their respective log files.

The *rsyslog* service is modular, allowing the modules listed in its configuration file to be dynamically loaded in the kernel as and when needed. Each module brings a new functionality to the system upon loading.

The *rsyslogd* daemon can be stopped manually using **systemctl stop rsyslog**. Replace stop with start, restart, reload, and status as appropriate.

A PID is assigned to the daemon at startup and a file by the name *rsyslogd.pid* is created in the */run* directory to save the PID. The reason this file is created and stores the PID is to prevent the initiation of multiple instances of this daemon.

The Syslog Configuration File

The *rsyslog.conf* is the primary syslog configuration file located in the */etc* directory . The default uncommented line entries from the file are shown below and explained thereafter. Section headings have been added to separate the directives in each section.

```
[user1@server1 ~]$ sudo grep '^#|^$' /etc/rsyslog.conf
#####
# Modules
module(load="imuxsock"      # provides support for local system logging (e.g. via logger)
       SysSock.Use="off") # Turn off message reception via local log socket;
                           # local messages are retrieved through imjournal now.
module(load="imjournal"      # provides access to the systemd journal
       StateFile="imjournal.state") # File to store the position in the journal
global(workDirectory="/var/lib/rsyslog")
#####
# Global Directives
global(workDirectory="/var/lib/rsyslog")
module(load="builtin:omfile" Template="RSYSLOG_TraditionalFileFormat")
include(file="/etc/rsyslog.d/*.conf" mode="optional")
#####
# Rules
*.info;mail.none;authpriv.none;cron.none          /var/log/messages
authpriv.*                                         /var/log/secure
mail.*                                            -/var/log/maillog
cron.*                                           /var/log/cron
*.emerg                                         :omusrmmsg:*
uucp,news.crit                                    /var/log/spooler
local7.*                                         /var/log/boot.log
```

As depicted, the syslog configuration file contains three sections: Modules, Global Directives, and Rules. The Modules section defines two modules—*imuxsock* and *imjournal*—and they are loaded on demand. The *imuxsock* module furnishes support for local system logging via the *logger* command, and the *imjournal* module allows access to the systemd journal.

The Global Directives section contains three active directives. The definitions in this section influence the overall functionality of the *rsyslog* service. The first directive sets the location for the storage of auxiliary files (*/var/lib/rsyslog*). The second directive instructs the *rsyslog* service to save captured messages

using traditional file formatting. The third directive directs the service to load additional configuration from files located in the `/etc/rsyslog.d/` directory.

The Rules section has many two-field line entries. The left field is called *selector*, and the right field is referred to as *action*. The selector field is further divided into two period-separated sub-fields called *facility* (left) and *priority* (right), with the former representing one or more system process categories that generate messages, and the latter identifying the severity associated with the messages. The semicolon (;) character is used as a distinction mark if multiple facility.priority groups are present. The action field determines the destination to send the messages to.

There are numerous supported facilities such as auth, authpriv, cron, daemon, kern, lpr, mail, news, syslog, user, uucp, and local0 through local7. The asterisk (*) character represents all of them.

Similarly, there are several supported priorities, and they include emerg, alert, crit, error, warning, notice, info, debug, and none. This sequence is in the descending criticality order. The asterisk (*) represents all of them. If a lower priority is selected, the daemon logs all messages of the service at that and higher levels.

Line 1 under the Rules section instructs the syslog daemon to catch and store informational messages from all services to the `/var/log/messages` file and ignore all messages generated by mail, authentication, and cron services.

Lines 2, 3, and 4 command the daemon to collect and log all messages produced by authentication, mail, and cron to the *secure*, *maillog*, and *cron* files, respectively. Line 5 orders the daemon to display emergency messages (*omusrmmsg* stands for *user message output module*) on the terminals of all logged-in users. Line 6 shows two comma-separated facilities that are set at a common priority. These facilities tell the daemon to gather critical messages from uucp and news facilities and log them to the `/var/log/spooler` file. Line 7 (the last line) is for recording the boot-time service startup status to the `/var/log/boot.log` file.

If you have made any modifications to the syslog configuration file, you need to run the `rsyslogd` command with the -N switch and specify a numeric verbosity level to inspect the file for any syntax or typing issues:

```
[user1@server1 ~]$ sudo rsyslogd -N 1
rsyslogd: version 8.37.0-9.el8, config validation run (level 1), master config /etc/rsyslog.conf
rsyslogd: End of config validation run. Bye.
```

The validation returns the version of the command, verbosity level used, and the configuration file path. With no issues reported, the `rsyslog` service can be restarted (or reloaded) in order for the changes to take effect.

Rotating Log Files

RHEL records all system activities in log files that are stored in a central location under the `/var/log` directory, as defined in the `rsyslog` configuration file. A long listing of this directory reveals the files along with subdirectories that may have multiple service-specific logs. Here is a listing from `server1`:

```
[user1@server1 ~]$ ls -l /var/log
total 6852
drwxr-xr-x. 2 root  root      250 Aug 23 06:27 anaconda
drwx----- 2 root  root      23 Aug 23 06:30 audit
-rw----- 1 root  root      0 Oct 15 03:33 boot.log
-rw-rw--- 1 root  utmp      0 Oct  1 03:23 btmp
-rw-rw--- 1 root  utmp      0 Aug 23 06:08 btmp-20191001
drwxr-xr-x. 2 chrony chrony     6 Aug 13 2018 chrony
-rw----- 1 root  root  15578 Oct 15 07:01 cron
drwxr-xr-x. 2 lp   sys       135 Sep 29 13:39 cups
-rw----- 1 root  root  656807 Oct 15 07:23 dnf.log
-rw----- 1 root  root  534386 Oct  6 03:05 dnf.log-20191006
-rw----- 1 root  root  212004 Oct 13 02:35 dnf.log-20191013
-rw----- 1 root  root  6424 Oct 15 07:23 dnf.rpm.log
-rw----- 1 root  root  2100 Oct  6 23:33 firewalld
drwx--x--x. 2 root  gdm      6 Feb 11 2019 gdm
drwxr-xr-x. 2 root  root      6 Mar  8 2019 glusterfs
drwxr-xr-x. 2 root  root      6 Aug 23 06:16 insights-client
-rw-rw-r-- 1 root  utmp  295504 Oct 14 17:49 lastlog
-rw----- 1 root  root      0 Oct 13 03:17 maillog
-rw----- 1 root  root  782565 Oct 15 07:23 messages
-rw----- 1 root  root  1430871 Oct  6 03:05 messages-20191006
-rw----- 1 root  root  3133990 Oct 13 02:35 messages-20191013
drwx----- 2 root  root      6 Aug 23 06:08 private
-rw----- 1 root  root  46185 Oct 15 07:56 secure
-rw----- 1 root  root      0 Oct 13 03:17 spooler
drwxr-xr-x. 3 root  root      21 Aug 23 06:15 swtpm
-rw-rw-r-- 1 root  utmp  77568 Oct 14 15:12 wtmp
-rw-r--r-- 1 root  root  18990 Aug 23 06:31 Xorg.9.log
```

The output shows log files for various services. Depending on the usage and the number of events generated and captured, log files may quickly fill up the `/var` file system, resulting in unpredictable system behavior. Also, they may grow to an extent that would make it difficult to load, read, send, or analyze them. To avoid getting into any unwanted situation, it's important to ensure that they're rotated on a regular basis and their archives are removed automatically.

To that end, a script called `logrotate` under `/etc/cron.daily/` invokes the `logrotate` command on a daily basis. Via the Anacron service, the command runs a rotation as per the schedule defined in the `/etc/logrotate.conf` file and the configuration files for various services located in the `/etc/logrotate.d` directory. The configuration files may be modified to alter the schedule or include additional tasks such as removing, compressing, and emailing selected log files.

Here is what the `/etc/cron.daily/logrotate` script contains:

```
[user1@server1 ~]$ cat /etc/cron.daily/logrotate
#!/bin/sh

/usr/sbin/logrotate /etc/logrotate.conf
EXITVALUE=$?
if [ $EXITVALUE != 0 ]; then
    /usr/bin/logger -t logrotate "ALERT exited abnormally with [$EXITVALUE]"
fi
exit $EXITVALUE
```

The following returns the default content of the */etc/logrotate.conf* file:

```
[user1@server1 ~]$ cat /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# use date as a suffix of the rotated file
dateext

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d
```

The file content shows the default log rotation frequency (weekly). It indicates the period of time (4 weeks) to retain the rotated logs before deleting them. Each time a log file is rotated, an empty replacement file is created with the date as a suffix to its name, and the *rsyslog* service is restarted. The script presents the option of compressing the rotated files using the *gzip* utility. During the script execution, the *logrotate* command checks for the presence of additional log configuration files in the */etc/logrotate.d* directory and includes them as necessary. The directives defined in the *logrotate.conf* file have a global effect on all log files. You can define custom settings for a specific log file in *logrotate.conf* or create a separate file in the */etc/logrotate.d* directory. Any settings defined in user-defined files overrides the global settings.

The */etc/logrotate.d* directory includes additional configuration files for other service logs, as shown below:

```
[user1@server1 ~]$ ls -l /etc/logrotate.d/
total 64
-rw-r--r--. 1 root root 91 Feb  8 2019 bootlog
-rw-r--r--. 1 root root 130 Feb 19 2018 btmp
-rw-r--r--. 1 root root 160 Apr  4 2018 chrony
-rw-r--r--. 1 root root 71 Dec 14 2018 cups
-rw-r--r--. 1 root root 526 Jan  4 2019 dnf
-rw-r--r--. 1 root root 172 Jul  9 2018 iscsiuiolog
-rw-r--r--. 1 root root 165 Feb 15 2019 libvirtd
-rw-r--r--. 1 root root 142 Feb 15 2019 libvirtd.gemu
-rw-r--r--. 1 root root 106 Jun  2 2015 numad
-rw-r--r--. 1 root root 408 Aug 12 2018 psacct
-rw-r--r--. 1 root root 115 Jan  7 2019 samba
-rw-r--r--. 1 root root 237 Feb 11 2019 sssd
-rw-r--r--. 1 root root 226 Dec 17 2018 syslog
-rw-r--r--. 1 root root 32 Feb 19 2018 up2date
-rw-r--r--. 1 root root 100 Feb  8 2019 wpa_supplicant
-rw-r--r--. 1 root root 145 Feb 19 2018 wtmp
```

Here there are log configuration files for a number of services—*chrony*, *cups*, *dnf*, and *samba*—all with their own rules defined. The following shows the file content for *btmp* (records of failed user login attempts) that is used to control the rotation behavior for the */var/log/btmp* file:

```
[user1@server1 ~]$ cat /etc/logrotate.d/btmp
# no packages own btmp -- we'll rotate it here
/var/log/btmp {
    missingok
    monthly
    create 0600 root utmp
    rotate 1
}
```

The rotation is once a month. The replacement file created will get read/write permission bits for the owner (*root*), the owning group will be set to *utmp*, and the *rsyslog* service will maintain one rotated copy of the *btmp* log file.

The Boot Log File

Logs generated during the system startup display the service startup sequence with a status showing whether the service was started successfully. This information may help in any post-boot troubleshooting if required. Boot logs are stored in the *boot.log* file under */var/log*. Here are a few lines from the beginning of the file:

```
[user1@server1 ~]$ sudo head /var/log/boot.log
----- Tue Oct 15 09:16:18 EDT 2019 -----
[ OK ] Stopped Journal Service.
      Starting Journal Service...
[ OK ] Listening on initctl Compatibility Named Pipe.
[ OK ] Listening on Process Core Dump Socket.
[ OK ] Created slice system-getty.slice.
      Starting Read and set NIS domainname from /etc/sysconfig/network...
[ OK ] Listening on udev Kernel Socket.
[ OK ] Set up automount Arbitrary Executable File Formats File System Automount Point.
[ OK ] Listening on Device-mapper event daemon FIFOs.
```

OK or FAILED within the square brackets indicates if the service was started successfully or not.

The System Log File

The default location for storing most system activities, as defined in the *rsyslog.conf* file, is the */var/log/messages* file. This file saves log information in plain text format and may be viewed with any file display utility, such as *cat*, *more*, *pg*, *less*, *head*, or *tail*. This file may be observed in real time using the *tail* command with the -f switch. The *messages* file captures the date and time of the activity, hostname of the system, name and PID of the service, and a short description of the event being logged.

EXAM TIP: It is helpful to “tail” the messages file when starting or restarting a system service or during testing to identify any issues encountered.

The following illustrates some recent entries from this file:

```
[user1@server1 ~]$ sudo tail /var/log/messages
Oct 15 09:36:24 server1 dnf[2303]: Metadata cache refreshed recently.
Oct 15 09:36:25 server1 systemd[1]: Started dnf makecache.
Oct 15 09:41:23 server1 systemd[1]: Starting Cleanup of Temporary Directories...
Oct 15 09:41:23 server1 systemd-tmpfiles[2349]: [/usr/lib/tmpfiles.d/libstoragemgmt.conf:1] Line references
path below legacy directory /var/run/, updating /var/run/lsm -> /run/lsm; please update the tmpfiles.d/ drop-
in file accordingly.
Oct 15 09:41:23 server1 systemd-tmpfiles[2349]: [/usr/lib/tmpfiles.d/libstoragemgmt.conf:2] Line references
path below legacy directory /var/run/, updating /var/run/lsm/ ipc -> /run/lsm/ ipc; please update the tmpfiles.
d/ drop-in file accordingly.
Oct 15 09:41:23 server1 systemd-tmpfiles[2349]: [/usr/lib/tmpfiles.d/mdadm.conf:1] Line references path belo
w legacy directory /var/run/, updating /var/run/mdadm -> /run/mdadm; please update the tmpfiles.d/ drop-in fi
le accordingly.
Oct 15 09:41:23 server1 systemd-tmpfiles[2349]: [/usr/lib/tmpfiles.d/radvd.conf:1] Line references path belo
w legacy directory /var/run/, updating /var/run/radvd -> /run/radvd; please update the tmpfiles.d/ drop-in fi
le accordingly.
```

Each line entry represents the detail for one event.

Logging Custom Messages

Many times it is worthwhile to add a manual note to the system log file to mark the start or end of an activity for future reference. This is especially important when you run a script to carry out certain tasks and you want to record the status or add comments at various stages throughout its execution. This is also beneficial in debugging the startup of an application to know where exactly it is failing.

The Modules section in the *rsyslog.conf* file provides the support via the *imuxsock* module to record custom messages to the *messages* file using the *logger* command. This command may be run by normal users or the *root* user. The following example shows how to add a note indicating the calling user has rebooted the system:

```
[user1@server1 ~]$ logger -i "System rebooted by $USER"
```

tail the last line from the *messages* file and you'll observe the message recorded along with the timestamp, hostname, and PID:

```
[user1@server1 ~]$ sudo tail -1 /var/log/messages
Oct 15 09:59:22 server1 user1[2533]: System rebooted by user1
```

You may add the *-p* option and specify a priority level either as a numerical value or in the facility.priority format. The default priority at which the events are recorded is user.notice. See the manual pages for the *logger* command for more details.

The systemd Journal

In addition to the *rsyslog* service, RHEL offers a systemd-based logging service for the collection and storage of logging data. This service is implemented via the *systemd-journald* daemon. The function of this service is to gather, store, and display logging events from a variety of sources such as the kernel, *rsyslog* and other services, initial RAM disk, and alerts generated during the early boot stage. It stores these messages in the binary format in files called *journals* that are located in the */run/log/journal* directory. These files are structured and indexed for faster and easier searches, and may be viewed and managed using the *journalctl* command. As you know, */run* is a virtual file system that is created in memory at system boot, maintained during system runtime, and destroyed at shutdown. Therefore, the data stored therein is non-persistent, but you can enable persistent storage for the logs if desired.

RHEL runs both *rsyslogd* and *systemd-journald* concurrently. In fact, the data gathered by *systemd-journald* may be forwarded to *rsyslogd* for further processing and persistent storage in text format.

The main configuration file for this service is */etc/systemd/journald.conf*, which contains numerous default settings that affect the overall functionality of the service. These settings may be modified as required.

Retrieving and Viewing Messages

RHEL provides the *journalctl* command to retrieve messages from the journal for viewing in a variety of ways using different options. One common usage is to run the command without any options to see all the messages generated since the last system reboot. The following shows a few initial entries from the journal:

```
[user1@server1 ~]$ sudo journalctl
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:15:01 EDT. --
Oct 15 09:25:41 server1.example.com kernel: Linux version 4.18.0-80.11.2.el8_0.x86_64 (mockbuild@x86-vm-07.>
Oct 15 09:25:41 server1.example.com kernel: Command line: BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-80.11.2.el8_0.x86_64 root=UUID=01caa6ef5448f53266b ro quiet
Oct 15 09:25:41 server1.example.com kernel: x86/fpu: Supporting XSAVE feature 0x0001: 'x87 floating point registers'
Oct 15 09:25:41 server1.example.com kernel: x86/fpu: Supporting XSAVE feature 0x0002: 'SSE registers'
Oct 15 09:25:41 server1.example.com kernel: x86/fpu: Supporting XSAVE feature 0x0004: 'AVX registers'
Oct 15 09:25:41 server1.example.com kernel: x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256
Oct 15 09:25:41 server1.example.com kernel: x86/fpu: Enabled xstate features 0x7, context size is 832 bytes
Oct 15 09:25:41 server1.example.com kernel: BIOS-provided physical RAM map:
```

Notice that the format of the messages is similar to that of the events logged to the `/var/log/messages` file that you saw earlier. Each line begins with a timestamp followed by the system hostname, process name with or without a PID, and the actual message.

Let's run the `journalctl` command with different options to produce various reports.

To display detailed output for each entry, use the `-o verbose` option:

```
[user1@server1 ~]$ sudo journalctl -o verbose
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:19:09 EDT. --
Tue 2019-10-15 09:25:41.923932 EDT [s=ffd2564870b64db7a6a4ca448f53266b;i=1;b=e75da4e8e6fc4ebb867dd01caa6ef5448f53266b]
  _SOURCE_MONOTONIC_TIMESTAMP=0
  _TRANSPORT=kernel
  PRIORITY=5
  SYSLOG_FACILITY=0
  SYSLOG_IDENTIFIER=kernel
  MESSAGE=Linux version 4.18.0-80.11.2.el8_0.x86_64 (mockbuild@x86-vm-07.build.eng.bos.redhat.com) (gcc v
  _BOOT_ID=e75da4e8e6fc4ebb867dd01caa6ef544
  _MACHINE_ID=2d06e883e31f434f95d9df6035e988e7
  _HOSTNAME=server1.example.com
Tue 2019-10-15 09:25:41.924033 EDT [s=ffd2564870b64db7a6a4ca448f53266b;i=2;b=e75da4e8e6fc4ebb867dd01caa6ef5448f53266b]
  _SOURCE_MONOTONIC_TIMESTAMP=0
  ....
```

To view all events since the last system reboot, use the `-b` options:

```
[user1@server1 ~]$ sudo journalctl -b
```

You may specify `-0` (default, since the last system reboot), `-1` (the previous system reboot), `-2` (two reboots before), and so on to view messages from previous system reboots.

To view only kernel-generated alerts since the last system reboot:

```
[user1@server1 ~]$ sudo journalctl -kb0
```

To limit the output to view a specific number of entries only (3 in the example below), use the `-n` option:

```
[user1@server1 ~]$ sudo journalctl -n3
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:29:36 EDT. --
Oct 15 12:29:36 server1.example.com sudo[4160]: user1 : TTY pts/0 ; PWD=/home/user1 ; USER=root ; COMMAND=/bin/su -
Oct 15 12:29:36 server1.example.com sudo[4160]: pam_systemd(sudo:session): Cannot create session: Already running session
Oct 15 12:29:36 server1.example.com sudo[4160]: pam_unix(sudo:session): session opened for user root by user1
```

To show all alerts generated by a particular service, such as `crond`:

```
[user1@server1 ~]$ sudo journalctl /usr/sbin/crond
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:31:15 EDT. --
Oct 15 09:25:56 server1.example.com crond[912]: (CRON) STARTUP (1.5.2)
Oct 15 09:25:56 server1.example.com crond[912]: (CRON) INFO (Syslog will be used instead of sendmail.)
Oct 15 09:25:56 server1.example.com crond[912]: (CRON) INFO (RANDOM_DELAY will be scaled with factor 9% if -->
Oct 15 09:25:59 server1.example.com crond[912]: (CRON) INFO (running with inotify support)
```

To retrieve all messages logged for a certain process, such as the PID associated with the *chrony* service:

```
[user1@server1 ~]$ sudo journalctl _PID=6 (pgrep chronyd)
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:35:39 EDT. --
Oct 15 09:25:51 server1.example.com chronyd[784]: chronyd version 3.3 starting (+CMDMON +NTP +REFCLOCK +RTC)
Oct 15 09:25:51 server1.example.com chronyd[784]: Frequency -33.628 +/- 4.368 ppm read from /var/lib/chrony
Oct 15 09:25:51 server1.example.com chronyd[784]: Using right/UTC timezone to obtain leap second data
Oct 15 09:26:10 server1.example.com chronyd[784]: Selected source 207.34.49.172
Oct 15 09:26:10 server1.example.com chronyd[784]: System clock TAI offset set to 37 seconds
```

To reveal all messages for a particular system unit, such as *sshd.service*:

```
[user1@server1 ~]$ sudo journalctl _SYSTEMD_UNIT=sshd.service
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:39:53 EDT. --
Oct 15 09:25:56 server1.example.com sshd[894]: Server listening on 0.0.0.0 port 22.
Oct 15 09:25:56 server1.example.com sshd[894]: Server listening on :: port 22.
Oct 15 09:27:40 server1.example.com sshd[2150]: Accepted password for user1 from 192.168.0.219 port 49876 s
Oct 15 09:27:41 server1.example.com sshd[2150]: pam_unix(sshd:session): session opened for user user1 by (u)
```

To view all error messages logged between a date range, such as October 10, 2019 and October 16, 2019:

```
[user1@server1 ~]$ sudo journalctl --since 2019-10-10 --until 2019-10-16 -p err
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:44:26 EDT. --
Oct 15 09:25:43 server1.example.com kernel: [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send log
Oct 15 09:25:43 server1.example.com kernel: [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send log
Oct 15 09:25:52 server1.example.com ssasd[788]: Could not open file [/var/log/sssd/sssd.log]. Error: [2][No
```

To get all warning messages that have appeared today and display them in reverse chronological order:

```
[user1@server1 ~]$ sudo journalctl --since today -p warning -r
-- Logs begin at Tue 2019-10-15 09:25:41 EDT, end at Tue 2019-10-15 12:46:28 EDT. --
Oct 15 12:29:31 server1.example.com packagekitd[4154]: g_path_get_basename: assertion 'file_name != NULL' failed
Oct 15 11:55:19 server1.example.com packagekitd[3672]: g_path_get_basename: assertion 'file_name != NULL' failed
Oct 15 09:26:32 server1.example.com kernel: [drm:vmw_sou_crtc_page_flip [vmwgfx]] *ERROR* Page flip error -1
Oct 15 09:26:32 server1.example.com gnome-shell[1076]: Failed to flip: Device or resource busy
Oct 15 09:26:30 server1.example.com gsd-color[1891]: unable to get EDID for xrandr-Virtual-1: unable to get
```

You can specify the time range in hh:mm:ss format, or yesterday, today, or tomorrow instead.

Similar to the *-f* (follow) option that is used with the *tail* command for real-time viewing of a log file, you can use the same switch with *journalctl* as well. Press *Ctrl+c* to terminate.

```
[user1@server1 ~]$ sudo journalctl -f
```

Check the manual pages of the *journalctl* command and the *systemd-journald* service for more details.

Preserving Journal Information

By default, journals are stored in the `/run/log/journal` directory for the duration of system runtime. This data is transient and it does not survive across reboots. The `journalctl` command examples demonstrated earlier retrieve journal information from this temporary location. The `rsyslogd` daemon, by default, reads the temporary journals and stores messages in the `/var/log/messages` file. You can enable a separate storage location for the journal to save all its messages there persistently. The default is under the `/var/log/journal` directory. This will make the journal information available for future reference.

The `systemd-journald` service supports four options with the Storage directive in its configuration file `journald.conf` to control how the logging data is handled. These options are described in [Table 12-4](#).

Option	Description
volatile	Stores data in memory only
persistent	Stores data permanently under <code>/var/log/journal</code> . Falls back to memory-only option if this directory does not exist or has a permission or other issue. The service creates <code>/var/log/journal</code> in case of its non-existence.
auto	Similar to “persistent” but does not create <code>/var/log/journal</code> if it does not exist. This is the default option.
none	Disables both volatile and persistent storage. Not recommended.

Table 12-4 Journal Data Storage Options

The default (auto) option appears more suitable as it stores data in both volatile and on-disk storage; however, you need to create the `/var/log/journal` directory manually. This option provides two fundamental benefits: faster query responses from in-memory storage and access to historical log data from on-disk storage.

Exercise 12-1: Configure Persistent Storage for Journal Information

This exercise should be done on `server1` as `user1` with `sudo` where required.

In this exercise, you will run the necessary steps to enable and confirm persistent storage for the journals.

1. Create a subdirectory called *journal* under the */var/log* directory and confirm:

```
[user1@server1 ~]$ sudo mkdir /var/log/journal  
[user1@server1 ~]$ ll -d /var/log/journal/  
drwxr-xr-x. 3 root root 46 Nov 5 16:11 /var/log/journal/
```

2. Restart the *systemd-journald* service and confirm:

```
[user1@server1 ~]$ sudo systemctl restart systemd-journald  
[user1@server1 ~]$ sudo systemctl status systemd-journald  
* systemd-journald.service - Journal Service  
  Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service; static; v|  
  Active: active (running) since Thu 2020-11-05 16:13:23 EST; 4s ago  
.....
```

3. List the new directory and observe a subdirectory matching the machine ID of the system as defined in the */etc/machine-id* file is created:

```
[user1@server1 ~]$ ll /var/log/journal/  
total 0  
drwxr-xr-x. 2 root root 53 Nov 5 16:12 2d06e883e31f434f95d9df6035e988e7  
[user1@server1 ~]$ cat /etc/machine-id  
2d06e883e31f434f95d9df6035e988e7
```

Compare the name of the subdirectory with the ID stored in the */etc/machine-id* file. They are identical.



This log file is rotated automatically once a month based on the settings in the *journald.conf* file. Check the manual pages of the configuration file for details and relevant directives.

This concludes the exercise.

System Tuning

RHEL uses a system tuning service called *tuned* to monitor storage, networking, processor, audio, video, and a variety of other connected devices, and adjusts their parameters for better performance or power saving based on a chosen profile. There are several predefined tuning profiles for common use cases shipped with RHEL that may be activated either statically or dynamically.

The *tuned* service activates a selected profile at service startup and continues to use it until it is switched to a different profile. This is the static behavior and it is enabled by default.

The dynamic alternative adjusts the system settings based on the live activity data received from monitored system components to ensure optimal performance. In most cases, the utilization of system components vary throughout the day. For example, a disk and processor's utilization increases during a program startup and the network connection use goes up during a large file transfer. A surge in a system component activity results in heightened power consumption.

Tuning Profiles

tuned includes nine predefined profiles to support a variety of use cases. In addition, you can create custom profiles from nothing or by using one of the existing profiles as a template. In either case, you need to store the custom profile under the `/etc/tuned` directory in order to be recognized by the *tuned* service.

Tuning profiles may be separated into three groups: (1) optimized for better performance, (2) geared more towards power consumption, and (3) that offers a balance between the other two and the maximum performance/power combination. [Table 12-5](#) lists and describes these profiles.

Profile	Description
Profiles Optimized for Better Performance	
Desktop	Based on the balanced profile for desktop systems. Offers improved throughput for interactive applications.
Latency-performance	For low-latency requirements
Network-latency	Based on the latency-performance focus on network throughput
Network-throughput	Based on the throughput-performance focus for maximum network throughput
Virtual-guest	Optimized for virtual machines
Virtual-host	Optimized for virtualized hosts
Profiles Optimized for Power Saving	
Powersave	Saves maximum power at the cost of performance
Balanced/Max Profiles	
Balanced	Preferred choice for systems that require balance between performance and power saving
Throughput-performance	Provides maximum performance and consumes maximum power

Table 12-5 Tuning Profiles

Predefined profiles are located in the `/usr/lib/tuned` directory in subdirectories matching their names. The following shows a long listing of the directory:

```
[user1@server1 ~]$ ls -l /usr/lib/tuned
total 16
drwxr-xr-x. 2 root root    24 Aug 23 06:16 balanced
drwxr-xr-x. 2 root root    24 Aug 23 06:16 desktop
-rw-r--r--. 1 root root 14413 Feb 22 2019 functions
drwxr-xr-x. 2 root root    24 Aug 23 06:16 latency-performance
drwxr-xr-x. 2 root root    24 Aug 23 06:16 network-latency
drwxr-xr-x. 2 root root    24 Aug 23 06:16 network-throughput
drwxr-xr-x. 2 root root    41 Aug 23 06:16 powersave
drwxr-xr-x. 2 root root   27 Aug 23 06:16 recommend.d
drwxr-xr-x. 2 root root    24 Aug 23 06:16 throughput-performance
drwxr-xr-x. 2 root root    24 Aug 23 06:16 virtual-guest
drwxr-xr-x. 2 root root    24 Aug 23 06:16 virtual-host
```

The default active profile set on `server1` and `server2` is the *virtual-guest* profile, as the two systems are hosted in a VirtualBox virtualized environment.

The tuned-adm Command

tuned comes with a single profile management command called *tuned-adm*. This tool can list active and available profiles, query current settings, switch between profiles, and turn the tuning off. This command can also recommend the best profile for the system based on many system attributes. Refer to the manual pages of the command for more details.

The following exercise demonstrates the use of most of the management operations listed above.

Exercise 12-2: Manage Tuning Profiles

This exercise should be done on `server1` as `user1` with `sudo` where required.

In this exercise, you will install the *tuned* service, start it now, and enable it for auto-restart upon future system reboots. You will display all available profiles and the current active profile. You will switch to one of the available profiles and confirm. You will determine the recommended profile for the system and switch to it. Finally, you will deactivate tuning and reactivate it. You will confirm the activation to conclude the exercise.

1. Install the *tuned* package if it is not already installed:

```
[user1@server1 ~]$ sudo dnf install -y tuned
Package tuned-2.10.0-15.el8.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

The output indicates that the software is already installed.

2. Start the *tuned* service and set it to autostart at reboots:

```
[user1@server1 ~]$ sudo systemctl --now enable tuned
```

3. Confirm the startup:

```
[user1@server1 ~]$ sudo systemctl status tuned
● tuned.service - Dynamic System Tuning Daemon
   Loaded: loaded (/usr/lib/systemd/system/tuned.service; enabled; vendor presen
   Active: active (running) since Tue 2019-12-10 23:16:19 EST; 19min ago
     Docs: man:tuned(8)
           man:tuned.conf(5)
           man:tuned-adm(8)
 Main PID: 881 (tuned)
    Tasks: 4 (limit: 5076)
   Memory: 2.9M
      CGroup: /system.slice/tuned.service
              └─881 /usr/libexec/platform-python -Es /usr/sbin/tuned -l -P

Dec 10 23:16:16 server1.example.com systemd[1]: Starting Dynamic System Tuning >
Dec 10 23:16:19 server1.example.com systemd[1]: Started Dynamic System Tuning D>
```

4. Display the list of available tuning profiles:

```
[user1@server1 ~]$ sudo tuned-adm list
Available profiles:
- balanced           - General non-specialized tuned profile
- desktop            - Optimize for the desktop use-case
- latency-performance - Optimize for deterministic performance at the cost of increased power consumption
- network-latency    - Optimize for deterministic performance at the cost of increased power consumption, focused on low latency network performance
- network-throughput - Optimize for streaming network throughput, generally only necessary on older CPUs or 40G+ networks
- powersave          - Optimize for low power consumption
- throughput-performance - Broadly applicable tuning that provides excellent performance across a variety of common server workloads
- virtual-guest       - Optimize for running inside a virtual guest
- virtual-host        - Optimize for running KVM guests
Current active profile: virtual-guest
```

The output shows the nine predefined profiles. It also exhibits the current active profile.

5. List only the current active profile:

```
[user1@server1 ~]$ sudo tuned-adm active
Current active profile: virtual-guest
```

6. Switch to the *powersave* profile and confirm:

```
[user1@server1 ~]$ sudo tuned-adm profile powersave
[user1@server1 ~]$ sudo tuned-adm active
Current active profile: powersave
```

The active profile is now *powersave*.

7. Determine the recommended profile for *server1* and switch to it:

```
[user1@server1 ~]$ sudo tuned-adm recommend
virtual-guest
[user1@server1 ~]$ sudo tuned-adm profile virtual-guest
[user1@server1 ~]$ sudo tuned-adm active
Current active profile: virtual-guest
```

The first instance of the command shows the best recommended profile for *server1* based on its characteristics, the second command instance switched to the recommended profile, and the third instance confirmed the switching.

8. Turn off tuning:

```
[user1@server1 ~]$ sudo tuned-adm off  
[user1@server1 ~]$ sudo tuned-adm active  
No current active profile.
```

The service will not perform any tuning until it is reactivated.

9. Reactivate tuning and confirm:

```
[user1@server1 ~]$ sudo tuned-adm profile virtual-guest  
[user1@server1 ~]$ sudo tuned-adm active  
Current active profile: virtual-guest
```

The tuning is re-enabled and the *virtual-guest* profile is in effect. This concludes the exercise.

Chapter Summary

This chapter started with a discussion of *systemd*, the default service management and system initialization scheme used in RHEL. We explored key components of *systemd*, its key directories, and examined unit and target configuration files. We utilized the lone *systemd* administration command to switch system operational states, identify and set default boot targets, and manage service start, stop, and status checking.

Next, we looked at the traditional system logging and newer *systemd* journaling services. Both mechanisms have similarities and differences in how they capture log data and where they direct it for storage and retrieval. We examined the system log configuration file and the configuration file that controls the log file rotation settings. The log subsystem proves valuable when records are needed for monitoring, troubleshooting, auditing, or reporting purpose.

Finally, we explored preconfigured tuning profiles and analyzed pros and cons associated with each one of them. We demonstrated how to determine a recommended profile for the system and how to set and activate it.

Review Questions

1. The *systemd* command may be used to rebuild a new kernel. True or False?
2. Which command is used to manage system services?

3. Which configuration file must be modified to ensure journal log entries are stored persistently?
4. What is the PID of the *systemd* process?
5. What is a target in *systemd*?
6. You need to append a text string “Hello world” to the system log file. What would be the command to achieve this?
7. What is the recommended location to store custom log configuration files?
8. What would the command *systemctl list-dependencies crond* do?
9. Name the two directory paths where *systemd* unit files are stored.
10. What would you run to identify the recommended tuning profile for the system?
11. What would the command *systemctl get-default* do?
12. *systemd* starts multiple services concurrently during system boot. True or False?
13. What is the name of the boot log file?
14. Which *systemctl* subcommand is executed after a unit configuration file has been modified to apply the changes?
15. Which other logging service complements the *rsyslog* service?
16. A RHEL system is booted up. You want to view all messages that were generated during the boot process. Which log file would you look at?
17. What would the command *systemctl restart rsyslog* do?
18. What are the two common *systemd* targets production RHEL servers are typically configured to run at?
19. By default, log files are rotated automatically every week. True or False?

Answers to Review Questions

1. False.
2. The *systemctl* command.
3. The *journald.conf* file under the */etc/systemd* directory.
4. The PID of the *systemd* process is 1.
5. A target is a collection of units.
6. The command to accomplish the desired result would be *logger -i "Hello world"*.
7. The recommended location to store custom log configuration files is */etc/rsyslog.d* directory.
8. The command provided will display all dependent units associated with the specified service.
9. The directory locations are */etc/systemd/system* and */usr/lib/systemd/system*.
10. The *tuned-adm recommend* command.

11. The command provided will reveal the current default boot target.
12. True.
13. The *boot.log* file in the */var/log* directory.
14. The *daemon-reload* subcommand.
15. The *systemd-journald* service.
16. The */var/log/boot.log* file.
17. The command provided will restart the *rsyslog* service.
18. The two common *systemd* boot targets are multi-user and graphical.
19. True.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 12-1: Modify Default Boot Target

As *user1* with *sudo* on *server1*, modify the default boot target from graphical to multi-user, and reboot the system to test it. Run the *systemctl* and *who* commands after the reboot for validation. Restore the default boot target back to graphical and reboot to verify. (Hint: System Initialization and Service Management).

Lab 12-2: Record Custom Alerts

As *user1* with *sudo* on *server1*, write the message “This is \$LOGNAME adding this marker on \$(date)” to */var/log/messages* file. Ensure that variable and command expansions work. Verify the entry in the file. (Hint: System Logging).

Lab 12-3: Apply Tuning Profile

As *user1* with *sudo* on *server1*, identify the current system tuning profile with the *tuned-adm* command. List all available profiles. List the recommended profile for *server1*. Apply the “balanced” profile and verify with *tuned-adm*. (Hint: System Tuning).

Chapter 13

Basic Storage Partitioning

This chapter describes the following major topics:

- Master Boot Record vs. GUID Partition Table
- Identify and understand disk partitions
- The concept of thin provisioning, and its benefits
- Create and delete partition on MBR disk
- Create and delete partition on GPT disk
- Overview of Virtual Data Optimizer and how it conserves storage
- Create and delete a Virtual Data Optimizer volume

RHCSA Objectives:

27. List, create, and delete partitions on MBR and GPT disks
37. Configure disk compression

Data is stored on disks that are logically divided into partitions. A partition can exist on a portion of a disk, on an entire disk, or it may span multiple disks. Each partition is accessed and managed independent of other partitions and may contain a file system or swap space. Partitioning information is stored at special disk locations that the system references at boot time. RHEL offers a number of tools for partition management. Partitions created with a combination of most of these tools can coexist on a single disk.

Thin provisioning is a powerful feature that guarantees an efficient use of storage space by allocating only what is needed and by storing data at adjacent locations. Many storage management solutions such as those we discuss later in this chapter and in the next incorporate thin provisioning technology in their core configuration.

Virtual Disk Optimizer is one of the newer storage management solutions incorporated in RHEL. It capitalizes on thin provisioning, de-duplication, and compression technologies to conserve storage space, improve data throughput, and save money.

This is the first of the three chapters that shed light on storage solutions. The next two chapters ([Chapters 14](#) and [15](#)) discuss advanced concepts and management tools.

Storage Management Overview

A disk in RHEL can be carved up into several partitions. This partition information is stored on the disk in a small region, which is read by the operating system at boot time. This region is referred to as the *Master Boot Record* (MBR) on the BIOS-based systems, and *GUID Partition Table* (GPT) on the UEFI-based systems. At system boot, the BIOS/UEFI scans all storage devices, detects the presence of MBR/GPT areas, identifies the boot disks, loads the bootloader program in memory from the default boot disk, executes the boot code to read the partition table and identify the `/boot` partition, loads the kernel in memory, and passes control over to it. Though MBR and GPT are designed for different PC firmware types, their job is essentially the same: to store disk partition information and the boot code.

Master Boot Record (MBR)

The MBR resides on the first sector of the boot disk. MBR was the preferred choice for saving partition table information on x86-based computers. However, with the arrival of bigger and larger hard drives, a newer firmware

specification (UEFI) was introduced. MBR is still widely used, but its use is diminishing in favor of UEFI.

MBR allows the creation of three types of partition—*primary*, *extended*, and *logical*—on a single disk. Of these, only primary and logical can be used for data storage; the extended is a mere enclosure for holding the logical partitions and it is not meant for data storage. MBR supports the creation of up to four primary partitions numbered 1 through 4 at a time. In case additional partitions are required, one of the primary partitions must be deleted and replaced with an extended partition to be able to add logical partitions (up to 11) within that extended partition. Numbering for logical partitions begins at 5. MBR supports a maximum of 14 usable partitions (3 primary and 11 logical) on a single disk.

MBR cannot address storage space beyond 2TB. This is due to its 32-bit nature and its 512-byte disk sector size. The MBR is non-redundant; the record it contains is not replicated, resulting in an unbootable system in the event of corruption. If your disk is smaller than 2TB and you don't intend to build more than 14 usable partitions, you can use MBR without issues. For more information on MBR, refer to [Chapter 11 “Boot Process, GRUB2, and the Linux Kernel”](#).

GUID Partition Table (GPT)

With the increasing use of disks larger than 2TB on x86 computers, a new 64-bit partitioning standard called *Globally Unique Identifiers (GUID) Partition Table* (GPT) was developed and integrated into the UEFI firmware. This new standard introduced plenty of enhancements, including the ability to construct up to 128 partitions (no concept of extended or logical partitions), utilize disks larger than 2TB, use 4KB sector size, and store a copy of the partition information before the end of the disk for redundancy.

Moreover, this standard allows a BIOS-based system to boot from a GPT disk using the bootloader program stored in a protective MBR at the first disk sector. In addition, the UEFI firmware also supports the secure boot feature, which only allows signed binaries to boot. For more information on UEFI and GPT, refer to [Chapter 11 “Boot Process, GRUB2, and the Linux Kernel”](#).

Disk Partitions

The space on a storage device can be sliced into partitions. Care must be taken when adding a new partition to elide data corruption with overlapping an extant partition or wasting storage by leaving unused space between adjacent partitions. On `server1`, the disk that was allocated at the time of installation is recognized as `sda` (`s` for SATA, SAS, or SCSI device) **disk a**,

with the first partition identified as *sda1* and the second partition as *sda2*. Any subsequent disks added to the system will be known as *sdb*, *sdc*, *sdd*, and so on, and will use 1, 2, 3, etc. for partition numbering.

RHEL offers a command called *lsblk* to list disk and partition information. The following graphic illustrates the current storage status on *server1*:

```
[user1@server1 ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0   10G  0 disk 
└─sda1     8:1    0   1G   0 part /boot
└─sda2     8:2    0   9G   0 part 
  ├─rhel-root 253:0  0   8G   0 lvm  /
  └─rhel-swap 253:1  0   1G   0 lvm  [SWAP]
sr0       11:0   1  6.6G  0 rom  /mnt
```

It reveals the presence of one 10GB disk, *sda*, with two partitions: *sda1* and *sda2*. The first partition holds */boot*, and the second one is an LVM object encapsulating *root* and *swap* logical volumes within it. Both *sda1* and *sda2* partitions occupy the entire disk capacity. The *sr0* represents the ISO image mounted as an optical medium.



LVM is discussed at length in [Chapter 14](#) “Advanced Storage Partitioning”.

There are additional tools such as *fdisk* and *parted* available that can be used to expose disk and partitioning information. Let’s run *fdisk* with *-l* and see what it reveals:

```
[user1@server1 ~]$ sudo fdisk -l
Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xae0aa336

Device      Boot  Start      End  Sectors Size Id Type
/dev/sda1    *    2048  2099199  2097152   1G 83 Linux
/dev/sda2        2099200 20971519 18872320   9G 8e Linux LVM

Disk /dev/mapper/rhel-root: 8 GiB, 8585740288 bytes, 16769024 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mapper/rhel-swap: 1 GiB, 1073741824 bytes, 2097152 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

The output depicts the size of *sda* in GBs, bytes, and sectors, the type of disk label (dos) the disk has, and the disk’s geometry in the top block. The second

block shows the two disk partitions: `sda1` as the bootable partition marked with an asterisk (*) and `sda2` as an LVM partition. It also exposes the starting and ending sector numbers, size in 1KB blocks, and type of each partition. The identifiers 83 and 8e are hexadecimal values for the partition types. The last two blocks are specific to the LVM logical volumes that exist within the `sda2` partition. A detailed coverage on LVM is provided in [Chapter 14](#) “Advanced Storage Partitioning”.

Storage Management Tools

RHEL offers numerous tools and toolsets for storage management, and they include `parted`, `gdisk`, VDO, LVM, and Stratis. There are other native tools available in the OS, but their discussion is beyond the scope of this book.

Partitions created with a combination of most of these tools and toolsets can coexist on the same disk. We look at `parted`, `gdisk`, and VDO in this chapter and LVM and Stratis in [Chapter 14](#) “Advanced Storage Partitioning”.

`parted` is a simple tool that understands both MBR and GPT formats. `gdisk` is designed to support the GPT format only, and it may be used as a replacement of `parted`. VDO is a disk optimizer software that takes advantage of certain technologies to minimize the overall data footprint on storage devices. LVM is a feature-rich logical volume management solution that gives flexibility in storage management. Stratis capitalizes on thin provisioning to create volumes much larger in size than the underlying storage devices they are built upon.

Thin Provisioning

Thin provisioning technology allows for an economical allocation and utilization of storage space by moving arbitrary data blocks to contiguous locations, which results in empty block elimination. With thin provisioning support in VDO, LVM, and Stratis, you can create a *thin pool* of storage space and assign volumes much larger storage space than the physical capacity of the pool. Workloads begin consuming the actual allocated space for data writing. When a preset custom threshold (80%, for instance) on the actual consumption of the physical storage in the pool is reached, expand the pool dynamically by adding more physical storage to it. The volumes will automatically start exploiting the new space right away. The thin provisioning technique helps prevent spending more money upfront.

Adding Storage for Practice

This and the next two chapters have a considerable number of exercises that require block storage devices for practice. In [Chapter 01](#) “Local Installation”

under “Lab Infrastructure for Practice”, we mentioned that *server2* will have 4x250MB, 1x4GB, and 2x1GB virtual disks for storage exercises. We presume that *server2* was built as part of Lab 1-1 and it is now available for use.

Exercise 13-1: Add Required Storage to *server2*

This exercise will add the required storage disks to *server2* (*RHEL8-VM2*) using VirtualBox.

In this exercise, you will start VirtualBox and add 4x250MB, 1x4GB, and 2x1GB disks to *server2* in preparation for exercises in this chapter and [Chapters 14 and 15](#).

1. Start VirtualBox on your Windows/Mac computer and highlight the *RHEL8-VM2* virtual machine that you created in Lab 1-1. See [Figure 13-1](#).

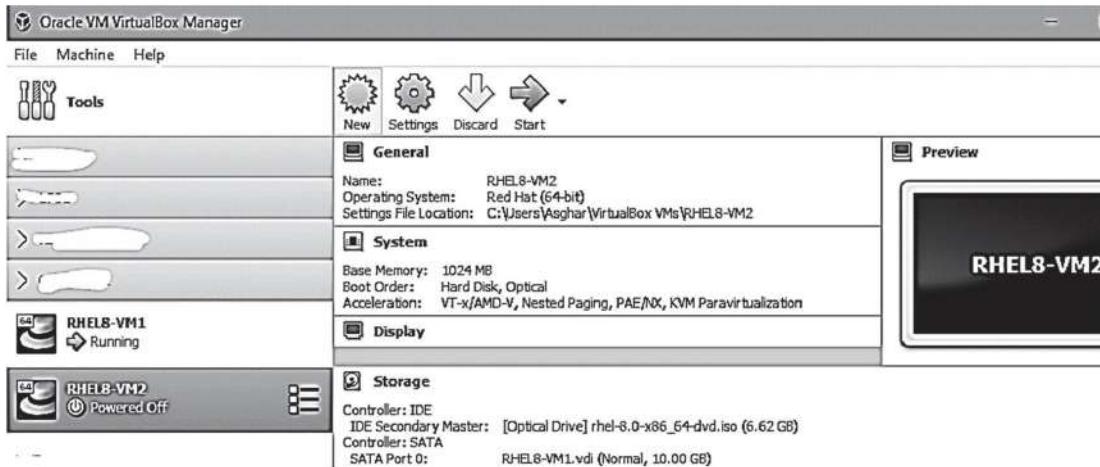


Figure 13-1 VirtualBox Interface

2. Click Settings at the top and then Storage on the window that pops up. Click on “Controller: SATA” to select it. [Figure 13-2](#).

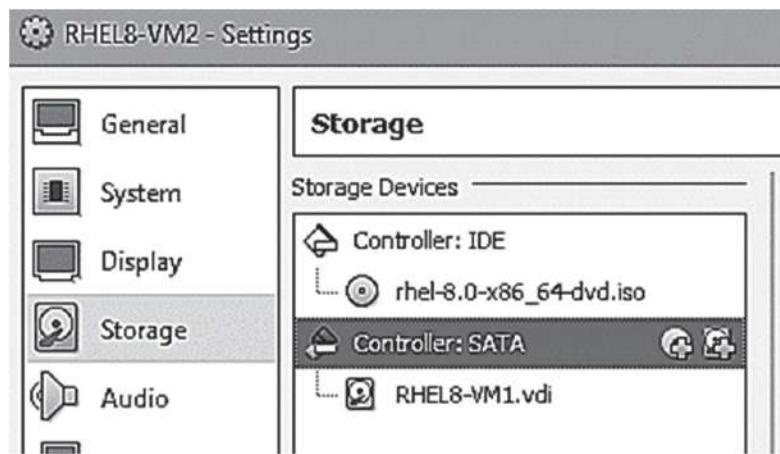


Figure 13-2 VirtualBox – Add Storage

3. Click on the right-side icon next to “Controller: SATA” to add a hard disk.
4. Follow this sequence to add a 250MB disk: Click “Create new disk”, “VDI (Virtualization Disk Image)”, “Dynamically allocated”, and adjust the size to 250MB. Assign the disk a unique name. [Figure 13-3](#).

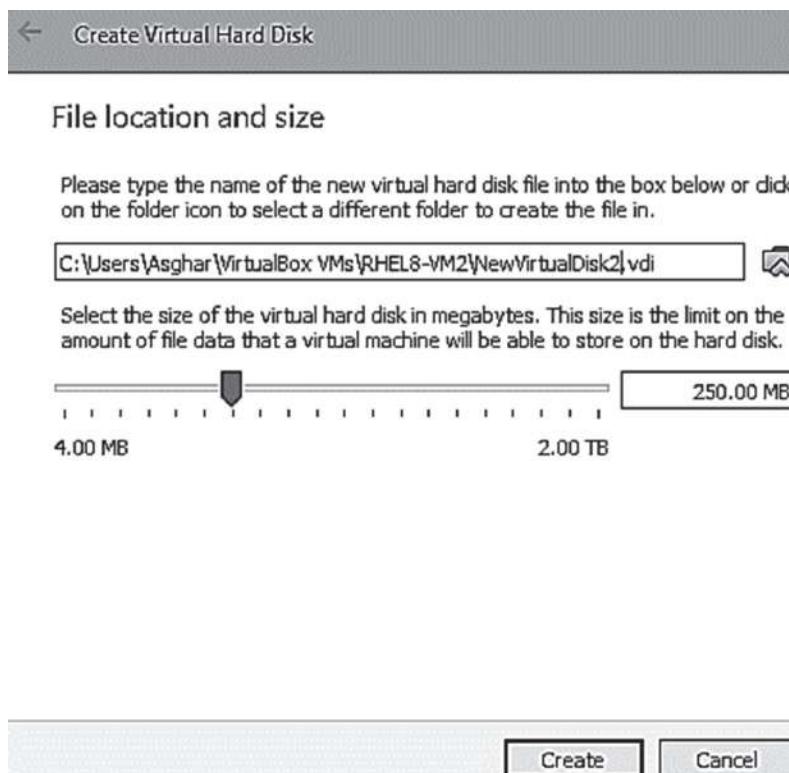


Figure 13-3 VirtualBox – Adjust Disk Name and Size

5. Click Create to create and attach the disk to the VM.

6. Repeat steps 3 through 5 three more times to add disks of the same size to the VM.
7. Repeat steps 3 through 5 one time to add a disk of size 4GB to the VM.
8. Repeat steps 3 through 5 two times to add two disks of size 1GB to the VM.
9. The final list of disks should look similar to what is shown in [Figure 13-4](#) after the addition of all seven drives. Disk names may vary.

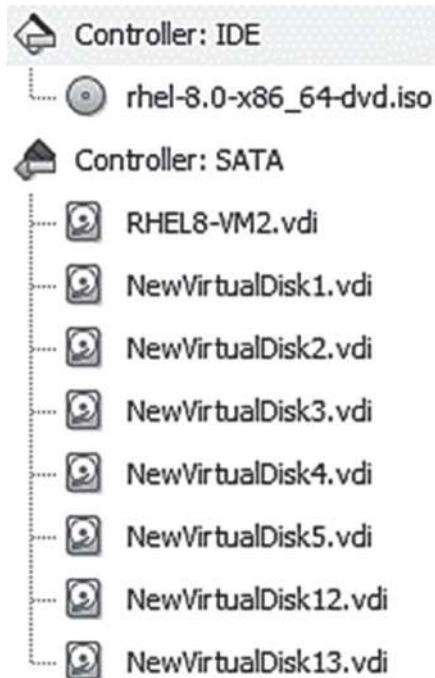


Figure 13-4 VirtualBox – 7 New Disks Added

10. Click OK to return to the main VirtualBox interface.
11. Power on *RHEL8-VM2* to boot RHEL 8 in it.
12. When the server is booted up, log on as *user1* and run the */sbin/k* command to verify the new storage:

```
[user1@server2 ~] $ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0   10G  0 disk 
└─sda1     8:1    0   1G   0 part /boot
└─sda2     8:2    0   9G   0 part
  └─rhel-root 253:0  0   8G   0 lvm   /
  └─rhel-swap 253:1  0   1G   0 lvm   [SWAP]
sdb        8:16   0  250M  0 disk 
sdc        8:32   0  250M  0 disk 
sdd        8:48   0  250M  0 disk 
sde        8:64   0  250M  0 disk 
sdf        8:80   0   4G   0 disk 
sdg        8:96   0   1G   0 disk 
sdh        8:112  0   1G   0 disk 
sr0       11:0   1  6.6G  0 rom   /mnt
```

13. The seven new disks added to `server2` are 250MB (`sdb`, `sdc`, `sdd`, and `sde`), 4GB (`sdf`), and 1GB (`sdg` and `sdh`).

This concludes the exercise for storage addition to `server2`.

MBR Storage Management with `parted`

`parted` (*partition editor*) is a popular tool in RHEL that can be used to partition disks. This program may be run interactively or directly from the command prompt. It understands and supports both MBR and GPT schemes, and can be used to create up to 128 partitions on a single GPT disk. `parted` provides an abundance of subcommands to perform disk management operations such as viewing, labeling, adding, naming, and deleting partitions. [Table 13-1](#) describes these subcommands in that sequence.

Subcommand	Description
print	Displays the partition table that includes disk geometry, partition number, start and end, size, type, file system, and relevant flags.
mklabel	Applies a label to the disk. Common labels are gpt and msdos.
mkpart	Makes a new partition
name	Assigns a name to a partition
rm	Removes the specified partition

Table 13-1 Common `parted` Subcommands

For the basic partition creation and deletion operations, Exercises 13-2 and 13-3 will show the use of this tool by directly invoking it from the command prompt. You will use the `/dev/sdb` disk for these exercises. After making a

partition, use the *print* subcommand to ensure you created what you wanted. The */proc/partitions* file is also updated to reflect the results of partition management operations.

Exercise 13-2: Create an MBR Partition

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will assign partition type “msdos” to */dev/sdb* for using it as an MBR disk. You will create and confirm a 100MB primary partition on the disk.

1. Execute *parted* on */dev/sdb* to view the current partition information:

```
[user1@server2 ~]$ sudo parted /dev/sdb print
Error: /dev/sdb: unrecognised disk label
Model: ATA VBOX HARDDISK (scsi)
Disk /dev/sdb: 262MB
Sector size (logical/physical): 512B/512B
Partition Table: unknown
Disk Flags:
```

There is an error on line 1 of the output, indicating an unrecognized label. This disk must be labeled before it can be partitioned.

2. Assign disk label “msdos” to the disk with *mklabel*. This operation is performed only once on a disk.

```
[user1@server2 ~]$ sudo parted /dev/sdb mklabel msdos
Information: You may need to update /etc/fstab.

[user1@server2 ~]$ sudo parted /dev/sdb print
Model: ATA VBOX HARDDISK (scsi)
Disk /dev/sdb: 262MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End    Size   Type    File system  Flags
```

The *print* subcommand confirms the successful application of the label.



To use the GPT partition table type, run “***sudo parted /dev/sdb mklabel gpt***” instead.

3. Create a 100MB primary partition starting at 1MB (beginning of the disk) using *mkpart*:

```
[user1@server2 ~]$ sudo parted /dev/sdb mkpart primary 1 101m
```

4. Verify the new partition with *print*:

```
[user1@server2 ~]$ sudo parted /dev/sdb print
Number  Start   End     Size    Type      File system  Flags
 1      1049kB  101MB  99.6MB  primary
```

Partition numbering begins at 1 by default.

5. Confirm the new partition with the *lsblk* command:

```
[user1@server2 ~]$ lsblk /dev/sdb
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdb      8:16    0  250M  0 disk
└─sdb1   8:17    0   95M  0 part
```

The device file for the first partition on the *sdb* disk is *sdb1* as identified on the bottom line. The partition size is 95MB.



Different tools will have variance in reporting partition sizes. You should ignore minor differences.

6. Check the */proc/partitions* file also:

```
[user1@server2 ~]$ cat /proc/partitions | grep sdb
 8        16      256000 sdb
 8        17      97280 sdb1
```

The virtual file is also updated with the new partition information. This completes the steps for creating and verifying an MBR partition using the *parted* command.

Exercise 13-3: Delete an MBR Partition

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will delete the *sdb1* partition that was created in [Exercise 13-2](#) and confirm the deletion.

1. Execute *parted* on */dev/sdb* with the *rm* subcommand to remove partition number 1:

```
[user1@server2 ~]$ sudo parted /dev/sdb rm 1
```

2. Confirm the partition deletion with *print*:

```
.....
```

Number	Start	End	Size	Type	File system	Flags
--------	-------	-----	------	------	-------------	-------

The partition no longer exists.

3. Check the */proc/partitions* file:

```
[user1@server2 ~]$ cat /proc/partitions | grep sdb
 8        16      256000 sdb
```

The virtual file has the partition entry deleted as well. You can also run the `/sbin/k` command for further verification. The partition has been removed successfully.

EXAM TIP: Knowing either parted or gdisk for the exam is enough.

We will recreate partitions for use in LVM in [Chapter 14](#) and then again in [Chapter 15](#) to construct file system and swap structures.

GPT Storage Management with gdisk

The *gdisk* (*GPT disk*) utility partitions disks using the GPT format. This text-based, menu-driven program can show, add, verify, modify, and delete partitions among other operations. *gdisk* can create up to 128 partitions on a single disk on systems with UEFI firmware.

The main interface of *gdisk* can be invoked by specifying a disk device name such as `/dev/sdc` with the command. Type *help* or `?` (question mark) at the prompt to view available subcommands.

```
[user1@server2 ~]$ sudo gdisk /dev/sdc
GPT fdisk (gdisk) version 1.0.3

Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.

Command (? for help): help
b      back up GPT data to a file
c      change a partition's name
d      delete a partition
i      show detailed information on a partition
l      list known partition types
n      add a new partition
o      create a new empty GUID partition table (GPT)
p      print the partition table
q      quit without saving changes
r      recovery and transformation options (experts only)
s      sort partitions
t      change a partition's type code
v      verify disk
w      write table to disk and exit
x      extra functionality (experts only)
?      print this menu

Command (? for help): _
```

The output illustrates that there is no partition table defined on the disk at the moment. There are several subcommands in the main menu followed

by a short description. Refer to the screenshot above for a list of subcommands.

Exercise 13-4: Create a GPT Partition

This exercise should be done on `server2` as `user1` with `sudo` where required. In this exercise, you will assign partition type “gpt” to `/dev/sdc` for using it as a GPT disk. You will create and confirm a 200MB partition on the disk.

1. Execute `gdisk` on `/dev/sdc` to view the current partition information:

```
[user1@server2 ~]$ sudo gdisk /dev/sdc
GPT fdisk (gdisk) version 1.0.3

Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.

Command (? for help): _
```

The disk currently does not have any partition table on it.

2. Assign “gpt” as the partition table type to the disk using the `o` subcommand. Enter “y” for confirmation to proceed. This operation is performed only once on a disk.

```
Command (? for help): o
This option deletes all partitions and creates a new protective MBR.
Proceed? (Y/N): Y
```

3. Run the `p` subcommand to view disk information and confirm the GUID partition table creation:

```
Command (? for help): p
Disk /dev/sdc: 512000 sectors, 250.0 MiB
Model: VBOX HARDDISK
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): CD1525A6-F747-4495-85F5-3C242603FB9A
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 511966
Partitions will be aligned on 2048-sector boundaries
Total free space is 511933 sectors (250.0 MiB)
```

Number	Start (sector)	End (sector)	Size	Code	Name
--------	----------------	--------------	------	------	------

The output returns the assigned GUID and states that the partition table can hold up to 128 partition entries.

4. Create the first partition of size 200MB starting at the default sector with default type “Linux filesystem” using the *n* subcommand:

```
Command (? for help): n
Partition number (1-128, default 1):
First sector (34-511966, default = 2048) or (+-)size(KMGTP):
Last sector (2048-511966, default = 511966) or (+-)size(KMGTP): +200M
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'
```

5. Verify the new partition with *p*:

```
Command (? For help): p
```

```
.....
Number  Start (sector)    End (sector)  Size       Code  Name
 1          2048           411647   200.0 MiB  8300  Linux filesystem
```

6. Run *w* to write the partition information to the partition table and exit out of the interface. Enter “y” to confirm when prompted.

```
Command (? for help): w
Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): Y
OK; writing new GUID partition table (GPT) to /dev/sdc.
The operation has completed successfully.
```



You may need to run the *partprobe* command after exiting the *gdisk* utility to update the kernel of the changes.

7. Verify the new partition by issuing either of the following at the command prompt:

```
[user1@server2 ~]$ grep sdc /proc/partitions
 8      32    256000  sdc
 8      33    204800  sdc1
[user1@server2 ~]$
[user1@server2 ~]$ lsblk /dev/sdc
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdc      8:32   0  250M  0 disk
└─sdc1   8:33   0  200M  0 part
```

The device file for the first partition on the *sdc* disk is *sdc1* and it is 200MB in size as reported in the above outputs. This completes the steps for creating and verifying a GPT partition using the *gdisk* command.

Exercise 13-5: Delete a GPT Partition

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will delete the `sdc1` partition that was created in [Exercise 13-4](#) and confirm the removal.

1. Execute `gdisk` on `/dev/sdc` and run `d1` at the utility's prompt to delete partition number 1:

```
Command (? for help): d1
Using 1
```

2. Confirm the partition deletion with `p`:

```
Number  Start (sector)    End (sector)  Size            Code  Name
```

The partition no longer exists.

3. Write the updated partition information to the disk with `w` and quit `gdisk`:

```
Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): Y
OK; writing new GUID partition table (GPT) to /dev/sdc.
The operation has completed successfully.
```

4. Verify the partition deletion by issuing either of the following at the command prompt:

```
[user1@server2 ~]$ grep sdc /proc/partitions
      8          32    256000  sdc
[user1@server2 ~]$
[user1@server2 ~]$ lsblk /dev/sdc
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdc   8:32    0   250M  0 disk
```

Both commands confirm the successful partition removal.

Storage Optimization with Virtual Data Optimizer (VDO)

One of the new features recently introduced in RHEL is a device driver layer that sits between the operating system kernel and the physical storage devices. The goals are to conserve disk space, improve data throughput, and save on storage cost. This feature is referred to as *Virtual Data Optimizer* (VDO). VDO employs thin provisioning, de-duplication, and compression technologies to help realize the goals.

How VDO Conserves Storage Space

VDO makes use of the thin provisioning technology to identify and eliminate empty (zero-byte) data blocks. This is referred to as *zero-block elimination*. VDO removes randomization of data blocks by moving in-use data blocks to contiguous locations on the storage device. This is the initial stage in the process.

Next, VDO keeps an eye on data being written to the disk. If it detects that the new data is an identical copy of some existing data, it makes an internal note of it but does not actually write the redundant data to the disk. VDO uses the technique called *de-duplication* to this end. This technique is implemented in RHEL with the inclusion of a kernel module called *UDS (Universal De-duplication Service)*. This is the second stage in the process.

In the third and final stage, VDO calls upon another kernel module called *kvdo*, which compresses the residual data blocks and consolidates them on a lower number of blocks. This results in a further drop in storage space utilization.

VDO runs in the background and processes inbound data through the three stages on VDO-enabled volumes. VDO is not a CPU- or memory-intensive process; it consumes a low amount of system resources.

Creating and Managing VDO Volumes

The concept of VDO volumes is similar to that of disk partitions, which you created in Exercises 13-1 and 13-3 using *parted* and *gdisk*. VDO volumes can be initialized for use just like disk partitions, or they can be used as LVM physical volumes.

VDO offers a set of commands to create, manage, and monitor volumes. Of these *vdo* and *vdostats* commands are discussed and used in this section. The *vdo* command is used to create and perform essential operations on VDO volumes, and the *vdostats* command is employed to monitor usage statistics of the underlying physical storage device.

Table 13-2 summarizes the subcommands available with *vdo*.

Subcommand	Description
create	Adds a new VDO volume on the specified block device
status	Returns the status and attributes of VDO volumes
list	Lists the names of all started VDO volumes
start	Starts a VDO volume
stop	Stops a VDO volume

Table 13-2 vdo Subcommands

The `vdostats` command has a couple of interesting options that you will use shortly.

Exercise 13-6: Install Software and Activate VDO

This exercise should be done on `server2` as `user1` with `sudo` where required.

In this exercise, you will install the VDO software packages, start the VDO service, and mark it for autostart on subsequent system reboots.

1. Install packages `vdo` and `kmod-kvdo`:

```
[user1@server2 ~]$ sudo dnf install vdo kmod-kvdo -y
```

2. Start the service and enable it to start automatically on future system reboots:

```
[user1@server2 ~]$ sudo systemctl --now enable vdo
```

3. Check the operational status of the service:

```
[user1@server2 ~]$ sudo systemctl status vdo
● vdo.service - VDO volume services
  Loaded: loaded (/usr/lib/systemd/system/vdo.service; enabled; vendor preset:disabled)
  Active: active (exited) since Tue 2019-11-05 11:35:38 EST; 39s ago
    Process: 8898 ExecStart=/usr/bin/vdo start --all --confFile /etc/vdoconf.yml
   Main PID: 8898 (code=exited, status=0/SUCCESS)
```

```
Nov 05 11:35:37 server2.example.com systemd[1]: Starting VDO volume services...
```

The relevant packages for VDO are installed, and the VDO service is started and activated. This concludes the exercise.

Exercise 13-7: Create a VDO Volume

This exercise should be done on `server2` as `user1` with `sudo` where required.

In this exercise, you will create a volume called *vdo-vol1* of logical size 16GB on */dev/sdf* disk (the actual size of */dev/sdf* is 4GB). You will list the volume and display its status information. You will also show the activation status of the compression and de-duplication features.

1. Create a volume called *vdo-vol1* (*--name*) on the */dev/sdf* device (*--device*) with a logical size of 16GB (*--vdoLogicalSize*) and slab size of 128MB (*--vdoSlabSize*):

```
[user1@server2 ~]$ sudo vdo create --name vdo-vol1 --device /dev/sdf --vdoLogica  
lSize 16G --vdoSlabSize 128  
Creating VDO vdo-vol1  
Starting VDO vdo-vol1  
Starting compression on VDO vdo-vol1  
VDO instance 1 volume is ready at /dev/mapper/vdo-vol1
```



If the logical size is not specified, the VDO volume will have the same size as the underlying disk (*/dev/sdf* in this case).



The slab size is the size of the increment by which VDO volumes grow. This value must be a power of two between 128MB and 32GB; the default is 2GB. The default unit of size specification is MB.

2. List the new volume using the *vdo* and *lsblk* commands:

```
[user1@server2 ~]$ sudo vdo list  
vdo-vol1  
[user1@server2 ~]$ lsblk /dev/sdf  
NAME      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
sdf        8:80    0   4G  0 disk  
└─vdo-vol1 253:2    0 16G  0 vdo
```

As indicated, the major number for the VDO volume is 253, which is associated with the device mapper kernel driver. The output also shows the logical volume size (16GB) and type (vdo). It also depicts the disk (*sdf*) that houses the volume, along with its actual size (4GB).

3. Display the usage status of the volume:

```
[user1@server2 ~]$ sudo vdostats --hu  
Device          Size     Used Available Use% Space saving%  
/dev/mapper/vdo-vol1    4.0G    2.8G     1.2G  68%       0%
```

The size of the actual disk is 4GB. Due to thin provisioning, the system allowed you to create the VDO volume much larger in size (4 times) than the physical disk capacity.

4. Show detailed statistics for the volume including configuration information:

```
[user1@server2 ~]$ sudo vdostats --verbose  
/dev/mapper/vdo-vol1 :  
    version          : 30  
    release version : 133524  
    data blocks used : 0  
    overhead blocks used : 723226  
    logical blocks used : 0  
    physical blocks   : 1048576  
    logical blocks     : 4194304  
    1K-blocks         : 4194304  
    1K-blocks used    : 2892904  
    1K-blocks available : 1301400  
    used percent      : 68  
    .....
```

The output will expose over one hundred different settings for the volume.

5. Display detailed statistics for the volume including configuration information:

```
[user1@server2 ~]$ sudo vdo status --name vdo-vol1  
VDO status:  
  Date: '2019-10-24 14:07:35-04:00'  
  Node: server2.example.com  
Kernel module:  
  Loaded: true  
  Name: kvdo  
  Version information:  
    kvdo version: 6.2.0.293  
Configuration:  
  File: /etc/vdoconf.yml  
  Last modified: '2019-10-24 14:03:13'  
VDOs:  
  vdo-vol1:  
    acknowledgement threads: 1  
  .....
```

The status includes volume, kernel module, and configuration information. It also provides a detailed look at volume-specific elements.

6. Show the activation status of the compression and de-duplication features:

```
[user1@server2 ~]$ sudo vdo status --name vdo-vol1 | grep -i compression  
  Compression: enabled  
[user1@server2 ~]$ sudo vdo status --name vdo-vol1 | grep -i deduplication  
  Deduplication: enabled
```

Both compression and de-duplication features are activated by default on new VDO volumes. This concludes the exercise.

Exercise 13-8: Delete a VDO Volume

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will delete the *vdo-vol1* volume that was created in [Exercise 13-7](#) and confirm the removal.

1. Verify that the volume still exists with the *vdo* and *lsblk* commands:

```
[user1@server2 ~]$ sudo vdo list
vdo-vol1
[user1@server2 ~]$ lsblk /dev/sdf
NAME      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdf        8:80    0   4G  0 disk
└─vdo-vol1 253:2    0  16G  0 vdo
```

2. Specify the name (*--name*) of the volume with the *vdo* command to delete:

```
[user1@server2 ~]$ sudo vdo remove --name vdo-vol1
Removing VDO vdo-vol1
Stopping VDO vdo-vol1
```

3. Confirm the removal with the *vdo* and *lsblk* commands:

```
[user1@server2 ~]$ sudo vdo list
[user1@server2 ~]$ lsblk /dev/sdf
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdf   8:80    0   4G  0 disk
```

The volume has been deleted successfully as reported in the above outputs.

EXAM TIP: Make sure that you run the *lsblk* command before attempting any storage task. The VDO task is usually performed on the biggest sized available disk.

You will recreate VDO volumes in [Chapter 15 “Local File Systems and Swap”](#) for use as file systems and swap areas.

Chapter Summary

This chapter started with an overview of how and where disk partitioning information is stored. It presented a comparison between the two common schemes and explained which one to use and in which situation. A little later, we touched briefly on the common storage management solutions available in RHEL.

We examined the concept of thin provisioning and realized the benefits associated with this technology.

Next, we carved up available disk devices using both MBR and GPT partitioning schemes. We demonstrated the partition creation, display, and delete operations by running one command directly at the command prompt and launching the other in interactive mode.

The last topic of the chapter focused on a storage management solution that was introduced in RHEL not too long ago. This solution exploits the underlying thin provisioning, de-duplication, and compression technologies to save cost, ensure efficient use of storage space, and improve data throughput. We performed a couple of exercises in the end to demonstrate the creation and deletion of volumes using this solution.

Review Questions

1. What is missing in the command `vdo create --name vdo1 --vdoLogicalSize 16GB --vdoSlabSize 128MB?`
2. What is the maximum number of usable partitions that can be created on a GPT disk?
3. Which kernel module is responsible for data block compression in VDO volumes?
4. What are the three techniques VDO volumes employ for storage conservation?
5. What would the command `parted /dev/sdc mkpart pri 1 200m` do?
6. Where is the partition table information stored on BIOS-based systems?
7. What is the name of the technology that VDO employs to remove randomization of data blocks?
8. What would `sdd3` represent?
9. Which command can be used to view the usage statistics of VDO volumes?
10. Thin provisioning technology allows us to create logical volumes of sizes larger than the actual physical storage size. True or False?
11. You have an unused VDO volume called `vdo1` on `/dev/sdf` disk and you try to delete it. You run `vdo remove --device /dev/sdf`, but the removal fails. What are you doing wrong?
12. What is the maximum number of usable partitions that can be created on an MBR disk?
13. What would the command `systemctl --now enable vdo` do?
14. The `gdisk` utility can be used to store partition information in MBR format. True or False?
15. What would the command `parted /dev/sdd mklabel msdos` do?

16. VDO is a memory-intensive storage optimization solution and should not be used. True or False?
17. Which file in the /proc file system stores the in-memory partitioning information?
18. You have created a VDO volume and you want to check whether compression is enabled. Which command would you use?
19. De-duplication is the process of zero-block elimination. True or False?

Answers to Review Questions

1. The storage device (--device) name is missing from the command provided.
2. 128.
3. The *kvdo* module is responsible for compressing data blocks in VDO volumes.
4. VDO volumes use thin provisioning, de-duplication, and compression techniques for storage conservation.
5. The command provided will create a primary partition of size 200MB on the *sdc* disk starting at the beginning of the disk.
6. The partition table information is stored on the Master Boot Record.
7. VDO employs thin provisioning technology to remove data block randomization.
8. *sdd3* represents the third partition on the fourth disk.
9. The *vdostats* command can be used to view VDO volume usage statistics.
10. True.
11. The correct command would be *vdo remove --name vdo1*.
12. 14.
13. The command provided will start the VDO service and enable it to autostart on system reboots.
14. False. The *gdisk* tool is only for GPT type tables.
15. The command provided will apply msdos label to the *sdd* disk.
16. False. VDO uses low memory and other compute resources.
17. The *partitions* file.
18. You can issue the *vdo status* command and pipe the output to *grep* for the pattern “compression”.
19. False. De-duplication is the process of removing blocks of identical data.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill

required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Add more storage to `server2` if required.

Lab 13-1: Create and Remove Partitions with `parted`

As `user1` with `sudo` on `server2`, create a 100MB primary partition on one of the available 250MB disks (`/sblk`) by invoking the `parted` utility directly at the command prompt. Apply label “msdos” if the disk is new. Create another 100MB partition by running `parted` interactively while ensuring that the second partition won’t overlap the first. Verify the label and the partitions. Remove both partitions from the command prompt. (Hint: MBR Storage Management with `parted`).

Lab 13-2: Create and Remove Partitions with `gdisk`

As `user1` with `sudo` on `server2`, create two 80MB partitions on one of the 250MB disks (`/sblk`) using the `gdisk` utility. Make sure the partitions won’t overlap. Verify the partitions. You may delete the partitions if you want. (Hint: GPT Storage Management with `gdisk`).

Lab 13-3: Create and Delete VDO Volumes

As `user1` with `sudo` on `server2`, check to see if VDO software is installed and the VDO service is enabled and started. Identify the 4GB disk with the `/sblk` command, and make sure that it is not in use. Create a volume `testvdo` with a logical size 16GB on the 4GB disk using the `vdo` command. Select an appropriate slab size for the volume. Verify the volume creation with the `vdo`, `/sblk`, and `vdostats` commands. (Hint: Storage Optimization with Virtual Data Optimizer).

Lab 13-4: Disable and Enable VDO Volume Features

As `user1` with `sudo` on `server2`, use the `vdostats` command to check whether compression and de-duplication are enabled for the volume created in Lab 13-3. Use the `vdo` command with `disableCompression` and `disableDeduplication` subcommands to disable compression and de-duplication, and verify with `vdostats`. Reactivate both features and confirm activation. You may delete the volume if you want. (Hint: Storage Optimization with Virtual Data Optimizer).

Chapter 14

Advanced Storage Partitioning

This chapter describes the following major topics:

- Describe Logical Volume Manager and its components
- Understand various Logical Volume Manager management operations
- Know Logical Volume Manager administration commands
- Create and confirm physical volumes, volume groups, and logical volumes
- Rename, reduce, extend, and remove logical volumes
- Extend, reduce, and remove volume groups
- Remove physical volumes
- Overview of Stratis (a volume-managing file system solution in RHEL) service and how it works
- Understand various Stratis management operations and the command
- Create, confirm, expand, rename, and destroy pools and file systems

RHCSA Objectives:

28. Create and remove physical volumes
29. Assign physical volumes to volume groups

30. Create and delete logical volumes
32. Add new partitions and logical volumes, and swap to a system non-destructively (the swap portion of this objective is covered in Chapter 15)
35. Extend existing logical volumes (additional coverage on this objective is available in Chapter 15)
38. Manage layered volumes

This chapter is the second of the three chapters (the other two being [Chapter 13](#) (previous) and [Chapter 15](#) (next)) that expounds upon storage management concepts and solutions available in RHEL. We discussed partitioning and thinly provisioned volumes in the previous chapter. This chapter presents a detailed coverage on Logical Volume Manager solution. LVM sets up an abstraction layer between the operating system and the storage hardware. It utilizes virtual objects for storage pooling and allocation, and offers a whole slew of management commands, each of which carries out a particular operation.

The other advanced storage management solution discussed in this chapter is a volume-managing file system that capitalizes on the proven features of LVM and the kernel device driver software. This solution dynamically adjusts the size of the underlying volume, eliminating the need for manual expansion.

Logical Volume Manager (LVM)

The *Logical Volume Manager* (LVM) solution is widely used for managing block storage in Linux. LVM provides an abstraction layer between the physical storage and the file system, enabling the file system to be resized, span across multiple physical disks, use arbitrary disk space, etc. LVM accumulates spaces taken from partitions or entire disks (called *Physical Volumes*) to form a logical container (called *Volume Group*), which is then divided into logical partitions (called *Logical Volumes*). The other key benefits of LVM include online resizing of volume groups and logical volumes, online data migration between logical volumes and between physical volumes, user-defined naming for volume groups and logical volumes, mirroring and striping across multiple physical disks, and snapshotting of logical volumes. [Figure 14-1](#) depicts the LVM components.

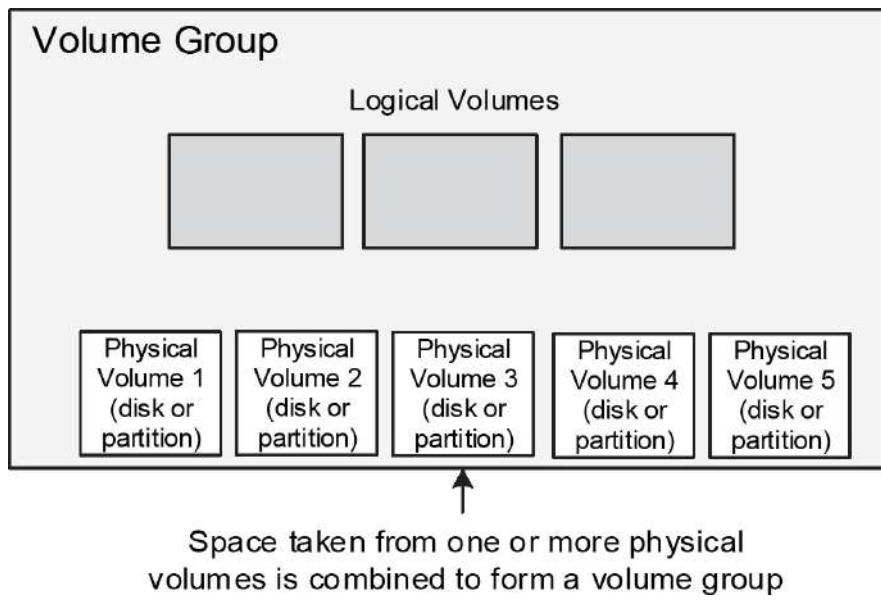


Figure 14-1 LVM Structure

As noted above, the LVM structure is made up of three key objects called physical volume, volume group, and logical volume. These objects are further virtually broken down into *Physical Extents* (PEs) and *Logical Extents* (LEs). The LVM components are explained in the following subsections.

Physical Volume

A *Physical Volume* (PV) is created when a block storage device such as a partition or an entire disk is initialized and brought under LVM control. This process constructs LVM data structures on the device, including a label on the second sector and metadata shortly thereafter. The label includes the UUID, size, and pointers to the locations of data and metadata areas. Given the criticality of metadata, LVM stores a copy of it at the end of the physical volume as well. The rest of the device space is available for use.

You can use an LVM command called *pvs* (*physical volume scan* or *summary*) to scan and list available physical volumes on *server2*:

```
[user1@server2 ~] $ sudo pvs
  PV        VG  Fmt Attr PSize  PFree
  /dev/sda2  rhel lvm2 a--  <9.00g    0
```

The output shows one physical volume (PV) */dev/sda2* of size 9GB in *rhel* volume group (VG). Additional information displays the metadata format (Fmt) used, status of the physical volume under the Attr column (a for allocatable), and the amount of free space available on the physical volume (PFree).

Try running this command again with the `-v` flag to view more information about the physical volume.

Volume Group

A *Volume Group* (VG) is created when at least one physical volume is added to it. The space from all physical volumes in a volume group is aggregated to form one large pool of storage, which is then used to build logical volumes. The physical volumes added to a volume group may be of varying sizes. LVM writes volume group metadata on each physical volume that is added to it. The volume group metadata contains its name, date and time of creation, how it was created, the extent size used, a list of physical and logical volumes, a mapping of physical and logical extents, etc. A volume group can have a custom name assigned to it at the time of its creation. For example, it may be called `vg01`, `vgora`, or `vgweb` that identifies the type of information it is constructed to store. A copy of the volume group metadata is stored and maintained at two distinct locations on each physical volume within the volume group.

You can use an LVM command called `vgs` (*volume group scan* or *summary*) to scan and list available volume groups on `server2`:

```
[user1@server2 ~]$ sudo vgs
  VG #PV #LV #SN Attr   VSize  VFree
  rhel   1    2    0 wz--n- <9.00g    0
```

The output shows one volume group (VG) `rhel` on `server2` containing one physical volume (#PV). Additional information displays the number of logical volumes (#LV) and snapshots (#SN) in the volume group, status of the volume group under the Attr column (w for writeable, z for resizable, and n for normal), size of the volume group (VSize), and the amount of free space available in the volume group (VFree).

Try running this command again with the `-v` flag to view more information about the volume group.

Physical Extent

A physical volume is divided into several smaller logical pieces when it is added to a volume group. These logical pieces are known as *Physical Extents* (PE). An extent is the smallest allocatable unit of space in LVM. At the time of volume group creation, you can either define the size of the PE or leave it to the default value of 4MB. This implies that a 20GB physical volume would have approximately 5,000 PEs. Any physical volumes added to this volume group thereafter will use the same PE size.

You can use an LVM command called `vgdisplay` (*volume group display*) on `server2` and `grep` for ‘PE Size’ to view the PE size used in the `rhel` volume group:

```
[user1@server2 ~] $ sudo vgdisplay rhel | grep 'PE Size'  
PE Size 4.00 MiB
```

The output reveals the PE size used for the `rhel` VG.

Logical Volume

A volume group consists of a pool of storage taken from one or more physical volumes. This volume group space is used to create one or more *Logical Volumes* (LVs). A logical volume can be created or weeded out online, expanded or shrunk online, and can use space taken from one or multiple physical volumes inside the volume group.

The default naming convention used for logical volumes is `/vol0`, `/vol1`, `/vol2`, and so on; however, you may assign custom names to them. For example, a logical volume may be called `system`, `undo`, or `webdata1` so as to establish the type of information it is constructed to store.

You can use an LVM command called `lvs` (*logical volume scan* or *summary*) to scan and list available logical volumes on `server2`:

```
[user1@server2 ~] $ sudo lvs  
  LV   VG   Attr       LSize  Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert  
  root  rhel -wi-aos--- <8.00g  
  swap  rhel -wi-aos---  1.00g
```

The output shows two logical volumes `root` and `swap` in `rhel` volume group. Additional information displays the status of the logical volumes under the Attr column (w for writeable, i for inherited allocation policy, a for active, and o for open) and their sizes.

Try running this command again with the `-v` flag to view more information about the logical volumes.

Logical Extent

A logical volume is made up of *Logical Extents* (LE). Logical extents point to physical extents, and they may be random or contiguous. The larger a logical volume is, the more logical extents it will have. Logical extents are a set of physical extents allocated to the logical volume.

The PE and LE sizes are normally kept the same within a volume group; however, a logical extent can be smaller or larger than a physical extent. The default LE size is 4MB, which corresponds to the default PE size.

You can use an LVM command called *lvdisplay* (*logical volume display*) on *server2* to view information about the *root* logical volume in the *rhel* volume group.

```
[user1@server2 ~] $ sudo lvdisplay /dev/rhel/root
--- Logical volume ---
LV Path          /dev/rhel/root
LV Name          root
VG Name          rhel
LV UUID          M6eGU8-wc5j-iJIC-rdUX-Uitm-NjjO-ZpyOIR
LV Write Access  read/write
LV Creation host, time server1.example.com, 2019-08-22 15:03:12 -0400
LV Status        available
# open           1
LV Size          <8.00 GiB
Current LE      2047
Segments         1
Allocation       inherit
Read ahead sectors  auto
- currently set to 256
Block device    253:0
```

The output does not disclose the LE size; however, you can convert the LV size in MBs (8,000) and then divide the result by the Current LE count (2,047) to get the LE size (which comes close to 4MB).

LVM Operations and Commands

The LVM toolset offers a multitude of administrative commands to carry out various disk and volume management operations. These operations include creating and removing a physical volume, volume group, and logical volume; extending and reducing a volume group and logical volume; renaming a volume group and logical volume; and listing and displaying physical volume, volume group, and logical volume information.

[Table 14-1](#) summarizes the common LVM tasks and the commands that are employed to accomplish them.

Command	Description
Create and Remove Operations	
pvcreate/pvremove	Initializes/uninitializes a disk or partition for LVM use
vgcreate/vgremove	Creates/removes a volume group
lvcreate/lvremove	Creates/removes a logical volume
Extend and Reduce Operations	
vgextend/vgreduce	Adds/removes a physical volume to/a volume group
lvextend/lvreduce	Extends/reduces the size of a logical volume
lvresize	Resizes a logical volume. With the -r option, this command calls the resize2fs command and resizes the underlying file system as well. Applies to Ext2/Ext3/Ext4 file system types only.
Rename Operations	
vgrename	Renames a volume group
lvrename	Renames a logical volume
Command	Description
List and Display Operations	
pvs/pvdisplay	Lists/displays physical volume information
vgs/vgdisplay	Lists/displays volume group information
lvs/lvdisplay	Lists/displays logical volume information

Table 14-1 Common LVM Operations and Commands

All the tools accept the -v switch to support verbosity. Refer to the manual pages of the commands for usage and additional details.

As noted earlier, there are seven disks available on `server2` for practice. Issue the `lsblk` command to confirm:

```
sdb      8:16    0  250M  0 disk
sdc      8:32    0  250M  0 disk
sdd      8:48    0  250M  0 disk
sde      8:64    0  250M  0 disk
 sdf     8:80    0     4G  0 disk
sdg      8:96    0     1G  0 disk
sdh     8:112   0     1G  0 disk
```

You will use the *sdd* and *sde* disks for LVM activities in the following exercises.

Exercise 14-1: Create a Physical Volume and Volume Group

This exercise should be done on *server2* as *user1* with *sudo* where required. In this exercise, you will initialize one partition *sdd1* (90MB) and one disk *sde* (250MB) for use in LVM. You will create a volume group called *vgbook* and add both physical volumes to it. You will use the PE size of 16MB and list and display the volume group and the physical volumes.

1. Create a partition of size 90MB on *sdd* using the *parted* command and confirm. You need to label the disk first, as it is a new disk.

```
[user1@server2 ~]$ sudo parted /dev/sdd mklabel msdos
[user1@server2 ~]$ sudo parted /dev/sdd mkpart primary 1 91m
[user1@server2 ~]$ sudo parted /dev/sdd print
Model: ATA VBOX HARDDISK (scsi)
Disk /dev/sdd: 262MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Partition Flags:

Number  Start   End     Size    Type      File system  Flags
 1       1049kB  91.2MB  90.2MB  primary
```

The *print* subcommand confirms the creation of the partition. It is the first partition on the disk.

2. Set (*set*) the flag on the partition (1) to “lvm” using the *parted* command:

```
[user1@server2 ~]$ sudo parted /dev/sdd set 1 lvm on
```

3. Verify flag activation using the *print* subcommand with *parted*:

```
Number  Start   End     Size    Type      File system  Flags
 1       1049kB  91.2MB  90.2MB  primary          lvm
```

The flag is applied and enabled on the partition as indicated under the Flags column.

4. Initialize the *sdd1* partition and the *sde* disk using the *pvcreate* command. Note that there is no need to apply a disk label on *sde* with *parted* as LVM does not require it.

```
[user1@server2 ~]$ sudo pvcreate /dev/sdd1 /dev/sde -v
Wiping signatures on new PV /dev/sdd1.
Wiping signatures on new PV /dev/sde.
Set up physical volume for "/dev/sdd1" with 176128 available sectors.
Zeroing start of device /dev/sdd1.
Writing physical volume data to disk "/dev/sdd1".
Physical volume "/dev/sdd1" successfully created.
Set up physical volume for "/dev/sde" with 512000 available sectors.
Zeroing start of device /dev/sde.
Writing physical volume data to disk "/dev/sde".
Physical volume "/dev/sde" successfully created.
```

The command generated a verbose output. You now have two physical volumes available for use.

5. Create *vgbook* volume group using the *vgcreate* command and add the two physical volumes to it. Use the *-s* option to specify the PE size in MBs.

```
[user1@server2 ~]$ sudo vgcreate -vs 16 vgbook /dev/sdd1 /dev/sde
Wiping signatures on new PV /dev/sdd1.
Wiping signatures on new PV /dev/sde.
Adding physical volume '/dev/sdd1' to volume group 'vgbook'
Adding physical volume '/dev/sde' to volume group 'vgbook'
Archiving volume group "vgbook" metadata (seqno 0).
Creating volume group backup "/etc/lvm/backup/vgbook" (seqno 1).
Volume group "vgbook" successfully created
```

The above command combines the two options with a single hyphen.

6. List the volume group information:

```
[user1@server2 ~]$ sudo vgs vgbook
VG      #PV #LV #SN Attr   VSize   VFree
vgbook   2    0    0 wz--n- 320.00m 320.00m

[user1@server2 ~]$ sudo vgs vgbook
VG      #PV #LV #SN Attr   VSize   VFree
vgbook   2    0    0 wz--n- 320.00m 320.00m
```

The total capacity available in the *vgbook* volume group is 320MB.

7. Display detailed information about the volume group and the physical volumes it contains:

```
[user1@server2 ~]$ sudo vgdisplay -v vgbook
--- Volume group ---
VG Name          vgbook
System ID
Format          lvm2
Metadata Areas   2
Metadata Sequence No  1
VG Access        read/write
VG Status         resizable
MAX LV           0
Cur LV            0
Open LV           0
Max PV            0
Cur PV            2
Act PV            2
VG Size          320.00 MiB
PE Size          16.00 MiB
Total PE          20
Alloc PE / Size  0 / 0
Free  PE / Size  20 / 320.00 MiB
VG UUID          CQVzWQ-myrd-OM2o-URWP-jHzE-sUBd-V9ZAOj

--- Physical volumes ---
PV Name          /dev/sdd1
PV UUID          0cxCYf-yGyp-pUzK-M1Sg-ZZ56-6d0L-d1ZJwV
PV Status         allocatable
Total PE / Free PE 5 / 5

PV Name          /dev/sde
PV UUID          GnnUDd-RrV2-4Y1q-KPJO-tgt3-cPyy-bqfVOD
PV Status         allocatable
Total PE / Free PE 15 / 15
```

The verbose output includes the physical volume attributes as well. There are a total of 20 PEs in the volume group (5 in *sdd1* and 15 in *sde*), and each PE is 16MB in size. The collective size of all the physical volumes represents the total size of the volume group, which is $20 \times 16 = 320$ MB.

8. List the physical volume information:

```
[user1@server2 ~]$ sudo pvs
PV      VG      Fmt  Attr PSize   PFree
/dev/sda2  rhel    lvm2 a--  <9.00g     0
/dev/sdc       lvm2 ---  250.00m 250.00m
/dev/sdd1  vgbook  lvm2 a--  80.00m  80.00m
/dev/sde   vgbook  lvm2 a-- 240.00m 240.00m
```

The output shows the physical volumes in *vgbook*, along with their utilization status.

9. Display detailed information about the physical volumes:

```
[user1@server2 ~]$ sudo pvdisplay /dev/sdd1
--- Physical volume ---
PV Name           /dev/sdd1
VG Name           vgbook
PV Size          86.00 MiB / not usable 6.00 MiB
Allocatable       yes
PE Size          16.00 MiB
Total PE         5
Free PE          5
Allocated PE     0
PV UUID          OcxCYf-yGYp-pUzK-M1Sg-ZZ56-6d0L-d1ZJwV

[user1@server2 ~]$ sudo pvdisplay /dev/sde
--- Physical volume ---
PV Name           /dev/sde
VG Name           vgbook
PV Size          250.00 MiB / not usable 10.00 MiB
Allocatable       yes
PE Size          16.00 MiB
Total PE         15
Free PE          15
Allocated PE     0
PV UUID          GnnUDd-RrV2-4Ylq-KPJO-tgt3-cPyy-bqfVOD
```

Once a partition or disk is initialized and added to a volume group, they are treated identically within the volume group. LVM does not prefer one over the other.

Exercise 14-2: Create Logical Volumes

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create two logical volumes, *lvol0* and *lvbook1*, in the *vgbook* volume group. You will use 120MB for *lvol0* and 192MB for *lvbook1* from the available pool of space. You will display the details of the volume group and the logical volumes.

1. Create a logical volume with the default name *lvol0* using the *lvcreate* command. Use the -L option to specify the logical volume size, 120MB. You may use the -v, -vv, or -vvv option with the command for verbosity.

```
[user1@server2 ~]$ sudo lvcreate -vL 120 vgbook
Rounding up size to full physical extent 128.00 MiB
Archiving volume group "vgbook" metadata (seqno 1).
Creating logical volume lvol0
Creating volume group backup "/etc/lvm/backup/vgbook" (seqno 2).
Activating logical volume vgbook/lvol0.
activation/volume_list configuration setting not defined: Checking only host
tags for vgbook/lvol0.
Creating vgbook-lvol0
Loading table for vgbook-lvol0 (253:2).
Resuming vgbook-lvol0 (253:2).
Wiping known signatures on logical volume "vgbook/lvol0"
Initializing 4.00 KiB of logical volume "vgbook/lvol0" with value 0.
Logical volume "lvol0" created.
```

The size for the logical volume may be specified in units such as MBs, GBs, TBs, or as a count of LEs; however, MB is the default if no unit is specified (see the previous command). The size of a logical volume is always in

multiples of the PE size. For instance, logical volumes created in *vgbook* with the PE size set at 16MB can be 16MB, 32MB, 48MB, 64MB, and so on. The output above indicates that the logical volume is 128MB (16x8), and not 120MB as specified.

2. Create *lvbook1* of size 192MB (16x12) using the *lvcreate* command. Use the -l switch to specify the size in logical extents and -n for the custom name. You may use -v for verbose information.

```
[user1@server2 ~]$ sudo lvcreate -l 12 -n lvbook1 vgbook
Logical volume "lvbook1" created.
```

3. List the logical volume information:

```
[user1@server2 ~]$ sudo lvs
  LV   VG     Attr   LSize   Pool Origin Data%  Meta%  Move Log Cpy*Sync Convert
root  rhel   -wi-ao---- <8.00g
swap  rhel   -wi-ao---- 1.00g
lvbook1 vgbook -wi-a---- 192.00m
lvol0 vgbook -wi-a---- 128.00m
```

Both logical volumes are listed in the output with their attributes and sizes.

4. Display detailed information about the volume group including the logical volumes and the physical volumes:

```
[user1@server2 ~]$ sudo vgdisplay -v vgbook
.....
--- Logical volume ---
LV Path          /dev/vgbook/lvol0
LV Name          lvol0
VG Name          vgbook
LV UUID          J902So-7x5x-2rhb-vaSM-vB2P-Ysfu-cpCpFu
LV Write Access  read/write
LV Creation host, time server2.example.com, 2019-10-27 20:07:14 -0400
LV Status        available
# open           0
LV Size          128.00 MiB
Current LE       8
Segments         1
Allocation       inherit
Read ahead sectors auto
- currently set to 8192
Block device    253:2

--- Logical volume ---
LV Path          /dev/vgbook/lvbook1
LV Name          lvbook1
VG Name          vgbook
LV UUID          ls46dQ-L3AN-UcS5-0k1D-ljs8-GwJ0-wOw7qu
LV Write Access  read/write
LV Creation host, time server2.example.com, 2019-10-27 20:19:11 -0400
LV Status        available
# open           0
LV Size          192.00 MiB
Current LE       12
Segments         2
Allocation       inherit
Read ahead sectors auto
- currently set to 8192
Block device    253:3

.....
```

Alternatively, you can run the following to view only the logical volume details:

```
[user1@server2 ~]$ sudo lvdisplay /dev/vgbook/lvol0  
[user1@server2 ~]$ sudo lvdisplay /dev/vgbook/lvbook1
```

Review the attributes of the logical volumes as detailed above.

Exercise 14-3: Extend a Volume Group and a Logical Volume

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will add another partition *sdd2* of size 158MB to *vgbook* to increase the pool of allocatable space. You will initialize the new partition prior to adding it to the volume group. You will increase the size of *lvbook1* to 336MB. You will display basic information for the physical volumes, volume group, and logical volume.

1. Create a partition of size 158MB on *sdd* and set the flag to “lvm” using the *parted* command. Display the new partition to confirm the partition number, size, and flag.

```
[user1@server2 ~]$ sudo parted /dev/sdd mkpart pri 92 250  
[user1@server2 ~]$ sudo parted /dev/sdd set 2 lvm on  
[user1@server2 ~]$ sudo parted /dev/sdd print  
  
.....  
2      92.3MB  250MB   157MB    primary          lvm
```

2. Initialize *sdd2* using the *pvcreate* command:

```
[user1@server2 ~]$ sudo pvcreate /dev/sdd2  
Physical volume "/dev/sdd2" successfully created.
```

3. Extend *vgbook* by adding the new physical volume to it:

```
[user1@server2 ~]$ sudo vgextend vgbook /dev/sdd2  
Volume group "vgbook" successfully extended
```

4. List the volume group:

```
[user1@server2 ~]$ sudo vgs  
  VG #PV #LV #SN Attr   VSize   VFree  
  rhel   1   2   0 wz--n- <9.00g     0  
  vgbook 3   2   0 wz--n- 464.00m 144.00m
```

The output reflects the addition of a third physical volume to *vgbook*. The total capacity of the volume group has now increased to 464MB with 144MB free.

5. Extend the size of *lvbook1* to 340MB by adding 144MB using the *lvextend* command:

```
[user1@server2 ~]$ sudo lvextend -L +144 /dev/vgbook/lvbook1
  Size of logical volume vgbook/lvbook1 changed from 192.00 MiB (12 extents) to 336.00 MiB (2
  1 extents).
  Logical volume vgbook/lvbook1 successfully resized.
```

EXAM TIP: Make sure the expansion of a logical volume does not affect the file system and the data it contains. More details in Chapter 15.

6. Issue *vgdisplay* on *vgbook* with the *-v* switch for the updated details:

```
[user1@server2 ~]$ sudo vgdisplay -v vgbook
<output truncated>
```

The output will show a lot of information about the volume group and the logical and physical volumes it contains. It will reflect the updates made in this exercise. In fact, each time a volume group or a logical volume is resized, *vgdisplay* will reflect those changes. The above output will display three physical volumes with the combined allocatable space grown to 464MB. The number of PEs will have increased to 29, with all of them allocated to logical volumes and 0 unused. The Logical Volume sections will display the updated information for the logical volumes. And at the very bottom, the three physical volumes will show with their device names, and total and available PEs in each.

7. View a summary of the physical volumes:

```
[user1@server2 ~]$ sudo pvs
  PV          VG     Fmt  Attr  PSize   PFree
  /dev/sda2    rhel   lvm2  a--   <9.00g    0
  /dev/sdc      -     lvm2  ---   250.00m  250.00m
  /dev/sdd1    vgbook lvm2  a--   80.00m    0
  /dev/sdd2    vgbook lvm2  a--   144.00m   0
  /dev/sde     vgbook lvm2  a--   240.00m   0
```

8. View a summary of the logical volumes:

```
[user1@server2 ~]$ sudo lvs
  LV      VG     Attr      LSize   Po
  root    rhel   -wi-ao---- <8.00g
  swap    rhel   -wi-ao----  1.00g
  lvbook1 vgbook -wi-a----- 336.00m
  lvol0   vgbook -wi-a----- 128.00m
```

This brings the exercise to an end.

Exercise 14-4: Rename, Reduce, Extend, and Remove Logical Volumes

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will rename *lv010* to *lvbook2*. You will decrease the size of *lvbook2* to 50MB using the *lvreduce* command and then add 32MB with the *lvresize* command. You will then remove both logical volumes. You will display the summary for the volume groups, logical volumes, and physical volumes.

1. Rename *lv010* to *lvbook2* using the *lvrename* command and confirm with *lvs*:

```
[user1@server2 ~]$ sudo lvrename vgbook lv010 lvbook2
Renamed "lv010" to "lvbook2" in volume group "vgbook"
[user1@server2 ~]$ sudo lvs
  LV      VG     Attr       LSize   Pool Origin Data%
  root    rhel   -wi-ao---- <8.00g
  swap    rhel   -wi-ao----  1.00g
  lvbook1 vgbook  -wi-a---- 336.00m
  lvbook2 vgbook  -wi-a---- 128.00m
```

2. Reduce the size of *lvbook2* to 50MB with the *lvreduce* command. Specify the absolute desired size for the logical volume. Answer “Do you really want to reduce vgbook/lvbook2?” in the affirmative.

```
[user1@server2 ~]$ sudo lvreduce -L 50 /dev/vgbook/lvbook2
Rounding size to boundary between physical extents: 64.00 MiB.
WARNING: Reducing active logical volume to 64.00 MiB.
THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce vgbook/lvbook2? [y/n]: y
Size of logical volume vgbook/lvbook2 changed from 128.00 MiB (8 extents) to 64.00 MiB (4 extents).
Logical volume vgbook/lvbook2 successfully resized.
```

3. Add 32MB to *lvbook2* with the *lvresize* command:

```
[user1@server2 ~]$ sudo lvresize -L +32 /dev/vgbook/lvbook2
Size of logical volume vgbook/lvbook2 changed from 64.00 MiB (4 extents) to 96.00 MiB (6 extents).
Logical volume vgbook/lvbook2 successfully resized.
```

4. Use the *pvs*, *lvs*, *vgs*, and *vgdisplay* commands to view the updated allocation.
5. Remove both *lvbook1* and *lvbook2* logical volumes using the *lvremove* command. Use the *-f* option to suppress the “Do you really want to remove active logical volume” message.

```
[user1@server2 ~]$ sudo lvremove /dev/vgbook/lvbook1 -f
Logical volume "lvbook1" successfully removed
[user1@server2 ~]$ sudo lvremove /dev/vgbook/lvbook2 -f
Logical volume "lvbook2" successfully removed
```



Removing a logical volume is a destructive task. You need to ensure that you perform a backup of any data in the target logical volume prior to deleting it. You will need to unmount the file system or disable swap in the logical volume. See [Chapter 15](#) on how to unmount a file system and disable swap.

6. Execute the `vgdisplay` command and `grep` for “Cur LV” to see the number of logical volumes currently available in `vgbook`. It should show 0, as you have removed both logical volumes.

```
[user1@server2 ~]$ sudo vgdisplay vgbook | grep 'Cur LV'  
Cur LV 0
```

This concludes the exercise.

Exercise 14-5: Reduce and Remove a Volume Group

This exercise should be done on `server2` as `user1` with `sudo` where required.

In this exercise, you will reduce `vgbook` by removing the `sdd1` and `sde` physical volumes from it, and then remove the volume group. Confirm the deletion of the volume group and the logical volumes at the end.

1. Remove `sdd1` and `sde` physical volumes from `vgbook` by issuing the `vgreduce` command:

```
[user1@server2 ~]$ sudo vgreduce vgbook /dev/sdd1 /dev/sde  
Removed "/dev/sdd1" from volume group "vgbook"  
Removed "/dev/sde" from volume group "vgbook"
```

2. Remove the volume group using the `vgremove` command. This will also remove the last physical volume, `sdd2`, from it.

```
[user1@server2 ~]$ sudo vgremove vgbook  
Volume group "vgbook" successfully removed
```



You can also use the `-f` option with the `vgremove` command to force the volume group removal even if it contains any number of logical and physical volumes in it.



Remember to proceed with caution whenever you perform reduce and erase operations.

3. Execute the `vgs` and `lvs` commands for confirmation:

```
[user1@server2 ~]$ sudo vgs
  VG #PV #LV #SN Attr   VSize  VFree
  rhel  1   2   0 wz--n- <9.00g    0
[user1@server2 ~]$ 
[user1@server2 ~]$ sudo lvs
  LV   VG   Attr       LSize  Pool Orig
  root rhel -wi-ao---- <8.00g
  swap rhel -wi-ao----  1.00g
```

This concludes the exercise.

Exercise 14-6: Uninitialize Physical Volumes

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will uninitialized all three physical volumes—*sdd1*, *sdd2*, and *sde*—by deleting the LVM structural information from them. Use the *pvs* command for confirmation. Remove the partitions from the *sdd* disk and verify that all disks used in Exercises 14-1 to 14-5 are now in their original raw state.

1. Remove the LVM structures from *sdd1*, *sdd2*, and *sde* using the *pvremove* command:

```
[user1@server2 ~]$ sudo pvremove /dev/sdd1 /dev/sdd2 /dev/sde
  Labels on physical volume "/dev/sdd1" successfully wiped.
  Labels on physical volume "/dev/sdd2" successfully wiped.
  Labels on physical volume "/dev/sde" successfully wiped.
```

2. Confirm the removal using the *pvs* command:

```
[user1@server2 ~]$ sudo pvs
  PV          VG   Fmt  Attr PSize  PFree
  /dev/sda2   rhel lvm2 a--   <9.00g    0
  /dev/sdc    lvm2 ---  250.00m 250.00m
```

The partitions and the disk are now back to their raw state and can be repurposed.

3. Remove the partitions from *sdd* using the *parted* command:

```
[user1@server2 ~]$ sudo parted /dev/sdd rm 1 ; sudo parted /dev/sdd rm 2
```

4. Verify that all disks used in previous exercises have returned to their original raw state using the *lsblk* command:

```
[user1@server2 ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0   10G  0 disk 
└─sda1     8:1    0    1G  0 part /boot
└─sda2     8:2    0    9G  0 part
  └─rhel-root 253:0  0    8G  0 lvm  /
  └─rhel-swap 253:1  0    1G  0 lvm  [SWAP]
sdb        8:16   0  250M 0 disk 
sdc        8:32   0  250M 0 disk 
sdd        8:48   0  250M 0 disk 
sde        8:64   0  250M 0 disk 
sdf        8:80   0    4G  0 disk 
sdg        8:96   0    1G  0 disk 
sdh        8:112  0    1G  0 disk
```

This brings the exercise to an end.

We will recreate logical volumes in [Chapter 15](#) and construct file system and swap structures in them.

Stratis Volume-Managing File System

RHEL 8 introduces a new simplified storage management solution called *Stratis*. Stratis capitalizes on three existing matured storage components: the *device mapper* (dm) kernel driver, the LVM solution, and the XFS file system. It implements the advanced features of these components to deliver file systems that are encapsulated within logical volumes. These logical volumes are created and expanded dynamically and transparently, hiding the underlying complexity. The Stratis solution delivers file systems that are referred to as *volume-managing file systems*. Stratis uses the thin provisioning technology as part of the solution.



File systems are explained in [Chapter 15](#) “Local File Systems and Swap”.

The central idea surrounding the Stratis solution is a storage *pool*. A storage pool is created using at least one disk or partition, which is referred to as a *blockdev*. There can be a combination of the two in a single pool and more can be added as the space requirement grows. The total capacity of the pool is the aggregate of the spaces taken from all the block devices that are part of the pool. Within a pool, *file systems* can be created, all sharing the entire pool capacity.



LVM logical volumes can also be used as block devices in a Stratis pool.

[Figure 14-2](#) provides a bird’s-eye view of the three major objects—pool, blockdev, and file system—used in Stratis.

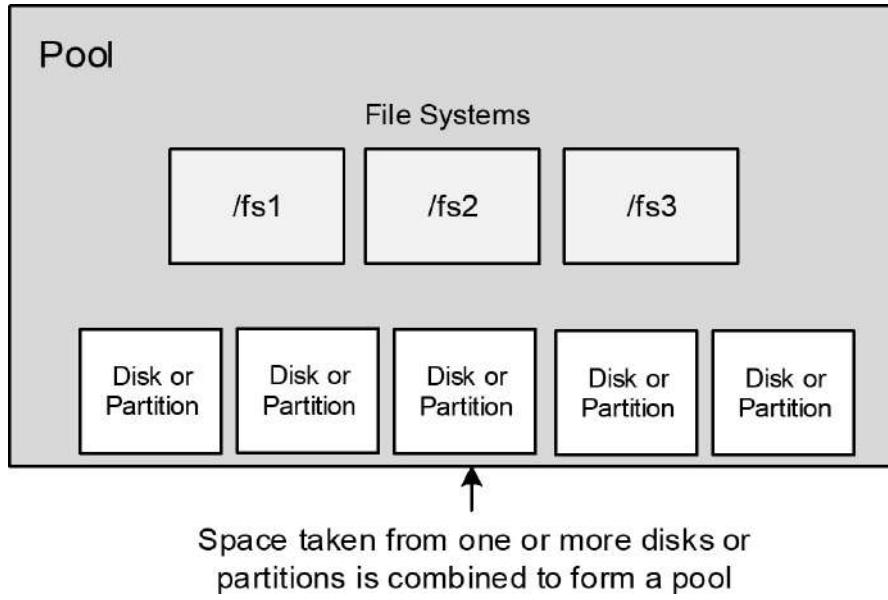


Figure 14-2 Stratis Structure

The diagram in [Figure 14-2](#) shows a pool containing multiple disks and partitions. The pool capacity is the aggregate of all the block storage devices included in the pool, and this capacity is shared among all the file systems. Each Stratis file system appears to occupy the entire pool capacity exclusively; however, they are thinly provisioned. Their actual size grows as the amount of data stored in them increases. Stratis takes care of the dynamic expansion of the file systems and the underlying volumes as needed. As Stratis handles the creation, formatting, and expansion of the file systems, they must not be manually initialized or reconfigured.

Stratis Management Operations and Command

The primary command to manage Stratis is called *stratis*. This command has a set of subcommands available to perform management operations such as creating, viewing, renaming, and destroying pools and file systems, and expanding pools.

[Table 14-2](#) summarizes the common Stratis subcommands that are employed to accomplish various management tasks.

Command	Description
pool	Administers storage pools. Subcommands are available to create, rename, expand, and destroy a pool.
blockdev	Lists block devices
filesystem	Administers file systems within storage pools. Subcommands are available to list, create, rename, and destroy a file system.

Table 14-2 Common Stratis Subcommands

Stratis runs as a service, so its operational state can be managed with the `systemctl` command. The `stratis` command interacts with the Stratis service to manage the pool and file systems dynamically.

For each pool added, Stratis creates a subdirectory under the `/stratis` directory matching the pool name. It then creates a symbolic link for each file system under that subdirectory to the actual device file located in the `/dev` directory.

Stratis file systems are not fixed-sized; however, pool space can be reserved to assure availability if multiple file systems share a pool.

Exercise 14-7: Install Software and Activate Stratis

This exercise should be done on `server2` as `user1` with `sudo` where required.

In this exercise, you will install the Stratis software packages, start the Stratis service, and mark it for autostart on subsequent system reboots.

1. Install the packages `stratisd` and `stratis-cli`:

```
[user1@server2 ~]$ sudo dnf install stratisd stratis-cli -y
.....
Installed:
  stratis-cli-1.0.2-1.el8.noarch
  stratisd-1.0.3-1.el8.x86_64
  python3-dbus-client-gen-0.4-1.el8.noarch
  python3-dbus-python-client-gen-0.6-2.el8.noarch
  python3-dbus-signature-pyparsing-0.03-2.el8.noarch
  python3-into-dbus-python-0.06-2.el8.noarch
  python3-justbases-0.9-6.el8.noarch
  python3-justbytes-0.11-2.el8.noarch
  python3-pyparsing-2.1.10-7.el8.noarch

Complete!
```

2. Start the service and enable it to start automatically on future system reboots:

```
[user1@server2 ~]$ sudo systemctl --now enable stratisd
```

3. Check the operational status of the service:

```
[user1@server2 ~]$ sudo systemctl status stratisd
* stratisd.service - A daemon that manages a pool of block devices to create flexible file systems
  Loaded: loaded (/usr/lib/systemd/system/stratisd.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2019-10-30 22:58:16 EDT; 10s ago
    Docs: man:stratisd(8)
 Main PID: 24310 (stratisd)
   Tasks: 1 (limit: 11516)
   Memory: 980.0K
      CGroup: /system.slice/stratisd.service
              └─24310 /usr/libexec/stratisd --debug
```

The relevant packages for the Stratis storage management solution are installed, and the Stratis service is started and activated. This concludes the exercise.

Exercise 14-8: Create and Confirm a Pool and File System

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create a Stratis pool and a file system in it. You will display information about the pool, file system, and device used.

1. You allocated 2x1GB disks: *sdg* and *sdh* for Stratis exercises. Use the *lsblk* command to confirm their availability:

```
[user1@server2 ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0 10G  0 disk 
└─sda1     8:1    0  1G  0 part /boot
└─sda2     8:2    0  9G  0 part
  ├─rhel-root 253:0  0  8G  0 lvm  /
  └─rhel-swap 253:1  0  1G  0 lvm  [SWAP]
sdb        8:16   0 250M 0 disk 
sdc        8:32   0 250M 0 disk 
sdd        8:48   0 250M 0 disk 
sde        8:64   0 250M 0 disk 
 sdf       8:80   0   4G  0 disk 
sdg        8:96   0   1G  0 disk 
sdh        8:112  0   1G  0 disk
```

The 2x1GB disks are available and unused.

2. Create a pool called *bookpool* using the *sdg* disk and verify the creation:

```
[user1@server2 ~]$ sudo stratis pool create bookpool /dev/sdg
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
bookpool           1 GiB                  52 MiB
```

The specified pool *bookpool* is created. The second command output returns the basic information about the pool, including the number of physical devices used in the pool and their sizes, and the amount of in-use space in MiBs.

3. Display the block device that is used to form the pool:

```
[user1@server2 ~]$ sudo stratis blockdev list bookpool
Pool Name  Device Node    Physical Size   State   Tier
bookpool   /dev/sdg           1 GiB     In-use   Data
```

The pool *bookpool* has a single disk of size 1GiB and it is currently in use.

4. Create a file system called *bookfs* in the *bookpool* and verify the creation:

```
[user1@server2 ~]$ sudo stratis filesystem create bookpool bookfs
[user1@server2 ~]$ sudo stratis filesystem list
Pool Name  Name      Used      Created          Device          UUID
bookpool   bookfs   546 MiB  Oct 30 2019 23:33  /stratis/bookpool/bookfs  516e345bc9fa4a90b1ec
dcc7a1d4d12f
```

The pool *bookpool* has a single file system *bookfs* of used size 546MiB. Its device file is */stratis/bookpool/bookfs*. The file system's UUID is also displayed.

5. Create a directory called */bookfs1* and mount the new file system on it:

```
[user1@server2 ~]$ sudo mkdir /bookfs1
[user1@server2 ~]$ sudo mount /stratis/bookpool/bookfs /bookfs1
```

6. Check the pool usage:

```
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
bookpool           1 GiB            598 MiB
```

Observe how the usage grew from 52MiB to 598MiB. This brings this exercise to an end.

Exercise 14-9: Expand and Rename a Pool and File System

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will expand the Stratis pool *bookpool* (created in [Exercise 14-8](#)) using the *sdh* disk. You will rename the pool and the file system it contains.

1. Expand the *bookpool* pool by adding the *sdh* disk to it:

```
[user1@server2 ~]$ sudo stratis pool add-data bookpool /dev/sdh
```

2. Verify the new capacity of the pool:

```
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
bookpool           2 GiB            602 MiB
```

The addition of another 1GB disk brings the total physical size to 2GiB.

3. Change the name of the pool from *bookpool* to *rhcsapool* and verify:

```
[user1@server2 ~]$ sudo stratis pool rename bookpool rhcsapool
[user1@server2 ~]$
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
rhcsapool           2 GiB            602 MiB
```

The new name is depicted in the first column.

4. Change the name of the file system from *bookfs* to *rhcsafs*, and verify:

```
[user1@server2 ~]$ sudo stratis filesystem rename rhcsapool bookfs rhcsafs
[user1@server2 ~]$ sudo stratis filesystem list
Pool Name  Name     Used     Created        Device          UUID
rhcsapool  rhcsafs  546 MiB  Oct 30 2019 23:33  /stratis/rhcsapool/rhcsafs  516e345bc9fa4a90b
1ecdcc7a1d4d12f
```

The file system name is changed and it's reflected in the output of the second command. The exercise is completed successfully.

Exercise 14-10: Destroy a File System and Pool

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will destroy the Stratis file system and the pool that was created, expanded, and renamed in Exercises 14-8 and 14-9. You will verify the deletion with appropriate commands.

1. The first step in the process is to unmount the file system *rhcsafs1* from its mount point */bookfs1*:

```
[user1@server2 ~]$ sudo umount /bookfs1
```

Unmounting a file system ensures that the file system is not busy.

2. Remove the file system *rhcsafs* from the pool:

```
[user1@server2 ~]$ sudo stratis filesystem destroy rhcsapool rhcsafs
```

3. Remove the pool *rhcsapool* from the system:

```
[user1@server2 ~]$ sudo stratis pool destroy rhcsapool
```

4. Confirm the removal of the file system and the pool:

```
[user1@server2 ~]$ sudo stratis filesystem list
Pool Name  Name  Used  Created  Device  UUID
[user1@server2 ~]$
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
```

The above outputs confirm the removal of the file system and the pool.

5. Verify that both *sdg* and *sdh* disks used in the previous Stratis exercises have returned to their original raw state using the *lsblk* command:

```
[user1@server2 ~]$ lsblk /dev/sdg /dev/sdh
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdg    8:96   0   1G  0 disk
sdh    8:112   0   1G  0 disk
```

This concludes the exercise.

We will recreate Stratis file systems in [Chapter 15](#) and add them for persistent mounting.

Chapter Summary

This chapter explicated two advanced storage management solutions: Logical Volume Manager and Stratis. The LVM solution has been around in RHEL for decades. The Stratis solution, on the other hand, is recently added.

We discovered how LVM works. We looked at various LVM objects and their relationship with one another. We explored LVM management commands and common options available with them. We performed a series of exercises to demonstrate the creation, expansion, renaming, reduction, and deletion of physical volumes, storage pools, and logical volumes.

The next advanced storage solution we looked at is called Stratis, which is a volume-managing file system. Stratis takes advantage of the underlying LVM functionality and the device mapper kernel driver to create and expand the XFS file system that it holds. Stratis dynamically expands the underlying LVM volume without the need for manual administrative intervention. We executed a couple of step-by-step exercises to explain the creation, growth, renaming, and removal of Stratis pools and file systems.

Review Questions

1. The *parted* utility may be used to create LVM logical volumes. True or False?
2. What are the two commands that you can use to reduce the number of logical extents from a logical volume?
3. Stratis uses LVM as the underlying logical volume management solution. True or False?
4. Provide the command to add physical volumes */dev/sdd1* and */dev/sdc* to *vg20* volume group.
5. What are the two commands that you can use to add logical extents to a logical volume?
6. Provide the command to create a volume group called *vg20* on */dev/sdd* disk with physical extent size 64MB.
7. Name the three Stratis objects?
8. Provide the command to remove *vg20* along with logical and physical volumes it contains.
9. What is the default size of a physical extent in LVM?
10. What is the default name for the first logical volume in a volume group?
11. What is one difference between the *pvs* and *pvdisplay* commands?
12. When can a disk or partition be referred to as a physical volume?
13. Provide the command to remove *webvol* logical volume from *vg20* volume group.
14. It is necessary to create file system structures in a logical volume before it can be used to store files in it. True or False?
15. What would the command *stratis pool create pool1 /dev/sdg* do?
16. Physical and logical extents are typically of the same size. True or False?
17. What is the purpose of the *pvremove* command?
18. What would the command *pvcreate /dev/sdd* do?
19. A disk or partition can be added to a volume group without being initialized. True or False?
20. What is the file system type that is created in a Stratis file system?
21. Provide the command to create a logical volume called *webvol* of size equal to 100 logical extents in *vg20* volume group.
22. A volume group can be created without any physical volume in it. True or False?
23. A single disk can be used by both *parted* and LVM solutions at the same time. True or False?
24. Provide the command to add */dev/sdh* to an existing pool called *pool1*.
25. Provide the command to erase */dev/sdd1* physical volume from *vg20* volume group.
26. A partition can be used as an LVM object. True or False?

27. Which command would you use to view the details of a volume group and its objects?

Answers to Review Questions

1. False.
2. The `lvreduce` and `lvresize` commands.
3. True.
4. `vgextend vg20 /dev/sdd1 /dev/sdc`
5. The `lvextend` and `lvresize` commands.
6. `vgcreate -s 64 vg20 /dev/sdd`
7. The three Stratis objects are pool, block device, and file system.
8. `vgremove -f vg20`
9. The default PE size is 4MB.
10. `lvol0` is the default name for the first logical volume created in a volume group.
11. The `pvs` command lists basic information about physical volumes whereas the `pvdisplay` command shows the details.
12. After the `pvcreate` command has been executed on it successfully.
13. `lvremove /dev/vg20/webvol`
14. True.
15. The command provided will create a Stratis pool called `pool1` containing `/dev/sdg` device.
16. True.
17. The `pvremove` command is used to remove LVM information from a physical volume.
18. The command provided will prepare the `/dev/sdd` disk for use in a volume group.
19. False. A disk or partition must be initialized before it can be added to a volume group.
20. XFS.
21. `lvcreate -I 100 -n webvol vg20`
22. False.
23. True. A single disk can be shared between `parted`-created partitions and LVM.
24. `stratis pool add-data pool1 /dev/sdh`
25. `vgreduce vg20 /dev/sdd1`
26. True.
27. The `vgdisplay` command with the `-v` option.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Add more storage to `server2` if required.

Lab 14-1: Create Volume Group and Logical Volumes

As `user1` with `sudo` on `server2`, initialize 1x250MB disk for use in LVM (use `lsblk` to identify available disks). Create volume group `vg100` with PE size 16MB and add the physical volume. Create two logical volumes `/vol0` and `swapvol` of sizes 100MB and 120MB. Use the `vgs`, `pvs`, `lvs`, and `vgdisplay` commands for verification. (Hint: Logical Volume Manager).

Lab 14-2: Expand Volume Group and Logical Volume

As `user1` with `sudo` on `server2`, create a partition on an available 250MB disk and initialize it for use in LVM (use `lsblk` to identify available disks). Add the new physical volume to `vg100`. Expand the `/vol0` logical volume to size 300MB. Use the `vgs`, `pvs`, `lvs`, and `vgdisplay` commands for verification. (Hint: Logical Volume Manager).

Lab 14-3: Reduce and Remove Logical Volumes

As `user1` with `sudo` on `server2`, reduce the size of `/vol0` logical volume to 80MB. Then erase both logical volumes `swapvol` and `/vol0`. Confirm the deletion with `vgs`, `pvs`, `lvs`, and `vgdisplay` commands. (Hint: Logical Volume Manager).

Lab 14-4: Remove Volume Group and Physical Volumes

As `user1` with `sudo` on `server2`, remove the volume group and uninitialized the physical volumes. Confirm the deletion with `vgs`, `pvs`, `lvs`, and `vgdisplay` commands. Use the `lsblk` command and verify that the disks used for the LVM labs no longer show LVM information. (Hint: Logical Volume Manager).

Lab 14-5: Create Stratis Pool

As *user1* with *sudo* on *server2*, check to see if Stratis software is installed and the Stratis service is enabled and started. Identify 2x1GB disks with the *lsblk* command and ensure they are not in use. Create pool *strpool* on one of the 1GB disks and verify the pool and the block device with the *stratis* command. (Hint: Stratis Volume-Managing File System).

Lab 14-6: Expand and Destroy Stratis Pool

As *user1* with *sudo* on *server2*, use the other 1GB disk and expand *strpool* using the *stratis* command. Verify the expansion. Finally, remove the entire pool and confirm the deletion. Use the *lsblk* command and verify that the disks used for the Stratis labs no longer show Stratis information. (Hint: Stratis Volume-Managing File System).

Chapter 15

Local File Systems and Swap

This chapter describes the following major topics:

- Understand file systems and their benefits, categories, and types
- Review file system types: Ext3/Ext4, XFS, VFAT, and ISO9660
- Know file system administration commandset
- Mount and unmount file systems manually and persistently
- Determine and use UUID
- Apply and use file system label
- Monitor file system and directory usage
- Create and mount different types of local file systems in partitions
- Create and mount XFS file system in VDO volume
- Create, mount, and resize Ext4 and XFS file systems in LVM
- Create, mount, and expand Stratis file system
- Understand, create, and activate swap in partitions and LVM

RHCSA Objectives:

31. Configure systems to mount file systems at boot by Universally Unique ID (UUID) or label
32. Add new partitions and logical volumes, and swap to a system nondestructively (the first part of this objective is covered in more detail in Chapter 14)
33. Create, mount, unmount, and use vfat, ext4, and xfs file systems
35. Extend existing logical volumes (more details in Chapter 14 also)

File systems are the most common structures created in partitions and volumes regardless of the underlying storage management solution employed. They are logical containers employed for file storage and can be optimized, resized, mounted, and unmounted independently. They must be connected to the root of the directory hierarchy in order to be accessed by users and applications. This may be accomplished automatically at system boot or manually when required. File systems can be mounted or unmounted using their unique identifiers, labels, or device files. There is a whole slew of commands available for file system creation and administration; some of them are file system type specific while others are general.

The other common structure created in partitions and logical volumes is the swap space. Swapping provides a mechanism to move out and in pages of idle data between the physical memory and the swap. Swap areas act as extensions to the physical memory, and they may be activated or deactivated independent of swap spaces located in other partitions and volumes.

This chapter is the last one in the three chapter series (the other two being [Chapters 13](#) and [14](#)) on storage management. It elaborates on file systems and swap, and demonstrates their creation and management in several exercises. It also highlights the tools to monitor their usage.

File Systems and File System Types

A *file system* is a logical container that stores files and directories. Each file system is created in a discrete partition, VDO volume, logical volume, or Stratis pool. A typical production RHEL system usually has numerous file systems. During OS installation, only two file systems—`/` and `/boot`—are created in the default disk layout, but you can design a custom disk layout and construct separate containers to store dissimilar information. Typical additional file systems created during an installation are `/home`, `/opt`, `/tmp`, `/usr`, and `/var`. The two mandatory file systems—`/` and `/boot`—are required for installation and booting.

Storing disparate data in distinct file systems versus storing all data in a single file system offers the following advantages:

- ✓ Make any file system accessible (mount) or inaccessible (unmount) to users independent of other file systems. This hides or reveals information contained in that file system.
- ✓ Perform file system repair activities on individual file systems

- ✓ Keep dissimilar data in separate file systems
- ✓ Optimize or tune each file system independently
- ✓ Grow or shrink a file system independent of other file systems

RHEL supports several types of file systems that may be categorized in three basic groups: *disk-based*, *network-based*, and *memory-based*. Disk-based file systems are typically created on physical drives using SATA, USB, Fibre Channel, and other technologies. Network-based file systems are essentially disk-based file systems shared over the network for remote access. Memory-based file systems are virtual; they are created at system startup and destroyed when the system goes down. Disk-based and network-based file systems store information persistently, while any data saved in virtual file systems does not survive across system reboots.

[Table 15-1](#) lists and explains various common disk- and network-based file system types supported in RHEL 8.

File System Type	Category	Description
Ext3	Disk	The third generation of the extended file system supports metadata journaling for faster recovery, superior reliability, allows the creation of up to 16 million subdirectories, and supports larger file systems and bigger files than its predecessor.
Ext4	Disk	The fourth generation of the extended file system was developed as the successor to Ext3. It supports all the features of Ext3 in addition to a larger file system size, a bigger file size, an unlimited number of subdirectories, metadata and quota journaling, and extended attributes.
XFS	Disk	XFS is a highly scalable and high-performing file system. It supports metadata journaling for fast crash recovery, and online defragmentation, expansion, quota journaling, and extended user attributes. XFS is the default file system type in RHEL 8.
VFAT	Disk	This file system is used for post-Windows 95 file system formats on hard disks, USB drives, and floppy disks.
ISO9660	Disk	This is used for optical file systems such as CD and DVD.
NFS	Network	Network File System. A shared directory or file system for remote access by other Linux systems.
AutoFS	Network	Auto File System. An NFS file system set to mount and unmount automatically on remote client systems.

Table 15-1 File System Types

This chapter covers Ext3, Ext4, XFS, and VFAT file systems at length. It also touches upon mounting and unmounting ISO9660. For a brief discussion on memory-based file systems, see [Chapter 02 “Initial Interaction with the System”](#). NFS and AutoFS are discussed in [Chapter 17 “Network File System”](#).

Extended File Systems

Extended file systems have been part of RHEL for many years. The first generation is obsolete and is no longer supported. The second, third, and

fourth generations are currently available and supported. The fourth generation is the latest in the series and is superior in features and enhancements to its predecessors.

The structure of an extended file system is built on a partition or logical volume at the time of file system creation. This structure is divided into two sets. The first set holds the file system's metadata and it is very tiny. The second set stores the actual data, and it occupies almost the entire partition or the logical volume (VDO, LVM, and Stratis) space.

The metadata includes the *superblock*, which keeps vital file system structural information, such as the type, size, and status of the file system, and the number of data blocks it contains. Since the superblock holds such critical information, it is automatically replicated and maintained at various known locations throughout the file system. The superblock at the beginning of the file system is referred to as the *primary superblock*, and all of its copies as *backup superblocks*. If the primary superblock is corrupted or lost, it renders the file system inaccessible. One of the backup superblocks is then used to supplant the corrupted or lost primary superblock to bring the file system back to its normal state.

The metadata also contains the *inode table*, which maintains a list of *index node (inode)* numbers. Each file is assigned an inode number at the time of its creation, and the inode number holds the file's attributes such as its type, permissions, ownership, owning group, size, and last access/modification time. The inode also holds and keeps track of the pointers to the actual data blocks where the file contents are located.

The Ext3 and Ext4 file systems support a journaling mechanism that provides them with the ability to recover swiftly after a system crash. Both Ext3 and Ext4 file systems keep track of recent changes in their metadata in a *journal* (or log). Each metadata update is written in its entirety to the journal after completion. The system peruses the journal of each extended file system following the reboot after a crash to determine if there are any errors, and it recovers the file system rapidly using the latest metadata information stored in its journal. The ext2 file system does not support journaling, but the support for journaling may be added to it if required.

In contrast to Ext3 that supports file systems up to 16TiB and files up to 2TiB, Ext4 supports very large file systems up to 1EiB (ExbiByte) and files up to 16TiB (TebiByte). Additionally, Ext4 uses a series of contiguous physical blocks on the hard disk called *extents*, resulting in improved read and write performance with reduced fragmentation. Ext4 supports extended user attributes, acl mount options (to support file permission allocation to specific users and groups), as well as metadata and quota journaling.

XFS File System

The *X File System* (XFS) is a high-performing 64-bit extent-based journaling file system type. XFS allows the creation of file systems and files up to 8EiB (ExbiByte). It does not run file system checks at system boot; rather, it relies on you to use the *xfs_repair* utility to manually fix any issues. XFS sets the extended user attributes and acl mount options by default on new file systems. It enables defragmentation on mounted and active file systems to keep as much data in contiguous blocks as possible for faster access. The only major caveat with using XFS is its inability to shrink.

Like Ext3 and Ext4, XFS also uses journaling for metadata operations, guaranteeing the consistency of the file system against abnormal or forced unmounting. The journal information is read and any pending metadata transactions are replayed when the XFS file system is remounted.

XFS uses sophisticated techniques in its architecture for speedy input/output performance. It can be snapshot in a mounted, active state. The snapshot can then be used for backup or other purposes.

VFAT File System

VFAT (*Virtual File Allocation Table*) is an extension to the legacy *FAT* file system type, also called *FAT16*, that was introduced in early versions of MS-DOS. The support for FAT16 was later added to Microsoft Windows, MacOS, and some UNIX versions, enabling them to read and write files written in that format. FAT16 had limitations; it was designed to use no more than 8.3 characters in filenames, limiting filenames to a maximum of eight characters plus three characters as an extension. Moreover, it only allowed filenames to begin with a letter or number and to not contain spaces. FAT16 treated lowercase and uppercase letters alike.

VFAT was introduced with Microsoft Windows 95 and it has since been available. It supports 255 characters in filenames including spaces and periods; however, it still does not differentiate between lowercase and uppercase letters. VFAT support was added to Linux several years ago. A VFAT file system may be created on hard drives, but it is primarily used on removable media, such as floppy and USB flash drives, for exchanging data between Linux and Windows.

ISO9660 File System

This file system type conforms to the ISO 9660 standard, hence the name. It is used for removable optical disc media such as CD/DVD drives for transporting software and patches, and operating system images in ISO

format between computers. The ISO9660 format originated from the *High-Sierra File System* (HSFS) format, and it has now been enhanced to include innovative features.

File System Management

Managing file systems involves such operations as creating, mounting, labeling, viewing, growing, shrinking, unmounting, and removing them. These management tasks are common to both Extended and XFS types. Most of these functions are also applicable to VFAT and a few to optical file systems.

In [Chapters 13](#) and [14](#), you created several partitions, VDO volumes, LVM logical volumes, and Stratis volumes. However, you did not initialize them with a file system type (except for Stratis), and therefore you could not mount or use them. Later, you destroyed all the partitions and the volumes that were created. You also deleted the LVM volume group and the Stratis pool. All the disks were returned to their unused state after the completion of the exercises.

Here is a listing of the block devices to confirm the current state of the disks:

```
[user1@server2 ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0  10G  0 disk 
└─sda1     8:1    0   1G  0 part /boot
  └─sda2     8:2    0   9G  0 part
    ├─rhel-root 253:0  0   8G  0 lvm  /
    └─rhel-swap 253:1  0   1G  0 lvm  [SWAP]
sdb        8:16   0 250M 0 disk 
sdc        8:32   0 250M 0 disk 
sdd        8:48   0 250M 0 disk 
sde        8:64   0 250M 0 disk 
sdf        8:80   0   4G  0 disk 
sdg        8:96   0   1G  0 disk 
sdh        8:112  0   1G  0 disk
```

The output verifies the unused state and availability status for all the disks—*sdb* through *sdh*. You should be able to reuse them in the exercises in this chapter.

File System Administration Commands

In order to create and manage file systems, RHEL offers a number of commands of which some are limited to their operations on the Extended, XFS, or VFAT file system type, while others are general and applicable to all file system types. [Table 15-2](#) describes common file system administration commands.

Command	Description
Extended File System	
e2label	Modifies the label of a file system
mke2fs	Creates a file system. Can also be invoked as mkfs.ext mkfs.ext4, mkfs -t ext3, and mkfs -t ext4.
resize2fs	Resizes a file system. This command is automatically run when the lvresize command is run with the -r switch.
tune2fs	Tunes or displays file system attributes
XFS	
mkfs.xfs	Creates a file system. Can also be invoked as mkfs -t xfs
xfs_admin	Tunes file system attributes
xfs_growfs	Extends the size of a file system
xfs_info	Exhibits information about a file system
VFAT	
mkfs.vfat	Creates a file system. It is equivalent to using mkfs -t vfat
General File System Commands	
blkid	Displays block device attributes including their UUIDs and labels
df	Reports file system utilization
du	Calculates disk usage of directories and file systems
lsblk	Lists block devices and file systems and their attributes including their UUIDs and labels
mount	Mounts a file system for user access. Displays currently mounted file systems.
umount	Unmounts a file system

Table 15-2 File System Management Commands

Most of these commands are used in this chapter.

Mounting and Unmounting File Systems

In order to enable users to access files and application programs in a file system, the file system must be connected to the directory structure at a desired attachment point, which is referred to as the *mount point*. A mount point in essence is any empty directory that is created and used for this purpose.

There are many file systems already mounted on your system, such as the root file system mounted on / and the boot file system mounted on /boot. Both of them are empty directories and are reserved to connect the two file systems to the directory hierarchy. You can use the *mount* command to view information about mounted file systems. The following shows the XFS file systems only:

```
[user1@server2 ~]$ mount -t xfs  
/dev/mapper/rhel-root on / type xfs (rw,relatime,seclabel,attr2,inode64,noquota)  
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

The “-t xfs” option makes the command to only show the file systems initialized with the XFS type.

The *mount* command is also used for mounting a file system to a mount point, and this action is performed with the *root* user privileges. The command requires the absolute pathnames of the file system block device and the mount point name. It also accepts the UUID or label of the file system in lieu of the block device name. Options are available with this command to mount all or a specific type of file system. The *mount* command is also used to mount other types of file systems such as those located in removable media. Upon successful mount, the kernel places an entry for the file system in the /proc/self/mounts file.



A mount point should be empty when an attempt is made to mount a file system on it, otherwise the content of the mount point will hide. As well, the mount point must not be in use or the mount attempt will fail.

The *mount* command supports numerous options that may be used as required to override its default behavior. We can also specify multiple comma-separated options. [Table 15-3](#) describes some common options.

Option	Description
acl (noacl)	Enables (disables) the support for ACLs
auto (noauto)	Mounts (does not mount) the file system when option is specified
defaults	Mounts a file system with all the default values (async, auto, rw, etc.)
_netdev	Used for a file system that requires network connectivity in place before it can be mounted and NFS are examples.
remount	Remounts an already mounted file system to change or disable an option
ro (rw)	Mounts a file system read-only (read/write)

Table 15-3 Common mount Command Options

The opposite of the *mount* command is *umount*, which is used to detach a file system from the directory hierarchy and make it inaccessible to users and applications. This command expects the absolute pathname to the block device containing the file system or its mount point name in order to detach it. Options are available with *umount* to unmount all or a specific type of file system. The kernel removes the corresponding file system entry from the */proc/self/mounts* file after it has been successfully disconnected.

Determining the UUID of a File System

Every Extended and XFS file system has a 128-bit (32 hexadecimal characters) UUID (*Universally Unique IDentifier*) assigned to it at the time of its creation. In contrast, UUIDs assigned to vfat file systems are 32-bit (8 hexadecimal characters) in length. Assigning a UUID makes the file system unique among many other file systems that potentially exist on the system. The primary benefit of using a UUID is the fact that it always stays persistent across system reboots. A UUID is used by default in RHEL 8 in the */etc/fstab* file for any file system that is created by the system in a standard partition.



RHEL attempts to mount all file systems listed in the */etc/fstab* file at reboots. Each file system has an associated device file and UUID, but may or may not have a corresponding label. The system checks for the presence of each file system's device file, UUID, or label, and then attempts to mount it.

The */boot* file system, for instance, is located in a partition and the device file associated with it is on *server2* is */dev/sda1*. You can use the *xfs_admin*

command, the *blkid* command, or the *lsblk* command as follows to determine its UUID:

```
[user1@server2 ~]$ sudo xfs_admin -u /dev/sda1
UUID = c1ff315e-4320-442c-a3c5-36db403b53f2
[user1@server2 ~]$ sudo blkid /dev/sda1
/dev/sda1: UUID="c1ff315e-4320-442c-a3c5-36db403b53f2" TYPE="xfs" PARTUUID="7685
6860-01"
[user1@server2 ~]$ sudo lsblk -f /dev/sda1
NAME FSTYPE LABEL UUID                                     MOUNTPOINT
sda1 xfs          c1ff315e-4320-442c-a3c5-36db403b53f2 /boot
```

The UUID reported by the above commands for the */boot* file system is "c1ff315e-4320-442c-a3c5-36db403b53f2". If you *grep* for the string "boot" on the */etc/fstab* file, you will see that the system uses this UUID to mount */boot*. A discussion on the */etc/fstab* file is provided later in this chapter.



For extended file systems, you can use the *tune2fs* command in addition to the *blkid* and *lsblk* commands to determine the UUID.

EXAM TIP: Knowing how to find the UUID of a file system created in a standard partition or with Stratis is important.

A UUID is also assigned to a file system that is created in a VDO or LVM volume; however, it need not be used in the *fstab* file, as the device files associated with the logical volumes are always unique and persistent.

Labeling a File System

A unique label may be used instead of a UUID to keep the file system association with its device file exclusive and persistent across system reboots. A label is limited to a maximum of 12 characters on the XFS file system and 16 characters on the Extended file system. By default, no labels are assigned to a file system at the time of its creation.

The */boot* file system is located in the */dev/sda1* partition and its type is XFS. You can use the *xfs_admin* command, the *blkid* command, or the *lsblk* command as follows to determine its label:

```
[user1@server2 ~]$ sudo xfs_admin -l /dev/sda1
label =
[user1@server2 ~]$ sudo blkid /dev/sda1
/dev/sda1: UUID="c1ff315e-4320-442c-a3c5-36db403b53f2" TYPE="xfs" PARTUUID="7685
6860-01"
[user1@server2 ~]$ sudo lsblk -f /dev/sda1
NAME FSTYPE LABEL UUID                                     MOUNTPOINT
sda1 xfs          c1ff315e-4320-442c-a3c5-36db403b53f2 /boot
```

The output discloses that there is currently no label assigned to the */boot* file system.

A label is not needed on a file system if you intend to use its UUID or if it is created in a VDO or LVM logical volume; however, you can still apply one using the `xfs_admin` command with the `-L` option. Labeling an XFS file system requires that the target file system be unmounted.

The following example demonstrates the steps to unmount `/boot`, set the label “bootfs” on its device file, and remount it:

```
[user1@server2 ~]$ sudo umount /boot
[user1@server2 ~]$ sudo xfs_admin -L bootfs /dev/sda1
writing all SBs
new label = "bootfs"
[user1@server2 ~]$ sudo mount /boot
```

You can confirm the new label by executing `sudo xfs_admin -l /dev/sda1`, `sudo blkid /dev/sda1`, or `sudo lsblk -f /dev/sda1`.



For extended file systems, you can use the `e2label` command to apply a label and the `tune2fs`, `blkid`, and `lsblk` commands to view and verify.

Now you can replace the `UUID="c1ff315e-4320-442c-a3c5-36db403b53f2"` for `/boot` in the `fstab` file with `LABEL=bootfs`, and unmount and remount `/boot` as demonstrated above for confirmation.

A label may also be applied to a file system created in a VDO or LVM volume; however, it is not recommended for use in the `fstab` file, as the device files for these logical volumes are always unique and remain persistent across system reboots.

Automatically Mounting a File System at Reboots

File systems defined in the `/etc/fstab` file are mounted automatically at reboots. This file must contain proper and complete information for each listed file system. An incomplete or inaccurate entry might leave the system in an undesirable or unbootable state. Another benefit of adding entries to this file is that you only need to specify one of the four attributes—block device name, UUID, label, or mount point—of the file system that you wish to mount manually with the `mount` command. The `mount` command obtains the rest of the information from this file. Similarly, you only need to specify one of these attributes with the `umount` command to detach it from the directory hierarchy.

The default `fstab` file contains entries for file systems that are created at the time of installation. On `server2`, for instance, this file currently has the following three entries:

```
/dev/mapper/rhel-root          /      xfs  defaults  0 0
UUID=c1ff315e-4320-442c-a3c5-36db403b53f2 /boot   xfs  defaults  0 0
/dev/mapper/rhel-swap          swap   swap  defaults  0 0
```

EXAM TIP: Any missing or invalid entry in this file may render the system unbootable. You will have to boot the system in emergency mode to fix this file. Ensure that you understand each field in the file for both file system and swap entries.

The format of this file is such that each row is broken out into six columns to identify the required attributes for each file system to be successfully mounted. Here is what the columns contain:

Column 1: Defines the physical or virtual device path where the file system is resident, or its associated UUID or label. There can be entries for network file systems here as well.

Column 2: Identifies the mount point for the file system. For swap partitions, use either “none” or “swap”.

Column 3: Specifies the type of file system such as Ext3, Ext4, XFS, VFAT, or ISO9660. For swap, the type “swap” is used. You may use “auto” instead to leave it up to the *mount* command to determine the type of the file system.

Column 4: Identifies one or more comma-separated options to be used when mounting the file system. See [Table 15-3](#) for a description of some of the options, consult the manual pages of the *mount* command or the *fstab* file for additional options and details.

Column 5: Is used by the *dump* utility to ascertain the file systems that need to be dumped. A value of 0 (or the absence of this column) disables this check. This field is applicable only on Extended file systems; XFS does not use it.

Column 6: Expresses the sequence number in which to run the *e2fsck* (file system check and repair utility for Extended file system types) utility on the file system at system boot. By default, 0 is used for memory-based, remote, and removable file systems, 1 for */*, and 2 for */boot* and other physical file systems. 0 can also be used for */*, */boot*, and other physical file systems you don’t want to be checked or repaired. This field is applicable only on Extended file systems; XFS does not use it.



A 0 in columns 5 and 6 for XFS, virtual, remote, and removable file system types has no meaning. You do not need to add them for these file system types.

This file is edited manually, so care must be observed to circumvent syntax and typing errors.

Monitoring File System Usage

On a live system, you'll often need to check file system usage to know if a mounted file system requires an expansion for growth or a clean up to generate free space. This involves examining the used and available spaces for a file system. The *df* (*disk free*) command has been used for this purpose. It reports usage details for mounted file systems. By default, this command reports the numbers in KBs unless the *-m* or *-h* option is specified to view the sizes in MBs or human-readable format.



This command may not produce correct information for VDO and Stratis file systems. Use their own tools for viewing usage.

Let's run this command with the *-h* option on *server2*:

```
[user1@server2 ~] $ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        900M    0  900M   0% /dev
tmpfs          915M    0  915M   0% /dev/shm
tmpfs          915M  9.2M  906M   1% /run
tmpfs          915M    0  915M   0% /sys/fs/cgroup
/dev/mapper/rhel-root  8.0G  3.9G  4.2G  48% /
/dev/sr0         6.7G  6.7G    0 100% /mnt
/dev/sda1       1014M 169M  846M  17% /boot
tmpfs          183M   28K  183M   1% /run/user/42
tmpfs          183M  4.0K  183M   1% /run/user/1000
```

The output shows the file system device file or type in column 1, followed by the total, used, and available spaces in columns 2, 3, and 4, and then the usage percentage and mount point in columns 5 and 6.

There are a few other useful flags available with the *df* command that can produce the desired output. These flags include:

- T to add the file system type to the output (example: **df -hT**)
- x to exclude the specified file system type from the output (example: **df -hx tmpfs**)
- t to limit the output to a specific file system type (example: **df -t xfs**)
- i to show inode information (example: **df -hi**)

You may use *-h* with any of these examples to print information in human-readable format.

Calculating Disk Usage

In contrast to the *df* command that returns usage information for an entire file system, the *du* command reports the amount of space a file or directory occupies. By default, it shows the output in KBs; however, you can use the *-m*

or -h option to view the output in MBs or human-readable format. In addition, you can view a usage summary with the -s switch and a grand total with -c.

Let's run this command on the `/usr/bin` directory to view the usage summary:

```
[user1@server2 ~]$ du -sh /usr/bin  
206M    /usr/bin
```

To add a “total” row to the output and with numbers displayed in KBs:

```
[user1@server2 ~]$ du -sc /usr/bin  
210540  /usr/bin  
210540  total
```

Try this command with different options on the `/usr/sbin/lvm` file and observe the results.

Exercise 15-1: Create and Mount Ext4, VFAT, and XFS File Systems in Partitions

This exercise should be done on `server2` as `user1` with `sudo` where required.

In this exercise, you will create 2 x 100MB partitions on the `/dev/sdb` disk, initialize them separately with the Ext4 and VFAT file system types, define them for persistence using their UUIDs, create mount points called `/ext4fs1` and `/vfatfs1`, attach them to the directory structure, and verify their availability and usage. Moreover, you will use the disk `/dev/sdc` and repeat the above procedure to establish an XFS file system in it and mount it on `/xfsfs1`.

1. Apply the label “msdos” to the `sdb` disk using the `parted` command:

```
[user1@server2 ~]$ sudo parted /dev/sdb mklabel msdos
```

2. Create 2 x 100MB primary partitions on `sdb` with the `parted` command:

```
[user1@server2 ~]$ sudo parted /dev/sdb mkpart primary 1 101m  
[user1@server2 ~]$ sudo parted /dev/sdb mkpart primary 102 201m
```

3. Initialize the first partition (`sdb1`) with Ext4 file system type using the `mkfs` command:

```
[user1@server2 ~]$ sudo mkfs -t ext4 /dev/sdb1
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 97280 1k blocks and 24384 inodes
Filesystem UUID: 681c0535-4e44-4043-8bef-17413c897f9b
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

4. Initialize the second partition (*sdb2*) with VFAT file system type using the *mkfs* command:

```
[user1@server2 ~]$ sudo mkfs -t vfat /dev/sdb2
mkfs.fat 4.1 (2017-01-24)
```

5. Initialize the whole disk (*sdc*) with the XFS file system type using the *mkfs.xfs* command. Add the *-f* flag to force the removal of any old partitioning or labeling information from the disk.

```
[user1@server2 ~]$ sudo mkfs.xfs /dev/sdc -f
meta-data=/dev/sdc              isize=512    agcount=4, agsize=16000 blks
                                =                      sectsz=512  attr=2, projid32bit=1
                                =                      crc=1      finobt=1, sparse=1, rmapbt=0
                                =                      reflink=1
data     =                      bsize=4096   blocks=64000, imaxpct=25
        =                      sunit=0      swidth=0 blks
naming   =version 2            bsize=4096   ascii-ci=0, ftype=1
log      =internal log         bsize=4096   blocks=1368, version=2
        =                      sectsz=512  sunit=0 blks, lazy-count=1
realtime =none                 extsz=4096   blocks=0, rtextents=0
```

6. Determine the UUIDs for all three file systems using the *lsblk* command:

NAME	FSTYPE	LABEL	UUID	MOUNTPOINT
sdb				
└─sdb1	ext4		681c0535-4e44-4043-8bef-17413c897f9b	
└─sdb2	vfat		E4BB-3A5E	
sdc	xfs		1ebce60e-d73d-4737-adcb-c1410b2b8c5f	

7. Open the */etc/fstab* file, go to the end of the file, and append entries for the file systems for persistence using their UUIDs:

```
UUID="681c0535-4e44-4043-8bef-17413c897f9b" /ext4fs1 ext4 defaults 0 0
UUID="E4BB-3A5E" /vfatfs1 vfat defaults 0 0
UUID="1ebce60e-d73d-4737-adcb-c1410b2b8c5f" /xfsfs1 xfs defaults 0 0
```

8. Create mount points */ext4fs1*, */vfatfs1*, and */xfsfs1* for the three file systems using the *mkdir* command:

```
[user1@server2 ~]$ sudo mkdir /ext4fs1 /vfatfs1 /xfsfs1
```

9. Mount the new file systems using the *mount* command. This command will fail if there are any invalid or missing information in the file.

```
[user1@server2 ~]$ sudo mount -a
```

10. View the mount and availability status as well as the types of all three file systems using the *df* command:

```
[user1@server2 ~]$ df -hT  
.....  
/dev/sdb1          ext4      88M   1.6M   80M   2% /ext4fs1  
/dev/sdb2          vfat     95M     0    95M   0% /vfatfs1  
/dev/sdc          xfs      245M   15M   231M   6% /xfsfs1
```

The output verifies the creation and availability status of the three file systems. They are added to the *fstab* file for persistence. A system reboot at this point will remount them automatically. These file systems may now be used to store files.

Exercise 15-2: Create and Mount XFS File System in VDO Volume

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create a VDO volume called *vdo1* of logical size 16GB on the *sdf* disk (the actual size of this disk is 4GB). You will initialize the volume with the XFS file system type, define it for persistence using its device files, create a mount point called */xfsvdo1*, attach it to the directory structure, and verify its availability and usage.

Prior to proceeding, ensure that the steps outlined in [Exercise 13-6](#) for VDO software installation and service startup have been accomplished.

1. Create a VDO volume *vdo1* on the *sdf* disk with a logical size of 16GB and a slab size of 128MB:

```
[user1@server2 ~]$ sudo vdo create --device /dev/sdf --name vdo1 --vdoSlabSize 128  
--vdoLogicalSize 16G  
Creating VDO vdo1  
Starting VDO vdo1  
Starting compression on VDO vdo1  
VDO instance 2 volume is ready at /dev/mapper/vdo1
```

2. List the new VDO volume using the *vdo* and *lsblk* commands:

```
[user1@server2 ~]$ sudo vdo list
vdo1
[user1@server2 ~]$ lsblk /dev/sdf
NAME   MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdf      8:80    0   4G  0 disk
└─vdo1 253:2    0  16G  0 vdo
```

The output shows the logical volume size (16GB), type (vdo), and the actual size (4GB) of the underlying disk.

3. Initialize the VDO volume with the XFS file system type with the *mkfs.xfs* command. The VDO volume device file is */dev/mapper/vdo1* as indicated in the output in step 1. Add the *-f* flag to force the removal of any old partitioning or labeling information from the disk.

```
[user1@server2 ~]$ sudo mkfs.xfs /dev/mapper/vdo1
meta-data=/dev/mapper/vdo1          isize=512    agcount=4, agsize=1048576 blks
        =                      sectsz=4096  attr=2, projid32bit=1
        =                      crc=1       finobt=1, sparse=1, rmapbt=0
        =                      reflink=1
data     =                      bsize=4096   blocks=4194304, imaxpct=25
        =                      sunit=0     swidth=0 blks
naming   =version 2             bsize=4096   ascii-ci=0, ftype=1
log      =internal log         bsize=4096   blocks=2560, version=2
        =                      sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none                  extsz=4096   blocks=0, rtextents=0
```

4. Open the */etc/fstab* file, go to the end of the file, and append the following entry for the file system for persistence using its device file:

```
/dev/mapper/vdo1          /xfsvdo1 xfs  x-systemd.requires=vdo.service 0 0
```

Make sure to include the option **x-systemd.requires=vdo.service** (or try **_netdev** instead) with the file system entry in the *fstab* file or your system will land into the emergency target on the next reboot. This option holds the *mount* command from mounting this file system until the *vdo.service* service has been operational.

5. Create the mount point */xfsvdo1* using the *mkdir* command:

```
[user1@server2 ~]$ sudo mkdir /xfsvdo1
```

6. Mount the new file system using the *mount* command. This command will fail if there are any invalid or missing information in the file.

```
[user1@server2 ~]$ sudo mount -a
```

The *mount* command with the *-a* flag is a validation test for the *fstab* file. It should always be executed after updating this file and before rebooting the server to avoid landing the system in an unbootable state.

7. View the mount and availability status as well as the types of the VDO file system using the *lsblk* and *df* commands:

```
[user1@server2 ~]$ lsblk /dev/sdf
NAME   MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdf      8:80    0   4G  0 disk
└vdo1 253:2    0  16G  0 vdo  /xfsvdo1
[user1@server2 ~]$
[user1@server2 ~]$ df -hT /xfsvdo1
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/mapper/vdo1 xfs   16G  147M  16G  1% /xfsvdo1
```

The *lsblk* command output illustrates the VDO volume name (*vdo1*), the disk it is located on (*sdf*), the actual (4GB) and logical (16GB) sizes, and the mount point (*/xfsvdo1*) where the file system is connected to the directory structure.

The *df* command shows the logical size of the file system and its usage status, but it does not reveal the underlying disk information. This file system is added to the *fstab* file for persistence, meaning a future system reboot will remount it automatically. This file system may now be used to store files.

Refer to [Chapter 13 “Basic Storage Partitioning”](#) for details on VDO.

Exercise 15-3: Create and Mount Ext4 and XFS File Systems in LVM Logical Volumes

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create a volume group called *vgfs* comprised of a 160MB physical volume created in a partition on the */dev/sdd* disk. The PE size for the volume group should be set at 16MB. You will create two logical volumes called *ext4vol* and *xfsvol* of sizes 80MB each and initialize them with the Ext4 and XFS file system types. You will ensure that both file systems are persistently defined using their logical volume device filenames. You will create mount points called */ext4fs2* and */xfsfs2*, mount the file systems, and verify their availability and usage.

1. Create a 150MB partition on the *sdd* disk using the *parted* command:

```
[user1@server2 ~]$ sudo parted /dev/sdd mkpart pri 1 181m
```

2. Initialize the *sdd1* partition for use in LVM using the *pvcreate* command:

```
[user1@server2 ~]$ sudo pvcreate /dev/sdd1
Physical volume "/dev/sdd1" successfully created.
```

3. Create the volume group *vgfs* with a PE size of 16MB using the physical volume *sdd1*:

```
[user1@server2 ~]$ sudo vgcreate -s 16 vgfs /dev/sdd1
Volume group "vgfs" successfully created
```

The PE size is not easy to alter after a volume group creation, so ensure it is defined as required at creation.

4. Create two logical volumes *ext4vol* and *xfsvol* of size 80MB each in *vgfs* using the *lvcreate* command:

```
[user1@server2 ~]$ sudo lvcreate -n ext4vol -L 80 vgfs
Logical volume "ext4vol" created.
[user1@server2 ~]$ sudo lvcreate -n xfsvol -L 80 vgfs
Logical volume "xfsvol" created.
```

5. Format the *ext4vol* logical volume with the Ext4 file system type using the *mkfs.ext4* command:

```
[user1@server2 ~]$ sudo mkfs.ext4 /dev/vgfs/ext4vol
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 81920 1k blocks and 20480 inodes
Filesystem UUID: 6a30d72f-003b-4797-a7ba-c6f6d36f7a58
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

You may alternatively use **sudo mkfs -t ext4 /dev/vgfs/ext4vol**.

6. Format the *xfsvol* logical volume with the XFS file system type using the *mkfs.xfs* command:

```
[user1@server2 ~]$ sudo mkfs.xfs /dev/vgfs/xfsvol
meta-data=/dev/vgfs/xfsvol      isize=512    agcount=4, agsize=5120 blks
                                =          sectsz=512   attr=2, projid32bit=1
                                =          crc=1     finobt=1, sparse=1, rmapbt=0
data      =          bsize=4096   blocks=20480, imaxpct=25
                                =          sunit=0    swidth=0 blks
naming    =version 2           bsize=4096   ascii-ci=0, ftype=1
log       =internal log        bsize=4096   blocks=1368, version=2
                                =          sectsz=512   sunit=0 blks, lazy-count=1
realtime  =none                extsz=4096   blocks=0, rtextents=0
```

You may use **sudo mkfs -t xfs /dev/vgfs/xfsvol** instead.

7. Open the */etc/fstab* file, go to the end of the file, and append entries for the file systems for persistence using their device files:

/dev/vgfs/ext4vol	/ext4fs2 ext4 defaults 0 0
/dev/vgfs/xfsvol	/xfsfs2 xfs defaults 0 0

8. Create mount points */ext4fs2* and */xfsfs2* using the *mkdir* command:

```
[user1@server2 ~]$ sudo mkdir /ext4fs2 /xfsfs2
```

9. Mount the new file systems using the *mount* command. This command will fail if there is any invalid or missing information in the file.

```
[user1@server2 ~]$ sudo mount -a
```

Fix any issues in the file if reported.

10. View the mount and availability status as well as the types of the new LVM file systems using the *lsblk* and *df* commands:

```
[user1@server2 ~]$ lsblk /dev/sdd
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdd        8:48   0  250M  0 disk 
└─sdd1     8:49   0 172M  0 part 
  ├─vgfs-ext4vol 253:3   0   80M  0 lvm   /ext4fs2
  └─vgfs-xfsvol 253:4   0   80M  0 lvm   /xfsfs2
[user1@server2 ~]$
[user1@server2 ~]$ df -hT | grep fs2
/dev/mapper/vgfs-ext4vol ext4      74M  1.6M  67M  3% /ext4fs2
/dev/mapper/vgfs-xfsvol  xfs      75M  4.9M  70M  7% /xfsfs2
```

The *lsblk* command output illustrates the LVM logical volumes (*ext4vol* and *xfsvol*), the disk they are located on (*sdd*), the sizes (80MB), and the mount points (*/ext4fs2* and */xfsfs2*) where the file system are connected to the directory structure.

The *df* command shows the size and usage information. Both file systems are added to the *fstab* file for persistence, meaning future system reboots will remount them automatically. They may now be used to store files.

Exercise 15-4: Resize Ext4 and XFS File Systems in LVM Logical Volumes

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will grow the size of the *vgfs* volume group that was created in [Exercise 15-3](#) by adding the whole *sde* disk to it. You will extend the *ext4vol* logical volume along with the file system it contains by 40MB using two separate commands. You will extend the *xfsvol* logical volume along with the file system it contains by 40MB using a single command. You will verify the new extensions.

1. Initialize the *sde* disk and add it to the *vgfs* volume group:

```
[user1@server2 ~]$ sudo pvcreate /dev/sde
Physical volume "/dev/sde" successfully created.
[user1@server2 ~]$ sudo vgextend vgfs /dev/sde
Volume group "vgfs" successfully extended
```

2. Confirm the new size of *vgfs* using the *vgs* and *vgdisplay* commands:

```
[user1@server2 ~]$ sudo vgs
  VG #PV #LV #SN Attr   VSize   VFree
  rhel  1   2   0 wz--n- <9.00g      0
  vgfs   2   2   0 wz--n- 400.00m 240.00m
[user1@server2 ~]$ sudo vgdisplay vgfs
--- Volume group ---
VG Name          vgfs
System ID
Format          lvm2
Metadata Areas   2
Metadata Sequence No 4
VG Access        read/write
VG Status        resizable
MAX LV           0
Cur LV            2
Open LV           2
Max PV           0
Cur PV            2
Act PV            2
VG Size          400.00 MiB
PE Size          16.00 MiB
Total PE         25
Alloc PE / Size  10 / 160.00 MiB
Free  PE / Size  15 / 240.00 MiB
VG UUID          OeTIVf-IByc-gVbN-oW9q-5k8L-sK5A-ixxotw
```

There are now two physical volumes in the volume group and the total size increased to 400MiB.

3. Grow the logical volume *ext4vol* and the file system it holds by 40MB using the *lvextend* and *fsadm* command pair. Make sure to use an uppercase L to specify the size. The default unit is MiB. The plus sign (+) signifies an addition to the current size.

```
[user1@server2 ~]$ sudo lvextend -L +40 /dev/vgfs/ext4vol
Rounding size to boundary between physical extents: 48.00 MiB.
Size of logical volume vgfs/ext4vol changed from 80.00 MiB (5 extents) to 128.
00 MiB (8 extents).
Logical volume vgfs/ext4vol successfully resized.

[user1@server2 ~]$ sudo fsadm resize /dev/vgfs/ext4vol
resize2fs 1.44.3 (10-July-2018)
Filesystem at /dev/mapper/vgfs-ext4vol is mounted on /ext4fs2; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/mapper/vgfs-ext4vol is now 131072 (1k) blocks long.
```

The *resize* subcommand instructs the *fsadm* command to grow the file system to the full length of the specified logical volume.

4. Grow the logical volume *xfsvol* and the file system (-r) it holds by (+) 40MB using the *lvresize* command:

```
[user1@server2 ~]$ sudo lvresize -r -L +40 /dev/vgfs/xfsvol
  Rounding size to boundary between physical extents: 48.00 MiB.
  Size of logical volume vgfs/xfsvol changed from 80.00 MiB (5 extents) to 128.0
0 MiB (8 extents).
  Logical volume vgfs/xfsvol successfully resized.
meta-data=/dev/mapper/vgfs-xfsvol isize=512    agcount=4, agsize=5120 blks
          =                      sectsz=512  attr=2, projid32bit=1
          =                      crc=1     finobt=1, sparse=1, rmapbt=0
          =                      reflink=1
data      =                      bsize=4096   blocks=20480, imaxpct=25
          =                      sunit=0    swidth=0 blks
naming    =version 2           bsize=4096   ascii-ci=0, ftype=1
log       =internal log       bsize=4096   blocks=1368, version=2
          =                      sectsz=512  sunit=0 blks, lazy-count=1
realtime  =none               extsz=4096   blocks=0, rtextents=0
data blocks changed from 20480 to 32768
```

5. Verify the new extensions to both logical volumes using the *lvs* command. You may also issue the *lvdisplay* or *vgdisplay* command instead.

```
[user1@server2 ~]$ sudo lvs | grep vol
  ext4vol vgfs -wi-ao---- 128.00m

  xfsvol  vgfs -wi-ao---- 128.00m
```

6. Check the new sizes and the current mount status for both file systems using the *df* and *lsblk* commands:

```
[user1@server2 ~]$ df -hT | grep -E 'ext4vol|xfsvol'
/dev/mapper/vgfs-xfsvol xfs        123M  5.4M  118M  5% /xfsfs2
/dev/mapper/vgfs-ext4vol ext4       120M  1.6M  110M  2% /ext4fs2
[user1@server2 ~]$
[user1@server2 ~]$ lsblk /dev/sdd /dev/sde
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdd        8:48   0  250M  0 disk
└─sdd1     8:49   0 172M  0 part
  ├─vgfs-ext4vol 253:2   0 128M  0 lvm  /ext4fs2
  └─vgfs-xfsvol  253:3   0 128M  0 lvm  /xfsfs2
sde        8:64   0  250M  0 disk
└─vgfs-ext4vol 253:2   0 128M  0 lvm  /ext4fs2
  └─vgfs-xfsvol  253:3   0 128M  0 lvm  /xfsfs2
```

The outputs reflect the new sizes (128MB) for both file systems. They also indicate their mount status.

This concludes the exercise.

Exercise 15-5: Create, Mount, and Expand XFS File System in Stratis Volume

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create a Stratis pool called *strpool* and a file system (*strfs2*) in it (the file system type will be XFS) by reusing the 1GB *sdg* disk from [Chapter 14](#). You will display information about the pool, file system, and device used. You will expand the pool to include another 1GB disk *sdh* and confirm.

For further details on Stratis storage management solution, refer to [Chapter 14](#) “Advanced Storage Partitioning”. Prior to proceeding, ensure that the steps outlined in [Exercise 14-7](#) for Stratis software installation and service startup have been accomplished.

1. Create a pool called *strpool* using the *sdg* disk and verify the creation:

```
[user1@server2 ~]$ sudo stratis pool create strpool /dev/sdg
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
strpool          1 GiB           52 MiB
```

The pool is created with a single disk included, and confirmed.

2. Show the block device used in the pool:

```
[user1@server2 ~]$ sudo stratis blockdev list strpool
Pool Name  Device Node      Physical Size   State   Tier
strpool    /dev/sdg          1 GiB        In-use  Data
```

3. Create a file system called *strfs2* in the pool and verify the creation:

```
[user1@server2 ~]$ sudo stratis filesystem create strpool strfs2
[user1@server2 ~]$ sudo stratis filesystem list
Pool Name  Name     Used     Created            Device          UUID
strpool    strfs2  546 MiB  Nov 05 2019 08:08  /stratis/strpool/strfs2  439eeef8
45f543788c2bf3f9d6923d92
```

4. Determine the UUID of the new file system to be added to the *fstab* file:

```
[user1@server2 ~]$ sudo lsblk /stratis/strpool/strfs2 -o UUID
UUID
439eeef8-45f5-4378-8c2b-f3f9d6923d92
```

5. Open the */etc/fstab* file, go to the end of the file, and append the following entry for the file system for persistence using the UUID:

```
UUID="439eeef8-45f5-4378-8c2b-f3f9d6923d92"  /strfs2  xfs  x-systemd.requires=stratisd.service 0 0
```

Make sure to include the option **x-systemd.requires=stratisd.service** (or try **netdev** instead) with the file system entry in the *fstab* file or your system will land into the emergency target on the next reboot. This option holds the

mount command from mounting this file system until the *stratisd.service* service has been started successfully.

6. Create the mount point */strfs2*:

```
[user1@server2 ~]$ sudo mkdir /strfs2
```

7. Mount the new file system using the *mount* command as follows. This command will fail if there are any invalid or missing information in the file.

```
[user1@server2 ~]$ sudo mount -a
```

8. Check the pool usage:

```
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
strpool          1 GiB            598 MiB
```

9. Check the file system usage:

```
[user1@server2 ~]$ sudo stratis filesystem list
Pool Name  Name     Used     Created           Device           UUID
strpool    strfs2   546 MiB  Nov 05 2019 08:08  /stratis/strpool/strfs2  439eeeef8
45f543788c2bf3f9d6923d92
```

10. Grow the pool by adding the *sdh* disk to it and confirm the growth:

```
[user1@server2 ~]$ sudo stratis pool add-data strpool /dev/sdh
[user1@server2 ~]$
[user1@server2 ~]$ sudo stratis pool list
Name      Total Physical Size  Total Physical Used
strpool          2 GiB            602 MiB
```

The file system *strfs2* will automatically expand to take advantage of the additional pool capacity when required.

This concludes the exercise.

Swap and its Management

Physical memory (or main memory) in the system is a finite temporary storage resource employed for loading kernel and running user programs and applications. *Swap space* is an independent region on the physical disk used for holding idle data until it is needed. The system splits the physical memory into small logical chunks called *pages* and maps their physical locations to virtual locations on the swap to facilitate access by system processors. This physical-to-virtual mapping of pages is stored in a data structure called *page table*, and it is maintained by the kernel.

When a program or process is spawned, it requires space in the physical memory to run and be processed. Although many programs can run concurrently, the physical memory cannot hold all of them at once. The kernel monitors the memory usage. As long as the free memory remains below a high threshold, nothing happens. However, when the free memory falls below that threshold, the system starts moving selected idle pages of data from physical memory to the swap space in an effort to make room to accommodate other programs. This piece in the process is referred to as *page out*. Since the system CPU performs the process execution in a round-robin fashion, when the system needs this paged-out data for execution, the CPU looks for that data in the physical memory and a *page fault* occurs, resulting in moving the pages back to the physical memory from the swap. This return of data to the physical memory is referred to as *page in*. The entire process of paging data out and in is known as *demand paging*.

RHEL systems with less physical memory but high memory requirements can become over busy with paging out and in. When this happens, they do not have enough cycles to carry out other useful tasks, resulting in degraded system performance. The excessive amount of paging that affects the system performance is called *thrashing*.

When thrashing begins, or when the free physical memory falls below a low threshold, the system deactivates idle processes and prevents new processes from being launched. The idle processes are only reactivated and new processes are only allowed to be started when the system discovers that the available physical memory has climbed above the threshold level and thrashing has ceased.

Determining Current Swap Usage

The size of a swap area should not be less than the amount of physical memory; however, depending on workload requirements, it may be twice the size or larger. It is also not uncommon to see systems with less swap than the actual amount of physical memory. This is especially witnessed on systems with a huge physical memory size.

RHEL offers the *free* command to view memory and swap space utilization. Use this command to view how much physical memory is installed (total), used (used), available (free), used by shared library routines (shared), holding data before it is written to disk (buffers), and used to store frequently accessed data (cached) on the system. The -h flag may be specified with the command to list the values in human-readable format, otherwise -k for KB, -m for MB, -g for GB, and so on are also supported. Add -t with the command to display a

line with the “total” at the bottom of the output. Here is a sample output from `server2`:

```
[user1@server2 ~]$ free -ht
              total        used         free      shared  buff/cache   available
Mem:       1.8Gi       895Mi      447Mi        20Mi      486Mi      749Mi
Swap:      1.0Gi        0B      1.0Gi
Total:     2.8Gi       895Mi      1.4Gi
```

The output indicates that the system has 1.8GiB of total memory of which 895MiB is in use and 447MiB is free. It also shows on the same line the current memory usages by temporary (`tmpfs`) file systems (20MiB) and kernel buffers and page cache (486MiB). It also illustrates an estimate of free memory available to start new processes (749MiB).

On the subsequent row, it reports the total swap space (1.0GiB) configured on the system with a look at used (0 Bytes) and free (1.0GiB) space. The last line prints the combined usage summary of the main memory and swap.

Try `free -hts 3` and `free -htc 2` to refresh the output every three seconds (-s) and to display the output twice (-c).

The `free` command reads memory and swap information from the `/proc/meminfo` file to produce the report. The values are shown in KBs by default, and they are slightly off from what is shown in the above screenshot with `free`. Here are the relevant fields from this file:

```
[user1@server2 ~]$ cat /proc/meminfo |grep -E 'Mem|Swap'
MemTotal:      1873316 kB
MemFree:       457848 kB
MemAvailable:  767288 kB
SwapCached:    0 kB
SwapTotal:     1048572 kB
SwapFree:      1048572 kB
```

This data depicts the system’s runtime memory and swap usage, as it is located in a virtual file.

Prioritizing Swap Spaces

On many production RHEL servers, you may find multiple swap areas configured and activated to meet the workload demand. The default behavior of RHEL is to use the first activated swap area and move on to the next when the first one is exhausted. The system allows us to prioritize one area over the other by adding the option “pri” to the swap entries in the `fstab` file. This flag supports a value between -2 and 32767 with -2 being the default. A higher value of “pri” sets a higher priority for the corresponding swap region. For swap areas with an identical priority, the system alternates between them.

Swap Administration Commands

In order to create and manage swap spaces on the system, the *mkswap*, *swapon*, and *swapoff* commands are available. Use *mkswap* to initialize a partition for use as a swap space. Once the swap area is ready, you can activate or deactivate it from the command line with the help of the other two commands, or set it up for automatic activation by placing an entry in the *fstab* file. The *fstab* file accepts the swap area's device file, UUID, or label.

Exercise 15-6: Create and Activate Swap in Partition and Logical Volume

This exercise should be done on *server2* as *user1* with *sudo* where required.

In this exercise, you will create one swap area in a new 40MB partition called *sdb3* using the *mkswap* command. You will create another swap area in a 144MB logical volume called *swapvol* in *vgfs*. You will add their entries to the */etc/fstab* file for persistence. You will use the UUID and priority 1 for the partition swap and the device file and priority 2 for the logical volume swap. You will activate them and use appropriate tools to validate the activation.

EXAM TIP: Use the *lsblk* command to determine available disk space.

1. Use the *parted*'s *print* subcommand on the *sdb* disk and the *vgs* command on the *vgfs* volume group to determine available space for a new 40MB partition and a 144MB logical volume:

```
[user1@server2 ~]$ sudo parted /dev/sdb print
Number  Start   End     Size    Type      File system  Flags
 1      1049kB  101MB   99.6MB  primary   ext4
 2      102MB   201MB   99.6MB  primary   fat16
[user1@server2 ~]$ sudo vgs vgfs
  VG #PV #LV #SN Attr   VSize   VFree
  vgfs 2    2    0  wz--n- 400.00m 144.00m
```

The outputs show 49MB (250MB minus 201MB) free space on the *sdb* disk and 144MB free space in the volume group.

2. Create partition called *sdb3* of size 40MB using the *parted* command:

```
[user1@server2 ~]$ sudo parted /dev/sdb mkpart primary 202 242
```

3. Create logical volume *swapvol* of size 144MB in *vgfs* using the *lvcreate* command:

```
[user1@server2 ~]$ sudo lvcreate -L 144 -n swapvol vgfs
Logical volume "swapvol" created.
```

4. Construct swap structures in `sdb3` and `swapvol` using the `mkswap` command:

```
[user1@server2 ~]$ sudo mkswap /dev/sdb3 ; sudo mkswap /dev/vgfs/swapvol
Setting up swapspace version 1, size = 38 MiB (39841792 bytes)
no label, UUID=94d86394-0eb3-4c52-a643-969ae193e1a9
Setting up swapspace version 1, size = 144 MiB (150990848 bytes)
no label, UUID=7ff47d80-a86c-467c-836d-5835977db11e
```

5. Edit the `fstab` file and add entries for both swap areas for auto-activation on reboots. Obtain the UUID for partition swap with `lsblk -f /dev/sdb3` and use the device file for logical volume. Specify their priorities.

```
UUID="94d86394-0eb3-4c52-a643-969ae193e1a9"    swap    swap    pri=1 0 0
/dev/vgfs/swapvol                                swap    swap    pri=2 0 0
```

EXAM TIP: You will not be given any credit for this work if you forget to add entries to the `fstab` file.

6. Determine the current amount of swap space on the system using the `swapon` command:

```
[user1@server2 ~]$ sudo swapon
NAME      TYPE      SIZE USED PRIO
/dev/dm-1  partition 1024M  0B   -2
```

There is one 1GB swap area on the system configured at the default priority of -2.

7. Activate the new swap regions using the `swapon` command:

```
[user1@server2 ~]$ sudo swapon -a
```

The command would display errors if there were any issues with swap entries in the `fstab` file.

8. Confirm the activation using the `swapon` command or by viewing the `/proc/swaps` file:

```
[user1@server2 ~]$ sudo swapon
NAME      TYPE      SIZE USED PRIO
/dev/dm-1  partition 1024M  0B   -2
/dev/sdb3  partition  38M   0B   1
/dev/dm-11 partition 144M   0B   2
[user1@server2 ~]$
[user1@server2 ~]$ cat /proc/swaps
Filename                Type      Size     Used   Priority
/dev/dm-1                partition 1048572  0       -2
/dev/sdb3                partition 38908   0       1
/dev/dm-11               partition 147452  0       2
```

The two new swap regions are activated and listed in the above outputs. Their sizes and priorities are also visible. The device mapper

device files for the logical volumes and the device file for the partition swap are also exhibited.

9. Issue the *free* command to view the reflection of swap numbers on the Swap and Total lines:

```
[user1@server2 ~]$ free -ht
              total        used         free      shared  buff/cache   available
Mem:       1.8Gi     889Mi     457Mi      20Mi     483Mi    757Mi
Swap:      1.2Gi        0B     1.2Gi
Total:     3.0Gi     889Mi     1.6Gi
```

The total swap is now 1.2GiB. This concludes the exercise.

Chapter Summary

This chapter covered two major storage topics: file systems and swap. These structures are created in partitions or logical volumes irrespective of the underlying storage management solution used to build them.

The chapter began with a detailed look at the concepts, categories, benefits, and types of file systems. We reviewed file system administration and monitoring utilities. We discussed the concepts around mounting and unmounting file systems. We examined the UUID associated with file systems and applied labels to file systems. We analyzed the file system table and added entries for auto-activating file systems at reboots. We explored tools for reporting file system usage and calculating disk usage. We performed a number of exercises on file system creation and administration in partitions and VDO, LVM, and Stratis volumes to reinforce the concepts and theory learned in this and the last two chapters.

We touched upon the concepts of swapping and paging, and studied how they work. We performed exercises on creating, activating, viewing, deactivating, and removing swap spaces, as well as configuring them for auto-activation at system reboots.

Review Questions

1. Which two file systems are created in a default RHEL 8 installation?
2. What would the command *lvresize -r -L +30 /dev/vg02/lvol2* do?
3. XFS is the default file system type in RHEL 8. True or False?
4. What type of information does the *b/blkid* command display?
5. What would the command *xfs_admin -L bootfs /dev/sda1* do?
6. The *lsblk* command cannot be used to view file system UUIDs. True or False?
7. What is the process of paging out and paging in known as?
8. What would the command *mkswap /dev/sdc2* do?

9. What would happen if you mount a file system on a directory that already contains files in it?
10. A UUID is always assigned to a file system at its creation time. True or False?
11. Arrange the activities to create and activate a swap space while ensuring persistence: (a) swapon, (b) update fstab, (c) mkswap, and (d) reboot?
12. The difference between the primary and backup superblocks is that the primary superblock includes pointers to the data blocks where the actual file contents are stored whereas the backup superblocks don't. True or False?
13. What would the command *mkfs.ext4 /dev/vgtest/lvoltest* do?
14. Arrange the tasks in correct sequence: umount file system, mount file system, create file system, remove file system.
15. Which of these statements is wrong with respect to file systems: (a) optimize each file system independently, (b) keep dissimilar data in separate file systems, (c) grow and shrink a file system independent of other file systems, and (d) file systems cannot be expanded independent of other file systems.
16. Which command can be used to create a label for an XFS file system?
17. What would the *mount* command do with the -a switch?
18. What would the command *df -t xfs* do?
19. What is the difference between the *mkfs.ext4* and *mke2fs* commands?
20. Which command can be used to determine the total and used physical memory and swap in the system?
21. Which virtual file contains information about the current swap?
22. The */etc/fstab* file can be used to activate swap spaces automatically at system reboots. True or False?
23. What is the default file system type used for optical media?
24. The *xfs_repair* command must be run on a mounted file system. True or False?
25. Provide two commands that can be used to activate and deactivate swap spaces manually.
26. Provide the *fstab* file entry for an Ext4 file system located in device */dev/mapper/vg20-lv1* and mounted with default options on the */ora1* directory.
27. What is the name of the virtual file that holds currently mounted file system information?
28. Which option must be specified with persistent VDO file system mounting to avoid landing the system in an unbootable state?
29. Both Ext3 and Ext4 file system types support journaling. True or False?

30. Name three commands that can be employed to view the UUID of an XFS file system?

Answers to Review Questions

1. / and /boot.
2. The command provided will expand the logical volume /vol2 in volume group vg02 along with the file system it contains by 30MB.
3. True.
4. The *blkid* command displays attributes for block devices.
5. The command provided will apply the specified label to the XFS file system in /dev/sda1.
6. False.
7. The process of paging out and in is known as demand paging.
8. The command provided will create swap structures in the /dev/vdc2 partition.
9. The files in the directory will hide.
10. True.
11. c/a/b or c/b/a.
12. False.
13. The command provided will format /dev/vgtest/lvoltest logical volume with Ext4 file system type.
14. Create, mount, unmount, and remove.
15. d is incorrect.
16. The *xfs_admin* command can be used to create a label for an XFS file system.
17. The command provided will mount all file systems listed in the /etc/fstab file but are not currently mounted.
18. The command provided will display all mounted file systems of type XFS.
19. No difference.
20. The *free* command.
21. The /proc/swaps file contains information about the current swap.
22. True.
23. The default file system type for optical devices is ISO9660.
24. False.
25. The *swapon* and *swapoff* commands.
26. /dev/mapper/vg20-lv1 /ora1 swap defaults 0 0
27. The *mounts* file under /proc/self directory.
28. The _netdev option.
29. True.

30. You can use the `xfs_admin`, `lsblk`, and `blkid` commands to view the UUID of an XFS file system.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

To use the same storage that was used in the labs for [Chapters 13 and 14](#), destroy all traces of the partitions, volume groups, and pools that were created before proceeding. Alternatively, you can add more storage to `server2` for the following labs.

Lab 15-1: Create VFAT, Ext4, and XFS File Systems in Partitions & Mount Persistently

As `user1` with `sudo` on `server2`, create three 70MB primary partitions on one of the available 250MB disks (`lsblk`) by invoking the `parted` utility directly at the command prompt. Apply label “msdos” if the disk is new. Initialize partition 1 with VFAT, partition 2 with Ext4, and partition 3 with XFS file system types. Create mount points `/vfatfs5`, `/ext4fs5`, and `/xfsfs5`, and mount all three manually. Determine the UUIDs for the three file systems, and add them to the `fstab` file. Unmount all three file systems manually, and execute `mount -a` to mount them all. Run `df -h` for verification. (Hint: File System Management).

Lab 15-2: Create XFS File System in VDO Volume and Mount Persistently

As `user1` with `sudo` on `server2`, ensure that VDO software is installed and the VDO service is enabled and started. Create a volume `vdo5` with a logical size 16GB on an available 4GB disk (`lsblk`) using the `vdo` command. Select an appropriate slab size for the volume. Verify the volume creation with the `vdo`, `lsblk`, and `vdostats` commands. Initialize the volume with XFS file system type. Create mount point `/vdofs5`, and mount it manually. Add the file system information to the `fstab` file, and use “`_netdev`” as a mount option. Unmount the file system manually, and execute `mount -a` to mount it back. Run `df -h` to confirm. (Hint: File System Management).

Lab 15-3: Create Ext4 and XFS File Systems in LVM Volumes and Mount Persistently

As *user1* with *sudo* on *server2*, initialize an available 250MB disk for use in LVM (*/sb/blk*). Create volume group *vg200* with PE size 8MB and add the physical volume. Create two logical volumes *lv200* and *lv300* of sizes 120MB and 100MB. Use the *vgs*, *pvs*, *lvs*, and *vgdisplay* commands for verification. Initialize the volumes with Ext4 and XFS file system types. Create mount points */vmfs5* and */vmfs6*, and mount them manually. Add the file system information to the *fstab* file using their device files. Unmount the file systems manually, and execute **mount -a** to mount them back. Run **df -h** to confirm. (Hint: File System Management).

Lab 15-4: Extend Ext4 and XFS File Systems in LVM Volumes

As *user1* with *sudo* on *server2*, initialize an available 250MB disk for use in LVM (*/sb/blk*). Add the new physical volume to volume group *vg200*. Expand logical volumes *lv200* and *lv300* along with the underlying file systems to 200MB and 250MB. Use the *vgs*, *pvs*, *lvs*, *vgdisplay*, and *df* commands for verification. (Hint: File System Management).

Lab 15-5: Create XFS File System in Stratis Volume and Mount Persistently

As *user1* with *sudo* on *server2*, confirm Stratis software is installed and the Stratis service is enabled and started. Create pool *strpool5* on an available 1GB disk and confirm. Create file system *strfs5* in the pool and verify. Create mount point */strfs5*. Determine the UUID of the Stratis file system and add it to the *fstab* file for persistence. Use the *x-systemd.requires=stratisd.service* as a mount option. Reboot the system and confirm mounting with **df -h**. (Hint: File System Management).

Lab 15-6: Create Swap in Partition and LVM Volume and Activate Persistently

As *user1* with *sudo* on *server2*, create two 100MB partitions on an available 250MB disk (*/sb/blk*) by invoking the *parted* utility directly at the command prompt. Apply label “msdos” if the disk is new. Initialize one of the partitions with swap structures. Apply label *swappart* to the swap partition, and add it to

the *fstab* file. Execute **swapon -a** to activate it. Run **swapon -s** to confirm activation.

Initialize the other partition for use in LVM. Expand volume group *vg200* (Lab 15-4) by adding this physical volume. Create logical volume *swapvol* of size 180MB. Use the *vgs*, *pvs*, *lvs*, and *vgdisplay* commands for verification.

Initialize the logical volume for swap. Add an entry to the *fstab* file for the new swap area using its device file. Execute **swapon -a** to activate it. Run **swapon -s** to confirm activation. (Hint: Swap and its Management).

Chapter 16

Networking, Network Devices, and Network Connections

This chapter describes the following major topics:

- Overview of basic networking concepts: hostname, IPv4, network classes, subnetting, subnet mask, CIDR, protocol, TCP/UDP, well-known ports, ICMP, Ethernet address, IPv6, IPv4/IPv6 differences, consistent device naming, etc.
- Change hostname of the system
- Understand the concepts of network device and connection
- Anatomy of a network connection profile
- Know network device and connection management tools and techniques
- Configure network connections by hand and using commands
- Describe the hosts table
- Test network connectivity using hostname and IP address

RHCSA Objectives:

47. Configure IPv4 and IPv6 addresses

A computer network is formed when two or more physical or virtual computers are connected together for sharing resources and data. The computers may be linked via wired or wireless means, and a device such as a switch is used to interconnect several computers to allow them to communicate with one another on the network. There are numerous concepts and terms that need to be grasped in order to work effectively and efficiently with network device and network connection configuration and troubleshooting, and several other network services. This chapter provides a wealth of that information.

For a system to be able to talk to other systems, one of its network devices must have a connection profile attached containing a unique IP address, hostname, and other essential network parameters. The network assignments may be configured statically or obtained automatically from a DHCP server. Few files are involved in the configuration, which may be modified by hand or using commands. Testing follows the configuration to confirm the system's ability to communicate.

Networking Fundamentals

The primary purpose of computer networks is to allow users to share data and resources. A simple network is formed when two computers are interconnected. Using a networking device such as a *switch*, this network can be expanded to include additional computers, as well as printers, scanners, storage, and other devices (collectively referred to as *nodes* or *entities*). A computer on the network can be configured to act as a file server, storage server, or as a gateway to the Internet for the rest of the networked computers. Nodes may be interconnected using wired or wireless means. A corporate network may have thousands of nodes linked via a variety of data transmission media. The Internet is the largest network of networks with millions of nodes interconnected.

There are many elementary concepts and terms that you need to grasp before being able to configure network interfaces, connection profiles, and client/server setups that are elaborated in this and other chapters. As well, there are many configuration files and commands related to various network services that you need to understand thoroughly in order to manage a RHEL-based environment effectively. Some of the concepts, terms, configuration files, and commands are explained in this chapter.

Hostname

A *hostname* is a unique alphanumeric label (the hyphen (-), underscore (_), and period (.) characters are also allowed) that is assigned to a node to identify it on the network. It can consist of up to 253 characters. It is normally allotted based on the purpose and principal use of the system. In RHEL, the hostname is stored in the */etc/hostname* file.

The hostname can be viewed with several different commands, such as *hostname*, *hostnamectl*, *uname*, and *nmcli*, as well as by displaying the content of the */etc/hostname* file. Let's run these commands on *server1*:

```
[user1@server1 ~]$ hostname  
server1.example.com  
[user1@server1 ~]$ hostnamectl --static  
server1.example.com  
[user1@server1 ~]$ uname -n  
server1.example.com  
[user1@server1 ~]$ nmcli general hostname  
server1.example.com  
[user1@server1 ~]$ cat /etc/hostname  
server1.example.com
```

All the above commands displayed the same output.

Exercise 16-1: Change System Hostname

This exercise should be done on both *server1* and *server2* as *user1* with *sudo* where required.

In this exercise, you will change the hostnames of both lab servers persistently. You will rename *server1.example.com* to *server10.example.com* by editing a file and restarting the corresponding service daemon. You will rename *server2.example.com* to *server20.example.com* using a command. You will validate the change on both systems.

On *server1*:

1. Open the */etc/hostname* file in a text editor and change the current entry to the following:

```
server10.example.com
```

2. Execute the *systemctl* command to restart the *systemd-hostnamed* service daemon and verify the new hostname with the *hostname* command:

```
[user1@server1 ~]$ sudo systemctl restart systemd-hostnamed  
[user1@server1 ~]$ hostname  
server10.example.com
```

3. To view the reflection of the new hostname in the command prompt, log off and log back in as *user1*. The new prompt will look like:

```
[user1@server10 ~] $
```

On server2:

1. Execute the *hostnamectl* command to change the hostname to [server20.example.com](#):

```
[user1@server2 ~] $ sudo hostnamectl set-hostname server20.example.com
```

2. To view the reflection of the new hostname in the command prompt, log off and log back in as *user1*. You can also use the *hostname* command to view the new name.

```
[user1@server20 ~] $  
[user1@server20 ~] $ hostname  
server20.example.com
```

You can also change the system hostname using the *nmcli* command. For instance, you could have used **nmcli general hostname [server20.example.com](#)** to rename [server2.example.com](#). The *nmcli* command is explained in detail later in this chapter.

EXAM TIP: You need to know only one of the available methods to change the hostname of the system.

Going forward, you will be using the new hostnames *server10* and *server20*.

IPv4 Address

IPv4 stands for *Internet Protocol version 4* and represents a unique 32-bit software address that every single entity on the network must have in order to communicate with other entities. It was the first version of IP that was released for public use. IPv4 addresses are also referred to as *dotted-quad* addresses, and they can be assigned on a temporary or permanent basis. Temporary addresses are referred to as *dynamic* addresses and are typically leased from a DHCP server for a specific period of time. Permanent addresses, on the other hand, are called *static* addresses and they are manually set.

You can use the *ip* command with the *addr* argument to view the current IP assignments on the system. Let's run this command on *server10* and see what it returns:

```
[user1@server10 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:bf:55:8e brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.110/24 brd 192.168.0.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:feb5:558e/64 scope link
        valid_lft forever preferred_lft forever
```

The output indicates one configured network connection (number 2 above) called *enp0s3* with IPv4 address 192.168.0.110 assigned to it. The other connection (number 1 above), represented as *lo*, is a special purpose software device reserved for use on every Linux system. Its IPv4 address is always 127.0.0.1, and it is referred to as the system's *loopback* (or *localhost*) address. Network programs and applications that communicate with the local system employ this hostname.

Network Classes

An IPv4 address is comprised of four period-separated octets ($4 \times 8 = 32$ bit address) that are divided into a *network* portion (or network ID/bits) comprising of the *Most Significant Bits* (MSBs) and a *node* portion (or node/host ID/bits) containing the *Least Significant Bits* (LSBs). The network portion identifies the correct destination network, and the node portion represents the correct destination node on that network. Public network addresses are classified into three categories: class A, class B, and class C. Private network addresses are classified into two categories: class D and class E. Class D addresses are multicast and they are employed in special use cases only. Class E addresses are experimental and are reserved for future use.

Class A

Class A addresses are used for large networks with up to 16 million nodes. This class uses the first octet as the network portion and the rest of the octets as the node portion. The total number of usable network and node addresses can be up to 126 and 16,777,214, respectively. The network address range for class A networks is between 0 and 127. See an example below of a random class A IP address, which also shows two reserved addresses:

10.121.51.209	(class A IP address)
10.121.51.0	(network address)

10.121.51.255 (broadcast address)

The 0 and 255 (highlighted) are network and broadcast addresses, and they are always reserved.

Class B

Class B addresses are used for mid-sized networks with up to 65 thousand nodes. This class employs the first two octets as the network portion and the other two as the node portion. The total number of usable network and node addresses can be up to 16,384 and 65,534, respectively. The network address range for class B networks is between 128 and 191. See an example below of a random class B IP address, which also shows two reserved addresses:

161.121.51.209	(class B IP address)
161.121.51.0	(network address)
161.121.51.255	(broadcast address)

The 0 and 255 (highlighted) are network and broadcast addresses, and they are always reserved.

Class C

Class C addresses are employed for small networks with up to 254 nodes. This class uses the first three octets as the network portion and the last octet as the node portion. The total number of usable network and node addresses can be up to 2,097,152 and 254, respectively. The network address range for class C networks is between 192 and 223. See an example below of an arbitrary class C IP address, which also shows two reserved addresses:

215.121.51.209	(class C IP address)
215.121.51.0	(network address)
215.121.51.255	(broadcast address)

The 0 and 255 (highlighted) are network and broadcast addresses, and they are always reserved.

Class D

Class D addresses range from 224 to 239.

Class E

Class E addresses range from 240 to 255.

Subnetting

Subnetting is a technique by which a large network address space is divided into several smaller and more manageable logical subnetworks, referred to as *subnets*. Subnetting results in reduced network traffic, improved network performance, and de-centralized and easier administration, among other benefits. Subnetting does not touch the network bits; it uses the node bits only.

The following should be kept in mind when dealing with subnetting:

- ✓ Subnetting does not increase the number of IP addresses in a network.
In fact, it reduces the number of usable addresses.
- ✓ All nodes in a given subnet have the same subnet mask.
- ✓ Each subnet acts as an isolated network and requires a router to talk to other subnets.
- ✓ The first and the last IP address in a subnet are reserved. The first address points to the subnet itself, and the last address is the broadcast address.

Subnet Mask

A *subnet mask* or *netmask* is the network portion plus the subnet bits. It segregates the network bits from the node bits. It is used by routers to pinpoint the start and end of the network/subnet portion and the start and end of the node portion for a given IP address.

The subnet mask, like an IP address, can be represented in either decimal or binary notation. The 1s in the subnet mask isolate the subnet bits from the node bits that contain 0s. The default subnet masks for class A, B, and C networks are 255.0.0.0, 255.255.0.0, and 255.255.255.0, respectively.

To determine the subnet address for an arbitrary IP address, such as 192.168.12.72 with netmask 255.255.255.224, write the IP address in binary format. Then write the subnet mask in binary format with all network and subnet bits set to 1 and all node bits set to 0. Then perform a logical AND operation. For each matching 1 you get a 1, otherwise you get a 0. The following highlights the ANDed bits:

11000000.10101000.00001100.01001000 (IP address 192.168.12.72)
11111111.11111111.11111111.11100000 (subnet mask 255.255.255.224)
=====
11000000.10101000.00001100.01000000 (subnet IP 192.168.12.64 in binary format)

192 . 168 . 12 . 64

(subnet IP in decimal format)

This calculation enables you to ascertain the subnet address from a given IP and subnet mask.

Classless Inter-Domain Routing (CIDR) Notation

Classless Inter-Domain Routing (CIDR) is a technique designed to control the quick depletion of IPv4 addresses and the rapid surge in the number of routing tables required to route IPv4 traffic on the network and the Internet. This technique was introduced as a substitute for the *classful* scheme, which was not scalable and had other limitations. Using CIDR, IPv4 addresses can be allocated in custom blocks suitable for networks of all sizes. This technique has resulted in smaller and less cluttered routing tables. CIDR was originally designed to address IPv4 needs; however, it has been extended to support IPv6 as well.

An IPv4 address written in CIDR notation has a leading forward slash (/) character followed by the number of routing bits. A sample class C IP address of 192.168.0.20 with the default class C subnet mask of 255.255.255.0 will be written as 192.168.0.20/24. This notation presents a compact method of denoting an IP address along with its subnet mask.

Protocol

A *protocol* is a set of rules governing the exchange of data between two network entities. These rules include how data is formatted, coded, and controlled. The rules also provide error handling, speed matching, and data packet sequencing. In other words, a protocol is a common language that all nodes on the network speak and understand. Protocols are defined in the */etc/protocols* file. An excerpt from this file is provided below:

```
[user1@server10 ~]$ cat /etc/protocols
ip      0      IP          # internet protocol, pseudo protocol number
hopopt  0      HOPOPT     # hop-by-hop options for ipv6
icmp    1      ICMP       # internet control message protocol
igmp    2      IGMP       # internet group management protocol
ggp     3      GGP         # gateway-gateway protocol
ipv4   4      IPv4       # IPv4 encapsulation
st     5      ST          # ST datagram mode
tcp    6      TCP         # transmission control protocol
cbt    7      CBT         # CBT, Tony Ballardie <A.Ballardie@cs.ucl.ac.uk>
egp    8      EGP         # exterior gateway protocol
....
```

Column 1 in the output lists the name of a protocol, followed by the associated port number, alias, and a short description in columns 2, 3, and 4.

Some common protocols are TCP, UDP, IP, and ICMP.

TCP and UDP Protocols

TCP (*Transmission Control Protocol*) and UDP (*User Datagram Protocol*) protocols are responsible for transporting data packets between network entities. TCP is reliable, connection-oriented, and point-to-point. It inspects for errors and sequencing upon a packet's arrival on the destination node, and returns an acknowledgement to the source node, establishing a point-to-point connection with the peer TCP layer on the source node. If the packet is received with an error or if it is lost in transit, the destination node requests a resend of the packet. This ensures guaranteed data delivery and makes TCP reliable. Due to its reliability and connection-oriented nature, TCP is widely implemented in network applications.

UDP, in contrast, is unreliable, connectionless, and multi-point. If a packet is lost or contains errors upon arrival at the destination, the source node is unaware of it. The destination node does not send an acknowledgment back to the source node. A common use of this protocol is in broadcast-only applications where reliability is not sought.

Well-Known Ports

Both TCP and UDP use ports for data transmission between a client and its server program. Ports are either well-known or private. A well-known port is reserved for an application's exclusive use, and it is standardized across all network operating systems. Well-known ports are defined in the `/etc/services` file, an excerpt of which is presented below:

```
[user1@server10 ~]$ cat /etc/services
.....
ftp          21/tcp
ftp          21/udp      fsp  fspd
ssh          22/tcp
ssh          22/udp      # The Secure Shell (SSH) Protocol
telnet       23/tcp
telnet       23/udp
# 24 - private mail system
lmtp         24/tcp      # LMTP Mail Delivery
lmtp         24/udp      # LMTP Mail Delivery
smtp         25/tcp      mail
smtp         25/udp      mail
time         37/tcp      timserver
....
```

Column 1 lists the official name of a network service, followed by the port number and transport layer protocol the service uses, optional aliases, and

comments in successive columns.

Some common services and the ports they listen on are FTP (*File Transfer Protocol*) 21, SSH (*Secure Shell*) 22, SMTP (*Simple Mail Transfer Protocol*) 25, DNS (*Domain Name System*) 53, HTTP (*HyperText Transfer Protocol*) 80, NTP (*Network Time Protocol*) 123, secure HTTP (*HyperText Transfer Protocol Secure*) 443, and rsyslog 514.

A private port, on the other hand, is an arbitrary number generated when a client application attempts to establish a communication session with its server process. This port number no longer exists after the session has ended.

ICMP Protocol

The *Internet Control Message Protocol* (ICMP) is a key protocol. It is primarily used for testing and diagnosing network connections. Commands such as *ping* uses this protocol to send a stream of messages to remote network devices to examine their health and report statistical and diagnostic information. The report includes the number of packets transmitted, received, and lost; a round-trip time for individual packets with an overall average; a percentage of packets lost during the communication; and so on. See a sample below that shows two packets (-c2) sent from *server10* to the IP address of *server20*:

```
[user1@server10 ~]$ ping -c2 192.168.0.120
PING 192.168.0.120 (192.168.0.120) 56(84) bytes of data.
64 bytes from 192.168.0.120: icmp_seq=1 ttl=64 time=0.325 ms
64 bytes from 192.168.0.120: icmp_seq=2 ttl=64 time=0.402 ms

--- 192.168.0.120 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 45ms
rtt min/avg/max/mdev = 0.325/0.363/0.402/0.042 ms
```

Other commands, such as *traceroute*, also employ this protocol for route determination and debugging between network entities. The IPv6 version of ICMP is referred to as *ICMPv6* and it is used by tools such as *ping6* and *tracepath6*.

Ethernet Address

An *Ethernet* address represents an exclusive 48-bit address that is used to identify the correct destination node for data packets transmitted from the source node. The data packets include hardware addresses for the source and the destination node. The Ethernet address is also referred to as the *hardware, physical, link layer, or MAC address*.

You can use the *ip* command to list all network interfaces available on the system along with their Ethernet addresses:

```
[user1@server10 ~]$ ip addr | grep ether  
link/ether 08:00:27:bf:55:8e brd ff:ff:ff:ff:ff:ff
```

IP and hardware addresses work hand in hand, and a combination of both is critical to identifying the correct destination node on the network. A network protocol called *Address Resolution Protocol* (ARP) is used to enable IP and hardware addresses to work in tandem. ARP determines the hardware address of the destination node when its IP address is known.

IPv6 Address

With the explosive growth of the Internet, the presence of an extremely large number of network nodes requiring an IP, and an ever-increasing demand for additional addresses—the conventional IPv4 address space, which provides approximately 4.3 billion addresses—has been exhausted. To meet the future demand, a new version of IP is now available and its use is on the rise. This new version is referred to as IPv6 (*IP version 6*) or IPng (*IP next generation*). By default, IPv6 is enabled in RHEL 8 on all configured network connections.

IPv6 is a 128-bit software address, providing access to approximately 340 undecillion (340 followed by 36 zeros) addresses. This is an extremely large space, and it is expected to fulfill the IP requirements for several decades to come.

IPv6 uses a messaging protocol called *Neighbor Discovery Protocol* (NDP) to probe the network to discover neighboring IPv6 devices, determine their reachability, and map their associations. This protocol also includes enhanced functionalities (provided by ICMP and ARP on IPv4 networks) for troubleshooting issues pertaining to connectivity, address duplication, and routing.

Unlike IPv4 addresses, which are represented as four dot-separated octets, IPv6 addresses contain eight colon-separated groups of four hexadecimal numbers. A sample v6 IP would be

1204:bab1:21d1:bb43:23a1:9bde:87df:bac9. It looks a bit daunting at first sight, but there are methods that will simplify their representation.

The *ip addr* command also shows IPv6 addresses for the interfaces:

```
[user1@server10 ~]$ ip addr | grep inet6  
inet6 ::1/128 scope host  
inet6 fe80::a00:27ff:febff:558e/64 scope link
```

It returns two IPv6 addresses. The first one belongs to the loopback interface, and the second one is assigned to the `enp0s3` connection.

Major Differences between IPv4 and IPv6

There are a number of differences between IPv4 and IPv6 protocols. Some of the major ones are highlighted in [Table 16-1](#).

IPv4	IPv6
Uses 4x8-bit, period-separated decimal number format for address representation. Example: 192.168.0.100	Uses 8x16-bit, colon-separated hexadecimal number format for representation. Example: fe80::a00:27ff:feae:f35b
Number of address bits: 32	Number of address bits: 128
Maximum number of addresses: ~4.3 billion.	Maximum number of addresses virtually unlimited
Common testing and troubleshooting tools: ping, traceroute, tracepath, etc.	Common testing and troubleshooting tools: ping6, traceroute6, tracepath6, etc.
Support for IP autoconfiguration: no	Support for IP autoconfiguration: yes
Packet size: 576 bytes	Packet size: 1280 bytes

Table 16-1 IPv4 vs IPv6

These and other differences not listed here are due to enhancements and new features added to IPv6.

Network Devices and Connections

Network Interface Cards (NICs) are hardware adapters that provide one or more Ethernet ports for network connectivity. NICs may also be referred to as *network adapters* and individual ports as *network interfaces* or *network devices*. NICs may be built-in to the system board or are add-on adapters. They are available in one, two, and four port designs on a single adapter.

Individual interfaces (devices) can have one or more connection profiles attached to them with different configuration settings. Each connection profile has a unique name and includes settings such as the device name, hardware address, activating the connection on reboot, and so on. A connection profile can be configured by editing files or using commands. A device can have

multiple connection profiles attached, but only one of them can be active at a time.

Consistent Network Device Naming

In RHEL versions earlier than 7, network interfaces were named *eth* (Ethernet), *em* (embedded), and *wlan* (wireless lan), and were numbered 0 and onwards as the interfaces were discovered during a system boot. This was the default scheme that had been in place for network device naming for years. Given a large number of interfaces located onboard and on add-on NICs, the number assignments could possibly change on the next boot due to failures or errors in their detection, which will result in connectivity and operational issues.

As of RHEL 7, the default naming scheme has been augmented to base on several rules governed by the *udevd* service. The default ruleset is to assign names using the device's location and topology, and the setting in firmware. The underlying virtualization layer (VMware, VirtualBox, KVM) also plays a role in the naming. Some sample device names are *enp0s3*, *ens160*, etc.

This advanced ruleset has resulted in consistent and predictable naming, eliminating the odds of reenumeration during a hardware rescan. Moreover, the designated names are not affected by the addition or removal of interface cards. This naming scheme helps in identifying, configuring, troubleshooting, and replacing the right adapter without hassle.

Understanding Interface Connection Profile

Each network connection has a configuration file that defines IP assignments and other relevant parameters for it. The networking subsystem reads this file and applies the settings at the time the connection is activated. Connection configuration files (or *connection profiles*) are stored in a central location under the */etc/sysconfig/network-scripts* directory. The filenames begin with *ifcfg-* and are followed by the name of the connection. Some instances of connection filenames are *ifcfg-enp0s3*, *ifcfg-ens160*, and *ifcfg-em1*.

On *server10* and *server20*, the device name for the first interface is *enp0s3* with connection name *enp0s3* and relevant connection information stored in the *ifcfg-enp0s3* file. This connection was established at the time of installation. The current content of the *ifcfg-enp0s3* file from *server10* are presented below with IPv6 directives excluded:

```
[user1@server10 ~]$ cat /etc/sysconfig/network-scripts/ifcfg-enp0s3 | grep -vi ipv6
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
NAME=enp0s3
UUID=5ece88f2-2b9c-420f-8e1d-a66d648ed171
DEVICE=enp0s3
ONBOOT=yes
IPADDR=192.168.0.110
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
```

These directives and a few others that can be defined in this file are listed in alphabetical order in [Table 16-2](#).

Directive	Description
BOOTPROTO	Defines the boot protocol to be used. Common values include dhcp to obtain IP assignments from a DHCP server and none or static to use a IPADDR as set with the IPADDR directive.
BROWSER_ONLY	Works if PROXY_METHOD is set to auto. Ignored otherwise.
DEFROUTE	Whether to use this connection as the default gateway.
DEVICE	Specifies the device name for the network connection.
DNS1	Defines the IP address or the hostname of the primary DNS server. This address/hostname is placed in the /etc/resolv.conf file if the PEERDNS directive is set to yes in this file.
GATEWAY	Specifies the gateway address for the connection. Ignored if the BOOTPROTO directive is set to none or static.
HWADDR	Describes the hardware address for the device.
IPADDR	Specifies the static IP for the connection if the BOOTPROTO directive is set to none or static.
IPV4_FAILURE_FATAL	Whether to disable the device if IPv4 configuration fails. Default is no.
IPV6INIT	Whether to enable IPv6 support for this connection.
NAME	Any description given to this connection. This default matches the device name.
NETMASK	Sets the netmask address for the connection. Ignored if the BOOTPROTO directive is set to none or static.
NM_CONTROLLED	Whether the NetworkManager service is to be allowed to modify the configuration for this connection. It should be turned off on computers that use static IP addresses. Default is yes.
ONBOOT	Whether to auto-activate this connection at boot.
PEERDNS	Whether to modify the DNS client resolver /etc/resolv.conf. Default is yes if BOOTPROTO is set to dhcp.
PREFIX	Defines the number of subnet bits. This directive may be used in lieu of NETMASK.
PROXY_METHOD	Method to be used for proxy setting. Default is none.
UUID	The UUID associated with this connection.

TYPE	Specifies the type of this connection
------	---------------------------------------

Table 16-2 Network Connection Configuration Directives

There are numerous other directives, including those for IPv6, that may be defined in connection profiles. Run **man nm-settings** for a description of additional directives.

Exercise 16-2: Add Network Devices to server10 and server20

This exercise will add one network interface to *server10* and one to *server20* using VirtualBox.

In this exercise, you will shut down *server10* and *server20*. You will launch VirtualBox and add one network interface to *server10* and one to *server20* in preparation for Exercises 17-3 and 17-4.

1. Execute **sudo shutdown now** on *server10*.
2. Start VirtualBox on your Windows/Mac computer and highlight the *RHEL8-VM1* virtual machine that you created in [Exercise 1-2](#). Make sure it is powered off.
3. Click Settings at the top and then Network on the window that pops up. Click on “Adapter 2” and check the “Enable Network Adapter” box. Select “Internal Network” from the drop down list next to “Attached to”. See [Figure 16-1](#).

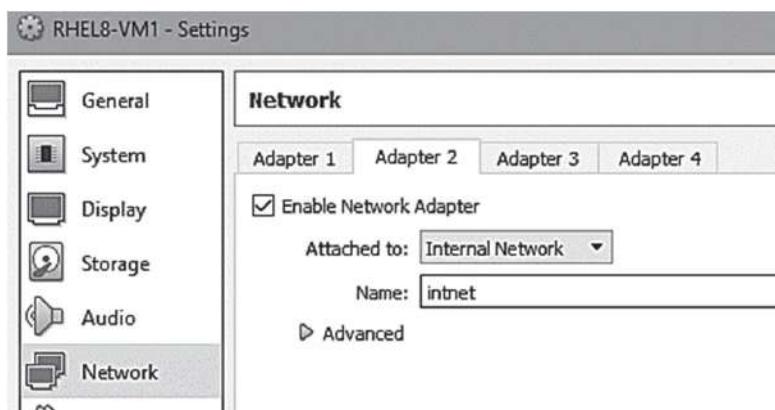


Figure 16-1 VirtualBox – Add Network Interface

4. Click OK to return to the main VirtualBox interface.
5. Power on *RHEL8-VM1* to boot RHEL 8 in it.

- When the server is up, log on as *user1* and run the *ip* command as follows to verify the new interface:

```
[user1@server10 ~] $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    qlen 1000
        link/ether 08:00:27:bf:55:8e brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.110/24 brd 192.168.0.255 scope global noprefixroute enp0s3
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:feb5:558e/64 scope link
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    qlen 1000
        link/ether 08:00:27:42:1e:1d brd ff:ff:ff:ff:ff:ff
```

The output reveals a new network device called *enp0s8*. This is the new interface you just added to the VM.

- Repeat steps 1 through 6 to add a network interface to *server20* and verify.

This completes the addition of new network interfaces to *server10* and *server20*. You are now ready to use them in the upcoming exercises.

Network Device and Connection Administration Tools

There are a few tools and methods available for configuring and administering network interfaces, connections, and connection profiles. The *NetworkManager* service includes a toolset for this purpose as well. Let's take a quick look at the basic management tools in [Table 16-3](#).

Command	Description
ip	A powerful and versatile tool for displaying, monitoring, managing interfaces, connections, routing, traffic, etc.
ifup	Activates a connection
ifdown	Deactivates a connection

Table 16-3 Basic Network Management Tools

You can manually create a connection profile and attach it to a network device. Many Linux administrators prefer this approach. RHEL also offers an alternative method for this purpose, which is discussed later in this chapter.

In [Exercise 16-3](#), you'll use the manual method to configure a connection profile for a new network device that was added in [Exercise 16-2](#) and employ the tools listed in [Table 16-3](#).

Exercise 16-3: Configure New Network Connection Manually

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will create a connection profile for the new network interface `enp0s8` using a text editing tool. You will assign the IP `172.10.10.110/24` with gateway `172.10.10.1` and set it to autoactivate at system reboots. You will deactivate and reactivate this interface at the command prompt.

1. Create a file called `ifcfg-enp0s8` in the `/etc/sysconfig/network-scripts` directory and enter the following information to establish a connection profile:

```
[user1@server10 ~]$ sudo cat /etc/sysconfig/network-scripts/ifcfg-enp0s8
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=no
NAME=enp0s8
DEVICE=enp0s8
ONBOOT=yes
IPADDR=172.10.10.110
PREFIX=24
GATEWAY=172.10.10.1
```

2. Deactivate and reactivate this interface using the `ifdown` and `ifup` commands:

```
[user1@server10 ~]$ sudo ifdown enp0s8
Connection 'enp0s8' successfully deactivated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
[user1@server10 ~]$
[user1@server10 ~]$ sudo ifup enp0s8
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/3)
```

3. Verify the activation of `enp0s8` connection:

```
[user1@server10 ~]$ ip a
.....
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:42:1e:1d brd ff:ff:ff:ff:ff:ff
    inet 172.10.10.110/24 brd 172.10.10.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe42:1e1d/64 scope link
        valid_lft forever preferred_lft forever
```

The new connection profile called `enp0s8` has been applied to the new network device called `enp0s8`. The connection is now ready for use. The connectivity to `server10` over this new connection is tested later in this chapter.

The NetworkManager Service

NetworkManager is the default interface and connection configuration, administration, and monitoring service used in RHEL 8. This service has a daemon program called *NetworkManager*, which is responsible for keeping available interfaces and connections up and active. It offers a powerful command line tool called *nmcli* to manage interfaces and connections, and to control the service. This utility offers many options for their effective management. The *NetworkManager* service also furnishes a text-based interface called *nmtui* and a graphical equivalent called *nm-connection-editor* that you may use in lieu of *nmcli*.

The nmcli Command

nmcli is a NetworkManager command line tool that is employed to create, view, modify, remove, activate, and deactivate network connections, and to control and report network device status. It operates on seven different object categories, with each category supporting several options to form a complete command. The seven categories are general, networking, connection, device, radio, monitor, and agent. This discussion only focuses on the connection and device object categories. They are described in [Table 16-4](#) along with management operations that they can perform.

Object	Description
Connection: activates, deactivates, and administers network connections	
show	Lists connections
up / down	Activates/deactivates a connection
add	Adds a connection
edit	Edits an existing connection or adds a new one
modify	Modifies one or more properties of a connection
delete	Deletes a connection
reload	Instructs NetworkManager to re-read all connection profiles
load	Instructs NetworkManager to re-read a connection profile
Device: displays and administers network interfaces	
status	Exhibits device status
show	Displays detailed information about all or the specified interface

Table 16-4 Network Connection and Device Administration Tools

Object categories and the objects within them may be written in an abridged form to save typing. For instance, the connection category may be abbreviated as a “c” or “con” and the device category as a “d” or “dev”. The same rule applies to object names as well. For instance, add may be specified as an “a”, delete as a “d”, and so on. Check the manual pages for *nmcli*-examples.

The *nmcli* command supports tab completion to make its use easier. Let’s run a few examples on *server10* to understand the command’s usage.

To show (s) all available connections (c) including both active and inactive:

```
[user1@server10 ~] $ nmcli c s
NAME      UUID              TYPE      DEVICE
enp0s3   5ece88f2-2b9c-420f-8e1d-a66d648ed171  ethernet  enp0s3
enp0s8   00cb8299-feb9-55b6-a378-3fdc720e0bc6  ethernet  enp0s8
```

The output lists two connection profiles (NAME) and the devices (DEVICE) they are attached to. It also shows their UUID and type.

To deactivate (down) the connection (c) *enp0s8*:

```
[user1@server10 ~]$ sudo nmcli c down enp0s8
Connection 'enp0s8' successfully deactivated (D-Bus active path: /org/freedesktop/
p/NetworkManager/ActiveConnection/3)
```

The connection profile is detached from the device, disabling the connection. You can check with **nmcli c s**.

To activate (up) the connection (c) enp0s8:

```
[user1@server10 ~]$ sudo nmcli c up enp0s8
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkMa
nager/ActiveConnection/4)
```

The connection profile is reattached to the device, enabling the connection. You can check with **nmcli c s**.

To display the status (s) of all available network devices (d):

```
[user1@server10 ~]$ nmcli d s
DEVICE  TYPE      STATE   CONNECTION
enp0s3  ethernet  connected enp0s3
enp0s8  ethernet  connected enp0s8
lo      loopback  unmanaged --
```

The output shows three devices and their types, states, and the connection profiles attached to them. The loopback interface is not managed by the NetworkManager service.

Exercise 16-4: Configure New Network Connection Using nmcli

This exercise should be done on `server20` as `user1` with `sudo` where required.

In this exercise, you will create a connection profile using the `nmcli` command for the new network interface `enp0s8` that was added to `server20` in [Exercise 16-2](#). You will assign the IP `172.10.10.120/24` with gateway `172.10.10.1`, and set it to autoactivate at system reboots. You will deactivate and reactivate this interface at the command prompt.

1. Check the running status of the `NetworkManager` service:

```
[user1@server20 ~]$ sudo systemctl status NetworkManager
● NetworkManager.service - Network Manager
  Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; ven|
    Active: active (running) since Sat 2019-11-16 23:31:47 EST; 13h ago
      Docs: man:NetworkManager(8)
   Main PID: 1183 (NetworkManager)
     Tasks: 3 (limit: 11516)
    Memory: 11.3M
       CPU: 0.100 us
      CPU: 0.100 us
```

The service is up and active on the server.

2. Check the presence of the new interface:

```
[user1@server20 ~]$ sudo nmcli d status | grep enp
enp0s3      ethernet connected    enp0s3
enp0s8      ethernet disconnected --
```

The output signifies the presence of a new network device called **enp0s8**. It does not have a connection profile attached to it.

3. Add (add) a connection profile (con) and attach it to the new interface. Use the type Ethernet, device name (ifname) **enp0s8** with a matching connection name (con-name), CIDR (ip4) 172.10.10.120/24, and gateway (gw4) 172.10.10.1:

```
[user1@server20 ~]$ sudo nmcli con add type Ethernet ifname enp0s8 con-name enp0s8
ip4 172.10.10.120/24 gw4 172.10.10.1
Connection 'enp0s8' (c9f81ca1-4418-4a8d-9f9a-b6e4368bba46) successfully added.
```

A new connection has been added, attached to the new interface, and activated. In addition, the command has saved the connection information in a new file called *ifcfg-enp0s8* and stored it in the */etc/sysconfig/network-scripts* directory.

4. Confirm the new connection status:

```
[user1@server20 ~]$ nmcli c s
          NAME      UUID                                  TYPE      DEVICE
        enp0s3  7426fe5c-9bd4-42ad-bb95-3d15491082b9  ethernet  enp0s3
        enp0s8  c9f81ca1-4418-4a8d-9f9a-b6e4368bba46  ethernet  enp0s8
```

The output indicates the association of the new connection with the network device.

5. Check the content of the *ifcfg-enp0s8* connection profile:

```
[user1@server20 ~]$ cat /etc/sysconfig/network-scripts/ifcfg-enp0s8
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=none
IPADDR=172.10.10.120
PREFIX=24
GATEWAY=172.10.10.1
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s8
UUID=c9f81ca1-4418-4a8d-9f9a-b6e4368bba46
DEVICE=enp0s8
ONBOOT=yes
```

There are a number of default directives added to the connection profile in addition to the configuration items you entered with the *nmcli* command above. The ONBOOT directive is also set to yes automatically. This setting is an indicative of the fact that the connection will be auto-enabled at system reboots.

6. Check the IP assignments for the new connection:

```
[user1@server20 ~]$ ip a
.....
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
    default qlen 1000
        link/ether 08:00:27:a1:15:b4 brd ff:ff:ff:ff:ff:ff
        inet 172.10.10.120/24 brd 172.10.10.255 scope global noprefixroute enp0s8
            valid_lft forever preferred_lft forever
        inet6 fe80::9d:bc7a:b38f:5bb3/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

The IP is assigned to the interface. The connection is tested later in this chapter.

7. Deactivate this connection to detach it from the interface:

```
[user1@server20 ~]$ sudo nmcli c down enp0s8
Connection 'enp0s8' successfully deactivated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/4)
```

The connection profile is now detached from the interface, deactivating the connection. You can check with **nmcli c s**.

8. Reactivate the connection to attach it to the interface:

```
[user1@server20 ~]$ sudo nmcli c up enp0s8
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/5)
```

The connection profile is now reattached to the interface, activating the connection. You can check with `nmcli c s`.

This brings the exercise to a conclusion.

EXAM TIP: You need to know only one of the available methods to set IP assignments on the system.

Understanding Hosts Table

Each IP address used on the system should have a hostname assigned to it. In an environment with multiple systems on the network, it is prudent to have some sort of a hostname to IP address resolution method in place to avoid typing the destination system IP repeatedly to access it. DNS is one such method. It is designed for large networks such as corporate networks and the Internet. For small, internal networks, the use of a local hosts table (the `/etc/hosts` file) is also common. This table is used to maintain hostname to IP mapping for systems on the local network, allowing us to access a system by simply employing its hostname. In this book, there are two systems in place: `server10.example.com` with IP 192.168.0.110 and alias `server10`, and `server20.example.com` with IP 192.168.0.120 and alias `server20`. You can append this information to the `/etc/hosts` file on both `server10` and `server20` as shown below:

192.168.0.110	<code>server10.example.com</code>	server10
192.168.0.120	<code>server20.example.com</code>	server20

Each row in the file contains an IP address in column 1 followed by the *official* (or *canonical*) hostname in column 2, and one or more optional aliases thereafter. The official hostname and one or more aliases give users the flexibility of accessing the system using any of these names.

EXAM TIP: In the presence of an active DNS with all hostnames resolvable, there is no need to worry about updating the hosts file.

As expressed above, the use of the `hosts` file is common on small networks and it should be updated on each individual system to reflect any changes for best inter-system connectivity experience.

Testing Network Connectivity

RHEL includes the `ping` command to examine network connectivity between two systems. It uses the IP address of the destination system to send a series of 64-byte *Internet Control Message Protocol* (ICMP) test packets to it. A

response from the remote system validates connectivity and health. With the `-c` option, you can specify the number of packets that you want transmitted.

The following sends two packets from `server10` to 192.168.0.120 (IP for `server20`):

```
[user1@server10 ~]$ ping 192.168.0.120 -c 2
PING 192.168.0.120 (192.168.0.120) 56(84) bytes of data.
64 bytes from 192.168.0.120: icmp_seq=1 ttl=64 time=0.406 ms
64 bytes from 192.168.0.120: icmp_seq=2 ttl=64 time=0.418 ms

--- 192.168.0.120 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 38ms
rtt min/avg/max/mdev = 0.406/0.412/0.418/0.006 ms
```

Under “192.168.0.120 ping statistics,” the output depicts the number of packets transmitted, received, and lost. The packet loss should be 0%, and the round trip time should not be too high for a healthy connection. In general, you can use this command to test connectivity with the system’s own IP, the loopback IP (127.0.0.1), a static route, the default gateway, and any other address on the local or remote network.

If a *ping* response fails, you need to check if the NIC is seated properly, its driver is installed, network cable is secured appropriately, IP and netmask values are set correctly, and the default or static route is accurate.

Exercise 16-5: Update Hosts Table and Test Connectivity

This exercise should be done on `server10` and `server20` as `user1` with `sudo` where required.

In this exercise, you will update the `/etc/hosts` file on both `server10` and `server20`. You will add the IP addresses assigned to both connections and map them to hostnames `server10`, `server10s8`, `server20`, and `server20s8` appropriately. You will test connectivity from `server10` to `server20` and from `server10s8` to `server20s8` using their IP addresses and then their hostnames.

On `server20`:

1. Open the `/etc/hosts` file and add the following entries:

```
192.168.0.110 server10.example.com server10
192.168.0.120 server20.example.com server20
172.10.10.110 server10s8.example.com server10s8
172.10.10.120 server20s8.example.com server20s8
```

The IP addresses for both connections are added for both servers.

On server10:

2. Open the /etc/hosts file and add the following entries:

```
192.168.0.110 server10.example.com server10
192.168.0.120 server20.example.com server20
172.10.10.110 server10s8.example.com server10s8
172.10.10.120 server20s8.example.com server20s8
```

The IP addresses for both connections are added for both servers.

3. Send two packets from server10 to the IP address of server20:

```
[user1@server10 ~]$ ping -c2 192.168.0.120
PING 192.168.0.120 (192.168.0.120) 56(84) bytes of data.
64 bytes from 192.168.0.120: icmp_seq=1 ttl=64 time=0.369 ms
64 bytes from 192.168.0.120: icmp_seq=2 ttl=64 time=0.446 ms

--- 192.168.0.120 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 45ms
rtt min/avg/max/mdev = 0.369/0.407/0.446/0.043 ms
```

4. Issue two ping packets on server10 to the hostname of server20:

```
[user1@server10 ~]$ ping -c2 server20
PING server20.example.com (192.168.0.120) 56(84) bytes of data.
64 bytes from server20.example.com (192.168.0.120): icmp_seq=1 ttl=64 time=0.351 ms
64 bytes from server20.example.com (192.168.0.120): icmp_seq=2 ttl=64 time=0.422 ms

--- server20.example.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 26ms
rtt min/avg/max/mdev = 0.351/0.386/0.422/0.040 ms
```

5. Send one packet from server10 to the IP address of server20s8:

```
[user1@server10 ~]$ ping -c1 172.10.10.120
PING 172.10.10.120 (172.10.10.120) 56(84) bytes of data.
64 bytes from 172.10.10.120: icmp_seq=1 ttl=64 time=0.430 ms

--- 172.10.10.120 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.430/0.430/0.430/0.000 ms
```

6. Issue one ping packet on server10 to the hostname of server20s8:

```
[user1@server10 ~]$ ping -c1 server20s8
PING server20s8.example.com (172.10.10.120) 56(84) bytes of data.
64 bytes from server20s8.example.com (172.10.10.120): icmp_seq=1 ttl=64 time=0.341 ms

--- server20s8.example.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.341/0.341/0.341/0.000 ms
```

Steps 3 through 6 verified the connectivity to the remote server over both connections. Each server has two IP addresses, and each IP address has a unique hostname assigned to it.

This concludes the exercise.

Chapter Summary

This chapter discussed the rudiments of networking. It began by providing an understanding of various essential networking terms and concepts to build the foundation for networking topics going forward. Topics such as hostname, IPv4, IPv6, network classes, subnetting, subnet mask, CIDR notation, protocol, port, Ethernet address, and consistent device naming were covered in sufficient detail.

We modified hostnames on both lab servers by modifying a configuration file and restarting the hostname service on one server and using a single command on the other. We employed two different methods to demonstrate multiple ways of doing the same thing. A third method was also mentioned to rename the hostname.

Next, we described the terms network devices and network connections, and realized the difference between the two. We examined a connection profile and looked at a number of directives that may be defined for a network connection. We added a new network device to each lab server and configured them by employing two different methods. We activated the new connections and performed a ping test for functional validation. Lastly, we populated the hosts table with the IP and hostname mapping on both lab servers.

Review Questions

1. What is the use of *ifup* and *ifdown* commands?
2. Which service is responsible for maintaining consistent device naming?
3. List three key differences between TCP and UDP protocols.
4. What is the significance of the NAME and DEVICE directives in a connection profile?
5. Which class of IP addresses has the least number of node addresses?
6. Which command can you use to display the hardware address of a network device?
7. Define protocol.
8. Which directory stores the network connection profiles?
9. True or False. A network device is a physical or virtual network port and a network connection is a configuration file attached to it.
10. IPv4 is a 32-bit software address. How many bits does an IPv6 address have?
11. Which file defines the port and protocol mapping?
12. What would the command *hostnamectl set-hostname host20* do?
13. Name the file that stores the hostname of the system.
14. What would the command *nmcli cs* do?

15. What is the purpose of the ONBOOT directive in the network connection profile?
16. The `/etc/hosts` file maintains hostname to hardware address mappings. True or False?
17. Which file contains service, port, and protocol mappings?
18. What would the `ip addr` command produce?
19. Which file would you consult to identify the port number and protocol associated with a network service?
20. Adding a connection profile with the `nmcli` command creates a connection profile in the `/etc/sysconfig/network-scripts` directory. True or False?
21. Name four commands that can be used to display the system hostname?
22. List any two benefits of subnetting.

Answers to Review Questions

1. The `ifup` and `ifdown` commands are used to enable and disable a network connection, respectively.
2. The `udevd` service handles consistent naming of network devices.
3. TCP is connection-oriented, reliable, and point-to-point; UDP is connectionless, unreliable, and multi-point.
4. The NAME directive sets the name for the network connection and the DEVICE directive defines the network device the connection is associated with.
5. The C class supports the least number of node addresses.
6. The `ip` command.
7. A set of rules that govern the exchange of information between two network entities.
8. The `/etc/sysconfig/network-scripts` directory.
9. True.
10. 128.
11. The `/etc/protocols` file.
12. The command provided will update the `/etc/hostname` file with the specified hostname and restart the `systemd-hostnamed` daemon for the change to take effect.
13. The `/etc/hostname` file.
14. The command provided will display the status information for all network connections.
15. The purpose of the ONBOOT directive is to direct the boot scripts whether to activate this connection.
16. False. This file maintains hostname to IP address mapping.

17. The `/etc/services` file.
18. This command provided will display information about network connections including IP assignments and hardware address.
19. The `/etc/services` file.
20. True.
21. The `hostname`, `uname`, `hostnamectl`, and `nmcli` commands can be used to view the system hostname.
22. Better manageability and less traffic.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 16-1: Add New Interface and Configure Connection Profile with nmcli

Add a new network interface to *RHEL8-VM1* in VirtualBox.

As *user1* with *sudo* on *server10*, run `ip a` and verify the presence of the new interface (`enp0s8`). Use the `nmcli` command and assign IP 192.168.0.210/24 and gateway 192.168.0.1. Set the network connection to auto-activate on system reboots. Deactivate and reactivate this connection manually. (Hint: Network Devices and Connections).

Lab 16-2: Add New Interface and Configure Connection Profile Manually

Add a new network interface to *RHEL8-VM2* in VirtualBox.

As *user1* with *sudo* on *server20*, run `ip a` and verify the presence of the new interface (`enp0s8`). Make a copy of the `ifcfg-enp0s3` file as `ifcfg-enp0s8` under the `/etc/sysconfig/network-scripts` directory. Remove the `HWADDR` and `UUID` directives, and set the values for `IPADDR`, `NETMASK`, and `GATEWAY` directives appropriately. Set the network connection to auto-activate on system reboots. Deactivate and reactivate this connection manually. (Hint: Network Devices and Connections).

Chapter 17

Network File System

This chapter describes the following major topics:

- Overview of Network File System service and key components
- Network File System benefits and versions
- Export a share on NFS server
- Mount the share on NFS client using standard mount method
- Understand the AutoFS service, and its benefits and functioning
- Analyze AutoFS configuration maps
- Mount the exported share on NFS client using AutoFS
- Configure NFS and AutoFS to share and mount user home directories

RHCSA Objectives:

34. Mount and unmount network file systems using NFS

Network

shares may be mounted on RHEL and accessed the same way as local file systems. This can be done manually using the same tools that are employed for mounting and unmounting local file systems. An alternative solution is to implement the AutoFS service to automatically mount and unmount them without the need to execute any commands explicitly. AutoFS monitors activities in mount points based on which it triggers a mount or unmount action.

RHEL exports shares using the Network File System service for remote mounting on clients. The combination of NFS and AutoFS/standard mount is prevalent in real world scenarios. This chapter elaborates on the benefits of the file sharing solution and expounds upon AutoFS maps. It demonstrates a series of exercises to detail the configuration of the NFS server, NFS client, and AutoFS service. It also explores the automatic mounting of user home directories on clients.

Network File System

Network File System (NFS) is a networking protocol that allows file sharing over the network. The Network File System service is based upon the client/server architecture whereby users on one system access files, directories, and file systems (collectively called “shares”) that reside on a remote system as if they are mounted locally on their system. The remote system that makes its shares available for network access is referred to as an NFS server, and the process of making the shares accessible is referred to as *exporting*. The shares the NFS server exports may be accessed by one or more systems. These systems are called NFS clients, and the process of making the shares accessible on clients is referred to as *mounting*. See [Figure 17-1](#) for a simple NFS client/server arrangement that shows two shares `/export1` and `/export2`—exported on the network to a remote system, which has them mounted there.

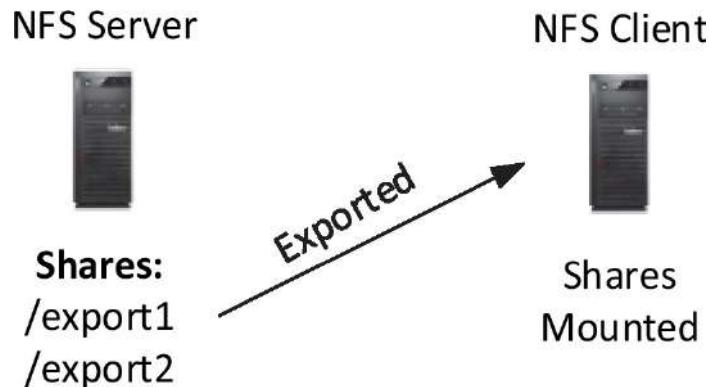


Figure 17-1 NFS Server/Client

A system can provide both server and client functionality concurrently. When a directory or file system share is exported, the entire directory structure beneath it becomes available for mounting on the client. A subdirectory or the parent directory of a share cannot be re-exported if it exists in the same file system. Similarly, a mounted share cannot be exported further. A single exported file share is mounted on a directory mount point.

Benefits of Using NFS

The use of NFS provides several benefits, some of which are highlighted below:

- ✓ Supports a variety of operating system platforms including Linux, UNIX, and Microsoft Windows.
- ✓ Multiple NFS clients can access a single share simultaneously.
- ✓ Enables the sharing of common application binaries and other read-only information, resulting in reduced administration overhead and storage cost.
- ✓ Gives users access to uniform data.
- ✓ Allows the consolidation of scattered user home directories on the NFS server and then exporting them to the clients. This way users will have only one home directory to maintain.

NFS Versions

RHEL 8 provides the support for NFS versions 3, 4.0, 4.1, and 4.2, with version 4.2 being the default. NFSv3 supports both TCP and UDP transport protocols, asynchronous writes, and 64-bit file sizes that gives clients the ability to access files of sizes larger than 2GB.

NFSv4.x are *Internet Engineering Task Force* (IETF) series of protocols that provide all the features of NFSv3, plus the ability to transit firewalls and work

on the Internet. They provide enhanced security and support for encrypted transfers and ACLs, as well as greater scalability, better cross-platform interoperability, and better system crash handling. They use usernames and group names rather than UIDs and GIDs for files located on network shares. NFSv4.0 and NFSv4.1 use the TCP protocol by default, but can work with UDP for backward compatibility. In contrast, NFSv4.2 only supports TCP.

NFS Server and Client Configuration

This section presents two exercises, one demonstrating how to export a share on a server (NFS server) and the other outlines the steps on mounting and accessing a share on a remote system (NFS client). The basic setup of the NFS service is straightforward. It requires adding an entry of the share to a file called `/etc/exports` and using a command called `exportfs` to make it available on the network. It also requires the addition of a firewall rule to allow access to the share by NFS clients.

The `mount` command employed on an NFS client is the same command that was used in [Chapter 15](#) to mount local file systems. Moreover, the `fstab` file requires an entry for the NFS share on the client for persistent mounting.

The exercises in this section illustrate the usage of both commands and the syntax of both files.

Exercise 17-1: Export Share on NFS Server

This exercise should be done on `server20` as `user1` with `sudo` where required.

In this exercise, you will create a directory called `/common` and export it to `server10` in read/write mode. You will ensure that NFS traffic is allowed through the firewall. You will confirm the export.

1. Install the NFS software called `nfs-utils`:

```
[user1@server2 ~]$ sudo dnf -y install nfs-utils
Package nfs-utils-1:2.3.3-14.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

2. Create `/common` directory to be exported as a share:

```
[user1@server20 ~]$ sudo mkdir /common
```

3. Add full permissions to `/common`:

```
[user1@server20 ~]$ sudo chmod 777 /common
```

4. Add the NFS service persistently to the firewalld configuration to allow the NFS traffic to pass through, and load the new rule:

```
[user1@server20 ~]$ sudo firewall-cmd --permanent --add-service nfs  
success  
[user1@server20 ~]$ sudo firewall-cmd --reload  
success
```



Refer to [Chapter 20](#) for details around the firewall service.

5. Start the NFS service and enable it to autostart at system reboots:

```
[user1@server20 ~]$ sudo systemctl --now enable nfs-server  
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service  
→ /usr/lib/systemd/system/nfs-server.service.
```

6. Verify the operational status of the NFS services:

```
[user1@server20 ~]$ sudo systemctl status nfs-server  
● nfs-server.service - NFS server and services  
  Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; vendor  
  Drop-In: /run/systemd/generator/nfs-server.service.d  
    └─order-with-mounts.conf  
    Active: active (exited) since Wed 2020-10-07 09:53:29 EDT; 1min 26s ago  
      Process: 4079 ExecStart=/bin/sh -c if systemctl -q is-active gssproxy; then >  
      Process: 4066 ExecStart=/usr/sbin/rpc.nfsd (code=exited, status=0/SUCCESS)  
      Process: 4065 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCE  
    Main PID: 4079 (code=exited, status=0/SUCCESS)  
  
Oct 07 09:53:29 server20.example.com systemd[1]: Starting NFS server and servi>  
Oct 07 09:53:29 server20.example.com systemd[1]: Started NFS server and servi>
```

7. Open the `/etc/exports` file in a text editor and add an entry for `/common` to export it to `server10` with read/write option:

```
/common server10(rw)
```

8. Export the entry defined in the `/etc/exports` file. The `-a` option exports all the entries listed in the file and `-v` displays verbose output.

```
[user1@server20 ~]$ sudo exportfs -av  
exporting server10.example.com:/common
```

The NFS service is now set up on `server20` with the `/common` share available for mounting on `server10` (NFS client in this case).

For practice, you can unexport the share by issuing the `exportfs` command with the `-u` flag as follows:

```
[user1@server20 ~]$ sudo exportfs -u server10:/common
```

Before proceeding, re-export the share using **`sudo exportfs -av`**.

Exercise 17-2: Mount Share on NFS Client

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will mount the `/common` share exported in [Exercise 17-1](#). You will create a mount point called `/local`, mount the remote share manually, and confirm the mount. You will add the remote share to the file system table for persistence. You will remount the share and confirm the mount. You will create a test file in the mount point and confirm the file creation on the NFS server (`server20`).

1. Install the NFS software called `nfs-utils`:

```
[user1@server2 ~]$ sudo dnf -y install nfs-utils
Package nfs-utils-1:2.3.3-14.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

2. Create `/local` mount point:

```
[user1@server10 ~]$ sudo mkdir /local
```

3. Mount the share manually using the `mount` command:

```
[user1@server10 ~]$ sudo mount server20:/common /local
```

The remote share is successfully mounted on `server10`, and it can be accessed as any other local file system.

4. Confirm the mount using either the `mount` or the `df` command:

```
[user1@server10 ~]$ mount | grep local
server20:/common on /local type nfs4 (rw,relatime,vers=4.2,rsize=262144,wszie=2
62144,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.168.
0.110,local_lock=none,addr=192.168.0.120)
[user1@server10 ~]$
[user1@server10 ~]$ df -h | grep local
server20:/common      8.0G  4.1G  4.0G  51% /local
```

The `mount` command output returns the NFS protocol version (NFS4.2) and all the default options used in mounting the share.

5. Open the `/etc/fstab` file and append the following entry for persistence:

```
server20:/common /local nfs _netdev 0 0
```



The `_netdev` option will make the system wait for networking to establish before attempting to mount this share.



A mount point should be empty and must not be in use when an attempt is made to mount a share on it.

6. Unmount the share manually using the *umount* command and remount it via the *fstab* file to validate the accuracy of the entry placed in the file:

```
[user1@server10 ~]$ sudo umount /local  
[user1@server10 ~]$ sudo mount -a
```

7. Run **mount** and **df -h** again to verify the remounting.
8. Create a file called *nfsfile* under */local* and verify:

```
[user1@server10 ~]$ touch /local/nfsfile  
[user1@server10 ~]$ ls -l /local  
total 0  
-rw-rw-r--. 1 user1 user1 0 Oct  7 10:50 nfsfile
```

9. Confirm the file creation on the NFS server (*server2*):

```
[user1@server20 ~]$ ls -l /common/  
total 0  
-rw-rw-r--. 1 user1 user1 0 Oct  7 10:50 nfsfile
```

EXAM TIP: Do not forget to update the */etc/fstab* file on the client.

This completes the setup and testing of mounting, unmounting, and remounting of an NFS share on the client.

Auto File System (AutoFS)

In the previous section, you learned how to attach (mount) an NFS share to the Linux directory tree manually for access by users and applications on an NFS client. Once attached, the share was treated just like any other local file system. You also learned how to detach (unmount) an NFS share manually from the directory tree to make it inaccessible to users and applications. You placed an entry for the share in the *fstab* file to guarantee a remount during system reboots.

RHEL offers an alternative way of mounting and unmounting a share on the clients during runtime as well as system reboots. This feature is delivered by a service called the *Auto File System* (AutoFS). AutoFS is a client-side service, which is employed to mount an NFS share on-demand. With a proper entry placed in AutoFS configuration files, the AutoFS service automatically mounts a share upon detecting an activity in its mount point with a command such as *ls* or *cd*. In the same manner, AutoFS unmounts the share automatically if it has not been accessed for a predefined period of time.



To avoid inconsistencies, mounts managed with AutoFS should not be mounted or unmounted manually or via the `/etc/fstab` file.

The use of AutoFS saves the kernel from dedicating system resources to maintain unused NFS shares, resulting in slight improvement in system performance.

Benefits of Using AutoFS

There are several benefits associated with using the AutoFS service over placing entries in the `/etc/fstab` file. Some of the key benefits are described below:

- ✓ AutoFS requires that NFS shares be defined in text configuration files called *maps*, which are located in the `/etc` or `/etc/auto.master.d` directory. AutoFS does not make use of the `/etc/fstab` file.
- ✓ AutoFS does not require *root* privileges to mount an NFS share; manual mounting and mounting via `fstab` do require that privilege.
- ✓ AutoFS prevents an NFS client from hanging if an NFS server is down or inaccessible. With the other method, the unavailability of the NFS server may cause the NFS client to hang.
- ✓ With AutoFS, a share is unmounted automatically if it is not accessed for five minutes by default. With the `fstab` method, the share stays mounted until it is either manually unmounted or the client shuts down.
- ✓ AutoFS supports wildcard characters and environment variables, which the other method does not support.

How AutoFS Works

The AutoFS service consists of a daemon called *automount* in the userland that mounts configured shares automatically upon access. This daemon is invoked at system boot. It reads the AutoFS master map and creates initial mount point entries, but it does not mount any shares yet. When the service detects a user activity under a mount point during runtime, it mounts the requested file system at that time. If a share remains idle for a certain time period, *automount* unmounts it by itself.

AutoFS Configuration File

The configuration file for the AutoFS service is `/etc/autofs.conf`, which AutoFS consults at service startup. Some key directives from this file are shown below along with preset values:

```
master_map_name = auto.master
```

```
timeout = 300
negative_timeout = 60
mount_nfs_default_protocol = 4
logging = none
```

There are additional directives available in this file and more can be added to modify the default behavior of the AutoFS service. [Table 17-1](#) describes the above directives.

Directive	Description
master_map_name	Defines the name of the master map. The default is auto.master located in the /etc directory.
timeout	Specifies, in seconds, the maximum idle time after which a share is automatically unmounted. The default is five minutes.
negative_timeout	Expresses, in seconds, a timeout value for failed mount attempts. The default is one minute.
mount_nfs_default_protocol	Sets the default NFS version to be used for mount shares.
logging	Configures a logging level. Options are verbose, warning, and debug. The default is none (disabled).

Table 17-1 AutoFS Directives

The directives in the *autofs.conf* file are normally left to their default values, but you can alter them if required.

AutoFS Maps

The AutoFS service needs to know the NFS shares to be mounted and their locations. It also needs to know any specific options to use with mounting them. This information is defined in AutoFS files called *maps*. There are three common AutoFS map types: *master*, *direct*, and *indirect*.

The Master Map

The *auto.master* file located in the */etc* directory is the default master map, as defined in the */etc/autofs.conf* configuration file with the *master_map_name* directive. This map may be used to define entries for indirect and direct maps. However, it is recommended to store user-defined map files in the */etc/auto.master.d* directory, which the AutoFS service automatically parses at

startup. The following presents two samples to explain the format of the map entries:

/-	/etc/auto.master.d/auto.direct
/misc	/etc/auto.misc

Line 1 defines a direct map and points to the `/etc/auto.master.d/auto.direct` file for mount details.

The second one is for an indirect map, notifying AutoFS to refer to the `/etc/auto.misc` file for mount details. The umbrella mount point `/misc` will precede all mount point entries listed in the `/etc/auto.misc` file. This indirect map entry is normally used to automount removable file systems, such as CD, DVD, external USB disks, and so on. Any custom indirect map file should be located in the `/etc/auto.master.d` directory.

You may append an option to either entry in the `auto.master` file; however, that option will apply globally to all subentries in the specified map file.

The Direct Map

The direct map is used to mount shares automatically on any number of unrelated mount points. Some key points to note when working with direct maps are:

- ✓ Direct mounted shares are always visible to users.
- ✓ Local and direct mounted shares can coexist under one parent directory.
- ✓ Accessing a directory containing many direct mount points mounts all shares.

Each direct map entry places a separate share entry to the `/etc/mtab` file, which maintains a list of all mounted file systems whether they are local or remote. This file is updated whenever a local file system, removable file system, or a network share is mounted or unmounted.

Exercise 17-3: Access NFS Share Using Direct Map

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will configure a direct map to automount the NFS share `/common` that is available from `server20`. You will install the relevant software, create a local mount point `/autodir`, and set up AutoFS maps to support the automatic mounting. Note that `/common` is already mounted on the `/local`

mount point on *server10* (NFS client) via the *fstab* file. There shouldn't be any conflict in configuration or functionality between the two.

1. Install the AutoFS software package called *autofs*:

```
[user1@server10 ~]$ sudo dnf install -y autofs  
.....  
Running transaction  
Preparing : 1/1  
Installing : autofs-1:5.1.4-29.el8.x86_64 1/1  
Running scriptlet: autofs-1:5.1.4-29.el8.x86_64 1/1  
Verifying : autofs-1:5.1.4-29.el8.x86_64 1/1  
2019-11-10 12:25:30,057 [WARNING] dnf:7057:MainThread @logutil.py:142 - logging already initialize  
Installed products updated.  
  
Installed:  
  autofs-1:5.1.4-29.el8.x86_64  
  
Complete!
```

2. Create a mount point */autodir* using the *mkdir* command:

```
[user1@server10 ~]$ sudo mkdir /autodir
```

3. Edit the */etc/auto.master* file and add the following entry at the beginning of the file. This entry will point the AutoFS service to the *auto.dir* file for additional information.

```
/ - /etc/auto.master.d/auto.dir
```

4. Create */etc/auto.master.d/auto.dir* file and add the mount point, NFS server, and share information to it:

```
/autodir server20:/common
```

5. Start the AutoFS service now and set it to autostart at system reboots:

```
[user1@server10 ~]$ sudo systemctl enable --now autofs  
Created symlink /etc/systemd/system/multi-user.target.wants/autofs.service → /u  
sr/lib/systemd/system/autofs.service.
```

6. Verify the operational status of the AutoFS service. Use the *-l* and *--no-pager* options to show full details without piping the output to a pager program (the *pg* command in this case).

```
[user1@server10 ~]$ sudo systemctl status autofs -l --no-pager
● autofs.service - Automounts filesystems on demand
    Loaded: loaded (/usr/lib/systemd/system/autofs.service; enabled; vendor pres
et: disabled)
      Active: active (running) since Wed 2020-10-07 11:01:59 EDT; 21s ago
        Main PID: 2843 (automount)
          Tasks: 6 (limit: 5076)
        Memory: 4.1M
       CGroup: /system.slice/autofs.service
               └─2843 /usr/sbin/automount --foreground --dont-check-daemon

Oct 07 11:01:59 server10.example.com systemd[1]: Starting Automounts filesystem
s on demand...
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(sss
): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(sss
): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(sss
): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(sss
): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com systemd[1]: Started Automounts filesystems
on demand.
```

- Run the `ls` command on the mount point `/autodir` and then issue the `mount` command to verify that the share is automounted and accessible:

```
[user1@server10 ~]$ ls /autodir
nfsfile
[user1@server10 ~]$
[user1@server10 ~]$ mount | grep autodir
/etc/auto.master.d/auto.dir on /autodir type autofs (rw,relatime,fd=5,pgrp=2843
,timeout=300,minproto=5,maxproto=5,direct,pipe_ino=50021)
server20:/common on /autodir type nfs4 (rw,relatime,vers=4.2,rsize=262144,wsize
=262144,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.16
8.0.110,local lock=none,addr=192.168.0.120)
```

Observe the above outcomes. The `mount` command output depicts the path of the AutoFS map (`/etc/auto.master.d/auto.dir`), the file system type (autofs), and the options used during the mount process. An activity in the mount point (the `ls` command) caused AutoFS to mount the share `/common` on `/autodir`. Wait five minutes and run the `mount` command again. You'll see that the auto file system has disappeared. A `cd`, `ls`, or some other activity in the mount point will bring it back.

This completes the AutoFS setup for an NFS share on the client using a direct map.

The Indirect Map

The indirect map is preferred over the direct map if you want to mount all of the shares under one common parent directory. Some key points to note when working with indirect maps are:

- ✓ Indirect mounted shares become visible only after they have been accessed.
- ✓ Local and indirect mounted shares cannot coexist under the same parent directory.
- ✓ Each indirect map puts only one entry in the `/etc/mtab` mount table.

- ✓ Accessing a directory containing many indirect mount points shows only the shares that are already mounted.

Both direct and indirect maps have their own merits and demerits. By comparing their features, it seems more prudent to use the indirect map for automounting NFS shares. However, this statement may not be true for every environment, as there could be specifics that would dictate which option to go with.

Exercise 17-4: Access NFS Share Using Indirect Map

This exercise should be done on *server10* as *user1* with *sudo* where required.

In this exercise, you will configure an indirect map to automount the NFS share */common* that is available from *server20*. You will install the relevant software and set up AutoFS maps to support the automatic mounting. You will observe that the specified mount point “autoindir” is created automatically under */misc*.

Note that */common* is already mounted on the */local* mount point via the *fstab* file and it is also configured via a direct map for automounting on */autodir*. There should occur no conflict in configuration or functionality among the three.

1. Install the AutoFS software package called *autofs*:

```
[user1@server10 ~]$ sudo dnf install -y autofs
.....
Package autofs-1:5.1.4-29.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

2. Confirm the entry for the indirect map */misc* in the */etc/auto.master* file exists:

```
[user1@server10 ~]$ grep ^/misc /etc/auto.master
/misc    /etc/auto.misc
```

3. Edit the */etc/auto.misc* file and add the mount point, NFS server, and share information to it:

```
autoindir  server20:/common
```

4. Start the AutoFS service now and set it to autostart at system reboots:

```
[user1@server10 ~]$ sudo systemctl enable --now autofs
Created symlink /etc/systemd/system/multi-user.target.wants/autofs.service → /u
sr/lib/systemd/system/autofs.service.
```

5. Verify the operational status of the AutoFS service. Use the `-l` and `--no-pager` options to show full details without piping the output to a pager program (the `pg` command in this case):

```
[user1@server10 ~]$ sudo systemctl status autofs -l --no-pager
* autofs.service - Automounts filesystems on demand
  Loaded: loaded (/usr/lib/systemd/system/autofs.service; enabled; vendor pres
  et: disabled)
  Active: active (running) since Wed 2020-10-07 11:01:59 EDT; 21s ago
    Main PID: 2843 (automount)
      Tasks: 6 (limit: 5076)
     Memory: 4.1M
        CGroup: /system.slice/autofs.service
                  └─2843 /usr/sbin/automount --foreground --dont-check-daemon

Oct 07 11:01:59 server10.example.com systemd[1]: Starting Automounts filesystems on demand...
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss
 ): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss
 ): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss
 ): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss
 ): setautomntent: No such file or directory
Oct 07 11:01:59 server10.example.com systemd[1]: Started Automounts filesystems on demand.
```

6. Run the `ls` command on the mount point `/misc/autoindir` and then `grep` for both `auto.misc` and `autoindir` on the `mount` command output to verify that the share is automounted and accessible:

```
[user1@server10 ~]$ ls /misc/autoindir
nfsfile
[user1@server10 ~]$
[user1@server10 ~]$ mount | egrep 'auto.misc|autoindir'
/etc/auto.misc on /misc type autofs (rw,relatime,fd=10,pgrp=2843,timeout=300,mi
nproto=5,maxproto=5,indirect,pipe_ino=50060)
server20:/common on /misc/autoindir type nfs4 (rw,relatime,vers=4.2,rsize=26214
4,wsize=262144,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr
=192.168.0.110,local lock=none,addr=192.168.0.120)
```

Observe the above outcomes. The `mount` command output illustrates the path of the AutoFS map (`/etc/auto.misc`), the auto-generated mount point (`/misc/autoindir`), file system type (autofs), and the options used during the mount process. An activity in the mount point (`ls` command in this case) caused AutoFS to mount the share `common` on `/misc/autoindir`. You can use the same umbrella mount point `/misc` to mount additional auto-generated mount points.

This mount point will automatically disappear after five minutes of idling. You can verify that by issuing the `mount` command again. A `cd`, `ls`, or some other activity in the mount point will bring it back.

This completes the AutoFS setup for an NFS share on the client using an indirect map.

Automounting User Home Directories

AutoFS allows us to automount user home directories by exploiting two special characters in indirect maps. The asterisk (*) replaces the references to specific mount points and the ampersand (&) substitutes the references to NFS servers and shared subdirectories. With user home directories located under `/home`, on one or more NFS servers, the AutoFS service will connect with all of them simultaneously when a user attempts to log on to a client. The service will mount only that specific user's home directory rather than the entire `/home`. The indirect map entry for this type of substitution is defined in an indirect map, such as `/etc/auto.master.d/auto.home`, and will look like:

```
* -rw &:/home/ &
```

With this entry in place, there is no need to update any AutoFS configuration files if NFS servers with `/home` shared are added or removed. Similarly, if user home directories are added or deleted, there will be no impact on the functionality of AutoFS. If there is only one NFS server sharing the home directories, you can simply specify its name in lieu of the first & symbol in the above entry.

Exercise 17-5: Automount User Home Directories Using Indirect Map

There are two portions for this exercise. The first portion should be done on `server20` (NFS server) and the second portion on `server10` (NFS client) as `user1` with `sudo` where required.

In the first portion, you will create a user account called `user30` with UID 3000. You will add the `/home` directory to the list of NFS shares so that it becomes available for remote mount.

In the second portion, you will create a user account called `user30` with UID 3000, base directory `/nfshome`, and no user home directory. You will create an umbrella mount point called `/nfshome` for mounting the user home directory from the NFS server. You will install the relevant software and establish an indirect map to automount the remote home directory of `user30` under `/nfshome`. You will observe that the home directory of `user30` is automounted under `/nfshome` when you sign in as `user30`.

On NFS server `server20`:

1. Create a user account called `user30` with UID 3000 (-u) and assign password “password1”:

```
[user1@server20 ~]$ sudo useradd -u 3000 user30
[user1@server20 ~]$ echo password1 | sudo passwd --stdin user30
Changing password for user user30.
passwd: all authentication tokens updated successfully.
```

2. Edit the `/etc/exports` file and add an entry for `/home` (do not remove the previous entry):

```
/home server10(rw)
```

3. Export all the shares listed in the `/etc/exports` file:

```
[user1@server20 ~]$ sudo exportfs -avr
exporting server10.example.com:/home
exporting server10.example.com:/common
```

On NFS client `server10`:

1. Install the AutoFS software package called `autoofs`:

```
[user1@server10 ~]$ sudo dnf install -y autoofs
.....
Package autoofs-1:5.1.4-29.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

2. Create a user account called `user30` with UID 3000 (-u), base home directory location `/nfshome` (-b), no home directory (-M), and password “`password1`”:

```
[user1@server10 ~]$ echo password1 | sudo passwd --stdin user30
Changing password for user user30.
passwd: all authentication tokens updated successfully.
```

This is to ensure that the UID for the user is consistent on the server and the client to avoid access issues.

3. Create the umbrella mount point `/nfshome` to automount the user’s home directory:

```
[user1@server10 ~]$ sudo mkdir /nfshome
```

4. Edit the `/etc/auto.master` file and add the mount point and indirect map location to it:

```
/nfshome  /etc/auto.master.d/auto.home
```

5. Create the `/etc/auto.master.d/auto.home` file and add the following information to it:

```
* -rw server20:/home/&
```

For multiple user setup, you can replace “user30” with the & character, but ensure that those users exist on both the server and the client with consistent UIDs.

6. Start the AutoFS service now and set it to autostart at system reboots. This step is not required if AutoFS is already running and enabled.

```
[user1@server10 ~]$ sudo systemctl enable --now autofs  
Created symlink /etc/systemd/system/multi-user.target.wants/autofs.service → /u  
sr/lib/systemd/system/autofs.service.
```

7. Verify the operational status of the AutoFS service. Use the `-l` and `--no-pager` options to show full details without piping the output to a pager program (the `pg` command):

```
[user1@server10 ~]$ sudo systemctl status autofs -l --no-pager  
● autofs.service - Automounts filesystems on demand  
  Loaded: loaded (/usr/lib/systemd/system/autofs.service; enabled; vendor pres  
  et: disabled)  
  Active: active (running) since Wed 2020-10-07 11:01:59 EDT; 21s ago  
    Main PID: 2843 (automount)  
      Tasks: 6 (limit: 5076)  
     Memory: 4.1M  
       CGroup: /system.slice/autofs.service  
           └─2843 /usr/sbin/automount --foreground --dont-check-daemon  
  
Oct 07 11:01:59 server10.example.com systemd[1]: Starting Automounts filesystem  
s on demand...  
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss  
): setautomntent: No such file or directory  
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss  
): setautomntent: No such file or directory  
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss  
): setautomntent: No such file or directory  
Oct 07 11:01:59 server10.example.com automount[2843]: setautomntent: lookup(ss  
): setautomntent: No such file or directory  
Oct 07 11:01:59 server10.example.com systemd[1]: Started Automounts filesystems  
on demand.
```

8. Log in as `user30` and run the `pwd`, `ls`, and `df` commands for verification:

```
[user1@server10 ~]$ su - user30  
Password:  
[user30@server10 ~]$  
[user30@server10 ~]$ pwd  
/nfshome/user30  
[user30@server10 ~]$ ls -l  
total 0  
[user30@server10 ~]$ df -h .  
Filesystem            Size  Used Avail Use% Mounted on  
server20:/home/user30  8.0G  4.1G  4.0G  51% /nfshome/user30
```

The user is successfully logged in with their home directory automounted from the NFS server. The `pwd` command confirms the path. The `df` command verifies the NFS server name and the source home directory path for `user30`,

as well as the mount location. You can also use the `mount` command and pipe the output to `grep` for `user30` to view mount details (`mount | grep user30`).

EXAM TIP: You may need to configure AutoFS for mounting a remote user home directory.

This completes the setup for an automounted home directory share for a user.

Chapter Summary

This chapter discussed the sharing and mounting of remote file systems using the Network File System protocol. It elucidated the concepts, benefits, and versions of the NFS service, and described the commands and configuration files involved in NFS management on the server and the client.

Next, we performed an exercise to demonstrate the configuration and sharing of a directory on one of the lab servers (NFS server) and another exercise on the second lab system (NFS client) to mount that share manually and persistently using the standard NFS mount method.

We explored the client-side service called AutoFS for automounting NFS shares. We discussed the concepts, benefits, and components associated with AutoFS, and analyzed its maps. We performed exercises to mount, confirm, and unmount the remote NFS share using both direct and indirect methods.

Finally, we described the AutoFS setting to automount user home directories from the NFS server.

Review Questions

1. What would the entry `* server10:/home/&` in an AutoFS indirect map imply?
2. Which command is used to export a share?
3. An NFS server exports a share and an NFS server mounts it. True or False?
4. Which command would you use to unexport a share?
5. What is the name of the NFS server configuration file?
6. What would the line entry `/dir1 *(rw)` in the `/etc/exports` file mean?
7. What type of AutoFS map would have the `/ - /etc/auto.media` entry in the `auto.master` file?
8. AutoFS requires `root` privileges to automatically mount a network file system. True or False?
9. What is the default timeout value for a file system before AutoFS unmounts it automatically?

10. Name the three common types of maps that AutoFS support.
11. Arrange the tasks in three different correct sequences to export a share using NFS: (a) update `/etc/exports`, (b) add service to firewalld, (c) run `exportfs`, (d) install `nfs-utils`, and (e) start nfs service.
12. What is the name of the AutoFS configuration file and where is it located?
13. The name of the AutoFS service daemon is `autofs`. True or False?

Answers to Review Questions

1. This indirect map entry would mount individual user home directories from `server10`.
2. The `exportfs` command.
3. True.
4. The `exportfs` command with the `-u` switch.
5. The `/etc/nfs.conf` file.
6. The line entry would export `/dir1` in read/write mode to all systems.
7. A direct map.
8. False.
9. Five minutes.
10. The three common AutoFS maps are master, direct, and indirect.
11. `d/e/b/a/c`, `d/e/a/c/b`, or `d/e/a/b/c`.
12. The name of the AutoFS configuration file is `autofs.conf` and it is located in the `/etc` directory.
13. False. The name of the AutoFS service daemon is `automount`.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 17-1: Configure NFS Share and Automount with Direct Map

As `user1` with `sudo` on `server10`, share directory `/share nfs` (create it) in read/write mode using NFS. On `server20` as `user1` with `sudo`, install the AutoFS software and start the service. Configure the master and a direct map to automount the share on `/mntauto` (create it). Run `ls` on `/mntauto` to trigger the mount. Use `df -h` to confirm. (Hint: NFS Server and Client Configuration, and Auto File System).

Lab 17-2: Automount NFS Share with Indirect Map

As *user1* with *sudo* on *server20*, configure the master and an indirect map to automount the share under */autoindir* (create it). Run *ls* on */autoindir/share nfs* to trigger the mount. Use **df -h** to confirm. (Hint: Auto File System).

Chapter 18

Time Synchronization and Hostname Resolution

This chapter describes the following major topics:

- Describe time synchronization and the role of Network Time Protocol
- Comprehend the terms: time source, NTP roles, and stratum levels
- Anatomy of the Chrony service configuration file
- Configure and verify NTP/Chrony client service
- View and set system date and time
- Overview of Domain Name System and hostname resolution
- Understand various DNS roles
- Analyze entries in resolver configuration files
- Perform name resolution using a variety of lookup tools

RHCSA Objectives:

43. Configure time service clients

48. Configure hostname resolution

The Chrony service, a new implementation of the Network Time Protocol, maintains the clock on the system and keeps it synchronized with a more accurate and reliable source of time. Providing accurate and uniform time for systems on the network allows time-sensitive applications such as monitoring software, backup tools, scheduling utilities, billing systems, file sharing protocols, and authentication programs to perform correctly and precisely. It also aids logging and auditing services to capture and record messages and alerts in log files with accurate timestamps.

Domain Name System is an OS- and hardware-independent network service employed for determining the IP address of a system when its hostname is known, and vice versa. This mechanism is implemented to map human-friendly hostnames to their assigned numeric IP addresses by consulting one or more servers offering the hostname resolution service. This service has been used on the Internet and corporate networks as the de facto standard for this purpose. DNS clients use this service to communicate with remote systems. There are several lookup programs that use DNS to obtain information.

Time Synchronization

Network Time Protocol (NTP) is a networking protocol for synchronizing the system clock with remote time servers for accuracy and reliability. This protocol has been in use with tens of millions of computing devices employing it to synchronize their clocks with tens of thousands of time servers deployed across the globe. Having steady and exact time on networked systems allows time-sensitive applications, such as authentication and email applications, backup and scheduling tools, financial and billing systems, logging and monitoring software, and file and storage sharing protocols, to function with precision.

NTP sends a stream of messages to configured time servers and binds itself to the one with least amount of delay in its responses, the most accurate, and may or may not be the closest distance-wise. The client system maintains a drift in time in a file and references this file for gradual drop in inaccuracy.

RHEL version 8 introduces a new implementation of NTP called *Chrony*. Chrony uses the UDP protocol over the well-known port 123. If enabled, it starts at system boot and continuously operates to keep the system clock in sync with a more accurate source of time.

Chrony performs well on computers that are occasionally connected to the network, attached to busy networks, do not run all the time, or have variations in temperature.

Time Sources

A *time source* is any reference device that acts as a provider of time to other devices. The most precise sources of time are the atomic clocks. They use *Universal Time Coordinated* (UTC) for time accuracy. They produce radio signals that radio clocks use for time propagation to computer servers and other devices that require correctness in time. When choosing a time source for a network, preference should be given to the one that takes the least amount of time to respond. This server may or may not be closest physically.

The common sources of time employed on computer networks are the local system clock, an Internet-based public time server, and a radio clock.

The local system clock can be used as a provider of time. This requires the maintenance of correct time on the server either manually or automatically via *cron*. Keep in mind that this server has no way of synchronizing itself with a more reliable and precise external time source. Therefore, using the local system clock as a time server is the least recommended option.

Several public time servers are available over the Internet for general use (visit www.ntp.org for a list). These servers are typically operated by government agencies, research and scientific organizations, large software vendors, and universities around the world. One of the systems on the local network is identified and configured to receive time from one or more public time servers. This option is preferred over the use of the local system clock.

The official ntp.org site also provides a common pool called pool.ntp.org for vendors and organizations to register their own NTP servers voluntarily for public use. Examples include rhel.pool.ntp.org

and ubuntu.pool.ntp.org for distribution-specific pools, and ca.pool.ntp.org and oceania.pool.ntp.org for country and continent/region-specific pools. Under these sub-pools, the owners maintain multiple time servers with enumerated hostnames such as 0.rhel.pool.ntp.org, 1.rhel.pool.ntp.org, 2.rhel.pool.ntp.org, and so on.

A radio clock is regarded as the perfect provider of time, as it receives time updates straight from an atomic clock. *Global Positioning System* (GPS), WWVB, and DCF77 are some popular radio clock methods. A direct use of signals from these sources requires connectivity of some hardware to the computer identified to act as an organizational or site-wide time server.

NTP Roles

From an NTP standpoint, a system can be configured to operate as a primary server, secondary server, peer, or client.

A *primary* server gets time from a time source and provides time to secondary servers or directly to clients.

A *secondary* server receives time from a primary server and can be configured to furnish time to a set of clients to offload the primary or for redundancy. The presence of a secondary server on the network is optional but highly recommended.

A *peer* reciprocates time with an NTP server. All peers work at the same stratum level, and all of them are considered equally reliable. Both primary and secondary servers can be peers of each other.

A *client* receives time from a primary or a secondary server and adjusts its clock accordingly.

Stratum Levels

As mentioned, there are different types of time sources available so you can synchronize the system clock. These time sources are categorized hierarchically into several levels that are referred to as *stratum levels* based on their distance from the reference clocks (atomic, radio, and GPS). The reference clocks operate at stratum level 0 and are the most accurate provider of time with little to no delay. Besides stratum 0, there are fifteen additional levels that range from 1

to 15. Of these, servers operating at stratum 1 are considered perfect, as they get time updates directly from a stratum 0 device. See [Figure 18-1](#) for a sample hierarchy.

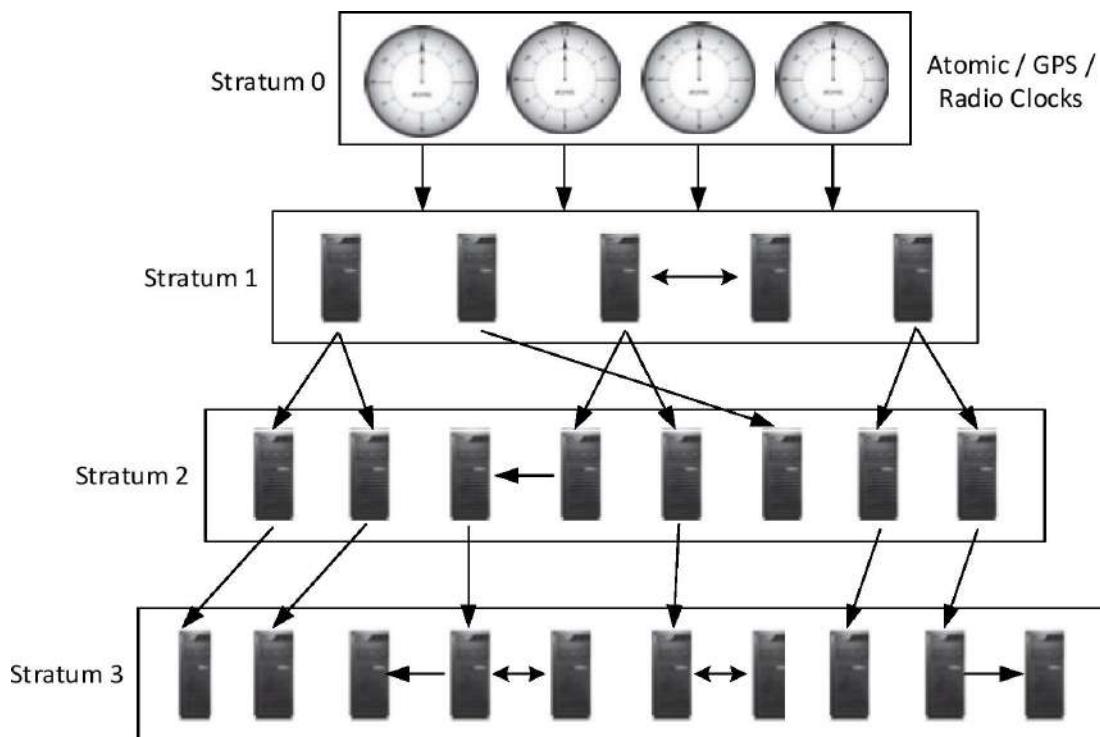


Figure 18-1 NTP Stratum Levels

A stratum 0 device cannot be used on the network directly. It is attached to a computer, which is then configured to operate at stratum 1. Servers functioning at stratum 1 are called *time servers* and they can be set up to deliver time to stratum 2 servers. Similarly, a stratum 3 server can be configured to synchronize its time with a stratum 2 server and deliver time to the next lower level servers, and so on. Servers sharing the same stratum can be configured as peers to exchange time updates with one another.



If a secondary server also gets time updates from a stratum 1 server, it will act as a peer to the primary server.

There are a number of public NTP servers available for free that synchronize time. They normally operate at higher stratum levels such as 2 and 3.

Chrony Configuration File

The key configuration file for the Chrony service is *chrony.conf* located in the */etc* directory. This file is referenced by the Chrony daemon at startup to determine the sources to synchronize the clock, the log file location, and other details. This file can be modified by hand to set or alter directives as required. Some common directives used in this file along with real or mock values are presented below with an explanation in [Table 18-1](#):

driftfile	/var/lib/chrony/drift
logdir	/var/log/chrony
pool	0.rhel.pool.ntp.org iburst
server	server20s8.example.com iburst
server	127.127.1.0
peer	prodntp1.abc.net

[Table 18-1](#) describes these directives.

Directive	Description
driftfile	Indicates the location and name of the drift file to be used to record the rate at which the system clock gains or losses time. This data is used by Chrony to maintain local system clock accuracy.
logdir	Sets the directory location to store the log files in
pool	Defines the hostname that represents a pool of time servers. Chrony binds itself with one of the servers to get updates. In case of a failure of that server, it automatically switches the binding to another server within the pool.
	The iburst option dictates the Chrony service to send the first four update requests to the time server every 2 seconds. This allows the daemon to quickly bring the local clock closer to the time server at startup.
server	Defines the hostname or IP address of a single time server. The IP 127.127.1.0 is a special address that epitomizes the local system clock.
peer	Identifies the hostname or IP address of a time server running at the same stratum level. A peer provides time to a server as well as receives time from the same server.

Table 18-1 Chrony Directives

There are plenty of other directives and options available with Chrony that may be defined in this file. Use **man chrony.conf** for details.

Chrony Daemon and Command

The Chrony service runs as a daemon program called *chronyd* that handles time synchronization in the background. It uses the configuration defined in the */etc/chrony.conf* file at startup and sets its behavior accordingly. If the local clock requires a time adjustment, Chrony takes multiple small steps toward minimizing the gap rather than doing it abruptly in a single step. There are a number of additional options available that may be passed to the service daemon if required.

The Chrony service has a command line program called *chronyc* available that can be employed to monitor the performance of the

service and control its runtime behavior. There are a few subcommands available with *chronyc*; the *sources* and *tracking* subcommands list current sources of time and view performance statistics, respectively.

Exercise 18-1: Configure NTP Client

This exercise should be done on *server10* as *user1* with *sudo* where required.

In this exercise, you will install the Chrony software package and activate the service without making any changes to the default configuration. You will validate the binding and operation.

1. Install the Chrony package using the *dnf* command:

```
[user1@server10 ~]$ sudo dnf -y install chrony
Package chrony-3.3-3.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

The software is already installed on the system.

2. Ensure that preconfigured public time server entries are present in the */etc/chrony.conf* file:

```
[user1@server10 ~]$ grep -E 'pool|server' /etc/chrony.conf | grep -v ^#
pool 2.rhel.pool.ntp.org iburst
```

There is a single pool entry set in the file by default. This pool name is backed by multiple NTP servers behind the scene.

3. Start the Chrony service and set it to autostart at reboots:

```
[user1@server10 ~]$ sudo systemctl --now enable chronyd
Created symlink /etc/systemd/system/multi-user.target.wants/chronyd.service → /usr/li
b/systemd/system/chronyd.service.
```

4. Examine the operational status of Chrony:

```
[user1@server10 ~]$ sudo systemctl status chronyd --no-pager
● chronyd.service - NTP client/server
   Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2019-11-18 10:33:29 EST; 2min 2s ago
       Docs: man:chronyd(8)
              man:chrony.conf(5)
   Process: 11145 ExecStartPost=/usr/libexec/chrony-helper update-daemon (code=exited, status=0/SUCCESS)
   Process: 11141 ExecStart=/usr/sbin/chronyd $OPTIONS (code=exited, status=0/SUCCESS)
 Main PID: 11143 (chronyd)
    Tasks: 1 (limit: 5076)
   Memory: 2.9M
      CGroup: /system.slice/chronyd.service
              └─11143 /usr/sbin/chronyd

Nov 18 10:33:28 server10.example.com systemd[1]: Starting NTP client/server...
Nov 18 10:33:29 server10.example.com chronyd[11143]: chronyd version 3.3 starting...UG)
Nov 18 10:33:29 server10.example.com chronyd[11143]: Frequency -39.266 +/- 1.653 _ift
Nov 18 10:33:29 server10.example.com chronyd[11143]: Using right/UTC timezone to _ata
Nov 18 10:33:29 server10.example.com systemd[1]: Started NTP client/server.
Nov 18 10:33:34 server10.example.com chronyd[11143]: Selected source 50.101.251.61
Nov 18 10:33:34 server10.example.com chronyd[11143]: System clock TAI offset set _nds
```

The service has started successfully and it is set for autostart.

5. Inspect the binding status using the *sources* subcommand with *chronyc*:

```
[user1@server10 ~]$ chronyc sources
210 Number of sources = 4
MS Name/IP address          Stratum Poll Reach LastRx Last sample
=====
^+ 64.ip-54-39-23.net        3      6    377     17    +147us[ +147us] +/-   12ms
^- up2.com                     2      6    377     17    +59us[ +59us] +/-   40ms
^+ bras-vprn-toroon2638w-1p> 2      6    377     82    -88us[ -107us] +/-   19ms
^* gpg.nizyy.com               2      6    377     18   -311us[ -344us] +/-   12ms
```

The output shows the number of available time sources in row 1 and rest of the information in eight columns. Columns 1 to 4—M, S, Name/IP, and Stratum—illustrate the mode, state, name/IP, and stratum level of the source. The ^ means server and the * implies current association.

Columns 4 to 8—Poll, Reach, LastRx, and Last Sample—display the polling rate (6 means 64 seconds), reachability register (377 indicates a valid response was received), how long ago the last sample was received, and the offset between the local clock and the source at the last measurement. Check out the manual pages of the *chronyc* command and search for the section ‘sources’ for additional details.

The last line in the output depicts the *server10* binding with time server gpg.nizyy.com. This association is identified with the asterisk character

(*) beside the time server.

6. Display the clock performance using the *tracking* subcommand with *chronyc*:

```
[user1@server10 ~]$ chronyc tracking
Reference ID      : C0630208 (gpg.nizzy.com)
Stratum          : 3
Ref time (UTC)   : Mon Nov 18 18:06:47 2019
System time      : 0.000128811 seconds fast of NTP time
Last offset      : +0.000033660 seconds
RMS offset       : 0.000096103 seconds
Frequency        : 39.044 ppm slow
Residual freq    : +0.001 ppm
Skew             : 0.051 ppm
Root delay       : 0.017188223 seconds
Root dispersion  : 0.003284285 seconds
Update interval  : 1041.1 seconds
Leap status      : Normal
```

Lines 1 and 2 in the above output identify the current source of time (Reference ID) and the stratum level it is configured at (Stratum). Line 3 shows the reference time at which the last measurement from the time source was processed (Ref time). Line 4 displays the local time offset from NTP time (System time). Line 5 depicts the last reported offset from the NTP server (Last offset). Line 6 identifies the frequency at which time adjustments are occurring (Frequency). The rest of the lines in the output show additional information. Check out the manual pages of the *chronyc* command and search for the section “tracking” for additional details.

EXAM TIP: You will not have access to the outside network during the exam, so you will need to point your system to an NTP server available on the exam network. Simply comment the default server/pool directive(s) and add a single directive “server <hostname>” to the file. Replace <hostname> with the NTP server name or its IP address as provided.

The concludes the exercise.

Displaying and Setting System Date and Time

System date and time can be viewed and manually adjusted with native Linux tools such as the *timedatectl* command. This command can

modify the date, time, and time zone. When executed without any option, as shown below, it outputs the local time, Universal time, RTC time (*real-time clock*, a battery-backed hardware clock located on the system board), time zone, and the status of NTP:

```
[user1@server10 ~]$ timedatectl
          Local time: Mon 2019-11-18 17:50:53 EST
          Universal time: Mon 2019-11-18 22:50:53 UTC
                  RTC time: Mon 2019-11-18 22:50:46
                  Time zone: America/New_York (EST, -0500)
System clock synchronized: yes
          NTP service: active
        RTC in local TZ: no
```

This command requires that the NTP/Chrony service is deactivated in order to make time adjustments. Run the *timedatectl* command as follows to turn off NTP and verify:

```
[user1@server10 ~]$ sudo timedatectl set-ntp false
[user1@server10 ~]$ timedatectl |grep NTP
          NTP service: inactive
```

To modify the current date to January 1, 2020, and confirm:

```
[user1@server10 ~]$ sudo timedatectl set-time 2020-1-1
[user1@server10 ~]$ timedatectl
          Local time: Wed 2020-01-01 00:00:05 EST
          Universal time: Wed 2020-01-01 05:00:05 UTC
                  RTC time: Wed 2020-01-01 05:00:05
                  Time zone: America/New_York (EST, -0500)
System clock synchronized: no
          NTP service: inactive
        RTC in local TZ: no
```

To change the time to 11:20 p.m. and date to November 18, 2019:

```
[user1@server10 ~]$ sudo timedatectl set-time "2019-11-18 23:00"
[user1@server10 ~]$ timedatectl
          Local time: Mon 2019-11-18 23:00:06 EST
          Universal time: Tue 2019-11-19 04:00:06 UTC
                  RTC time: Tue 2019-11-19 04:00:06
                  Time zone: America/New_York (EST, -0500)
System clock synchronized: no
          NTP service: inactive
        RTC in local TZ: no
```

To reactivate NTP:

```
[user1@server10 ~]$ sudo timedatectl set-ntp true  
[user1@server10 ~]$ timedatectl | grep NTP  
    NTP service: active
```

Check out the manual pages of the *timedatectl* command for more subcommands and usage examples.

Alternatively, you can use the *date* command to view or modify the system date and time.

To view current date and time:

```
[user1@server10 ~]$ date  
Tue Nov 19 11:01:08 EST 2019
```

To change the date and time to November 22, 2019 1:00 p.m.:

```
[user1@server10 ~]$ sudo date --set "2019-11-22 13:00"  
Fri Nov 22 13:00:00 EST 2019
```

There are many options available with the *date* command. Consult its manual pages for details.

DNS and Name Resolution

Domain Name System (DNS) is an inverted tree-like structure employed on the Internet and private networks (including home and corporate networks) as the de facto standard for resolving hostnames to their numeric IP addresses. DNS is platform-independent with support integrated in every operating system. DNS is also referred to as BIND, *Berkeley Internet Name Domain*, which is an implementation of DNS, and it has been the most popular DNS application in use.

Name resolution is the technique that uses DNS/BIND for hostname lookups.

In order to understand DNS, a brief discussion of its components and roles is imperative. The following subsections provide a look at the client-side configuration files and commands, along with examples on how to use the tools for resolving hostnames.

DNS Name Space and Domains

The DNS *name space* is a hierarchical organization of all the domains on the Internet. The root of the name space is represented by a period (.). The hierarchy below the root (.) denotes the *top-level domains* (TLDs) with names such as .com, .net, .edu, .org, .gov, .ca, and .de. A DNS *domain* is a collection of one or more systems. Subdomains fall under their parent domains and are separated by a period (.). For example, [redhat.com](#) is a second-level subdomain that falls under .com, and [bugzilla.redhat.com](#) is a third-level subdomain that falls under [redhat.com](#).

[Figure 18-2](#) exhibits a sample hierarchy of the name space, showing the top three domain levels.

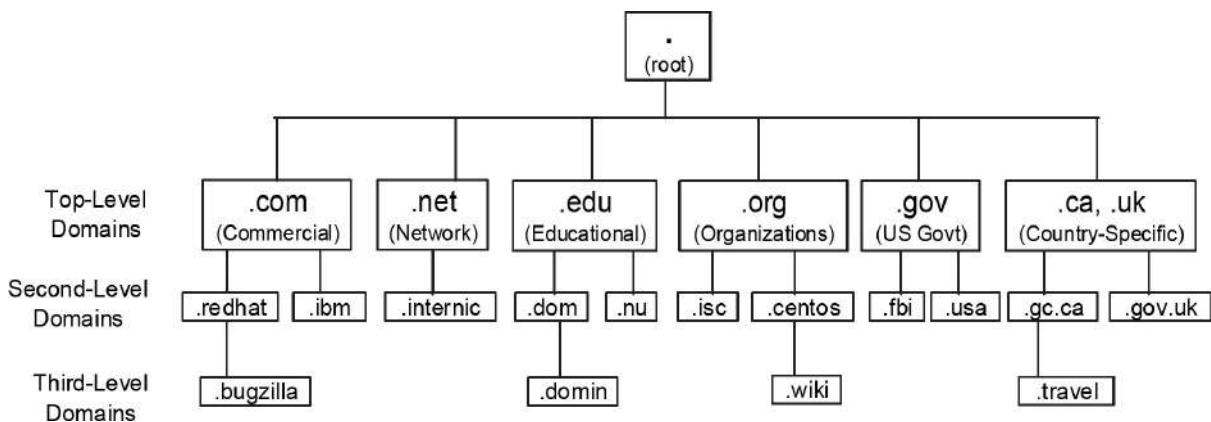


Figure 18-2 Sample DNS Hierarchy

At the deepest level of the hierarchy are the *leaves* (systems, nodes, or any device with an IP address) of the name space. For example, a network switch *net01* in *.travel.gc.ca* subdomain will be known as *net01.travel.gc.ca*. If a period (.) is added to the end of this name to look like *net01.travel.gc.ca.*, it will be referred to as the *Fully Qualified Domain Name* (FQDN) for *net01*.

DNS Roles

From a DNS perspective, a system can be configured to operate as a primary server, secondary server, or client. A DNS server is also referred to as a *nameserver*.

A *primary* (a.k.a. *master*) server is responsible for its domain (or subdomain). It maintains a master database of all the hostnames and

their associated IP addresses that are included in that domain. Any changes in the database is done on this server. Each domain must have one primary server with one or more optional *secondary* (a.k.a. *slave*) servers for load balancing and redundancy. A secondary server also stores an updated copy of the master database and it continues to provide name resolution service in the event the primary server becomes unavailable or inaccessible.

A *DNS client* queries nameservers for name lookups. Every system with access to the Internet or other external networks will have the DNS client functionality configured and operational. Setting up DNS client on Linux involves two text files that are discussed in the next two subsections.

Understanding Resolver Configuration File

The *resolv.conf* file under */etc* is the DNS resolver configuration file where information to support hostname lookups is defined. This file may be edited manually with a text editor. It is referenced by resolver utilities to construct and transmit queries. There are three key directives set in this file—domain, nameserver, and search—and they are described in [Table 18-2](#).

Directive	Description
domain	Identifies the default domain name to be searched for queries
nameserver	Declares up to three DNS server IP addresses to be queried one at a time in the order in which they are listed. Nameserver entries may be defined as separate line items with the directive or on a single line.
search	Specifies up to six domain names, of which the first must be the local domain. No need to define the domain directive if the search directive is used.

Table 18-2 The Resolver Configuration File

A sample entry showing the syntax is provided below for reference:

```
domain      example.com
search      example.net example.org example.edu example.gov
nameserver  192.168.0.1 8.8.8.8 8.8.4.4
```

A variation of the above would be:

```
domain      example.com
search      example.net example.org example.edu example.gov
nameserver  192.168.0.1
nameserver  8.8.8.8
nameserver  8.8.4.4
```

Currently, there are two entries “search `example.com`” and “nameserver 192.168.0.1” defined in the `resolv.conf` file on `server10` and `server20`.

```
[user1@server10 ~] $ cat /etc/resolv.conf
# Generated by NetworkManager
search example.com
nameserver 192.168.0.1
```

On a system with this file absent, the resolver utilities only query the nameserver configured on the localhost, determine the domain name from the hostname of the system, and construct the search path based on the domain name.

Viewing and Adjusting Name Resolution Sources and Order The `nsswitch.conf` file under `/etc` directs the lookup utilities to the correct source to get hostname information. In the presence of multiple sources, this file also identifies the order in which to consult them and an action to be taken next. There are four keywords—`success`, `notfound`, `unavail`, and `tryagain`—that oversee this behavior, and are described along with default actions in [Table 18-3](#).

Keyword	Meaning	Default Action
success	Information found in source and provided to the requester	return (do not try the next source)
notfound	Information not found in source	continue (try the next source)
unavail	Source down or not responding; service disabled or not configured	continue (try the next source)
tryagain	Source busy, retry later	continue (try the next source)

Table 18-3 Name Service Source and Order Determination

The following example entry shows the syntax of a relevant entry from the *nsswitch.conf* file. It shows two sources for name resolution: files (*/etc/hosts*) and DNS (*/etc/resolv.conf*).

hosts:	files	dns
--------	-------	-----

Based on the default behavior, the search will terminate if the requested information is found in the *hosts* table. However, you can alter this behavior and instruct the lookup programs to return if the requested information is not found there. The modified entry will look like:

hosts:	files [notfound=return]	dns
--------	-------------------------	-----

This altered entry will ignore the DNS.



See [Chapter 16](#) for details on the */etc/hosts* file.

Once the *resolv.conf* and *nsswitch.conf* files are configured appropriately, you can use any of the native client resolver tools for lookups. Common query tools available in RHEL 8 include *dig*, *host*, *nslookup*, and *getent*.

Performing Name Resolution with *dig*

dig (domain information groper) is a DNS lookup utility. It queries the nameserver specified at the command line or consults the *resolv.conf* file to determine the nameservers to be queried. This tool may be used to troubleshoot DNS issues due to its flexibility and verbosity. The following shows a few usage examples.

To get the IP for redhat.com using the nameserver listed in the *resolv.conf* file:

```
[user1@server10 ~]$ dig redhat.com  
.....  
;; ANSWER SECTION:  
redhat.com.      3152      IN      A      209.132.183.105  
  
;; Query time: 13 msec  
;; SERVER: 192.168.0.1#53(192.168.0.1)  
;; WHEN: Tue Nov 19 09:38:50 EST 2019  
;; MSG SIZE  rcvd: 55
```

The output shows the total time (13 milliseconds) it took to get the result, the IP address (209.132.183.105) of redhat.com, the nameserver IP (192.168.0.1) used for the query, the DNS port number (53), query timestamp, the size of the received message, and other information.

To perform a reverse lookup on the redhat.com IP (209.132.183.105), use the **-x** option with the command:

```
[user1@server10 ~]$ dig -x 209.132.183.105  
.....  
;; ANSWER SECTION:  
105.183.132.209.in-addr.arpa. 600 IN PTR redirect.redhat.com.  
  
;; AUTHORITY SECTION:  
183.132.209.in-addr.arpa. 600 IN NS ns2.redhat.com.  
183.132.209.in-addr.arpa. 600 IN NS ns3.redhat.com.  
183.132.209.in-addr.arpa. 600 IN NS ns4.redhat.com.  
183.132.209.in-addr.arpa. 600 IN NS ns1.redhat.com.  
  
;; Query time: 67 msec  
;; SERVER: 192.168.0.1#53(192.168.0.1)  
;; WHEN: Tue Nov 19 09:46:38 EST 2019  
;; MSG SIZE  rcvd: 162
```

Reference the command's manual pages for details and options.

Performing Name Resolution with host

host is an elementary DNS lookup utility that works on the same principles as the *dig* command in terms of nameserver determination. This tool produces lesser data in the output by default; however, you can add the *-v* option for verbosity.

To perform a lookup on redhat.com:

```
[user1@server10 ~]$ host redhat.com
redhat.com has address 209.132.183.105
redhat.com mail is handled by 10 us-smtp-inbound-2.mimecast.com.
redhat.com mail is handled by 10 us-smtp-inbound-1.mimecast.com.
```

Rerun the above with *-v* added. The output will be similar to that of the *dig* command.

To perform a reverse lookup on the IP of redhat.com using the *-v* flag to add details:

```
[user1@server10 ~]$ host -v 209.132.183.105
Trying "105.183.132.209.in-addr.arpa"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31347
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;105.183.132.209.in-addr.arpa. IN PTR

;; ANSWER SECTION:
105.183.132.209.in-addr.arpa. 595 IN PTR redirect.redhat.com.

Received 79 bytes from 192.168.0.1#53 in 4 ms
```

Refer to the command's manual pages for options and more information.

Performing Name Resolution with nslookup

nslookup queries the nameservers listed in the *resolv.conf* file or specified at the command line. The following shows a few usage examples.

To get the IP for redhat.com using nameserver 8.8.8.8 instead of the nameserver defined in *resolv.conf*:

```
[user1@server10 ~]$ nslookup redhat.com 8.8.8.8
Server:      8.8.8.8
Address:     8.8.8.8#53

Non-authoritative answer:
Name:   redhat.com
Address: 209.132.183.105
```

To perform a reverse lookup on the IP of [redhat.com](#) using the nameserver from the resolver configuration file:

```
[user1@server10 ~]$ nslookup 209.132.183.105
105.183.132.209.in-addr.arpa      name = redirect.redhat.com.
```

Consult the command's manual pages on how to use it in interactive mode.

Performing Name Resolution with `getent`

The `getent` (*get entries*) command is a rudimentary tool that can fetch matching entries from the databases defined in the `nsswitch.conf` file. This command reads the corresponding database and displays the information if found. For name resolution, use the `hosts` database and `getent` will attempt to resolve the specified hostname or IP address. For instance, run the following for forward and reverse lookups:

```
[user1@server10 ~]$ getent hosts redhat.com
209.132.183.105 redhat.com
[user1@server10 ~]$ getent hosts 209.132.183.105
209.132.183.105 redirect.redhat.com
```

Check the command's manual pages for available flags and additional information.

Chapter Summary

The focus of this chapter was on two topics: network time synchronization and hostname resolution.

The chapter began with a discussion of Network Time Protocol, what role it plays in keeping the clocks synchronized, and what is its

relationship with the Chrony service. We explored various sources for obtaining time, different roles that systems could play, and the strata paradigm. We analyzed the configuration file to understand some key directives and their possible settings. We performed an exercise to configure the service, display clock association, and analyze the results. We also employed a couple of other RHEL tools to display the system time and set it instantly.

We concluded the chapter with a deliberation of DNS and name resolution. We discussed the concepts and roles, analyzed the resolver configuration file, and examined the source/order determination file. We added required entries to the resolver configuration file and tested the functionality by employing client tools for hostname lookup.

Review Questions

1. Chrony is an implementation of the Network Time Protocol. True or False?
2. What is the name and location of the DNS resolver file?
3. What stratum level do two peer systems operate on a network?
4. Provide the maximum number of nameservers that can be defined in the resolver configuration file.
5. What is the purpose of a drift file in Chrony/NTP?
6. What is a relative distinguished name?
7. What would you add to the Chrony configuration file if you want to use the local system clock as the provider of time?
8. BIND is an implementation of Domain Name System. True or False?
9. Define time source.
10. Name the three DNS roles that a RHEL system can play.
11. What would you run to check the NTP bind status with time servers?
12. Define DNS name space.
13. Name the four Chrony/NTP roles that a RHEL system can play.
14. Which file defines the name resolution sources and controls the order in which they are consulted?
15. What is the filename and directory location for the Chrony configuration file?

16. The Chrony client is preconfigured when the *chrony* software package is installed. You just need to start the service to synchronize the clock. True or False?
17. List any three DNS lookup tools.
18. List two utilities that you can use to change system time.

Answers to Review Questions

1. True.
2. The DNS resolver file is called *resolv.conf* and it is located in the */etc* directory.
3. Two peers on a network operate at the same stratum level.
4. Three.
5. The purpose of a drift file is to keep track of the rate at which the system clock gains or losses time.
6. A relative distinguished name represents individual components of a distinguished name.
7. You will add “server 127.127.1.0” to the Chrony configuration file and restart the service.
8. True.
9. A time source is a reference device that provides time to other devices.
10. From a DNS perspective, a RHEL machine can be a primary server, a secondary server, or a client.
11. You will run *chronyc sources* to check the binding status.
12. DNS name space is a hierarchical organization of all the domains on the Internet.
13. From a Chrony/NTP standpoint, a RHEL machine can be a primary server, a secondary server, a peer, or a client.
14. The */etc/nsswitch.conf* file.
15. The name of the Chrony configuration file is *chrony.conf* and it is located in the */etc* directory.
16. True.
17. Name resolution tools are *dig*, *host*, *getent*, and *nslookup*.
18. The *timedatectl* and *date* commands can be used to modify the system time.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 18-1: Modify System Date and Time

As *user1* with *sudo* on *server10*, execute the *date* and *timedatectl* commands to check the current system date and time. Identify the distinctions between the two outputs. Use *timedatectl* and change the system date to a future date. Issue the *date* command and change the system time to one hour ahead of the current time. Observe the new date and time with both commands. Reset the date and time to the current actual time using either the *date* or the *timedatectl* command. (Hint: Time Synchronization).

Lab 18-2: Configure Chrony

As *user1* with *sudo* on *server10*, install Chrony and mark the service for autostart on reboots. Edit the Chrony configuration file and comment all line entries that begin with “pool” or “server”. Go to the end of the file, and add a new line “server 127.127.1.0”. Start the Chrony service and run **chronyc sources** to confirm the binding. (Hint: Time Synchronization).

Chapter 19

The Secure Shell Service

This chapter describes the following major topics:

- Understand the OpenSSH service, versions, and algorithms
- Overview of encryption techniques and authentication methods
- Describe OpenSSH administration commands and configuration files
- Configure private/public key-based authentication
- Access OpenSSH server from other Linux systems
- Use OpenSSH client tools to transfer files
- Synchronize files remotely over OpenSSH

RHCSA Objectives:

04. Access remote systems using ssh
26. Securely transfer files between systems
57. Configure key-based authentication for SSH

Secure Shell is a network service that delivers a secure mechanism for data transmission between source and destination systems over insecure network paths. It provides a set of utilities that allows users to generate key pairs and use them to set up trusted logins between systems for themselves. Additional utilities in the set gives remote users the ability to log in, execute commands, and transfer files securely over encrypted network channels. These tools have predominantly supplanted their insecure counterparts in the corporate world.

The OpenSSH Service

Secure Shell (SSH) delivers a secure mechanism for data transmission between source and destination systems over IP networks. It was designed to replace the old remote login programs that transmitted user passwords in clear text and data unencrypted. SSH employs digital signatures for user authentication with encryption to secure a communication channel. As a result, this makes it extremely hard for unauthorized people to gain access to passwords or the data in transit. It also monitors the data being transferred throughout a session to ensure integrity. SSH includes a set of utilities for remote users to log in, transfer files, and execute commands securely. Due to strong security features, SSH utilities have supplanted their conventional, unsecure login and file transfer counterpart programs.

OpenSSH is a free, open source implementation of the proprietary SSH. Once applied successfully on the system, the unsecure services—*telnet*, *rlogin*, *rcp*, and *ftp*—can be disabled after a careful examination to eliminate potential impact. The secure command that has substituted *telnet* and *rlogin* remote login services is called *ssh*, and those for *rcp* and *ftp* are called *scp* and *sftp*, respectively.

Common Encryption Techniques

Encryption is a way of scrambling information with the intent to conceal the real information from unauthorized access. OpenSSH can utilize various encryption techniques during an end-to-end communication session between two entities (client and server). The two common techniques are *symmetric* and *asymmetric*. They are also referred to as *secret key encryption* and *public key encryption* techniques.

Symmetric Technique

This technique uses a single key called a *secret* key that is generated as a result of a negotiation process between two entities at the time of their initial contact. Both sides use the same secret key during subsequent communication for data encryption and decryption.

Asymmetric Technique

This technique uses a combination of *private* and *public* keys, which are randomly generated and mathematically related strings of alphanumeric characters attached to messages being exchanged. The client transmutes the information with a *public* key and the server decrypts it with the paired *private* key. The private key must be kept secure since it is private to a single sender; the public key is disseminated to clients. This technique is used for channel encryption as well as user authentication.

Authentication Methods

Once an encrypted channel is established between the client and server, additional negotiations take place between the two to authenticate the user trying to access the server. OpenSSH offers several methods for this purpose; they are listed below in the order in which they are attempted during the authentication process:

- GSSAPI-based (*Generic Security Service Application Program Interface*) authentication
- Host-based authentication
- Public key-based authentication
- Challenge-response authentication
- Password-based authentication

Let's review each one in detail.

GSSAPI-Based Authentication

GSSAPI provides a standard interface that allows security mechanisms, such as Kerberos, to be plugged in. OpenSSH uses this interface and the underlying Kerberos for authentication. With this method, an exchange of tokens takes place between the client and server to validate user identity.

Host-Based Authentication

This type of authentication allows a single user, a group of users, or all users on the client to be authenticated on the server. A user may be configured to log in with a matching username on the server or as a different user that already exists there. For each user that requires an automatic entry on the

server, a `~/.shosts` file is set up containing the client name or IP address, and, optionally, a different username.

The same rule applies to a group of users or all users on the client that require access to the server. In that case, the setup is done in the `/etc/ssh/shosts.equiv` file on the server.

Private/Public Key-Based Authentication

This method uses a private/public key combination for user authentication. The user on the client has a public key and the server stores the corresponding private key. At the login attempt, the server prompts the user to enter the passphrase associated with the key and logs the user in if the passphrase and key are validated.

Challenge-Response Authentication

This method is based on the response(s) to one or more arbitrary challenge questions that the user has to answer correctly in order to be allowed to log in to the server.

Password-Based Authentication

This is the last fall back option. The server prompts the user to enter their password. It checks the password against the stored entry in the `shadow` file and allows the user in if the password is confirmed.

Of the five authentication methods, the password-based method is common and requires no further explanation. The GSSAPI-based, host-based, and challenge-response methods are beyond the scope of this book. The public/private authentication and encryption methods will be the focus in the remainder of this chapter.

OpenSSH Protocol Version and Algorithms

OpenSSH has evolved over the years. Its latest and the default version in RHEL 8, version 2, has numerous enhancements, improvements, and sophisticated configuration options. It supports various algorithms for data encryption and user authentication (digital signatures) such as RSA, DSA, and ECDSA. RSA is more common and it is widely employed partly because it supports both encryption and authentication. In contrast, DSA and ECDSA are restricted to authentication only. These algorithms are used to generate public and private key pairs for the asymmetric technique.

RSA stands for *Rivest-Shamir-Adleman*, who first published this algorithm, DSA for *Digital Signature Algorithm*, and ECDSA is an acronym for *Elliptic Curve Digital Signature Algorithm*.

OpenSSH Packages

OpenSSH has three software packages that are of interest. These are *openssh*, *openssh-clients*, and *openssh-server*. The *openssh* package provides the *ssh-keygen* command and some library routines; the *openssh-clients* package includes commands, such as *scp*, *sftp*, *ssh*, and *ssh-copy-id*, and a client configuration file */etc/ssh/ssh_config*; and the *openssh-server* package contains the *sshd* service daemon, server configuration file */etc/ssh/sshd_config*, and library routines. By default, all three packages are installed during OS installation.

OpenSSH Server Daemon and Client Commands

The OpenSSH server program *sshd* is preconfigured and operational on new RHEL installations, allowing remote users to log in to the system using an ssh client program such as PuTTY or the *ssh* command. This daemon listens on well-known TCP port 22 as documented in the */etc/ssh/sshd_config* file with the Port directive.

The client software includes plenty of utilities such as those listed and described in [Table 19-1](#).

Command	Description
<i>scp</i>	The secure remote copy command that replaces non-secure <i>rcp</i> command
<i>sftp</i>	The secure remote copy command that replaces non-secure <i>ftp</i> command
<i>ssh</i>	The secure remote login command that replaces non-secure <i>telnet</i> and <i>rlogin</i> commands
<i>ssh-copy-id</i>	Copies public key to remote systems
<i>ssh-keygen</i>	Generates and manages private and public keys

Table 19-1 OpenSSH Client Tools

The use of these commands is demonstrated in the following subsections.

Server Configuration File

The OpenSSH server daemon *sshd* has a configuration file that defines default global settings on how it should operate. This file is located in the */etc/ssh* directory and called *sshd_config*. There are a number of directives

preset in this file that affect all inbound ssh communication and are tuned to work as-is for most use cases. In addition, the `/var/log/secure` log file is used to capture authentication messages.

A few directives with their default values from the `sshd_config` file are displayed below:

```
[user1@server10 ~] $ sudo cat /etc/ssh/sshd_config
```

#Port	22
#Protocol	2
ListenAddress	0.0.0.0
SyslogFacility	AUTHPRIV
#LogLevel	INFO
PermitRootLogin	yes
#PubkeyAuthentication	yes
AuthorizedKeysFile	.ssh/authorized_keys
PasswordAuthentication	yes
#PermitEmptyPasswords	no
ChallengeResponseAuthentication	no
UsePAM	yes
X11Forwarding	yes

The above directives are elaborated in [Table 19-2](#).

Directive	Description
Port	Specifies the port number to listen on. Default is 22.
Protocol	Specifies the default protocol version to use.
ListenAddress	Sets the local addresses the sshd service should listen on. Default is to listen on all local addresses.
SyslogFacility	Defines the facility code to be used when logging messages to the /var/log/secure file. This is based on the configuration in the /etc/rsyslog.conf file. Default is AUTHPRIV.
LogLevel	Identifies the level of criticality for the messages to be logged. Default is INFO.
PermitRootLogin	Allows or disallows the root user to log in directly to the system. Default is yes.
PubKeyAuthentication	Enables or disables public key-based authentication. Default is yes.
AuthorizedKeysFile	Sets the name and location of the file containing a user's authorized keys. Default is ~/.ssh/authorized_keys.
PasswordAuthentication	Enables or disables local password authentication. Default is yes.
PermitEmptyPasswords	Allows or disallows the use of null passwords. Default is no.
ChallengeResponseAuthentication	Enables or disables challenge-response authentication mechanism. Default is yes.
UsePAM	Enables or disables user authentication via PAM. If enabled, only root will be able to run the sshd daemon. Default is yes.
X11Forwarding	Allows or disallows remote access to graphical applications. Default is yes.

Table 19-2 OpenSSH Server Configuration File

There are many more settings available that may be added to the file for additional control. Check out the manual pages of the `sshd_config` file ([man 5 sshd_config](#)) for details.

Client Configuration File

Each RHEL client machine that uses ssh to access a remote OpenSSH server has a local configuration file that directs how the client should behave. This file, `ssh_config`, is located in the `/etc/ssh` directory. There are a number of directives preset in this file that affect all outbound ssh communication and are tuned to work as-is for most use cases.

A few directives with their default values from the `ssh_config` file are displayed below:

```
[user1@server10 ~] $ sudo more /etc/ssh/ssh_config
```

# Host *	
# ForwardX11	no
# PasswordAuthentication	yes
# StrictHostKeyChecking	ask
# IdentityFile	<code>~/.ssh/id_rsa</code>
# IdentityFile	<code>~/.ssh/id_dsa</code>
# Port	22
# Protocol	2

The above directives are described in [Table 19-3](#).

Directive	Description
Host	Container that declares directives applicable to one host, a group of hosts, or all hosts. It ends when another occurrence of Host or Match is encountered. Default is *, which sets global defaults for all hosts.
ForwardX11	Enables or disables automatic redirection of X11 traffic over SSH connections. Default is no.
PasswordAuthentication	Allows or disallows password authentication. Default is yes.
StrictHostKeyChecking	Controls (1) whether to add host keys (host fingerprints) to <code>~/.ssh/known_hosts</code> when accessing a host for the first time, and (2) what to do when the keys of a previously-accessed host mismatch with what is stored in <code>~/.ssh/known_hosts</code> . Options are: no : adds new host keys and ignores changes to existing keys. yes : adds new host keys and disallows connections to hosts with non-matching keys. accept-new : adds new host keys and disallows connections to hosts with non-matching keys. ask (default) : prompts whether to add new host keys and disallows connections to hosts with non-matching keys.
IdentityFile	Defines the name and location of a file that stores a user's private key for their identity validation. Defaults are <code>id_rsa</code> , <code>id_dsa</code> , and <code>id_ecdsa</code> based on the type of algorithm used. Their corresponding public key files with <code>.pub</code> extension are also stored at the same directory location.
Port	Sets the port number to listen on. Default is 22.
Protocol	Specifies the default protocol version to use

Table 19-3 OpenSSH Client Configuration File

The `~/.ssh` directory does not exist by default; it is created when a user executes the `ssh-keygen` command for the first time to generate a key pair or connects to a remote ssh server and accepts its host key for the first time. In the latter case, the client stores the server's host key locally in a file called `known_hosts` along with its hostname or IP address. On subsequent access attempts, the client will use this information to verify the server's authenticity.

There are a lot more settings available that may be added to the file for additional control. Check out the manual pages of the `ssh_config` file ([man 5 ssh_config](#)) for details.

System Access and File Transfer

A user must log in to the Linux system in order to use it or transfer files. The login process identifies the user to the system. For accessing a RHEL system remotely, use the `ssh` command, and the `scp` or the `sftp` command for copying files. These commands require either a resolvable hostname of the target system or its IP address in order to try to establish a connection. All these commands are secure and may be used over secure and unsecure network channels to exchange data.

The following subsections and exercises describe multiple access scenarios including accessing a RHEL system (`server20`) from another RHEL system (`server10`) and a Windows computer, accessing a RHEL system (`server20`) using `ssh` keys, and transferring files using `scp` and `sftp`.

Exercise 19-1: Access RHEL System from Another RHEL System

This exercise should be done on `server10` and `server20` as `user1`.

This exercise works under two assumptions: (1) `user1` exists on both `server10` and `server20`, and (2) hostname and IP mapping is in place in the `/etc/hosts` file ([Chapter 16](#)). Use the IP address in lieu of the hostname if the mapping is unavailable for `server20`.

In this exercise, you will issue the `ssh` command as `user1` on `server10` to log in to `server20`. You will run appropriate commands on `server20` for validation. You will log off and return to the originating system.

1. Issue the `ssh` command as `user1` on `server10`:

```
[user1@server10 ~]$ ssh server20
The authenticity of host 'server20 (192.168.0.120)' can't be established.
ECDSA key fingerprint is SHA256:fDOWHphHOYEOMrB2SUGKNEkWONABCPBHf8nb6y8nnbh0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server20,192.168.0.120' (ECDSA) to the list of known hosts.
user1@server20's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Nov 22 11:02:49 2019 from 192.168.0.110
[user1@server20 ~]$
```

Answer ‘yes’ to the question presented and press Enter to continue. This step adds the hostname of `server20` to a file called `known_hosts` under `/home/user1/.ssh` directory on the originating computer (`server10`). This message will not reappear on subsequent login attempts to `server20` for this

user. Enter the correct password for *user1* to be allowed in. You will be placed in the home directory of *user1* on *server20*. The command prompt will reflect that information.

2. Issue the basic Linux commands *whoami*, *hostname*, and *pwd* to confirm that you are logged in as *user1* on *server20* and placed in the correct home directory:

```
[user1@server20 ~]$ whoami  
user1  
[user1@server20 ~]$ hostname  
server20.example.com  
[user1@server20 ~]$ pwd  
/home/user1  
[user1@server20 ~]$
```

3. Run the *logout* or the *exit* command or simply press the key combination *Ctrl+d* to log off *server20* and return to *server10*:

```
[user1@server20 ~]$ exit  
logout  
Connection to server20 closed.  
[user1@server10 ~]$
```

This concludes the exercise.

If you wish to log on as a different user such as *user2* (assuming *user2* exists on the target server *server20*), you may run the *ssh* command in either of the following ways:

```
[user1@server10 ~]$ ssh -l user2 server20  
[user1@server10 ~]$ ssh user2@server20
```

The above will allow you to log in if the password entered for *user2* is valid.

Exercise 19-2: Generate, Distribute, and Use SSH Keys

This exercise should be done on *server10* and *server20* as *user1* and *sudo* where required.

In this exercise, you will generate a password-less ssh key pair using RSA algorithm for *user1* on *server10*. You will display the private and public file contents. You will distribute the public key to *server20* and attempt to log on to *server20* from *server10*. You will show the log file message for the login attempt.

1. Log on to *server10* as *user1*.
2. Generate RSA keys without a password (-N) and without detailed output (-q). Press Enter when prompted to provide the filename to store the

private key.

```
[user1@server10 ~]$ ssh-keygen -N "" -q  
Enter file in which to save the key (/home/user1/.ssh/id_rsa):  
[user1@server10 ~]$
```

The content of the *id_rsa* (private key) file is shown below:

```
[user1@server10 ~]$ cat .ssh/id_rsa  
-----BEGIN OPENSSH PRIVATE KEY-----  
b3B1bnNzaC1rZXktdjEAAAAABG5vbmuAAAAAEbm9uZQAAAAAAAAABAAABFwAAAAdzc2gtcn  
NhAAAAAWEAQQAAQEApG1Hp95XPXCoXfs8qQWDsYnQYiTfAb8VqP46mXSztKoYNHsC+FQa  
n+6XNh5wOxG70+11ZN15y/RBIRwOT5WxrgKlm9u6EREOkP2F36vFWWJT12Cm3e5LFeuBwt  
MmF3yBgsOjf2oquUIX8hI/NJCrc1qut/oP31Om4Ci0/GrFn9FJgS8A1/KFgYn+H/pY9u6  
xEV5ZH286RvrcGISyyQr6q63+P+ojzR4n2gO26HNh0FpE8ZDeA+12516rdaS9iqkieQhrk  
uTr4Pe0gXgFfGLQxAAAAGnVzZXIxQHN1cnZlcjEwLmV4YW1wbGUuY29t  
-----END OPENSSH PRIVATE KEY-----
```

The content of the *id_rsa.pub* (public key) file is displayed below:

```
[user1@server10 ~]$ cat .ssh/id_rsa.pub  
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCKaUen3lc9cKhd+zypBYOxitBiJN8BvxWo/jqZdJm0  
qhg0ewL4VBqf7pc2HnA7EbVt6XVr2XnL9EEhHA5P1bGuAqSb27oREQ6Q/YXfq8VZYlPXVKbd7ksV64HC  
OFavt0JY1OkXWj21sBFuqFvPnfqmyzuZ6O+Hb66K4fvmgTZcfjfDksU7czWhXQ36rt25Ih/7k1WHO92  
8p8naQ8tZvEB7hxHqeArHHJcbS0d4SdrhXNImGgYe9deyWSRT4eG6KcIcN8+vo267UGpLTngFDy3kUMI  
xzd5R1Q/fmmeK3MDF5AtpNjAPoGcFuns3drQbwM+gTrncemqBWHq2XSD7KM3 user1@server10.exam  
ple.com
```

3. Copy the public key file to *server20* under */home/user1/.ssh* directory. Accept the fingerprints for *server20* when prompted (only presented on the first login attempt). Enter the password for *user1* set on *server20* to continue with the file copy. The public key will be copied as *authorized_keys*.

```
[user1@server10 ~]$ ssh-copy-id server20  
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/user1/.ssh/  
id_rsa.pub"  
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter  
out any that are already installed  
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt  
ed now it is to install the new keys  
user1@server20's password:  
  
Number of key(s) added: 1  
  
Now try logging into the machine, with: "ssh 'server20'"  
and check to make sure that only the key(s) you wanted were added.  
[user1@server10 ~]$ _
```

At the same time, this command also creates or updates the *known_hosts* file on *server10* and stores the fingerprints for *server20* in it. Here is what is currently stored in it:

```
[user1@server10 ~]$ cat .ssh/known_hosts
server20,192.168.0.120 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAlb
m1zdH AyNTYAAABBBJwhgvOsYtOShMjVKWycS7HJQb54i5eASbt9kBjE9DK6gOMYrII4mIR/s5NVdhu9o
zrPrV7AIaxx8jSG0smvRjk=
```

- On *server10*, run the *ssh* command as *user1* to connect to *server20*. You will not be prompted for a password because there was none assigned to the ssh keys.

```
[user1@server10 ~]$ ssh server20
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sun Nov 24 17:11:37 2019 from 192.168.0.13
[user1@server20 ~]$
```

You can view this login attempt in the */var/log/secure* file on *server20*:

```
[user1@server20 ~]$ sudo tail /var/log/secure
.....
Nov 24 17:41:31 server20 sshd[19228]: Accepted publickey for user1 from 192.168.
0.110 port 56374 ssh2: RSA SHA256:OLtCqjQPiXk2fNecJOEbR8KqP5w59k0IOIopluz5OIO
Nov 24 17:41:31 server20 sshd[19228]: pam_unix(sshd:session): session opened for
user user1 by (uid=0)
```

The log entry shows the timestamp, hostname, process name and PID, username and source IP, and other relevant information. This file will log all future login attempts for this user.

Executing Commands Remotely Using ssh

The *ssh* command is a secure replacement for the legacy unsecure tools *telnet*, *rlogin*, and *rsh*. It allows you to securely sign in to a remote system or execute a command without actually logging on to it. [Exercise 19-2](#) demonstrated how a user can log in using this command. The following shows a few basic examples on how to use *ssh* to execute a command on a remote system.

Invoke the *ssh* command on *server10* to execute the *hostname* command on *server20*:

```
[user1@server10 ~]$ ssh server20 hostname
server20.example.com
```

Run the *nmcli* command on *server20* to show (s) active network connections (c):

```
[user1@server10 ~]$ ssh server20 nmcli c s
NAME      UUID                                     TYPE      DEVICE
enp0s3    7426fe5c-9bd4-42ad-bb95-3d15491082b9  ethernet  enp0s3
enp0s8    c9f81ca1-4418-4a8d-9f9a-b6e4368bba46  ethernet  enp0s8
virbr0   58c12da2-fbba-488b-bcccd-280b67ef176b   bridge    virbr0
```

You can run any command on *server20* this way without having to log in to it.

Copying Files Remotely Using *scp*

Similar to *ssh*, a user can execute the *scp* command to transfer files from *server10* to *server20*, and vice versa. This program can be run by a normal user as long as the user has the required read and write permissions on the source and destination, or by the *root* user. Here are a few examples to understand the program's syntax and usage.

To transfer the */etc/chrony.conf* file from *server20* to */tmp* on *server10* and confirm:

```
[user1@server10 ~]$ scp server20:/etc/chrony.conf /tmp  
chrony.conf                                100% 1103    830.3KB/s  00:00  
[user1@server10 ~]$ ls -l /tmp/chrony.conf  
-rw-r--r--. 1 user1 user1 1103 Nov 24 18:33 /tmp/chrony.conf
```

The program took not even a second to transfer the file. The file size is 1103 bytes. The *ls* command confirms the pull.

Now let's transfer the entire */etc/sysconfig* directory (-r) from *server10* into */tmp* on *server20* and confirm. Ignore any permission errors reported in the output.

```
[user1@server10 ~]$ scp -r /etc/sysconfig server20:/tmp  
/etc/sysconfig/ip6tables-config: Permission denied  
/etc/sysconfig/iptables-config: Permission denied  
/etc/sysconfig/ebtables-config: Permission denied  
nftables.conf                                100% 507    377.2KB/s  00:00  
ifcfg-enp0s3                                  100% 344    264.3KB/s  00:00  
ifcfg-enp0s8                                  100% 166    123.2KB/s  00:00  
samba                                         100% 428    360.1KB/s  00:00  
selinux                                       100% 548    599.5KB/s  00:00  
run-parts                                    100% 0      0.0KB/s   00:00  
crond                                         100% 110    94.4KB/s  00:00  
chronyd                                       100% 46     41.0KB/s  00:00  
cpupower                                      100% 150    133.1KB/s  00:00  
raid-check                                     100% 2915   2.1MB/s   00:00  
wpa_supplicant                                100% 258    177.8KB/s  00:00  
grub                                           100% 329    218.5KB/s  00:00  
up2date                                       100% 1796   1.8MB/s   00:00  
.....
```

Run the *ls* command on *server20* for verification:

```
[user1@server10 ~]$ ssh server20 ls -l /tmp/sysconfig
total 88
-rw-r--r--. 1 user1 user1 162 Jan  8 23:33 anaconda
-rw-r--r--. 1 user1 user1 403 Jan  8 23:33 atd
-rw-r--r--. 1 user1 user1 339 Jan  8 23:33 autofs
drwxr-xr-x. 2 user1 user1  43 Jan  8 23:33 cbq
-rw-r--r--. 1 user1 user1  46 Jan  8 23:33 chronyd
drwxr-xr-x. 2 user1 user1   6 Jan  8 23:33 console
-rw-r--r--. 1 user1 user1 150 Jan  8 23:33 cpupower
-rw-r--r--. 1 user1 user1 110 Jan  8 23:33 crond
-rw-r--r--. 1 user1 user1  73 Jan  8 23:33 firewalld
-rw-r--r--. 1 user1 user1 329 Jan  8 23:33 grub
-rw-r--r--. 1 user1 user1 903 Jan  8 23:33 irqbalance
-rw-r--r--. 1 user1 user1 1722 Jan  8 23:33 kdump
-rw-r--r--. 1 user1 user1 185 Jan  8 23:33 kernel
-rw-r--r--. 1 user1 user1 310 Jan  8 23:33 man-db
drwxr-xr-x. 2 user1 user1   6 Jan  8 23:33 modules
-rw-r--r--. 1 user1 user1   22 Jan  8 23:33 network
....
```

The output verifies the directory copy.

In the above examples, the user account that was used on the source and target servers is the same user, *user1*. To transfer a file or directory using a different user account on the target server, you need to include that user's name with the command. You must know the password for the user on the target server. Here is the syntax:

```
[user1@server10 ~]$ scp /etc/hosts user2@server20:/tmp
```

Check the manual pages of the *scp* command for more details and usage examples.

Transferring Files Remotely Using *sftp*

The *sftp* command is an interactive file transfer tool that can be used instead of *scp*. This tool can be launched as follows on *server10* to connect to *server20*:

```
[user1@server10 ~]$ sftp server20
Connected to server20.
sftp> ?
```

Type **?** at the prompt to list available commands along with a short description:

```
sftp> ?
Available commands:
bye
cd path
chgrp grp path
chmod mode path
chown own path
df [-hi] [path]

exit
get [-afPpRr] remote [local]
reget [-fFpRr] remote [local]
reput [-fFpRr] [local] remote
help
lcd path
lls [ls-options [path]]
lmkdir path
ln [-s] oldpath newpath
lpwd
ls [-1afhlnrSt] [path]
lumask umask
mkdir path
progress
put [-afPpRr] local [remote]
pwd
quit
rename oldpath newpath
rm path
rmdir path
symlink oldpath newpath
version
!command
!
?

Quit sftp
Change remote directory to 'path'
Change group of file 'path' to 'grp'
Change permissions of file 'path' to 'mode'
Change owner of file 'path' to 'own'
Display statistics for current directory or
filesystem containing 'path'
Quit sftp
Download file
Resume download file
Resume upload file
Display this help text
Change local directory to 'path'
Display local directory listing
Create local directory
Link remote file (-s for symlink)
Print local working directory
Display remote directory listing
Set local umask to 'umask'
Create remote directory
Toggle display of progress meter
Upload file
Display remote working directory
Quit sftp
Rename remote file
Delete remote file
Remove remote directory
Symlink remote file
Show SFTP version
Execute 'command' in local shell
Escape to local shell
Synonym for help
```

As shown in the above screenshot, there are many common commands available at the sftp> prompt. These include *cd* to change directory, *get/put* to download/upload a file, *ls* to list files, *pwd* to print working directory, *mkdir* to create a directory, *rename* to rename a file, *rm* to remove a file, and *bye/quit/exit* to exit the program and return to the command prompt. These commands will run on the remote server (*server20*). The following screenshot shows how these commands are used:

Furthermore, there are four commands beginning with an ‘l’—*/cd*, */ls*, */pwd*, and */mkdir*—at the *sftp>* prompt. These commands are intended to be run on the source server (*server10*). Other Linux commands are also available at the *sftp>* prompt that you may use for basic file management operations on the remote server.

Type *quit* at the *sftp>* prompt to exit the program when you’re done.

You may use either *sftp* or *scp* for transferring files depending on your comfort level. Consult the manual pages of the commands for options and additional details.

Synchronizing Files Remotely Using *rsync*

The *rsync* (*remote synchronization*) program works in a manner similar to the *cp*, *rcp*, and *scp* commands to copy files between the source and destination. With *rsync*, the source and destination could be on the same system or different systems. The first initiation of *rsync* copies all files from the source to the destination with subsequent executions copy only the updated files. The *rsync* command uses the ssh protocol by default.

The following examples explain the usage of the program and introduce some common flags.

To copy a single file such as *grub.conf* to */tmp* on the same system:

```
[user1@server10 ~] $ sudo rsync -av /boot/grub2/grub.cfg /tmp
sending incremental file list
grub.cfg

sent 5,126 bytes  received 35 bytes  10,322.00 bytes/sec
total size is 5,032  speedup is 0.98
```

The *-a* option in the above example instructs the command to perform an archive operation and preserve all file attributes such as permissions, ownership, symlinks, and timestamps. The *-v* switch is used for verbosity.

The actual size of the *grub.cfg* file is 5,032 bytes. The additional bytes sent (5,126 minus 5,032 = 94 bytes) contain the metadata and other overhead, and the received bytes signify the metadata received. The output displays the files being copied and the file transfer rate as well.

Subsequent invocations of the above would produce an output similar to the following if the file has not been modified:

```
[user1@server10 ~]$ sudo rsync -avz /boot/grub2/grub.cfg /tmp
sending incremental file list

sent 47 bytes received 12 bytes 118.00 bytes/sec
total size is 5,032 speedup is 85.29
```

It shows no filenames under the file list, as there was no transfer occurred.

To copy */etc/rsyslog.conf* to */tmp* to *server20* using in-transit compression (-z) and displaying the transfer progress (-P):

```
[user1@server10 ~]$ rsync -avPz /etc/rsyslog.conf server20:/tmp
sending incremental file list
rsyslog.conf
      3,185 100%    0.00kB/s   0:00:00 (xfr#1, to-chk=0/1)

sent 1,478 bytes received 35 bytes 3,026.00 bytes/sec
total size is 3,185 speedup is 2.11
```

To copy the entire */home/user1* directory recursively (-r) from *server20* to */tmp/trans* directory on *server10* (create */tmp/trans* before running the *rsync* command):

```
[user1@server10 ~]$ mkdir /tmp/trans
[user1@server10 ~]$ rsync -aPvzr server20:/home/user1 /tmp/trans
receiving incremental file list
user1/
user1/.ICEauthority
      1,244 100%    1.19MB/s   0:00:00 (xfr#1, to-chk=150/152)
user1/.bash_history
      19,437 100%   18.54MB/s   0:00:00 (xfr#2, to-chk=149/152)
user1/.bash_logout
        18 100%    2.20kB/s   0:00:00 (xfr#3, to-chk=148/152)
user1/.bash_profile
       141 100%   10.59kB/s   0:00:00 (xfr#4, to-chk=147/152)
.....
sent 1,394 bytes received 897,769 bytes 599,442.00 bytes/sec
total size is 8,639,186 speedup is 9.61
```

The *rsync* command is fast and versatile, and has numerous other options available. Refer to the command's manual pages for a description of options and usage examples.

Chapter Summary

This chapter discussed the open source version of the secure shell service. It started with an overview of the service, and described what it is, how it works, available versions, and algorithms employed. We skimmed through various encryption techniques and authentication methods. We touched upon the service daemon, client and server configuration files, and commands. We demonstrated accessing a lab server from another lab server. We generated

and distributed password-less private/public key pair and employed ssh utilities to remote execute commands and transfer files.

Lastly, we examined a program that may be put into action to keep files synchronized between two systems over an ssh channel.

Review Questions

1. What is the secure equivalent for the *rcp* command?
2. What would the command *ssh-keygen -N “”* do?
3. The primary secure shell server configuration file is *ssh_config*. True or False?
4. Which three common algorithms are used with SSH version 2 for encryption and/or authentication?
5. What is the secure shell equivalent for the *telnet* command?
6. True or False? By default, the *root* user can directly log on to a RHEL system.
7. What is the default location to store user SSH keys?
8. What would the command *ssh-copy-id* do?
9. What is the *rsync* command used for?
10. Which two of the five authentication methods mentioned in this chapter are more prevalent?
11. What is the use of the *ssh-keygen* command?
12. Name the default algorithm used with SSH.
13. What kind of information does the *~/.ssh/known_hosts* file store?
14. List the two encryption techniques described in this chapter.
15. What is the default port used by the secure shell service?
16. Which log file stores authentication messages?
17. The *ssh* tool provides a non-secure tunnel over a network for accessing a RHEL system. True or False?
18. Name the SSH client-side configuration file.
19. What would the command *ssh server10 ls* do?

Answers to Review Questions

1. The *scp* command is the secure equivalent for the *rcp* command.
2. The command provided will generate a password-less ssh key pair using the default RSA algorithm.
3. False. The primary secure shell configuration file is *sshd_config*.
4. The SSH version 2 uses RSA, DSA, and ECDSA algorithms.
5. The secure equivalent for *telnet* is the *ssh* command.
6. True.
7. Under the *~/.ssh* directory.

8. The *ssh-copy-id* command is used to distribute the public key to remote systems.
9. The *rsync* command is used to maintain a copy of source files at remote location.
10. The public key-based and password-based authentication methods are more prevalent.
11. The *ssh-keygen* command is used to generate public/private key combination for use with ssh.
12. The default algorithm used with ssh is RSA.
13. The *~/.ssh/known_hosts* file stores fingerprints of remote servers.
14. The two encryption techniques are symmetric (secret key) and asymmetric (public key).
15. The default port used by the secure shell service is 22.
16. The */var/log/secure* file stores authentication messages.
17. False. The *ssh* command provides a secure tunnel over a network.
18. The client-side SSH configuration file is *ssh_config* and it is located in the */etc/ssh* directory.
19. The command provided will run the */s* command on the specified remote ssh server without the need for the user to log in.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 19-1: Establish Key-Based Authentication

As *user1* with *sudo* on *server10* and *server20*, create user account *user20* and assign a password. As *user20* on *server10*, generate a private/public key pair without a passphrase using the *ssh-keygen* command. Distribute the public key to *server20* with the *ssh-copy-id* command. Log on to *server20* as *user20* and accept the fingerprints for the server if presented. On subsequent log in attempts from *server10* to *server20*, *user20* should not be prompted for their password. (Hint: System Access and File Transfer).

Lab 19-2: Test the Effect of PermitRootLogin Directive

As *user1* with *sudo* on *server20*, edit the */etc/ssh/sshd_config* file and change the value of the directive *PermitRootLogin* to “no”. Use the *systemctl* command to activate the change. As *root* on *server10*, run **ssh server20** (or its IP address). You’ll get permission denied message. Reverse the change on *server20* and retry **ssh server20**. You should be able to log in. (Hint: The OpenSSH Service).

Chapter 20

The Linux Firewall

This chapter describes the following major topics:

- Describe Linux firewall for host-based security control
- Overview of the firewalld service
- Understand the concepts of firewalld zones and services
- Analyze zone and service configuration files
- Control access to network services and ports through firewalld
- Use firewall-cmd command to manage firewall rules

RHCSA Objectives:

50. Restrict network access using firewall-cmd/firewall
55. Configure firewall settings using firewall-cmd/firewalld

Running a system in a networked or an Internet-facing environment requires that some security measures be taken to tighten access to the system in general and individual services in particular. This can be accomplished by implementing a firewall and restricting inbound traffic to allowed ports from valid source IP addresses only. We discuss a host-based firewall solution in this chapter.

Firewall Overview

A *firewall* is a protective layer implemented at the network or server level to secure, monitor, and control inbound and outbound traffic flow. Firewalls employed at the network level use either dedicated hardware or sophisticated software appliances to form a shield around the network. Server level firewalls are referred to as *host-based firewalls* and they run in a computer operating system to monitor and manage traffic in and out. Firewalls defend a network or an individual server from undesired traffic.

RHEL is shipped with a host-based firewall solution that works by filtering data packets. A data packet is formed as a result of a process called *encapsulation* whereby the header information is attached to a message (called *payload*) during packet formation. The header includes information such as source and destination IP addresses, port, and type of data. Based on predefined *rules*, a firewall intercepts each inbound and outbound data packet, inspects its header, and decides whether to allow the packet to pass through.

Ports are defined in the `/etc/services` file for common network services that are standardized across all network operating systems, including RHEL. Some common services and the ports they listen on are FTP (*File Transfer Protocol*) on port 21, SSH (*Secure Shell*) 22, Postfix (an email service) 25, HTTP (*HyperText Transfer Protocol*) 80, and NTP (*Network Time Protocol*) on port 123.

The host-based firewall solution employed in RHEL uses a kernel module called *netfilter* together with a filtering and packet classification framework called *nftables* for policing the traffic movement. It also supports other advanced features such as *Network Address Translation* (NAT) and *port forwarding*. This firewall solution inspects, modifies, drops, or routes incoming, outgoing, and forwarded network packets based on defined rulesets.

Overview of firewalld

firewalld is the default host-based firewall management service in RHEL 8. One of the major advantages is its ability to add, modify, or delete firewall rules immediately without disrupting current network connections or restarting the service process. This is especially useful in testing and troubleshooting scenarios. *firewalld* also allows to save rules persistently so that they are activated automatically at system reboots.

The *firewalld* service lets you perform management operations at the command line using the *firewall-cmd* command, graphically using the web console, or manually by editing rules files. *firewalld* stores the default rules in files located in the */usr/lib/firewalld* directory, and those that contain custom rules in the */etc/firewalld* directory. The default rules files may be copied to the custom rules directory and modified.

firewalld Zones

firewalld uses the concept of *zones* for easier and transparent traffic management. Zones define policies based on the trust level of network connections and source IP addresses. A network connection can be part of only one zone at a time; however, a zone can have multiple network connections assigned to it. Zone configuration may include services, ports, and protocols that may be open or closed. It may also include rules for advanced configuration items such as masquerading, port forwarding, NATting, ICMP filters, and rich language. Rules for each zone are defined and manipulated independent of other zones.

firewalld inspects each incoming packet to determine the source IP address and applies the rules of the zone that has a match for the address. In the event no zone configuration matches the address, it associates the packet with the zone that has the network connection defined, and applies the rules of that zone. If neither works, *firewalld* associates the packet with the default zone, and enforces the

rules of the default zone on the packet.

The *firewalld* software installs several predefined zone files that may be selected or customized. These files include templates for traffic that must be blocked or dropped, and for traffic that is public-facing, internal, external, home, public, trusted, and work-related. Of these, the *public* zone is the default zone, and it is activated by default when the *firewalld* service is started. [Table 20-1](#) lists and describes the predefined zones sorted based on the trust level from trusted to untrusted.

Zone	Description
trusted	Allow all incoming
internal	Reject all incoming traffic except for what is allowed. Intended for use on internal networks.
home	Reject all incoming traffic except for what is allowed. Intended for use in homes.
work	Reject all incoming traffic except for what is allowed. Intended for use at workplaces.
dmz	Reject all incoming traffic except for what is allowed. Intended for use in publicly-accessible demilitarized zones.
external	Reject all incoming traffic except for what is allowed. Outgoing traffic forwarded through this zone is masqueraded to look originated from the IPv4 address of an outgoing network interface. Intended for use on external networks with masquerading.
public	Reject all incoming traffic except for what is allowed. It is the default zone for any newly added network interfaces. Intended for use in public places.
block	Reject all incoming traffic with icmp-host-prohibited message returned. Intended for use in secure places.
drop	Drop all incoming traffic without responding with ICMP error message. Intended for use in highly secure places.

Table 20-1 firewalld Default Zones

For all the predefined zones, outgoing traffic is allowed by default.

Zone Configuration Files

`firewalld` stores zone rules in XML format at two locations: the system-defined rules in the `/usr/lib/firewalld/zones` directory, and the user-defined rules in the `/etc/firewalld/zones` directory. The files at the former location can be used as templates for adding new rules, or applied instantly to any available network connection. A system zone configuration file is automatically copied to the `/etc/firewalld/zones` directory if it is modified with a management tool.

Alternatively, you can copy the required zone file to the `/etc/firewalld/zones` directory manually, and make the necessary changes. The `firewalld` service reads the files saved in this location, and applies the rules defined in them. A listing of the system zone files is presented below:

```
[user1@server10 ~]$ ls -l /usr/lib/firewalld/zones/
total 36
-rw-r--r--. 1 root root 299 Jan 14 2019 block.xml
-rw-r--r--. 1 root root 293 Jan 14 2019 dmz.xml
-rw-r--r--. 1 root root 291 Jan 14 2019 drop.xml
-rw-r--r--. 1 root root 304 Jan 14 2019 external.xml
-rw-r--r--. 1 root root 397 Jan 14 2019 home.xml
-rw-r--r--. 1 root root 412 Jan 14 2019 internal.xml
-rw-r--r--. 1 root root 343 Jan 14 2019 public.xml
-rw-r--r--. 1 root root 162 Jan 14 2019 trusted.xml
-rw-r--r--. 1 root root 339 Jan 14 2019 work.xml
```

The default *public* zone file is displayed below:

```
[user1@server10 ~]$ cat /usr/lib/firewalld/zones/public.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on
networks to not harm your computer. Only selected incoming connections are accep
ted.</description>
  <service name="ssh"/>
  <service name="dhcpcv6-client"/>
  <service name="cockpit"/>
</zone>
```

As depicted in the screenshot, the zone has a name and description, and it contains a list of all the allowed services—*ssh*, *dhcpcv6-client*, and *cockpit*. See the manual pages for *firewalld.zone* for details on zone configuration files.

firewalld Services

In addition to the concept of zones, *firewalld* also uses the idea of *services* for easier activation and deactivation of specific rules. *firewalld* services are preconfigured firewall rules delineated for various services and stored in different files. The rules consist of necessary settings, such as the port number, protocol, and possibly helper modules, to support the loading of the service. *firewalld* services can be added to a zone. By default, *firewalld* blocks all traffic unless a service or port is explicitly opened.

Service Configuration Files

firewalld stores service rules in XML format at two locations: the system-defined rules in the */usr/lib/firewalld/services* directory, and the user-defined rules in the */etc/firewalld/services* directory. The files at the former location can be used as templates for adding new service rules, or activated instantly. A system service configuration file is automatically copied to the */etc/firewalld/services* directory if it is modified with a management tool. Alternatively, you can copy the required service file to the */etc/firewalld/services* directory manually, and make the necessary changes.

The *firewalld* service reads the files saved in this location, and applies the rules defined in them. A listing of the system service files is presented below:

```
[user1@server10 ~]$ ls -l /usr/lib/firewalld/services/
total 616
-rw-r--r--. 1 root root 412 Jan 14 2019 amanda-client.xml
-rw-r--r--. 1 root root 447 Jan 14 2019 amanda-k5-client.xml
-rw-r--r--. 1 root root 283 Jan 14 2019 amqps.xml
-rw-r--r--. 1 root root 273 Jan 14 2019 amqp.xml
-rw-r--r--. 1 root root 285 Jan 14 2019 apcupsd.xml
-rw-r--r--. 1 root root 301 Jan 14 2019 audit.xml
-rw-r--r--. 1 root root 320 Jan 14 2019 bacula-client.xml
-rw-r--r--. 1 root root 346 Jan 14 2019 bacula.xml
-rw-r--r--. 1 root root 339 Jan 14 2019 bgp.xml
-rw-r--r--. 1 root root 275 Jan 14 2019 bitcoin-rpc.xml
-rw-r--r--. 1 root root 307 Jan 14 2019 bitcoin-testnet-rpc.xml
-rw-r--r--. 1 root root 281 Jan 14 2019 bitcoin-testnet.xml
-rw-r--r--. 1 root root 244 Jan 14 2019 bitcoin.xml
-rw-r--r--. 1 root root 294 Jan 14 2019 ceph-mon.xml
-rw-r--r--. 1 root root 329 Jan 14 2019 ceph.xml
-rw-r--r--. 1 root root 168 Jan 14 2019 cfengine.xml
-rw-r--r--. 1 root root 211 Jan 14 2019 cockpit.xml
....
```

The following shows the content of the *ssh* service file:

```
[user1@server10 ~]$ cat /usr/lib/firewalld/services/ssh.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
    <short>SSH</short>
    <description>Secure Shell (SSH) is a protocol for logging into and executing commands on remote machines. It provides secure encrypted communications. If you plan on accessing your machine remotely via SSH over a firewalled interface, enable this option. You need the openssh-server package installed for this option to be useful.</description>
    <port protocol="tcp" port="22"/>
</service>
```

As depicted in the screenshot, the service has a name and description, and it defines the port and protocol for the service. See the manual pages for *firewalld.service* for details on service configuration files.

Firewall Management

Managing the *firewalld* service involves a number of operations, such as listing, querying, adding, changing, and removing zones, services, ports, IP sources, and network connections. There are three methods available in RHEL 8 to perform the management tasks. They include the *firewall-cmd* command for those who prefer to work at the command line and the web interface for graphical administration. The third management option is to make use of the zone and service templates, and edit them as desired. We use the command line method in this book.

The `firewall-cmd` Command

The `firewall-cmd` command is a powerful tool to manage the `firewalld` service at the command prompt. This tool can be used to add or remove rules from the runtime configuration, or save any modifications to service configuration for persistence. It supports numerous options for the management of zones, services, ports, connections, and so on; [Table 20-2](#) lists and describes the common options only.

Option	Description
General	
--state	Displays the running status of firewalld
--reload	Reloads firewall rules from zone files. All run changes are lost.
--permanent	Stores a change persistently. The change only becomes active after a service reload or restart.
Zones	
--get-default-zone	Shows the name of the default/active zone
--set-default-zone	Changes the default zone for both runtime and permanent configuration
--get-zones	Prints a list of available zones
--get-active-zones	Displays the active zone and the assigned interfaces
--list-all	Lists all settings for a zone
--list-all-zones	Lists the settings for all available zones
--zone	Specifies the name of the zone to work on. Without this option, the default zone is used.
Services	
--get-services	Prints predefined services
--list-services	Lists services for a zone
--add-service	Adds a service to a zone
--remove-service	Removes a service from a zone
--query-service	Queries for the presence of a service
Ports	
--list-ports	Lists network ports
--add-port	Adds a port or a range of ports to a zone
--remove-port	Removes a port from a zone
--query-port	Queries for the presence of a port
Network Connections	
--list-interfaces	Lists network connections assigned to a zone
--add-interface	Binds a network connection to a zone
--change-interface	Changes the binding of a network connection

	different zone
--remove-interface	Unbinds a network connection from a zone
IP Sources	
--list-sources	Lists IP sources assigned to a zone
--add-source	Adds an IP source to a zone
--change-source	Changes an IP source
--remove-source	Removes an IP source from a zone

Table 20-2 Common firewall-cmd Options

With all the --add and --remove options, the --permanent switch may be specified to ensure the rule is stored in the zone configuration file under the `/etc/firewalld/zones` directory for persistence. Some of the options from [Table 20-2](#) are used in the upcoming exercises; the rest are beyond the scope of this book. Consult the manual pages of the command for details on the usage of these and other options.

Querying the Operational Status of firewalld

You can check the running status of the `firewalld` service using either the `systemctl` or the `firewall-cmd` command. Both commands will produce different outputs, but the intent here is to ensure the service is in the running state.

```
[user1@server10 ~]$ sudo firewall-cmd --state
running
[user1@server10 ~]$ sudo systemctl status firewalld -l --no-pager
* firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pr
eset: enabled)
    Active: active (running) since Mon 2019-11-25 22:47:22 EST; 1min 0s ago
      Docs: man:firewalld(1)
     Main PID: 15403 (firewalld)
        Tasks: 2 (limit: 5076)
       Memory: 25.1M
      CGroup: /system.slice/firewalld.service
              └─15403 /usr/libexec/platform-python -s /usr/sbin/firewalld --nofork
                --nopid

Nov 25 22:47:21 server10.example.com systemd[1]: Starting firewalld - dynamic fi
rewall daemon...
Nov 25 22:47:22 server10.example.com systemd[1]: Started firewalld - dynamic fir
ewall daemon.
```

The output indicates that the `firewalld` service is in the running state on `server10`. The other command outcome also reports that the service is marked for autostart at system reboots. In case `firewalld` is not enabled or is inactive,

issue **sudo systemctl --now enable firewalld** to start it immediately, and mark it for autostart on reboots.

You are ready to perform the exercises presented next.

Exercise 20-1: Add Services and Ports, and Manage Zones

This exercise should be done on *server10* as *user1* with *sudo* where required.

In this exercise, you will determine the current active zone. You will add and activate a permanent rule to allow HTTP traffic on port 80, and then add a runtime rule for traffic intended for TCP port 443 (the HTTPS service). You will add a permanent rule to the *internal* zone for TCP port range 5901 to 5910. You will confirm the changes and display the contents of the affected zone files. Lastly, you will switch the default zone to the *internal* zone and activate it.

1. Determine the name of the current default zone:

```
[user1@server10 ~]$ sudo firewall-cmd --get-default-zone  
public
```

2. Add a permanent rule to allow HTTP traffic on its default port:

```
[user1@server10 ~]$ sudo firewall-cmd --permanent --add-service http  
success
```

The command made a copy of the *public.xml* file from */usr/lib/firewalld/zones* directory into the */etc/firewalld/zones* directory, and added the rule for the HTTP service.

3. Activate the new rule:

```
[user1@server10 ~]$ sudo firewall-cmd --reload  
success
```

4. Confirm the activation of the new rule:

```
[user1@server10 ~]$ sudo firewall-cmd --list-services  
cockpit dhcpcv6-client http ssh
```

5. Display the content of the default zone file to confirm the addition of the permanent rule:

```
[user1@server10 ~]$ sudo cat /etc/firewalld/zones/public.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.</description>
  <service name="ssh"/>
  <service name="dhcpv6-client"/>
  <service name="cockpit"/>
  <service name="http"/>
</zone>
```

6. Add a runtime rule to allow traffic on TCP port 443 and verify:

```
[user1@server10 ~]$ sudo firewall-cmd --add-port 443/tcp
success
[user1@server10 ~]$ sudo firewall-cmd --list-ports
443/tcp
```

7. Add a permanent rule to the *internal* zone for TCP port range 5901 to 5910:

```
[user1@server10 ~]$ sudo firewall-cmd --add-port 5901-5910/tcp --permanent --zone internal
success
```

8. Display the content of the *internal* zone file to confirm the addition of the permanent rule:

```
[user1@server10 ~]$ sudo cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other computers on the net works to not harm your computer. Only selected incoming connections are accepted.</description>
  <service name="ssh"/>
  <service name="mdns"/>
  <service name="samba-client"/>
  <service name="dhcpv6-client"/>
  <service name="cockpit"/>
  <port port="5901-5910" protocol="tcp"/>
</zone>
```

The *firewall-cmd* command makes a backup of the affected zone file with a *.old* extension whenever an update is made to a zone.

9. Switch the default zone to internal and confirm:

```
[user1@server10 ~]$ sudo firewall-cmd --set-default-zone internal
success
[user1@server10 ~]$ sudo firewall-cmd --get-default-zone
internal
```

10. Activate the rules defined in the *internal* zone and list the port range added earlier:

```
[user1@server10 ~]$ sudo firewall-cmd --reload
success
[user1@server10 ~]$ sudo firewall-cmd --list-ports
5901-5910/tcp
```

This completes the exercise.

Exercise 20-2: Remove Services and Ports, and Manage Zones

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will remove the two permanent rules that were added in [Exercise 20-1](#). You will switch the `public` zone back as the default zone, and confirm the changes.

1. Remove the permanent rule for HTTP from the `public` zone:

```
[user1@server10 ~]$ sudo firewall-cmd --remove-service=http --zone public --permanent  
success
```

Notice the equal sign (=) used with the `--remove-service` option. The `firewall-cmd` command supports the specification of add, remove, change, and zone options with and without an equal sign (=). The `--zone` option is used to specify the `public` zone as it is currently the non-default.

2. Remove the permanent rule for ports 5901 to 5910 from the `internal` zone:

```
[user1@server10 ~]$ sudo firewall-cmd --remove-port 5901-5910/tcp --permanent  
success
```

The `--zone` option is not used, as ‘internal’ is currently the default zone.

3. Switch the default zone to `public` and validate:

```
[user1@server10 ~]$ sudo firewall-cmd --set-default-zone=public  
success  
[user1@server10 ~]$ sudo firewall-cmd --get-default-zone  
public
```

4. Activate the `public` zone rules, and list the current services:

```
[user1@server10 ~]$ sudo firewall-cmd --reload  
success  
[user1@server10 ~]$  
[user1@server10 ~]$ sudo firewall-cmd --list-services  
cockpit dhcpcv6-client ssh
```

The `public` zone reflects the removal of the `http` service. This concludes the exercise.

Exercise 20-3: Test the Effect of Firewall Rule

This exercise should be done on `server10` and `server20` as `user1` with `sudo` where required.

In this exercise, you will remove the `sshd` service rule from the runtime configuration on `server10`, and try to access the server from `server20` using the `ssh` command.

1. Remove the rule for the `sshd` service on `server10`:

```
[user1@server10 ~] $ sudo firewall-cmd --remove-service ssh  
success
```

2. Issue the `ssh` command on `server20` to access `server10`:

```
[user1@server20 ~] $ ssh server10  
ssh: connect to host server10 port 22: No route to host
```

The error displayed is because the firewall on `server10` blocked the access. Put the rule back on `server10` and try to access it from `server20` again:

3. Add the rule back for `sshd` on `server10`:

```
[user1@server10 ~] $ sudo firewall-cmd --add-service=ssh  
success
```

4. Issue the `ssh` command on `server20` to access `server10`. Enter “yes” if prompted and the password for `user1`.

```
[user1@server20 ~] $ ssh server10  
The authenticity of host 'server10 (192.168.0.110)' can't be established.  
ECDSA key fingerprint is SHA256:UavWZeF2HXWcfDrPUsDnb8uHT04/uQ2pWJBI6LVGDOQ.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'server10,192.168.0.110' (ECDSA) to the list of known hosts.  
user1@server10's password:  
Activate the web console with: systemctl enable --now cockpit.socket  
  
Last login: Sat Nov 30 20:21:41 2019 from 192.168.0.13  
[user1@server10 ~] $
```

This brings the exercise to an end.

Chapter Summary

We discussed a native host-based firewall solution for system protection in this chapter. We explored the concepts around firewall and described how it works. We looked at the `firewalld` service and examined the concepts of zones and services. We reviewed predefined zones and services, and analyzed their configuration files. We studied the lone firewall management command and reviewed options for listing and administering zones, services, ports, network connections, and source IP addresses.

We learned how to change and check the operational state of the firewalld service. We performed exercises to add and remove services and ports persistently and non-persistently, and manage zones. Finally, we tested the effect of deleting a port from the firewall configuration and adding it back.

Review Questions

1. A firewall can be configured between two networks but not between two hosts. True or False?
2. Which directory stores the configuration file for modified firewalld zones?
3. After changing the default firewalld zone to internal, what would you run to verify?
4. What is the process of data packet formation called?
5. What would the command `firewall-cmd --permanent --add-service=nfs --zone=external` do?
6. If you have a set of firewall rules defined for a service stored under both `/etc/firewalld` and `/usr/lib/firewalld` directories, which of the two sets will take precedence?
7. What is the kernel module in RHEL 8 that implements the host-level protection called?
8. firewalld is the firewall management solution in RHEL 8. True or False?
9. Name the default firewalld zone.
10. What is the purpose of firewalld service configuration files?
11. What would the command `firewall-cmd --remove-port=5000/tcp` do?
12. What is the primary command line tool for managing firewalld called?

Answers to Review Questions

1. False. A firewall can also be configured between two host computers.
2. The modified `firewalld` zone files are stored under `/etc/firewalld/zones` directory.
3. You run `firewall-cmd --get-default-zone` for validation.
4. The process of data packet formation is called encapsulation.
5. The command provided will add the `nfs` service to `external` firewalld zone persistently.
6. The ruleset located in the `/etc/firewalld` directory will have precedence.
7. The kernel module that implements the host-level protection is called netfilter.
8. True.
9. The default firewalld zone is the `public` zone.
10. firewalld service configuration files store service-specific port, protocol, and other details, which makes it easy to activate and deactivate them.

11. The command provided will remove the runtime firewall rule for TCP port 5000.
12. The primary command line tool for managing *firewalld* is called *firewall-cmd*.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 20-1: Add Service to Firewall

As *user1* with *sudo* on *server10*, add and activate a permanent rule for HTTPs traffic to the default zone. Confirm the change by viewing the zone's XML file and running the *firewall-cmd* command. (Hint: Firewall Management).

Lab 20-2: Add Port Range to Firewall

As *user1* with *sudo* on *server10*, add and activate a permanent rule for the UDP port range 8000 to 8005 to the *trusted* zone. Confirm the change by viewing the zone's XML file and running the *firewall-cmd* command. (Hint: Firewall Management).

Chapter 21

Security Enhanced Linux

This chapter describes the following major topics:

- Describe Security Enhanced Linux and its terminology
- Understand SELinux contexts for users, processes, files, and ports
- Copy, move, and archive files with and without SELinux context
- How domain transitioning works
- Overview of SELinux Booleans
- Query and manage SELinux via management tools
- Modify SELinux contexts for files and ports
- Add SELinux rules to policy database
- View and analyze SELinux alerts

RHCSA Objectives:

58. Set enforcing and permissive modes for SELinux
59. List and identify SELinux file and process context
60. Restore default file contexts
61. Use Boolean settings to modify system SELinux settings
62. Diagnose and address routine SELinux policy violations

Security

Enhanced Linux is a mechanism that controls who can access and do what on the system. It is part and parcel of the Linux kernel. It handles access beyond what the traditional access control system delivers including file and directory permissions, user and group-level permissions, shadow password and password aging mechanisms, and ACLs. The goal of SELinux is to limit the possible damage that could occur to the system due to unauthorized user or program access. This chapter covers SELinux in reasonable detail.

Security Enhanced Linux

Security Enhanced Linux (SELinux) is an implementation of the *Mandatory Access Control* (MAC) architecture developed by the U.S. *National Security Agency* (NSA) in collaboration with other organizations and the Linux community for flexible, enriched, and granular security controls in Linux. MAC is integrated into the Linux kernel as a set of patches using the *Linux Security Modules* (LSM) framework that allows the kernel to support various security implementations, including SELinux.

MAC provides an added layer of protection above and beyond the standard Linux *Discretionary Access Control* (DAC) security architecture. DAC includes the traditional file and directory permissions, ACLs, extended attribute settings, setuid/setgid bits, su/sudo privileges, and other controls. MAC limits the ability of a *subject* (Linux user or process) to access an *object* (file, directory, file system, device, network interface/connection, port, pipe, socket, etc.) to reduce or eliminate the potential damage the subject may be able to inflict on the system if compromised due to the exploitation of vulnerabilities in services, programs, or applications.

MAC controls are fine-grained; they protect other services in the event one service is negotiated. For instance, if the HTTP service process is compromised, the attacker can only damage the files the hacked process will have access to, and not the other processes running on the system, or the objects the other processes will have access to. To ensure this coarse-grained control, MAC uses a set of defined authorization rules called *policy* to examine security attributes associated with subjects and objects when a subject tries to access an object, and decides whether to permit the access attempt. These attributes are stored in *contexts* (a.k.a. *labels*), and are applied to both subjects and objects.

SELinux decisions are stored in a special cache area called *Access Vector Cache* (AVC). This cache area is checked for each access attempt by a process to determine whether the access attempt was previously allowed. With this mechanism in place, SELinux does not have to check the policy ruleset repeatedly, thus improving performance.

By default, SELinux controls are enabled at the time of RHEL installation with the default configuration, which confines the processes to the bare minimum privileges that they need to function.

Terminology

To comprehend SELinux, an understanding of some key terms is essential. These terms are useful in explaining the concepts and SELinux functionality in the remainder of this chapter.

Subject

A *subject* is any user or process that accesses an object. Examples include `system_u` for the SELinux system user, and `unconfined_u` for subjects that are not bound by the SELinux policy. The subject is stored in field 1 of the context.

Object

An *object* is a resource, such as a file, directory, hardware device, network interface/connection, port, pipe, or socket, that a subject accesses. Examples include `object_r` for general objects, `system_r` for system-owned objects, and `unconfined_r` for objects that are not bound by the SELinux policy.

Access

An *access* is an action performed by the subject on an object. Examples include creating, reading, or updating a file, creating or navigating a directory, and accessing a network port or socket.

Policy

A *policy* is a defined ruleset that is enforced system-wide, and is used to analyze security attributes assigned to subjects and objects. This ruleset is referenced to decide whether to permit a subject's access attempt to an object, or a subject's attempt to interact with another subject. The default behavior of SELinux in the absence of a rule is to deny the access. Two standard preconfigured policies are `targeted` and `m/s` with `targeted` being the default.

The targeted policy dictates that any process that is targeted runs in a confined domain, and any process that is not targeted runs in an unconfined domain. For instance, SELinux runs logged-in users in the unconfined

domain, and the *httpd* process in a confined domain by default. Any subject running unconfined is more vulnerable than the one running confined.

The *mls* policy places tight security controls at deeper levels.

A third preconfigured policy called *minimum* is a light version of the targeted policy, and it is designed to protect only selected processes.

Context

A *context* (a.k.a. *label*) is a tag to store security attributes for subjects and objects. In SELinux, every subject and object has a context assigned, which consists of a SELinux user, role, type (or domain), and sensitivity level. SELinux uses this information to make access control decisions.

Labeling

Labeling is the mapping of files with their stored contexts.

SELinux User

SELinux policy has several predefined SELinux user identities that are authorized for a particular set of roles. SELinux policy maintains Linux user to SELinux user identity mapping to place SELinux user restrictions on Linux users. This controls what roles and levels a process (with a particular SELinux user identity) can enter. A Linux user, for instance, cannot run the *su* and *sudo* commands or the programs located in their home directories if they are mapped to the SELinux user *user_u*.

Role

A *role* is an attribute of the *Role-Based Access Control* (RBAC) security model that is part of SELinux. It classifies who (subject) is allowed to access what (domains or types). SELinux users are authorized for roles, and roles are authorized for domains and types. Each subject has an associated role to ensure that the system and user processes are separated. A subject can transition into a new role to gain access to other domains and types.

Examples roles include *user_r* for ordinary users, *sysadm_r* for administrators, and *system_r* for processes that initiate under the *system_r* role. The role is stored in field 2 of the context.

Type Enforcement

Type enforcement (TE) identifies and limits a subject's ability to access domains for processes, and types for files. It references the contexts of the subjects and objects for this enforcement.

Type and Domain

A *type* is an attribute of type enforcement. It is a group of objects based on uniformity in their security requirements. Objects such as files and directories with common security requirements, are grouped within a specific type.

Examples of types include `user_home_dir_t` for objects located in user home directories, and `usr_t` for most objects stored in the `/usr` directory. The type is stored in field 3 of a file context.

A *domain* determines the type of access that a process has. Processes with common security requirements are grouped within a specific domain type, and they run confined within that domain. Examples of domains include `init_t` for the `systemd` process, `firewalld_t` for the `firewalld` process, and `unconfined_t` for all processes that are not bound by SELinux policy. The domain is stored in field 3 of a process context.

SELinux policy rules outline how types can access each other, domains can access types, and domains can access each other.

Level

A *level* is an attribute of *Multi-Level Security* (MLS) and *Multi-Category Security* (MCS). It is a pair of sensitivity:category values that defines the level of security in the context. A category may be defined as a single value or a range of values, such as `c0:c4` to represent `c0` through `c4`. In RHEL 8, the targeted policy is used as the default, which enforces MCS (MCS supports only one sensitivity level (`s0`) with 0 to 1023 different categories).

SELinux Contexts for Users

SELinux contexts define security attributes placed on subjects and objects. Each context contains a type and a security level with subject and object information. Use the `id` command with the `-Z` option to view the context set on Linux users. The following example shows the context for `user1`:

```
[user1@server10 ~]$ id -Z  
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

The output indicates that `user1` is mapped to the SELinux `unconfined_u` user, and that there are no SELinux restrictions placed on this user. You'll get the same result if you run this command for other users. This entails that all Linux users, including `root`, run unconfined by default, which gives them full access to the system.

In addition to the unconfined user with unlimited privileges, SELinux includes seven confined user identities with restricted access to objects. These accounts are mapped to Linux users via SELinux policy. This regulated

access helps safeguard the system from potential damage that Linux users might inflict on the system.

You can use the `seinfo` query command to list the SELinux users; however, the `setools-console` software package must be installed before doing so.

```
[user1@server10 ~]$ sudo seinfo -u

Users: 8
  guest_u
  root
  staff_u
  sysadm_u
  system_u
  unconfined_u
  user_u
  xguest_u
```

The output shows the eight predefined SELinux users. You can use the `semanage` command to view the mapping between Linux and SELinux users:

```
[user1@server10 ~]$ sudo semanage login -l

Login Name          SELinux User      MLS/MCS Range      Service
__default__
root               unconfined_u      s0-s0:c0.c1023    *
                  unconfined_u      s0-s0:c0.c1023    *
```

The output displays Linux users in column 1 (Login Name) and SELinux users they are mapped to in column 2 (SELinux User). Columns 3 and 4 show the associated security level (MLS/MCS Range), and the context for the Linux user (the * represents all services). By default, all non-root Linux users are represented as `__default__`, which is mapped to the `unconfined_u` user in the policy.

SELinux Contexts for Processes

You can determine the context for processes using the `ps` command with the `-Z` flag. The following example shows only the first two lines from the command output:

```
[user1@server10 ~]$ ps -eZ
LABEL                      PID TTY      TIME CMD
system_u:system_r:init_t:s0  1 ?        00:00:04 systemd
```

In the output, the subject (`system_u`) is a SELinux username (mapped to Linux user `root`), object is `system_r`, domain (`init_t`) reveals the type of protection applied to the process, and level of security (`s0`). Any process that is unprotected will run in the `unconfined_t` domain.

SELinux Contexts for Files

You can spot the context for files and directories. To this end, use the `ls` command with the `-Z` switch. The following shows the four attributes set on the `/etc/passwd` file:

```
[user1@server10 ~]$ ls -lZ /etc/passwd
-rw-r--r--. 1 root root system_u:object_r:passwd_file_t:s0 2809 Nov 11 07:24 /etc/passwd
```

The outcome indicates the subject (`system_u`), object (`object_r`), type (`passwd_file_t`), and security level (`s0`) for the `passwd` file. Contexts for system-installed and user-created files are stored in the `file_contexts` and `file_contexts.local` files located in the `/etc/selinux/targeted/contexts/files` directory. These policy files can be updated using the `semanage` command.

Copying, Moving, and Archiving Files with SELinux Contexts

As mentioned, all files in RHEL are labeled with an SELINUX security context by default. New files inherit the parent directory's context at the time of creation. However, three common file management operations—copy, move, and archive—require special attention. There are certain rules to be kept in mind during their use to ensure correct contexts on affected files. These rules are:

1. If a file is copied to a different directory, the destination file will receive the destination directory's context, unless the `--preserve=context` switch is specified with the `cp` command to retain the source file's original context.
2. If a copy operation overwrites the destination file in the same or different directory, the file being copied will receive the context of the overwritten file, unless the `--preserve=context` switch is specified with the `cp` command to preserve the source file's original context.
3. If a file is moved to the same or different directory, the SELinux context will remain intact, which may differ from the destination directory's context.
4. If a file is archived with the `tar` command, use the `--selinux` option to preserve the context.

Later in the chapter, we will perform an exercise to confirm the behavior of the three operations.

SELinux Contexts for Ports

SELinux contexts define security attributes for network ports, which can be viewed with the `semanage` command . The following illustrates a few entries from the output of this command:

```
[user1@server10 ~]$ sudo semanage port -l
SELinux Port Type          Proto  Port Number
chronyd_port_t              udp    323
dns_port_t                  tcp    53, 853
ftp_port_t                  tcp    21, 989, 990
http_port_t                 tcp    80, 81, 443, 488, 8008, 8009, 8443, 9000
ntp_port_t                  udp    123
syslogd_port_t              udp    514, 601, 20514
```

The output is displayed in three columns. Column 1 shows the SELinux type, column 2 depicts the protocol, and column 3 indicates the port number(s). By default, SELinux allows services to listen on a restricted set of network ports only. This is evident from the above output.

Domain Transitioning

SELinux allows a process running in one domain to enter another domain to execute an application that is restricted to run in that domain only, provided a rule exists in the policy to support such transition. SELinux defines a permission setting called *entrypoint* in its policy to control processes that can transition into another domain. To understand how this works, a basic example is provided below that shows what happens when a Linux user attempts to change their password using the `/usr/bin/passwd` command.

The `passwd` command is labeled with the `passwd_exec_t` type, which can be confirmed as follows:

```
[user1@server10 ~]$ ls -lZ /usr/bin/passwd
-rwsr-xr-x. 1 root root system_u:object_r:passwd_exec_t:s0 34512 Aug 12 2018 /usr/bin/passwd
```

The `passwd` command requires access to the `/etc/shadow` file in order to modify a user password. The `shadow` file has a different type set on it (`shadow_t`):

```
[user1@server10 ~]$ ls -lZ /etc/shadow
-----. 1 root root system_u:object_r:shadow_t:s0 1613 Nov 11 07:25 /etc/shadow
```

The SELinux policy has rules that specifically allow processes running in domain `passwd_t` to read and modify the files with type `shadow_t`, and allow them entrypoint permission into domain `passwd_exec_t`. This rule enables the user's shell process executing the `passwd` command to switch into the `passwd_t` domain and update the `shadow` file.

Open two terminal windows. In window 1, issue the `passwd` command as `user1` and wait at the prompt:

```
[user1@server10 ~]$ passwd  
Changing password for user user1.  
Current password: _
```

In window 2, run the `ps` command:

```
[user1@server10 ~]$ ps -eZ | grep passwd  
unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023 14188 pts/1 00:00:00 passwd
```

As you can see, the `passwd` command (process) transitioned into the `passwd_t` domain to change the user password. A process running in this domain is allowed to modify the content of the `/etc/shadow` file.

SELinux Booleans

Booleans are on/off switches that SELinux uses to determine whether to permit an action. Booleans activate or deactivate certain rule in the SELinux policy immediately and without the need to recompile or reload the policy. For instance, the `ftpd_anon_write` Boolean can be turned on to enable anonymous users to upload files. This privilege can be revoked by turning this Boolean off. Boolean values are stored in virtual files located in the `/sys/fs/selinuxBOOLEANS` directory. The filenames match the Boolean names. A sample listing of this directory is provided below:

```
[user1@server10 ~]$ ls -l /sys/fs/selinuxBOOLEANS  
total 0  
-rw-r--r--. 1 root root 0 Nov 30 20:20 abrt_anon_write  
-rw-r--r--. 1 root root 0 Nov 30 20:20 abrt_handle_event  
-rw-r--r--. 1 root root 0 Nov 30 20:20 abrt_upload_watch_anon_write  
-rw-r--r--. 1 root root 0 Nov 30 20:20 antivirus_can_scan_system  
-rw-r--r--. 1 root root 0 Nov 30 20:20 antivirus_use_jit  
-rw-r--r--. 1 root root 0 Nov 30 20:20 auditadm_exec_content  
-rw-r--r--. 1 root root 0 Nov 30 20:20 authlogin_nsswitch_use_ldap  
.....
```

On a typical server, you'll see hundreds of Boolean files in the output.

The manual pages of the Booleans are available through the `selinux-policy-doc` package. Once installed, use the `-K` option with the `man` command to bring the pages up for a specific Boolean. For instance, issue `man -K abrt_anon_write` to view the manual pages for the `abrt_anon_write` Boolean.

Boolean values can be viewed, and flipped temporarily or for permanence. The new value takes effect right away. Temporary changes are stored as a "1" or "0" in the corresponding Boolean file in the `/sys/fs/selinuxBOOLEANS` directory and permanent changes are saved in the policy database.

One of the exercises in the next section demonstrates how to display and change Boolean values.

SELinux Administration

Managing SELinux involves plentiful tasks, including controlling the activation mode, checking operational status, setting security contexts on subjects and objects, and switching Boolean values. RHEL provides a set of commands to perform these operations. These commands are available through multiple packages, such as *libselinux-utils* provides *getenforce*, *getsebool*, and *getsebool* commands, *policycoreutils* contains *sestatus*, *setsebool*, and *restorecon* commands, *policycoreutils-python-utils* provides the *semanage* command, and *setools-console* includes the *seinfo* and *seseach* commands.

For viewing alerts and debugging SELinux issues, a graphical tool called *SELinux Alert Browser* is available, which is part of the *setroubleshoot-server* package. In order to fully manage SELinux, you need to ensure that all these packages are installed on the system. Besides this toolset, there are additional utilities available to accomplish specific SELinux administration tasks, but their use is not as frequent.

Management Commands

SELinux delivers a variety of commands for effective administration. [Table 21-1](#) lists and describes the commands mentioned above plus a few more under various management categories.

Command	Description
Mode Management	
getenforce	Displays the current mode of operation
sestatus	Shows SELinux runtime status and Boolean values
setenforce	Switches the operating mode between enforcing and permissive temporarily
Context Management	
chcon	Changes context on files (changes do not survive file system relabeling)
restorecon	Restores default contexts on files by referencing the file /etc/selinux/targeted/contexts/files directory
semanage	Changes context on files with the fcontext subcommand (changes survive file system relabeling)
Policy Management	
seinfo	Provides information on policy components
semanage	Manages policy database
sesearch	Searches rules in the policy database
Boolean Management	
getsebool	Displays Booleans and their current settings
setsebool	Modifies Boolean values temporarily, or in the policy database
semanage	Modifies Boolean values in the policy database with the boolean subcommand
Troubleshooting	
sealert	The graphical troubleshooting tool

Table 21-1 SELinux Management Commands

Most of these commands are employed in this chapter.

Viewing and Controlling SELinux Operational State

One of the key configuration files that controls the SELinux operational state, and sets its default type is the *config* file located in the */etc/selinux* directory. The default content of the file are displayed below:

```
[user1@server10 ~]$ cat /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#       targeted - Targeted processes are protected,
#       minimum - Modification of targeted policy. Only selected processes are protected
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

The SELINUX directive in the file sets the activation mode for SELinux. *Enforcing* activates it and allows or denies actions based on the policy rules. *Permissive* activates SELinux, but permits all actions. It records all security violations. This mode is useful for troubleshooting and in developing or tuning the policy. The third option is to completely turn SELinux off. When running in enforcing mode, the SELINUXTYPE directive dictates the type of policy to be enforced. The default SELinux type is targeted.

Issue the `getenforce` command to determine the current operating mode:

```
[user1@server10 ~]$ getenforce
Enforcing
```

The output returns enforcing as the current active policy. You may flip the state to permissive using the `setenforce` command, and verify the change with `getenforce`:

```
[user1@server10 ~]$ sudo setenforce permissive
[user1@server10 ~]$ getenforce
Permissive
```



You may alternatively use a “0” for permissive and a “1” for enforcing.

The change takes effect at once; however, it will be lost at the next system reboot. To make it persistent, edit the `/etc/selinux/config` file and set the SELINUX directive to the desired mode.

EXAM TIP: You may switch SELinux to permissive for troubleshooting a non-functioning service. Change it back to enforcing when the issue is resolved.

To disable SELinux completely, the SELINUX directive needs to be set to disabled in the `config` file, and the system must be rebooted. Reactivation in the future to either enforcing or permissive will require the mode resetting in

the *config* file followed by a system reboot. The reboot will take longer than normal, as SELinux will go through the process of relabeling all the files.

Querying Status

The current runtime status of SELinux can be viewed with the *sestatus* command. This command also displays the location of principal directories, the policy in effect, and the activation mode.

```
[user1@server10 ~] $ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      31
```

The output reveals that SELinux is enabled (SELinux status), and it is running in permissive mode (Current mode) with the targeted policy in effect (Loaded policy name). It also indicates the current mode setting in the *config* file (Mode from config file) along with other information.

The *sestatus* command may be invoked with the *-v* switch to report on security contexts set on files and processes, as listed in the */etc/sestatus.conf* file. The default content of this file is shown below:

```
[user1@server10 ~] $ cat /etc/sestatus.conf
[files]
/etc/passwd
/etc/shadow
/bin/bash
/bin/login
/bin/sh
/sbin/agetty
/sbin/init
/sbin/mingetty
/usr/sbin/sshd
/lib/libc.so.6
/lib/ld-linux.so.2
/lib/ld.so.1

[process]
/sbin/mingetty
/sbin/agetty
/usr/sbin/sshd
```

Run the *sestatus* command with *-v*:

```
[user1@server10 ~]$ sestatus -v
SELinux status:                    enabled
SELinuxfs mount:                  /sys/fs/selinux
SELinux root directory:          /etc/selinux
Loaded policy name:                targeted
Current mode:                     enforcing
Mode from config file:           enforcing
Policy MLS status:                enabled
Policy deny_unknown status:       allowed
Memory protection checking:      actual (secure)
Max kernel policy version:        31

Process contexts:
Current context:                 unconfined_u:unconfined_r:unconfined_t:s0-s0:
c1023
Init context:                     system_u:system_r:init_t:s0

File contexts:
Controlling terminal:             unconfined_u:object_r:user_devpts_t:s0
/etc/passwd                      system_u:object_r:passwd_file_t:s0
/etc/shadow                        system_u:object_r:shadow_t:s0

....
```

With `-v` included, the command reports the contexts for the current process (Current context) and the `init` (`systemd`) process (Init context) under Process Contexts. It also reveals the file contexts for the controlling terminal and associated files under File Contexts.

Exercise 21-1: Modify SELinux File Context

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will create a directory `sedir1` under `/tmp` and a file `sefile1` under `sedir1`. You will check the context on the directory and file. You will change the SELinux user and type to `user_u` and `public_content_t` on both and verify.

1. Create the hierarchy `sedir1/sefile1` under `/tmp`:

```
[user1@server10 ~]$ cd /tmp
[user1@server10 tmp]$ mkdir sedir1
[user1@server10 tmp]$ touch sedir1/sefile1
```

2. Determine the context on the new directory and file:

```
[user1@server10 tmp]$ ls -ldZ sedir1
drwxrwxr-x. 2 user1 user1 unconfined_u:object_r:user_tmp_t:s0 21 Dec  3 11:38 sedir1
[user1@server10 tmp]$ ls -lZ sedir1/sefile1
-rw-rw-r--. 1 user1 user1 unconfined_u:object_r:user_tmp_t:s0 0 Dec  3 11:38 sedir1/sefile1
```

The directory and the file get `unconfined_u` and `user_tmp_t` as the SELinux user and type.

3. Modify the SELinux user (-u) on the directory to `user_u` and type (-t) to `public_content_t` recursively (-R) with the `chcon` command:

```
[user1@server10 tmp]$ sudo chcon -vu user_u -t public_content_t sedir1 -R
changing security context of 'sedir1/sefile1'
changing security context of 'sedir1'
```

4. Validate the new context:

```
[user1@server10 tmp]$ ls -ldZ sedir1
drwxrwxr-x. 2 user1 user1 user_u:object_r:public_content_t:s0 21 Dec  3 11:38 sedir1
[user1@server10 tmp]$ ls -lZ sedir1/sefile1
-rw-rw-r--. 1 user1 user1 user_u:object_r:public_content_t:s0 0 Dec  3 11:38 sedir1/sefile1
```

This concludes the exercise.

Exercise 21-2: Add and Apply File Context

This exercise should be done on *server10* as *user1* with *sudo* where required.

In this exercise, you will add the current context on *sedir1* to the SELinux policy database to ensure a relabeling will not reset it to its previous value (see [Exercise 21-1](#)). Next, you will change the context on the directory to some random values. You will restore the default context from the policy database back to the directory recursively.

1. Determine the current context:

```
[user1@server10 tmp]$ ls -ldZ sedir1
drwxrwxr-x. 2 user1 user1 user_u:object_r:public_content_t:s0 21 Dec  3 11:38 sedir1
[user1@server10 tmp]$ ls -lZ sedir1/sefile1
-rw-rw-r--. 1 user1 user1 user_u:object_r:public_content_t:s0 0 Dec  3 11:38 sedir1/sefile1
```

The output indicates the current SELinux user (*user_u*) and type (*public_content_t*) set on the directory and the file.

2. Add (-a) the directory recursively to the policy database using the *semanage* command with the *fcontext* subcommand:

```
[user1@server10 tmp]$ sudo semanage fcontext -a -s user_u -t public_content_t '/tmp/sendir1(/.*)?'
```

The regular expression *(/.)?* instructs the command to include all files and subdirectories under */tmp/sendir1*. This expression is needed only if recursion is required.

The above command added the context to the */etc/selinux/targeted/contexts/files/file_contexts.local* file. You can use the *cat* command to view the content.

3. Validate the addition by listing (-l) the recent changes (-C) in the policy database:

```
[user1@server10 tmp]$ sudo semanage fcontext -Cl | grep sedir
/tmp/sendir1(/.*)?          all files          user_u:object_r:public_content_t:s0
```

4. Change the current context on `sedir1` to something random (`staff_u/etc_t`) with the `chcon` command:

```
[user1@server10 tmp]$ sudo chcon -vu staff_u -t etc_t sedir1 -R  
changing security context of 'sedir1/sefile1'  
changing security context of 'sedir1'
```

5. The security context is changed successfully. Confirm with the `ls` command:

```
[user1@server10 tmp]$ ls -ldZ sedir1 ; ls -lZ sedir1/sefile1  
drwxrwxr-x. 2 user1 user1 staff_u:object_r:etc_t:s0 21 Dec 3 11:38 sedir1  
-rw-rw-r--. 1 user1 user1 staff_u:object_r:etc_t:s0 0 Dec 3 11:38 sedir1/sefile1
```

6. Reinstate the context on the `sedir1` directory recursively (-R) as stored in the policy database using the `restorecon` command:

```
[user1@server10 tmp]$ sudo restorecon -Rv sedir1  
Relabeled /tmp/sendir1 from staff_u:object_r:etc_t:s0 to staff_u:object_r:public_content_t:s0  
Relabeled /tmp/sendir1/sefile1 from staff_u:object_r:etc_t:s0 to staff_u:object_r:public_content_t:s0
```

The output confirms the restoration of the default context on the directory and the file.

EXAM TIP: Use the combination of `semanage` and `restorecon` commands to add a file context to the SELinux policy and then apply it. This will prevent the context on file to reset to the original value in the event of SELinux relabeling (disabled to enforcing/permissive).

Exercise 21-3: Add and Delete Network Ports

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will add a non-standard network port 8010 to the SELinux policy database for the `httpd` service and confirm the addition. You will then remove the port from the policy and verify the deletion.

1. List (-l) the ports for the `httpd` service as defined in the SELinux policy database:

```
[user1@server10 ~]$ sudo semanage port -l | grep ^http_port  
http_port_t          tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
```

The output reveals eight network ports the `httpd` process is currently allowed to listen on.

2. Add (-a) port 8010 with type (-t) `http_port_t` and protocol (-p) `tcp` to the policy:

```
[user1@server10 ~]$ sudo semanage port -at http_port_t -p tcp 8010
```

3. Confirm the addition:

```
[user1@server10 ~]$ sudo semanage port -l | grep ^http_port
http_port_t          tcp      8010, 80, 81, 443, 488, 8008, 8009, 8443, 9000
```

The new network port is visible in the outcome.

4. Delete (-d) port 8010 from the policy and confirm:

```
[user1@server10 ~]$ sudo semanage port -d -p tcp 8010
[user1@server10 ~]$ sudo semanage port -l | grep ^http_port
http_port_t          tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
```

The port is removed from the policy database.

EXAM TIP: Any non-standard port you want to use for any service, make certain to add it to the SELinux policy database with the correct type.

Exercise 21-4: Copy Files with and without Context

This exercise should be done on *server10* as *user1* with *sudo* where required.

In this exercise, you will create a file called *sefile2* under */tmp* and display its context. You will copy this file to the */etc/default* directory, and observe the change in the context. You will remove *sefile2* from */etc/default*, and copy it again to the same destination, ensuring that the target file receives the source file's context.

1. Create file *sefile2* under */tmp* and show context:

```
[user1@server10 ~]$ touch /tmp/sefile2
[user1@server10 ~]$ ls -lZ /tmp/sefile2
-rw-rw-r--. 1 user1 user1 unconfined_u:object_r:user_tmp_t:s0 0 Dec  3 17:54 /tmp/sefile2
```

The context on the file is *unconfined_u*, *object_r*, and *user_tmp_t*.

2. Copy this file to the */etc/default* directory, and check the context again:

```
[user1@server10 ~]$ sudo cp /tmp/sefile2 /etc/default
[user1@server10 ~]$ ls -lZ /etc/default/sefile2
-rw-r--r--. 1 root root unconfined_u:object_r:etc_t:s0 0 Dec  3 17:57 /etc/default/sefile2
```

The target file (*/etc/default/sefile2*) received the default context of the destination directory (*/etc/default*).

3. Erase the */etc/default/sefile2* file, and copy it again with the --preserve=context option:

```
[user1@server10 ~]$ sudo rm /etc/default/sefile2
[user1@server10 ~]$ sudo cp /tmp/sefile2 /etc/default --preserve=context
```

4. List the file to view the context:

```
[user1@server10 ~]$ ls -lZ /etc/default/sefile2
-rw-r--r--. 1 root root unconfined_u:object_r:user_tmp_t:s0 0 Dec  3 18:02 /etc/default/sefile2
```

The original context (`user_tmp_t`) is preserved on the target file after the copy operation has finished.

Exercise 21-5: View and Toggle SELinux Boolean Values

This exercise should be done on `server10` as `user1` with `sudo` where required.

In this exercise, you will display the current state of the Boolean `nfs_export_all_rw`. You will toggle its value temporarily, and reboot the system. You will flip its value persistently after the system has been back up.

1. Display the current setting of the Boolean `nfs_export_all_rw` using three different commands—`getsebool`, `sestatus`, and `semanage`:

```
[user1@server10 ~]$ sudo getsebool -a | grep nfs_export_all_rw
nfs_export_all_rw --> on
[user1@server10 ~]$ sudo sestatus -b | grep nfs_export_all_rw
nfs_export_all_rw
on
[user1@server10 ~]$ sudo semanage boolean -l | grep nfs_export_all_rw
nfs_export_all_rw      (on ,   on)  Allow nfs to export all rw
```

2. Turn off the value of `nfs_export_all_rw` using the `setsebool` command by simply furnishing “off” or “0” with it and confirm:

```
[user1@server10 ~]$ sudo setsebool nfs_export_all_rw 0
[user1@server10 ~]$ sudo getsebool -a | grep nfs_export_all_rw
nfs_export_all_rw --> off
```

3. Reboot the system and rerun the `getsebool` command to check the Boolean state:

```
[user1@server10 ~]$ sudo getsebool -a | grep nfs_export_all_rw
nfs_export_all_rw --> on
```

The value reverted to its previous state.

4. Set the value of the Boolean persistently (-P or -m as needed) using either of the following:

```
[user1@server10 ~]$ sudo setsebool -P nfs_export_all_rw off  
[user1@server10 ~]$ sudo semanage boolean -m -o nfs_export_all_rw
```

5. Validate the new value using the *getsebool*, *sestatus*, or *semanage* command:

```
[user1@server10 ~]$ sudo getsebool nfs_export_all_rw  
nfs_export_all_rw --> off  
[user1@server10 ~]$ sudo sestatus -b | grep nfs_export_all_rw  
nfs_export_all_rw off  
[user1@server10 ~]$ sudo semanage boolean -l | grep nfs_export_all_rw  
nfs_export_all_rw (off , off) Allow nfs to export all rw
```

The command outputs confirm the permanent change.

Monitoring and Analyzing SELinux Violations

SELinux generates alerts for system activities when it runs in enforcing or permissive mode. It writes the alerts to the */var/log/audit/audit.log* file if the *auditd* daemon is running, or to the */var/log/messages* file via the *rsyslog* daemon in the absence of *auditd*. SELinux also logs the alerts that are generated due to denial of an action, and identifies them with a type tag AVC (*Access Vector Cache*) in the *audit.log* file. It also writes the rejection in the *messages* file with a message ID, and how to view the message details.



If it works with SELinux in permissive mode and not in enforcing, something needs to be adjusted in SELinux.

SELinux denial messages are analyzed and the audit data is examined to identify the potential cause of the rejection. The results of the analysis are recorded with recommendations on how to fix it. These results can be reviewed to aid in troubleshooting, and recommended actions taken to address the issue. SELinux runs a service daemon called *setroubleshootd* that performs this analysis and examination in the background. This service also has a client interface called *SELinux Troubleshooter* (the *sealert* command) that reads the data and displays it for assessment. The client tool has both text and graphical interfaces. The server and client components are part of the *setroubleshoot-server* software package that must be installed on the system prior to using this service.

The following shows a sample allowed record in raw format from the */var/log/audit/audit.log* file:

```
type=USER_ACCT msg=audit(1575462873.869:744): pid=7496 uid=0 auid=1000 ses=2 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:accounting grantors=pam_succeed_if acct="root" exe="/usr/bin/su" hostname=server10.example.com addr=? terminal=pts/0 res=success'UID="root" AUID="user1"
```

The record shows a successful *user1* attempt to *su* into the *root* user account on *server10*.

The following is a sample denial record from the same file in raw format:

```
type=AVC msg=audit(1575464056.692:782): avc: denied { create } for pid=7677 comm="passwd" name="nshadow" scontext=unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023 tcontext=system_u:object_r:etc_t:s0 tclass=file permissive=0
```

The message has the AVC type, and it is related to the *passwd* command (comm) with source context (scontext)

“*unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023*”, and the *nshadow* file (name) with file type (tclass) “file”, and target context (tcontext) “*system_u:object_r:etc_t:s0*”. It also indicates the SELinux operating mode, which is enforcing (permissive=0). This message indicates that the */etc/shadow* file does not have the correct context set on it, and that’s why SELinux prevented the *passwd* command from updating the user’s password.

You can also use the *sealert* command to analyze (-a) all AVC records in the *audit.log* file. This command produces a formatted report with all relevant details:

```
[user1@server10 ~]$ sudo sealert -a /var/log/audit/audit.log
100% done
found 1 alerts in /var/log/audit/audit.log
-----
SELinux is preventing /usr/bin/passwd from create access on the file nshadow.

Additional Information:
Source Context           unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023
Target Context          system_u:object_r:etc_t:s0
Target Objects          nshadow [ file ]
Source                 passwd
Source Path             /usr/bin/passwd
Port                  <Unknown>
Host                  <Unknown>
Source RPM Packages   selinux-policy-3.14.1-61.el8.noarch
Target RPM Packages   None
Policy RPM             selinux-policy-3.14.1-61.el8.noarch
Selinux Enabled         True
Policy Type            targeted
Enforcing Mode         Enforcing
Host Name              server10.example.com
Platform               Linux server10.example.com
                        4.18.0-80.11.2.el8_0.x86_64 #1 SMP Sun Sep 15
                        11:24:21 UTC 2019 x86_64 x86_64
Alert Count             1
First Seen              2019-12-04 07:54:16 EST
Last Seen               2019-12-04 07:54:16 EST
Local ID                2cbcdcb9-c14d-460f-a012-cc56888c5d27
-----
```

The above SELinux denial was due to the fact that I produced the scenario by changing the SELinux type on the *shadow* file to something random (*etc_t*). I then issued the *passwd* command as *user1* to modify the password. As expected, SELinux disallowed the *passwd* command to write the new password to the *shadow* file, and it logged the password rejection attempt to the audit log. I then restored the type on the *shadow* file with **restorecon /etc/shadow**. I re-tried the password change and it worked.

Chapter Summary

In this chapter, we discussed security enhanced Linux in fair detail. We looked at the concepts, features, and terminology at length. We examined how security contexts are associated with users, processes, files, and ports, and viewed and modified contexts for them. We analyzed the configuration file that controls its state and defines the policy to be enforced. We examined how domain transitioning works. We learned several SELinux administrative commands and performed tasks such as checking and switching activation mode and operational status. We studied the concept of Booleans and learned how to modify certain parts of the SELinux policy temporarily and persistently. Finally, we reviewed the SELinux Troubleshooter program and used it to view and analyze SELinux related messages.

Review Questions

1. What is the name and location of the SELinux configuration file?
2. What would the command *semanage fcontext -C/* do?
3. Which command can be used to ensure modified contexts will survive file system relabeling?
4. What would the command *semanage login -a -s user_u user10* do?
5. Name the two commands that can be used to modify a Boolean value.
6. Which option is used with the *ps* command to view the security contexts for processes?
7. What would the command *restorecon -F /etc/sysconfig* do?
8. What is the name of the default SELinux policy used in RHEL 8?
9. What would the command *sestatus -b | grep nfs_export_all_rw* do?
10. Name the directory that stores SELinux Boolean files.
11. What are the two commands to display and modify the SELinux mode?
12. Name the two SELinux subjects.
13. SELinux is an implementation of discretionary access control. True or False?
14. Where are SELinux denial messages logged in the absence of the *auditd* daemon?
15. Name the four parts of a process context.

16. What one task must be done to change the SELinux mode from enforcing to disabled?
17. Which option with the `cp` command must be specified to preserve SELinux contexts?
18. What is the purpose of the command `sestatus`?
19. With SELinux running in enforcing mode and one of the services on the system is compromised, all other services will be affected. True or False?
20. Name the command that starts the SELinux Troubleshooter program.

Answers to Review Questions

1. The SELinux configuration filename is `config` and it is located in the `/etc/selinux` directory.
2. The command provided will show recent changes made to the SELinux policy database.
3. The `semanage` command.
4. This command provided will map Linux user `user10` with SELinux user `user_u`.
5. The `semanage` and `setsebool` commands.
6. The `-Z` option.
7. The command provided will restore the default SELinux contexts on the specified directory.
8. The default SELinux policy used in RHEL 8 is targeted.
9. The command provided will display the current value of the specified Boolean.
10. The `/sys/fs/selinux/booleans` directory.
11. The `getenforce` and `setenforce` commands.
12. User and process are two SELinux subjects.
13. False. SELinux is an implementation of mandatory access control.
14. The SELinux denial messages are logged to the `/var/log/messages` file.
15. The four parts of a process context are user, role, type/domain, and sensitivity level.
16. The system must be rebooted.
17. The `--preserve=context` option.
18. The `sestatus` command displays SELinux status information.
19. False.
20. The command name is `sealert`.

Do-It-Yourself Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform the labs without

external help. A step-by-step guide is not supplied, as the knowledge and skill required to implement the lab has already been disseminated in the chapter; however, hints to the relevant major topic(s) are included.

Lab 21-1: Disable and Enable the SELinux Operating Mode

As *user1* with *sudo* on *server10*, check and make a note of the current SELinux operating mode. Modify the configuration file and set the mode to disabled. Reboot the system to apply the change. Run **sudo getenforce** to confirm the change when the system is up. Restore the directive's value to enforcing in the configuration file, and reboot to apply the new mode. Run **sudo getenforce** to confirm the mode when the system is up. (Hint: SELinux Administration).

Lab 21-2: Modify Context on Files

As *user1* with *sudo* on *server10*, create directory hierarchy */tmp/d1/d2*. Check the contexts on */tmp/d1* and */tmp/d1/d2*. Change the SELinux type on */tmp/d1* to *etc_t* recursively with the *chcon* command and confirm. Add */tmp/d1* to the policy database with the *semanage* command to ensure the new context is persistent on the directory hierarchy. (Hint: SELinux Administration).

Lab 21-3: Add Network Port to Policy Database

As *user1* with *sudo* on *server10*, add network port 9001 to the SELinux policy database for the secure HTTP service using the *semanage* command. Verify the addition. (Hint: SELinux Administration).

Lab 21-4: Copy Files with and without Context

As *user1* with *sudo* on *server10*, create file *sef1* under */tmp*. Copy the file to the */usr/local* directory. Check and compare the contexts on both source and destination files. Create another file *sef2* under */tmp* and copy it to the */var/local* directory using the *--preserve=context* option with the *cp* command. Check and compare the contexts on both source and destination files. (Hint: SELinux Administration).

Lab 21-5: Flip SELinux Booleans

As *user1* with *sudo* on *server10*, check the current value of Boolean *ssh_use_tcpd* using the *getsebool* and *sestatus* commands. Use the *setsebool* command and toggle the value of the directive. Confirm the new value with the *getsebool*, *semanage*, or *sestatus* command. (Hint: SELinux Administration).

Chapter 22

Shell Scripting

This chapter describes the following major topics:

- Overview of shell scripts
- Write scripts to display basic system information, employ shell and environment variables, use command substitution, and manipulate special and positional parameters
- Execute and debug scripts
- Know exit codes and test conditions
- Understand logical constructs: if-then-fi, if-then-else-fi, and if-then-elif-fi
- Write scripts using logical statements
- Know arithmetic test conditions
- Comprehend looping construct: for-do-done
- Write scripts using looping statements

RHCSA Objectives:

12. Conditionally execute code (use of: if, test, [], etc.)
13. Use Looping constructs (for, etc.) to process file, command line input
14. Process script inputs (\$1, \$2, etc.)
15. Processing output of shell commands within a script

16. Processing shell command exit codes

Shell scripts are essentially a group of Linux commands along with control structures and optional comments stored in a text file. Their primary purpose of creation is the automation of long and repetitive tasks. Scripts may include any simple to complex command and can be executed directly at the Linux command prompt. They do not need to be compiled as they are interpreted by the shell line by line. This chapter presents example scripts and analyzes them to solidify the learning. These scripts begin with simple programs and advance to more complicated ones. As with any other programming language, the scripting skill develops over time as more and more scripts are read, written, and examined. This chapter also discusses a debug technique that may be helpful in troubleshooting the code.

Shell Scripts

Shell scripts (a.k.a. *shell programs* or simply *scripts*) are text files that contain Linux commands and control structures to automate lengthy, complex, or repetitive tasks, such as managing packages and users, administering partitions and file systems, monitoring file system utilization, trimming log files, archiving and compressing files, finding and removing unnecessary files, starting and stopping database services and applications, and producing reports. Commands in a script are interpreted and run by the shell one at a time in the order in which they are listed. Each line is executed as if it is typed and run at the command prompt. Control structures are utilized for creating and managing conditional and looping constructs. Comments are also generally included to add information about the script such as the author name, creation date, previous modification dates, purpose, and usage. If the script encounters an error during execution, the error message is printed on the screen.

Scripts presented in this chapter are written in the bash shell and may be used in other Linux shells with slight modifications.

You can use any available text editor to write the scripts; however, it is suggested to use the *vim* editor so that you have an opportunity to practice it as you learn scripting. To quickly identify where things are in your scripts, you can use the *nl* command to enumerate the lines. You can store your scripts in the */usr/local/bin* directory, which is included in the PATH of all users by default.

Script01: Displaying System Information

Let's create the first script called `sys_info.sh` on `server10` in the `/usr/local/bin` directory and examine it line by line. Use the `vim` editor with `sudo` to write the script. Type what you see below. Do not enter the line numbers, as they are used for explanation and reference.

```
[user1@server10 ~]$ nl /usr/local/bin/sys_info.sh
 1 #!/bin/bash
 2 # The script name is sys_info.sh, written by Asghar Ghori on November 4
 , 2020
 3 # The script should be located in under /usr/local/bin
 4 # The script shows basic system information
 5 echo "Display Basic System Information"
 6 echo "===="
 7 echo
 8 echo "The hostname, hardware, and OS information is:"
 9 /usr/bin/hostnamectl
10 echo
11 echo "The following users are currently logged in:"
12 /usr/bin/who
```



Within vim, press the `ESC` key and then type `:set nu` to view line numbers associated with each line entry.

In this script, comments and commands are used as follows:

The first line indicates the shell that will run the commands in the script. This line must start with the “`#!`” character combination (called *shebang*) followed by the full pathname to the shell file.

The next three lines contain comments: the script name, author name, creation time, default directory location for storage, and purpose. The number sign (#) implies that anything to the right of it is informational and will be ignored during script execution. Note that the first line also uses the number character (#), but it is followed by the exclamation mark (!); that combination has a special meaning to the shell.

The fifth line has the first command of the script. The `echo` command prints on the screen whatever follows it (“Display Basic System Information” in this case). This may include general comments, errors, or script usage.

The sixth line will highlight the text “Display Basic System Information” by underlining it.

The seventh line has the `echo` command followed by nothing. This will insert an empty line in the output.

The eighth line will print “The hostname, hardware, and OS information is:”.

The ninth line will execute the `hostnamectl` command to display basic information about the system.

The tenth line will insert an empty line.

The eleventh line will print “The following users are currently logged in:” on the screen.

The twelfth line will execute the *who* command to list the logged-in users.

Here is the listing of the *sys_info.sh* file created in the */usr/local/bin* directory:

```
[user1@server10 ~]$ ll /usr/local/bin/sys_info.sh
-rw-r--r--. 1 root root 489 Nov  4 18:01 /usr/local/bin/sys_info.sh
```

Executing a Script

The script created above does not have the execute permission bit since the default umask for the *root* user is set to 0022, which allows read/write access to the owner, and read-only access to the rest. You will need to run the *chmod* command on the file and add an execute bit for everyone:

```
[user1@server10 ~]$ sudo chmod +x /usr/local/bin/sys_info.sh
[user1@server10 ~]$ ll /usr/local/bin/sys_info.sh
-rwxr-xr-x. 1 root root 489 Nov  4 18:01 /usr/local/bin/sys_info.sh
```

Any user on the system can now run this script using either its name or the full path.

Let's run the script and see what the output will look like:

```
[user1@server10 ~]$ sys_info.sh
Display Basic System Information
=====

The hostname, hardware, and OS information is:
  Static hostname: server10.example.com
  Icon name: computer-vm
  Chassis: vm
  Machine ID: 2d06e883e31f434f95d9df6035e988e7
  Boot ID: d28449e756484b349ccbcfc7c0a2daf0
  Virtualization: oracle
  Operating System: Red Hat Enterprise Linux 8.0 (Ootpa)
    CPE OS Name: cpe:/o:redhat:enterprise_linux:8.0:GA
    Kernel: Linux 4.18.0-80.11.2.el8_0.x86_64
    Architecture: x86-64

The following users are currently logged in:
  user1      pts/0          2020-11-04 21:37 (192.168.0.146)
```

The output reflects the execution of the commands as scripted.

The *hostnamectl* command displays the hostname of the system, type of platform (physical, virtual) it is running on, hypervisor vendor name, operating

system name and version, current kernel version, hardware architecture, and other information.

Debugging a Script

Before you have a perfectly working script in place, you may have to run and modify it more than once. You can use a debugging technique that will help identify where the script might have failed or did not function as expected. You can either append the `-x` option to the `"#!/bin/bash"` at the beginning of the script to look like `"#!/bin/bash -x"`, or execute the script as follows:

```
[user1@server10 ~]$ bash -x sys_info.sh
+ echo 'Display Basic System Information'
Display Basic System Information
+ echo =====
=====
+ echo

+ echo 'The hostname, hardware, and OS information is:'
The hostname, hardware, and OS information is:
+ /usr/bin/hostnamectl
....
```

The above output now also includes the actual lines from the script prefixed by the `+` sign and followed by the command execution result. It also shows the line number of the problem line in the output if there is any. This way you can identify any issues pertaining to the path, command name, use of special characters, etc., and address it quickly. Try changing any of the `echo` commands in the script to `"iecho"` and re-run the script in the debug mode to confirm what has just been said.

Script02: Using Local Variables

You had worked with variables earlier in the book and seen their usage. To recap, there are two types of variables: *local* (also called *private* or *shell*) and *environment*. Both can be defined and used in scripts and at the command line.

The following script called `use_var.sh` will define a local variable and display its value on the screen. You will re-check the value of this variable after the script execution has completed. The comments have been excluded for brevity.

```
[user1@server10 ~]$ cat /usr/local/bin/use_var.sh
#!/bin/bash
echo "Setting a Local Variable"
echo =====
SYSNAME=server10.example.com
echo "The hostname of this system is $SYSNAME"
```

Add the execute bit to the script. The following output will be generated when you run it:

```
[user1@server10 ~]$ use_var.sh
Setting a Local Variable
=====
The hostname of this system is server10.example.com
```

If you run the `echo` command to see what is stored in the `SYSNAME` variable, you will get nothing:

```
[user1@server10 ~]$ echo $SYSNAME

[user1@server10 ~]$
```

The output is self-explanatory.

Script03: Using Pre-Defined Environment Variables

The following script called `pre_env.sh` will display the values of SHELL and LOGNAME environment variables:

```
[user1@server10 ~]$ cat /usr/local/bin/pre_env.sh
#!/bin/bash
echo "The location of my shell command is:"
echo $SHELL
echo "I am logged in as $LOGNAME"
```

Add the execute bit to the script, and run to view the result:

```
[user1@server10 ~]$ pre_env.sh
The location of my shell command is:
/bin/bash
I am logged in as user1
```

The output is self-explanatory.

Script04: Using Command Substitution

During the execution of a script, you can use the command substitution feature of the bash shell and store the output generated by the command into a variable. For example, the following script called *cmd_out.sh* will run the *hostname* and *uname* commands and save their outputs in variables. This script shows two different ways to use command substitution. Make sure to use the backticks (normally located with the ~ character on the keyboard) to enclose the *uname* command.

```
[user1@server10 ~]$ cat /usr/local/bin/cmd_out.sh
#!/bin/bash
SYSNAME=$(hostname)
KERNVER=`uname -r`
echo "The hostname is $SYSNAME"
echo "The kernel version is $KERNVER"
```

Add the execute bit and run the script:

```
[user1@server10 ~]$ cmd_out.sh
The hostname is server10.example.com
The kernel version is 4.18.0-80.11.2.el8_0.x86_64
```

The output is self-explanatory.

Understanding Shell Parameters

A *shell parameter* (or simply a *parameter*) is an entity that holds a value such as a name, special character, or number. The parameter that holds a name is referred to as a *variable*; a special character is referred to as a *special parameter*; and one or more digits, except for 0 is referred to as a *positional parameter* (a.k.a. a *command line argument*). A special parameter represents the command or script itself (\$0), count of supplied arguments (\$* or \$@), all arguments (\$#), and PID of the process (\$\$). A positional parameter (\$1, \$2, \$3 . . .) is an argument supplied to a script at the time of its invocation, and its position is determined by the shell based on its location with respect to the calling script. [Figure 22-1](#) gives a pictorial view of the special and positional parameters.

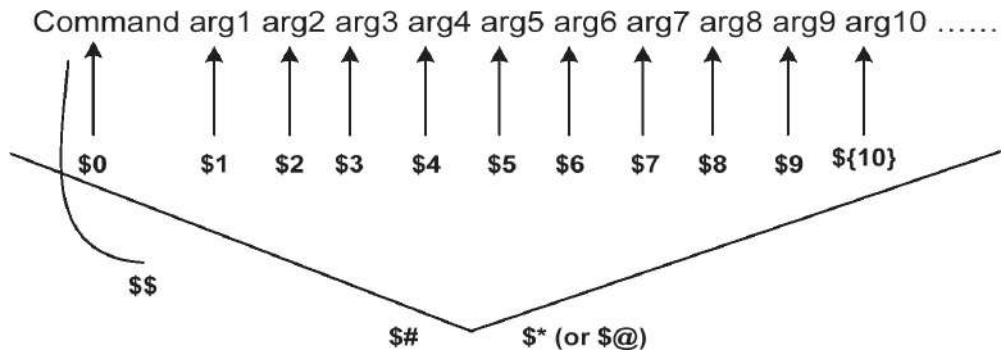


Figure 22-1 Shell Parameters

Figure 22-1 also shows that positional parameters beyond 9 are to be enclosed in curly brackets. Just like the variable and command substitutions, the shell uses the dollar (\$) sign for special and positional parameter expansions as well.

Script05: Using Special and Positional Parameters

The script *com_line_arg.sh* below will show the supplied arguments, total count, value of the first argument, and PID of the script:

```
[user1@server10 ~]$ cat /usr/local/bin/com_line_arg.sh
#!/bin/bash
echo "There are $# arguments specified at the command line"
echo "The arguments supplied are: $*"
echo "The first argument is: $1"
echo "The Process ID of the script is: $$"
```

The result will be as follows when the script is executed with four arguments. Do not forget to add the execute bit prior to running it.

```
[user1@server10 ~]$ com_line_arg.sh Baku Timbuktu Xingyang Quito
There are 4 arguments specified at the command line
The arguments supplied are: Baku Timbuktu Xingyang Quito
The first argument is: Baku
The Process ID of the script is: 6493
```

The output is self-explanatory.

Script06: Shifting Command Line Arguments

The *shift* command is used to move arguments one position to the left. During this move, the value of the first argument is lost. The *com_line_arg_shift.sh* script below is an extension to the *com_line_arg.sh* script. It uses the *shift* command to show what happens when arguments are moved.

```
[user1@server10 ~]$ cat /usr/local/bin/com_line_arg_shift.sh
#!/bin/bash
echo "There are $# arguments specified at the command line"
echo "The arguments supplied are: $*"
echo "The first argument is: $1"
echo "The Process ID of the script is: $$"
shift
echo "The new first argument after the first shift is: $1"
shift
echo "The new first argument after the second shift is: $1"
```

Let's execute the script with the same four arguments. Notice that a new value is assigned to \$1 after each shift.

```
[user1@server10 ~]$ com_line_arg_shift.sh Baku Timbuktu Xingyang Quito
There are 4 arguments specified at the command line
The arguments supplied are: Baku Timbuktu Xingyang Quito
The first argument is: Baku
The Process ID of the script is: 6504
The new first argument after the first shift is: Timbuktu
The new first argument after the second shift is: Xingyang
```

Multiple shifts in a single attempt may be performed by furnishing a count of desired shifts to the *shift* command as an argument. For example, “*shift 2*” will carry out two shifts, “*shift 3*” will make three shifts, and so on.

Logical Constructs

So far, we have talked about simple scripts that run the code line by line. The shell lets us employ logical constructs to control the flow of scripts. It does this by allowing us to use test conditions, which decides what to do next based on the true or false status of the condition.

The shell offers two logical constructs: the *if-then-fi* construct and the *case* construct. The if-then-fi construct has a few variations and those will be covered as well. A discussion on the case construct is beyond the scope.

Before starting to look at the example scripts and see how logical constructs are used, let's discuss exit codes and various test conditions. You will use them later in the example scripts.

Exit Codes

Exit codes, or *exit values*, refer to the value returned by a command when it finishes execution. This value is based on the outcome of the command. If the

command runs successfully, you typically get a zero exit code, otherwise you get a non-zero value. This code is also referred to as a *return code*, and it is stored in a special shell parameter called ? (question mark). Let's look at the following two examples to understand their usage:

```
[user1@server10 ~]$ pwd  
/home/user1  
[user1@server10 ~]$ echo $?  
0  
[user1@server10 ~]$ man  
What manual page do you want?  
[user1@server10 ~]$ echo $?  
1  
[user1@server10 ~]$
```

In the first example, the *pwd* command ran successfully and it produced the desired result, hence a zero exit code was returned and stored in the ?. In the second example, the *man* command did not run successfully because of a missing argument, therefore a non-zero exit code was returned and stored in the ?.

You can define exit codes within a script at different locations to help debug the script by knowing where exactly it terminated.

Test Conditions

Test conditions are used in logical constructs to decide what to do next. They can be set on integer values, string values, or files using the *test* command or by enclosing them within the square brackets []. [Table 22-1](#) describes various test condition operators.

Operation on Integer Value	Description
integer1 -eq (-ne) integer2	Integer1 is equal (not equal) to integer2
integer1 -lt (-gt) integer2	Integer1 is less (greater) than integer2
integer1 -le (-ge) integer2	Integer1 is less (greater) than or equal to integer2
Operation on String Value	Description
string1=(!=)string2	Tests whether the two strings are identical (not identical)
-l string or -z string	Tests whether the string length is zero
string or -n string	Tests whether the string length is non-zero
Operation on File	Description
-b (-c) file	Tests whether the file is a block (character) device file
-d (-f) file	Tests whether the file is a directory (file)
-e (-s) file	Tests whether the file exists (non-empty)
-L file	Tests whether the file is a symlink
-r (-w) (-x) file	Tests whether the file is readable (writable/executable)
-u (-g) (-k) file	Tests whether the file has the setuid (sticky) bit
file1 -nt (-ot) file2	Tests whether file1 is newer (older) than file2
Logical Operators	Description
!	The logical NOT operator
-a or && (two ampersand characters)	The logical AND operator. Both operands must be true for the condition to be true. Syntax: [-b file1 -a -r file2]
-o or (two pipe characters)	The logical OR operator. Either of the both operands must be true for the condition to be true. Syntax: [(x == 1 -o y == 2)]

Table 22-1 Test Conditions

Having described the exit codes and test conditions, let's look at a few example scripts and observe their effects.

The if-then-fi Construct

The if-then-fi statement evaluates the condition for true or false. It executes the specified action if the condition is true; otherwise, it exits the construct. The if-then-fi statement begins with an “if” and ends with a “fi”, as depicted in the flow diagram in [Figure 22-2](#):

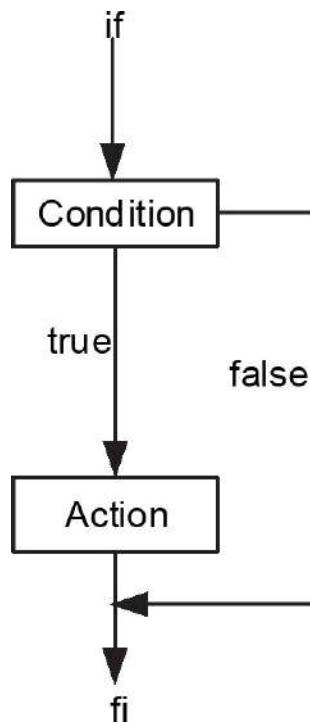


Figure 22-2 The if-then-fi Construct

The general syntax of this statement is as follows:

```
if           condition  
then         action  
fi
```

This construct is used in the following script.

Script07: The if-then-fi Construct

You saw earlier how to check the number of arguments supplied at the command line. The following script called *if_then_fi.sh* determines the number of arguments and prints an error message if there are none provided:

```
[user1@server10 ~]$ cat /usr/local/bin/if_then_fi.sh
#!/bin/bash
if [ $# -ne 2 ] # Ensure there is a space after [ and before ]
then
    echo "Error: Invalid number of arguments supplied"
    echo "Usage: $0 source_file destination_file"
exit 2
fi
echo "Script terminated"
```

This script will display the following messages on the screen if it is executed without exactly two arguments specified at the command line:

```
[user1@server10 ~]$ if_then_fi.sh a
Error: Invalid number of arguments supplied
Usage: /usr/local/bin/if_then_fi.sh source_file destination_file
```

A value of 2 will appear upon examining the return code as follows. This value reflects the exit code that you defined in the script on line 6.

```
[user1@server10 ~]$ echo $?
2
```

Conversely, the return code will be 0 and the message will be as follows if you supply a pair of arguments:

```
[user1@server10 ~]$ if_then_fi.sh a b
Script terminated
[user1@server10 ~]$ echo $?
0
```

The if-then-else-fi Construct

The if-then-fi statement has a limitation and it can execute an action only if the specified condition is true. It quits the statement if the condition is untrue. The if-then-else-fi statement, in contrast, is more advanced in the sense that it can execute an action if the condition is true and another action if the condition is false. The flow diagram for this structure is shown in [Figure 22-3](#):

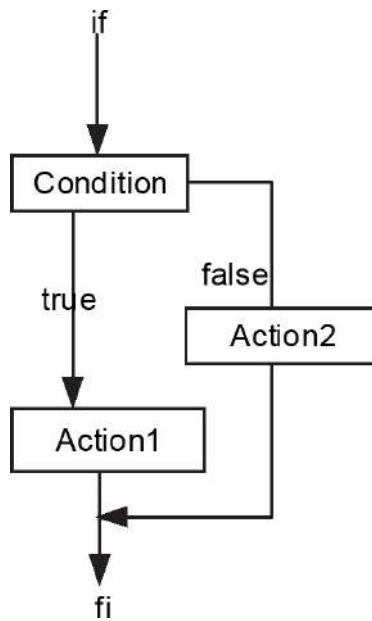


Figure 22-3 The if-then-else-fi Construct

The general syntax of this statement is as follows:

```

if           condition
then
else          action1
else          action2
fi

```

action1 or action2 is performed based on the true or false evaluation of the condition.

Script08: The if-then-else-fi Construct

The following script called *if_then_else_fi.sh* will accept an integer value as an argument and tell if the value is positive or negative:

```
[user1@server10 ~]$ cat /usr/local/bin/if_then_else_fi.sh
#!/bin/bash
if [ $1 -gt 0 ]
then
    echo "$1 is a positive integer value"
else
    echo "$1 is a negative integer value"
fi
```

Run this script one time with a positive integer value and the next time with a negative value:

```
[user1@server10 ~]$ if_then_else_fi.sh 10  
10 is a positive integer value  
[user1@server10 ~]$ if_then_else_fi.sh -10  
-10 is a negative integer value
```

Try the script again but with a non-integer value and see what it does.

The if-then-elif-fi Construct

The if-then-elif-fi is a more sophisticated construct than the other two conditional statements. You can define multiple conditions and associate an action with each one of them. During the evaluation, the action corresponding to the true condition is performed. The flow diagram for this structure is shown in [Figure 22-4](#):

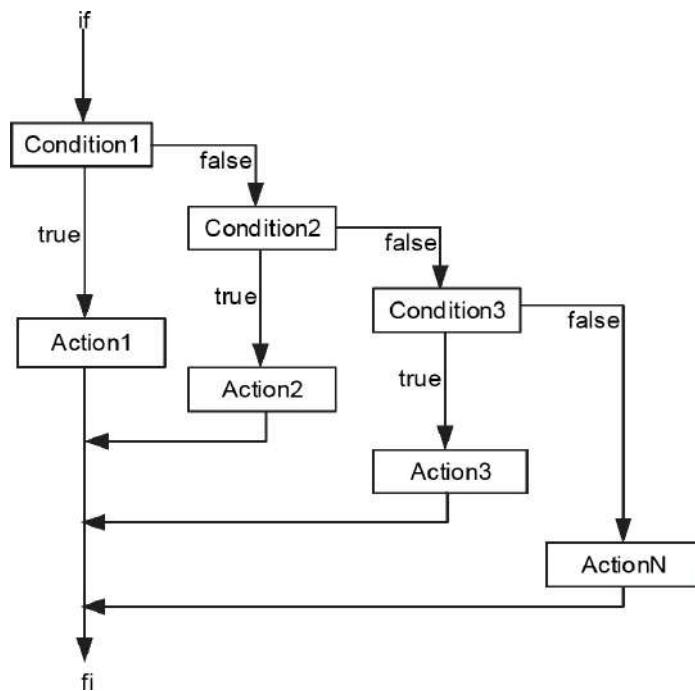


Figure 22-4 The if-then-elif-fi Construct

The general syntax of this statement is as follows:

if	condition1
then	
action1	
elif	condition2

```

then
        action2
elif
        condition3
then
        action3
.....
else
        action(n)
fi

```

Let's use the if-then-elif-fi construct in the following two example scripts.

Script09: The if-then-elif-fi Construct (Example 1)

The *if_then_elif_fi.sh* script is an enhanced version of the *if_then_else_fi.sh* script. It accepts an integer value as an argument and tells if it is positive, negative, or zero. If a non-integer value or no argument is supplied, the script will complain. Notice that the script employs the *exit* command after each action to help you identify where it exited.

```
[user1@server10 ~]$ cat /usr/local/bin/if_then_elif_fi.sh
#!/bin/bash
if [ $1 -gt 0 ]
then
    echo "$1 is a positive integer value"
    exit 1
elif [ $1 -eq 0 ]
then
    echo "$1 is a zero integer value"
    exit 2
elif [ $1 -lt 0 ]
then
    echo "$1 is a negative integer value"
    exit 3
else
    echo "$1 is not an integer value. Please supply an integer."
    exit 4
fi
```

Run this script four times: the first time with a positive integer, the second time with 0, the third time with a negative integer, and the fourth time with a non-integer value. Check the exit code after each execution to know where the script exited.

```
[user1@server10 ~]$ if_then_elif_fi.sh 10
10 is a positive integer value
[user1@server10 ~]$ echo $?
1
[user1@server10 ~]$
[user1@server10 ~]$ if_then_elif_fi.sh 0
0 is a zero integer value
[user1@server10 ~]$ echo $?
2
[user1@server10 ~]$
[user1@server10 ~]$ if_then_elif_fi.sh -10
-10 is a negative integer value
[user1@server10 ~]$ echo $?
3
[user1@server10 ~]$
[user1@server10 ~]$ if_then_elif_fi.sh abd
abd is not an integer value. Please supply an integer.
[user1@server10 ~]$ echo $?
4
```

The outputs and exit values reflect the program code.

Script10: The if-then-elif-fi Construct (Example 2)

The script `ex200_ex294.sh` will display the name of the Red Hat exam RHCSA or RHCE in the output based on the input argument (ex200 or ex294). If a random or no argument is provided, it will print “Usage: Acceptable values are ex200 and ex294”. Make sure to add white spaces in the conditions as shown.

```
[user1@server10 ~]$ cat /usr/local/bin/ex200_ex294.sh
#!/bin/bash
if [ "$1" = ex200 ]
then
    echo "RHCSA"
elif [ "$1" = ex294 ]
then
    echo "RHCE"
else
    echo "Usage: Acceptable values are ex200 and ex294"
fi
```

Run this script three times: the first time with argument ex200, the second time with argument ex294, and the third time with something random as an argument:

```
[user1@server10 ~]$ ex200_ex294.sh ex200
RHCSA
[user1@server10 ~]$ ex200_ex294.sh ex294
RHCE
[user1@server10 ~]$ ex200_ex294.sh ex300
Usage: Acceptable values are ex200 and ex294
```

The results are as expected.

EXAM TIP: A good understanding of the usage of logical statements is important.

Looping Constructs

As a Linux user and administrator, you often want to perform certain task on a number of given elements or repeatedly until a specified condition becomes true or false. For instance, if plenty of disks need to be initialized for use in LVM, you can either run the *pvcreate* command on each disk one at a time manually or employ a loop to do it for you. Likewise, based on a condition, you may want a program to continue to run until that condition becomes true or false.

There are three constructs—*for-do-done*, *while-do-done*, and *until-do-done*—that you can use to implement looping.



The for loop is also referred to as the foreach loop.

The for loop iterates on a list of given values until the list is exhausted. The while loop runs repeatedly until the specified condition becomes false. The until loop does just the opposite of the while loop; it performs an operation repeatedly until the specified condition becomes true. This chapter will discuss the for loop only; the other two are out of scope.

Test Conditions

The *let* command is used in looping constructs to evaluate a condition at each iteration. It compares the value stored in a variable against a pre-defined value. Each time the loop does an iteration, the variable value is altered. You can enclose the condition for arithmetic evaluation within a pair of parentheses (()) or quotation marks (" ") instead of using the *let* command explicitly.

Table 22-2 lists operators that can be used in test conditions.

Operator	Description
	Negation

!	
+ / - / * / /	Addition / subtraction / multiplication / division
%	Remainder
< / <=	Less than / less than or equal to
> / >=	Greater than / greater than or equal to
=	Assignment
== / !=	Comparison for equality / non-equality

Table 22-2 Arithmetic Operators

Having described various test condition operators, let's look at the syntax of the for loop and a few example scripts and observe the implications of some of the test conditions.

The for Loop

The for loop is executed on an array of elements until all the elements in the array are consumed. Each element is assigned to a variable one after the other for processing. The flow diagram for this construct is displayed in [Figure 22-5](#):

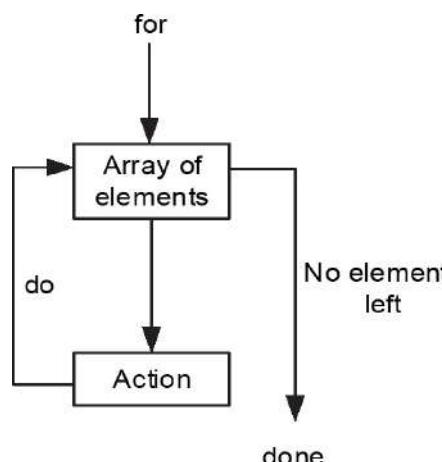


Figure 22-5 The for Loop The general syntax of this construct is as follows:

```

for VAR in list
do
    action
done
  
```

Script11: Print Alphabets Using for Loop

The *for_do_done.sh* script initializes the variable COUNT to 0. The for loop will read each letter sequentially from the range placed within curly brackets (no spaces before the letter A and after the letter Z), assign it to another variable LETTER, and display the value on the screen. The *expr* command is an arithmetic processor and it is used here to increment the COUNT by 1 at each loop iteration.

```
[user1@server10 ~]$ cat /usr/local/bin/for_do_done.sh
#!/bin/bash
COUNT=0
for LETTER in {A..Z}
do
    COUNT=`/usr/bin/expr $COUNT + 1`
    echo "Letter $COUNT is [$LETTER]"
done
```

The output of the script will be:

```
[user1@server10 ~]$ for_do_done.sh
Letter 1 is [A]
Letter 2 is [B]
Letter 3 is [C]
Letter 4 is [D]
Letter 5 is [E]

.....
Letter 25 is [Y]
Letter 26 is [Z]
```

Script12: Create Users Using for Loop

The *create_user.sh* script can create several Linux user accounts. As each account is created, the value of the variable ? is checked. If the value is 0, a message saying the account is created successfully will be displayed, otherwise the script will terminate. In case of a successful account creation, the *passwd* command will be invoked to assign the user the same password as their username.

```
[user1@server10 ~]$ cat /usr/local/bin/create_user.sh
#!/bin/bash
for USER in user{10..12}
do
    echo "Creating account for user $USER"
    /usr/sbin/useradd $USER
    if [ $? = 0 ]
    then
        echo $USER | /usr/bin/passwd --stdin $USER
        echo "$USER is created successfully"
    else
        echo "Failed to create account $USER"
    exit
fi
done
```

The result of the script execution below confirms the addition of three new user accounts:

```
[user1@server10 ~]$ sudo /usr/local/bin/create_user.sh
Creating account for user user10
Changing password for user user10.
passwd: all authentication tokens updated successfully.
user10 is created successfully
Creating account for user user11
Changing password for user user11.
passwd: all authentication tokens updated successfully.
user11 is created successfully
Creating account for user user12
Changing password for user user12.
passwd: all authentication tokens updated successfully.
user12 is created successfully
```

If this script is re-executed without modifying the list of elements (user names), the following will appear:

```
[user1@server10 ~]$ sudo /usr/local/bin/create_user.sh
Creating account for user user10
useradd: user 'user10' already exists
Failed to create account user10
```

EXAM TIP: A good understanding of the looping construct will help on the exam.

Chapter Summary

In this chapter, we learned the basics of bash shell scripting. This chapter began with an overview of scripting and then demonstrated how to write and analyze test scripts using various built-in features of the bash shell. We wrote and examined simple code and gradually advanced to more advanced scripts, including those that employed logical and looping constructs. We learned how

to identify problem lines in our scripts. After understanding and practicing the scripts presented in this chapter, you should be able to write your own programs, debug them, and examine those authored by others.

Review Questions

1. What are the three major components in a shell script?
2. Which looping construct can be used to perform an action on listed items?
3. What is the function of the *shift* command?
4. You can script the startup and shutdown of a database. True or False?
5. What does the *echo* command do without any arguments?
6. What would the command *echo \$?* do?
7. When would you want to use an exit code in your script?
8. What would you modify in a shell script to run it in the debug mode?
9. What are the two types of logical constructs mentioned in this chapter?
10. What would != imply in a looping condition?
11. What is the difference between a line in a script starting with a “#” and a “#!”?
12. What comments may you want to include in a shell script? Write any six.
13. What is one benefit of writing shell scripts?
14. What would the command *bash -x /usr/local/bin/script1.sh* do?

Answers to Review Questions

1. The three major components in a shell script are commands, control structures, and comments.
2. The for loop.
3. The *shift* command moves an argument to the left.
4. True.
5. The *echo* command inserts an empty line in the output when used without arguments.
6. This command will display the exit code of the last command executed.
7. The purpose of using an exit code is to determine exactly where the script quits.
8. We would specify -x as an argument to the shell path.
9. The if and case constructs.
10. != would check the value for non-equality.
11. The former is used to include general comments in the script and the latter combination dictates the full path to the shell file that is to be used to execute the script.
12. The author name, creation date, last modification date, location, purpose, and usage.

13. One major benefit of writing shell scripts is to automate lengthy and repetitive tasks.
14. This command will execute `script1.sh` in debug mode.

DIY Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform these labs without any additional help. A step-by-step guide is not provided, as the implementation of these labs requires the knowledge that has been presented in this chapter. Use defaults or your own thinking for missing information.

Lab 22-1: Write a Script to Create Logical Volumes

For this lab, present 2x1GB virtual disks to your system in VirtualBox VM Manager. Write a single bash shell script to create 2x800MB partitions on each disk using *parted* and then bring both partitions into LVM control with the *pvcreate* command. Create a volume group called *vgscript* and add both PVs to it. Create three logical volumes each of size 500MB and name them *lvscript1*, *lvscript2*, and *lvscript3*.

Lab 22-2: Write a Script to Create File Systems

This lab is a continuation of Lab 22-1. Write another bash shell script to create xfs, ext4, and vfat file system structures in each logical volume, respectively. Create mount points */mnt/xfs*, */mnt/ext4*, and */mnt/vfat*, and mount the file systems. Include the *df* command with -h to list the mounted file systems.

Lab 22-3: Write a Script to Configure a New Network Profile

For this lab, present a new network interface to your system in VirtualBox VM Manager. Write a single bash shell script to run the *nmcli* command to configure custom IP assignments (choose your own settings) on the new network device and ensure that they are automatically applied to it on future system reboots. Make a copy of the */etc/hosts* file as part of this script. Choose a hostname of your choice and add a mapping to the */etc/hosts* file without overwriting existing file content.

Chapter 23

Containers

This chapter describes the following major topics:

- Understand container technology
- Identify key Linux features that establish the foundation to run containers
- Analyze container benefits
- A better home for containers
- Grasp the concepts of container images and registries
- Compare pros and cons of root and rootless containers
- Examine registry configuration file
- Work with container images (find, inspect, pull, list, and delete)
- Administer basic containers (start, list, stop, remove, interact with, run commands from outside, attach to, run custom entry point commands, etc.)
- Implement advanced container features (port mapping, environment variables, and persistent storage)
- Control container operational states via systemd

RHCSA Objectives:

63. Find and retrieve container images from a remote registry
64. Inspect container images
65. Perform container management using commands such as podman and skopeo
66. Perform basic container management such as running, starting, stopping, and listing running containers
67. Run a service inside a container
68. Configure a container to start automatically as a systemd service
69. Attach persistent storage to a container

Containers

and containerization technologies such as Docker and Kubernetes have received an overwhelming appreciation and massive popularity in recent years. They are now part of many new deployments. Containers offer an improved method to package distributed applications, deploy them in a consistent manner, and run them in isolation from one another on the same or different virtual or physical server(s). Containers take advantage of the native virtualization features available in the Linux kernel. Each container typically encapsulates one self-contained application that includes all dependencies such as library files, configuration files, software binaries, and services.

This chapter presents an overview of container images, container registries, and containers. It shows how to interact with images and registries. It demonstrates how to launch, manage, and interact with containers. It discusses advanced topics such as mapping a host port with a container port, passing and setting environment variables, and attaching host storage for data persistence. The chapter ends with a detailed look at controlling the operational states of containers via the `systemd` service. There are numerous exercises in the chapter to support the concepts learned.

Introduction to Containers

A *container* is essentially a set of processes that runs in complete seclusion on a Linux system. A single Linux system running on bare metal hardware or in a virtual machine may have tens or hundreds of containers running. The underlying hardware may be located either on the ground or in the cloud.

Traditionally, one or more software applications are deployed on a single server. These applications may have conflicting requirements in terms of shared library files, package dependencies, and software version. Moreover, patching or an update to the operating system may result in breaking an application functionality. To address these and other potential challenges, developers perform an analysis on their current deployments before they decide whether to collocate a new application with an existing one or to go with a new server without taking the risk of breaking the current operation.

Fortunately, there is a better choice available now in the form of containers. Developers can now package their application alongside dependencies, shared library files, environment variables, and other specifics in a single image file and use that file to run the application in a unique, isolated “virtual computer” called container. Each container is treated as a complete whole,

which can be tagged, started, stopped, restarted, or even transported to another server independent of other containers. This way any conflicts that may exist among applications, within application components, or with the operating system can be evaded. Applications encapsulated to run inside containers are called *containerized applications*, and this is a growing trend for architecting and deploying applications, application components, and databases in real world environments.

Containers and the Linux Features

The container technology employs some of the core features available in the Linux kernel. These include control groups, namespaces, seccomp (*secure computing mode*), and SELinux. A short description of each feature is provided below:

Control Groups

Control groups (abbreviated as *cgroups*) splits processes into groups to set limits on their consumption of compute resources—CPU, memory, disk, and network I/O. These restrictions result in controlling individual processes from over utilizing available resources.

Namespaces

Namespaces restrict the ability of process groups from seeing or accessing system resources—PIDs, network interfaces, mount points, hostname, etc.—thus instituting a layer of isolation between process groups and the rest of the system. This feature guarantees a secure, performant, and stable environment for containerized applications as well as the host operating system.

Secure Computing Mode (seccomp) and SELinux

These Linux features impose security constraints thereby protecting processes from one another and the host operating system from running processes.

The container technology employs these characteristics to run processes isolated in a highly secure environment with full control over what they can or cannot do.

Benefits of Using Containers

There are several benefits linked with using containers. These benefits range from security to manageability and from independence to velocity. The following provides a quick look at common containerization benefits:

Isolation

Containers are not affected due to changes in the host operating system or in other hosted or containerized applications, as they run fully isolated from the rest of the environment.

Loose Coupling

Containerized applications are loosely coupled with the underlying operating system due to their self-containment and minimal level of dependency.

Maintenance Independence

Maintenance is performed independently on individual containers.

Less Overhead

Containers require fewer system resources than do bare metal and virtual servers.

Transition Time

Containers require a few seconds to start and stop.

Transition Independence

Transitioning from one state to another (start or stop) is independent of other containers, and it does not affect or require a restart of any underlying host operating system service.

Portability

Containers can be migrated to other servers without modifications to the contained applications. The target servers may be bare metal or virtual and located on-premises or in the cloud.

Reusability

The same container image can be used to run identical containers in development, test, preproduction, and production environments. There is no need to rebuild the image.

Rapidity

The container technology allows for accelerated application development, testing, deployment, patching, and scaling. Also, there is no need for an exhaustive testing.

Version Control

Container images can be version-controlled, which gives users the flexibility in choosing the right version to run a container.

Container Home: Bare Metal or Virtual Machine

A hypervisor software such as VMware ESXi, Oracle VirtualBox, Microsoft Hyper-V, or KVM allows multiple virtual machines to run on the same bare metal physical server. Each virtual machine runs an isolated, independent instance of its own guest operating system. All virtual machines run in parallel and share the resources of the underlying hardware of the bare metal server. Each virtual machine may run multiple applications that share the virtualized resources allocated to it. The hypervisor runs a set of services on the bare metal server to enable virtualization and support virtual machines, which introduces management and operational overheads. Besides, any updates to the guest operating system may require a reboot or result in an application failure.

Containers, in contrast, run directly on the underlying operating system whether it be running on a bare metal server or in a virtual machine. They share hardware and operating system resources securely among themselves. This allows containerized applications to stay lightweight and isolated, and run in parallel. Containers share the same Linux kernel and require far fewer hardware resources than do virtual machines, which contributes to their speedy start and stop. Given the presence of an extra layer of hypervisor services, it may be more beneficial and economical to run containers directly on non-virtualized physical servers.

[Figure 23-1](#) depicts how applications are placed on traditional bare metal hardware, in virtual machines, and to run as containers on bare metal servers.

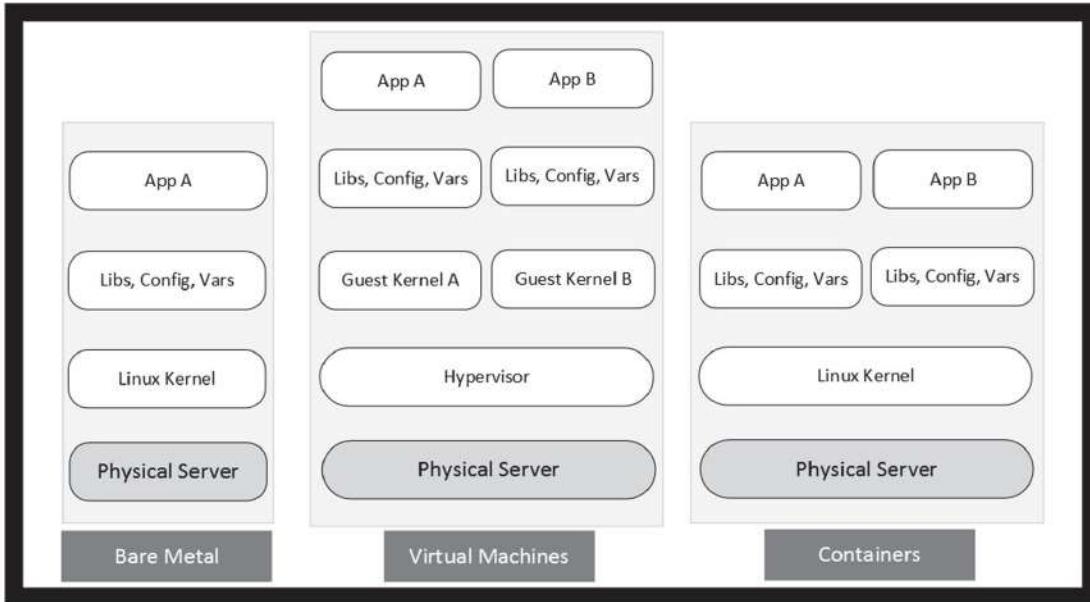


Figure 23-1 Container Home: Bare Metal or Virtual Machine

The added layer of hypervisor is shown in the middle stack. Any of the above implementation can run multiple applications concurrently. A decision as to which option to go with requires a careful use case study; however, the benefits of running containers on bare metal servers are obvious.

Container Images and Container Registries

Launching a container requires a pre-packaged image to be available. A container *image* is essentially a file that is built with all necessary components—application binaries, library files, configuration settings, environment variables, static data files, etc.—required by an application to run smoothly, securely, and successfully included. RHEL follows the *open container initiative* (OCI) to allow users to build images based on industry standard specifications. An OCI-compliant image can be executed and managed with OCI-compliant tools such as *podman* (*pod manager*) and Docker. Images can be version-controlled giving users the suppleness to use the latest or any of the previous versions to launch their containers. A single image can be used to run several containers at once.

Container images adhere to a standard naming convention for identification. This is referred to as *fully qualified image name* (FQIN). An FQIN is comprised of four components: (1) the storage location (registry_name), (2) the owner or organization name (user_name), (3) a unique repository name (repo_name), and (4) an optional version (tag). The syntax of an FQIN is *registry_name/user_name/repo_name:tag*.

Images are stored and maintained in public or private *registries*; however, they need to be downloaded and made locally available for consumption. There are several registries available on the Internet. These include Red Hat Container Catalog at registry.redhat.io (or registry.access.redhat.com), Red Hat Quay at quay.io, and Docker Hub at hub.docker.com. Additional registries may be added as required. Private registries may require authentication for access.

Root vs. Rootless Containers

Containers can be launched with the *root* user privileges. This gives containers full access to perform administrative functions including the ability to map privileged network ports (1024 and below). However, launching containers with superuser rights opens a gate to potential unauthorized access if a container is compromised due to a vulnerability or misconfiguration.

To secure containers and the underlying operating system, the concept of *rootless* container was instituted. Rootless containers allow normal, unprivileged Linux users to run containers and interact with them just like the *root* user, but without the ability to perform tasks that require privileged access. Running containers without *root* is gaining traction.

Working with Images and Containers

To work with images and containers, a good comprehension of the management commands and the configuration files involved is crucial. It is also important to have the minimum required version of the necessary software installed on the system to ensure the features you’re looking to implement are available. RHEL offers two commands—*podman* and *skopeo*—to manage and interact with images, registries, and containers. These tools can also be used to troubleshoot issues and perform advanced management tasks; however, a discussion on those is beyond the scope of this book.

Exercise 23-1: Install Necessary Container Support

This exercise should be done on *server20* as *user1* with *sudo* where required.

In this exercise, you will install the necessary software to set the foundation for completing the exercises in the remainder of the chapter. The standard RHEL 8.0 (and RHEL 8.2) image includes a module called *container-tools* that consists of all the required components and commands; however, a newer version is available in the Red Hat Subscription Management (RHSM) service

and that's what you will be installing here to take advantage of the latest features and bug fixes. You will use the standard *dnf* command to install the module.

1. Register the system with RHSM to access the latest version of the *container-tools* module. Use the credentials you created in [Chapter 01](#) “Local Installation” to connect to the service. Enter your username at the command line with the --username option. Enter the *root* user password first if you have invoked the *subscription-manager* command as a normal user and then your Red Hat password.

```
[user1@server20 ~]$ subscription-manager register --username .
You are attempting to run "subscription-manager" which requires administrative
privileges, but more information is needed in order to do so.
Authenticating as "root"
Password:
Registering to: subscription.rhsm.redhat.com:443/subscription
Password:
The system has been registered with ID: ab468655-2f86-4338-8618-21c1d8e99732
The registered system name is: server20.example.com
```

The system has been registered as indicated.

2. Next, attach the server with the RHEL subscription using the *attach* subcommand:

```
[user1@server20 ~]$ subscription-manager attach
You are attempting to run "subscription-manager" which requires administrative
privileges, but more information is needed in order to do so.
Authenticating as "root"
Password:
Installed Product Current Status:
Product Name: Red Hat Enterprise Linux for x86_64
Status: Subscribed
```

server20 is attached to the indicated subscription (Product Name) and it has access to the Red Hat-maintained BaseOS and AppStream repositories, which contain newer versions of the packages that the corresponding repositories on the DVD image provide. The Red Hat repos also incorporate a lot more software packages than do the DVD repos.

3. Pull the metadata information for the Red Hat repositories:

```
[user1@server20 ~]$ sudo dnf repolist
Updating Subscription Management repositories.
Red Hat Enterprise Linux 8 for x86_64 - AppStre 3.3 MB/s | 19 MB 00:05
Red Hat Enterprise Linux 8 for x86_64 - BaseOS 3.3 MB/s | 22 MB 00:06
Last metadata expiration check: 0:00:16 ago on Sun 25 Oct 2020 08:52:40 PM EDT.
repo id          repo name          status
AppStream        AppStream          4,670
BaseOS          BaseOS            1,658
rhel-8-for-x86_64-appstream-rpms Red Hat Enterprise Linux 8 for x86_64 - 11,134
rhel-8-for-x86_64-baseos-rpms   Red Hat Enterprise Linux 8 for x86_64 - 5,076
```

4. Install the *container-tools* module:

```
[user1@server20 ~]$ sudo dnf module install container-tools
Updating Subscription Management repositories.
Last metadata expiration check: 0:06:05 ago on Sun 25 Oct 2020 08:52:40 PM EDT.
Dependencies resolved.
=====
 Package          Arch    Version      Repository      Size
=====
Upgrading:
containers-common x86_64 1:1.0.0-1.module+el8.2.1+6676+604e1b26
                           rhel-8-for-x86_64-appstream-rpms 52 k
slirp4netns       x86_64 1.0.1-1.module+el8.2.1+6595+03641d72
                           rhel-8-for-x86_64-appstream-rpms 45 k
containernetworking-plugins
                         x86_64 0.8.6-1.module+el8.2.1+6626+598993b4
                           rhel-8-for-x86_64-appstream-rpms 20 M
buildah           x86_64 1.14.9-1.module+el8.2.1+6689+748e6520
                           rhel-8-for-x86_64-appstream-rpms 9.0 M
.....
Installed:
oci-umount-2:2.3.4-2.git87f9237.module+el8+2769+577ad176.x86_64
skopeo-1:1.0.0-1.module+el8.2.1+6676+604e1b26.x86_64
podman-1.9.3-2.module+el8.2.1+6867+366c07d6.x86_64
common-2:2.0.17-1.module+el8.2.1+6771+3533eb4c.x86_64
libslirp-4.3.0-3.module+el8.2.1+6816+bedf4f91.x86_64
libvarlink-18-3.el8.x86_64
Complete!
```

See the *skopeo* and *podman* commands and their versions on the list of installed packages above. They are now available on the system for managing images and containers.

5. Verify the module installation:

```
[user1@server20 ~]$ dnf module list container-tools:rhel8 installed
Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMS)
Name           Stream      Profiles Summary
container-tools rhel8 [d][e] common [ Common tools and dependencies for containe
d] [i]   r runtimes
Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The output confirms a successful installation ([i] beside the common profile) of the *container-tools* module from RHSM.

This concludes the exercise.

The podman Command

Managing images and containers involves several operations such as finding, inspecting, retrieving, and deleting images and running, stopping, listing, and deleting containers. The *podman* command is used for most of these operations. It supports numerous subcommands and options. [Table 23-1](#) describes the ones that are used in this chapter.

Subcommand	Description
Image Management	
images	Lists downloaded images from local storage
inspect	Examines an image and displays its details
login/logout	Logs in/out to/from a container registry. A login may be required to access private and protected registries
pull	Downloads an image to local storage from a registry
rmi	Removes an image from local storage
search	<p>Searches for an image. The following options can be included with this subcommand:</p> <ol style="list-style-type: none"> 1. A partial image name in the search will produce all images containing the partial name. 2. The --no-trunc option makes the command exhibit without truncating it. 3. The --limit <number> option limits the displayed to the specified number.
tag	Adds a name to an image. The default is 'latest' to the image as the latest version. Older images may specific version identifiers.
Container Management	
attach	Attaches to a running container
exec	Runs a process in a running container
generate	Generates a systemd unit configuration file that can be used to control the operational state of a container. This new option is important and is employed in later exercises.
info	Reveals system information, including the defined registries
inspect	Exhibits the configuration of a container
ps	Lists running containers (includes stopped containers with the -a option)
rm	Removes a container
run	Launches a new container from an image. Some options such as -d (detached), -i (interactive), and -t (terminal) are important and are employed in exercises where needed.
start/stop/restart	Starts, stops, or restarts a container

Table 23-1 Common podman Subcommands

All the subcommands described in [Table 23-1](#) are used in the upcoming exercises. Consult the manual pages of the *podman* command for details on the usage of these and other subcommands.

EXAM TIP: A solid understanding of the usage of the *podman* command is key to completing container tasks.

The *skopeo* Command

The *skopeo* command is utilized for interacting with local and remote images and registries. It has numerous subcommands available; however, you will be using only the *inspect* subcommand to examine the details of an image stored in a remote registry. View the manual pages of *skopeo* for details on the usage of the *inspect* and other subcommands.

The *registries.conf* File

The system-wide configuration file for image registries is the *registries.conf* file and it resides in the */etc/containers* directory. Normal Linux users may store a customized copy of this file, if required, under the *~/.config/containers* directory. The settings stored in the per-user file will take precedence over those stored in the system-wide file. This is especially useful for running rootless containers.

This file defines searchable and blocked registries. There are three sections—*registries.search*, *registries.insecure*, and *registries.block*, as evident from the below output:

```
[user1@server20 ~]$ grep -Ev '^#|^$' /etc/containers/registries.conf
[registries.search]
registries = ['registry.redhat.io', 'quay.io', 'docker.io']
[registries.insecure]
registries = []
[registries.block]
registries = []
```

The *registries.search* section lists the registries that are searched if an FQIN is not specified at the command line. By default, there are three registries on the list (see the above output). You will be using *registry.redhat.io* (old name [registry.access.redhat.com](#)) primarily and the other two if needed. If access to an additional registry is necessary, simply add it to the list and place it according to the preference. For instance, if you want a private registry called *registry.private.myorg.io* to be added with the highest priority, edit the *registries.conf* file and make the following change:

```
[registries.search]
registries = ['registry.private.myorg.io', 'registry.redhat.io', 'quay.io',
'docker.io']
```

If this private registry is the only one to be used, you can take the rest of the registry entries out of the list. In that case, the line entry will look like: registries = registry.private.myorg.io.

The registries.insecure section lists the registries that do not have valid SSL/TLS certificates. Insecure registries may also be added to the registries.search section. There are none included in the file by default.

The registries.block section lists registries that must not be used.

EXAM TIP: As there is no Internet access provided during Red Hat exams, you may have to access a network-based registry to download images.

The default content of the file is good for many use cases; however, you may see additional or different entries on busy systems.

Viewing Podman Configuration and Version

The *podman* command references various system runtime and configuration files and runs certain Linux commands in the background to gather and display information. For instance, it looks for registries and storage data in the system-wide and per-user configuration files, pulls memory information from the */proc/meminfo* file, executes **uname -r** to obtain the kernel version, and so on. *podman's* *info* subcommand shows all this information. Here is a sample when this command is executed as a normal user (*user1*):

```
[user1@server20 ~]$ podman info
host:
  arch: amd64
  buildahVersion: 1.14.9
  cgroupVersion: v1
  common:
    package: common-2.0.17-1.module+el8.2.1+6771+3533eb*
    path: /usr/bin/common
    version: 'common version 2.0.17, commit: 3c703d9f17fc
cb36a0'
  cpus: 1
  distribution:
    distribution: '"rhel"'
    version: "8.0"
  eventLogger: file
  hostname: server20.example.com
```

```

.....
kernel: 4.18.0-80.e18.x86_64
memFree: 162566144
memTotal: 1918275584
ociRuntime:
  name: runc
  package: runc-1.0.0-66.rc10.module+el8.2.1+6465+1a51e8b6.x86_64
  path: /usr/bin/runc
  version: 'runc version spec: 1.0.1-dev'
os: linux
rootless: true
.....
registries:
  search:
    - registry.access.redhat.com
    - registry.redhat.io
    - docker.io
store:
  configFile: /home/user1/.config/containers/storage.conf
  containerStore:
    number: 2
    paused: 0
    running: 2
    stopped: 0
  graphDriverName: overlay
.....
imageStore:
  number: 1
runRoot: /run/user/1000
volumePath: /home/user1/.local/share/containers/storage/volumes
.....

```

The above output is self-explanatory.

Now, re-run the command as *root* (precede by *sudo* if running as *user1*) and compare the values for the settings “rootless” under host and “ConfigFile” and “ImageStore” under store. The differences lie between where the root and rootless (normal) users store and obtain configuration data, the number of container images they have locally available, and so on.

Similarly, you can run the *podman* command as follows to check its version:

```
[user1@server20 ~]$ podman version
Version:          1.9.3
RemoteAPI Version: 1
Go Version:       go1.13.4
OS/Arch:          linux/amd64
```

The output reveals the version (1.9.3) of the *podman* utility and this is what you will be using to perform the forthcoming exercises.

Image Management

Images are stored in private or public registries and are pulled locally for consumption. They can be searched through one or more registries and their metadata can be examined before download. Downloaded images can be removed when no longer needed to conserve local storage. The same pair of commands—*podman* and *skopeo*—is employed for these operations.

Exercise 23-2: Search, Examine, Download, and Remove an Image

This exercise should be done on *server20* as *user1* with *sudo* where required. In this exercise, you will look for an image called *mysql* in the *quay.io* registry, examine its details, pull it to your system, confirm the retrieval, and finally erase it from the local storage. You will use the *podman* and *skopeo* commands as required for these operations.

1. Find the *mysql* image in the specified registry (*quay.io*) using *podman search*. You may add the --no-trunc option to view full output.

```
[user1@server20 ~]$ podman search quay.io/rhel8/mysql
INDEX      NAME                                     STARS  OFFICIAL  AUTOMATED  DES
CRIPTION
quay.io    quay.io/presslabs/mysql-operator        0          Doc
ker images for presslabs mysql-operator. ...
quay.io    quay.io/freshbooks/liquibase-mysql     0
quay.io    quay.io/presslabs/mysql-operator-sidecar 0          Doc
ker images for presslabs mysql operator's...
quay.io    quay.io/cosee-concourse/mysql-resource 0
quay.io    quay.io/app-sre/mysql                   0          Use
d by the SQL-QUERY integration
.....
```

There are several images containing the searched name. You will pick *app-sre/mysql* for use in this exercise.

2. Inspect the image without downloading it using *skopeo inspect*. A long output will be generated. The command uses the docker:// mechanism to access the image.

```
[user1@server20 ~]$ skopeo inspect docker://quay.io/app-sre/mysql
{
  "Name": "quay.io/app-sre/mysql",
  "Digest": "sha256:77b7e09c906615c1bb59b2e9d7703f728b1186a5a70e547ce2f1079ef4
c1c5ca",
  "RepoTags": [
    "5.5.45",
    "5.5.46",
    "5.5.47",
    "5.5.48",
  .....
```

```

        "5.7.7"
],
"Created": "2020-10-13T08:03:01.091875544Z",
"DockerVersion": "18.09.7",
"Labels": null,
"Architecture": "amd64",
"Os": "linux",
"Layers": [
    "sha256:bb79b6b2107fea8e8a47133a660b78e3a546998fcf0427be39ac9a0af4a97e90
",
    "sha256:49e22f6fb9f7713028f8ed9b0beaa2ebac38d73ff6fd60532031e4a257f314c0
.....

```

The output shows older versions under RepoTags, creation timestamp for the latest version, the Docker version that was used to build this image, and other information. It is a good practice to analyze the metadata of an image prior to downloading and consuming it.

- Download the image by specifying the fully qualified image name using *podman pull*:

```
[user1@server20 ~]$ podman pull docker://quay.io/app-sre/mysql
Trying to pull docker://quay.io/app-sre/mysql...Getting image source signatures
Copying blob bb79b6b2107f: 25.84 MiB / 25.84 MiB [=====] 38s
Copying blob 49e22f6fb9f7: 1.70 KiB / 1.70 KiB [=====] 38s
Copying blob 842b1255668c: 3.98 MiB / 3.98 MiB [=====] 38s
Copying blob 9f48d1f43000: 1.35 MiB / 1.35 MiB [=====] 38s
Copying blob c693f0615bce: 115 B / 115 B [=====] 38s
Copying blob 8a621b9dbed2: 12.82 MiB / 12.82 MiB [=====] 38s
Copying blob 0807d32aef13: 2.34 KiB / 2.34 KiB [=====] 38s
Copying blob 9eb4355ba450: 225 B / 225 B [=====] 38s
Copying blob 6879faad3b6c: 107.28 MiB / 107.28 MiB [=====] 38s
Copying blob 164ef92f3887: 843 B / 843 B [=====] 38s
Copying blob 6e4a6e666228: 5.02 KiB / 5.02 KiB [=====] 38s
Copying blob d45dea7731ad: 121 B / 121 B [=====] 38s
Copying config 8e85dd5c3255: 6.97 KiB / 6.97 KiB [=====] 0s
Writing manifest to image destination
Storing signatures
8e85dd5c32558ea5c22cc4786cff512c1940270a50e7dbc21ad2df42f0637de4
```

- List the image to confirm the retrieval using *podman images*:

```
[user1@server20 ~]$ podman images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
quay.io/app-sre/mysql     latest   8e85dd5c3255   7 days ago   549 MB
```

The output indicates the FQIN of the image, version (tag), image ID, when it was created, and size.

- Display this image's details using *podman inspect*. The command will generate a long output. You may pipe the output to the *less* command to view one screenful of information at a time.

```
[user1@server20 ~]$ podman inspect mysql
[
  {
    "Id": "8e85dd5c32558ea5c22cc4786cff512c1940270a50e7dbc21ad2df42f0637de4",
    "Digest": "sha256:77b7e09c906615c1bb59b2e9d7703f728b1186a5a70e547ce2f1079ef4c1c5ca",
    "RepoTags": [
      "quay.io/app-sre/mysql:latest"
    ],
    "RepoDigests": [
      "quay.io/app-sre/mysql@sha256:77b7e09c906615c1bb59b2e9d7703f728b1186a5a70e547ce2f1079ef4c1c5ca"
    ],
    "Parent": "",
    "Comment": "",
    "Created": "2020-10-13T08:03:01.091875544Z",
    "Config": {
      "ExposedPorts": {
        "3306/tcp": {},
        "33060/tcp": {}
      }
    },
    ...
  }
]
```

6. Remove the mysql image from local storage using *podman rmi*:

```
[user1@server20 ~]$ podman rmi mysql
8e85dd5c32558ea5c22cc4786cff512c1940270a50e7dbc21ad2df42f0637de4
```

The *podman* command shows the ID of the image after deletion.

7. Confirm the removal using *podman images*:

```
[user1@server20 ~]$ podman images
[user1@server20 ~]$
```

The image is no longer available in the local storage, which confirms the removal.

This concludes the exercise.

Basic Container Management

Managing containers involves common tasks such as starting, stopping, listing, viewing information about, and deleting them. Depending on the use case, containers can be launched in different ways. They can have a name assigned or be nameless, have a terminal session opened for interaction, execute an entry point command (the command specified at the launch time) and be auto-terminated right after, and so on. Running containers can be stopped and restarted, or discarded if no longer needed. The *podman* command is utilized to start containers and manage their lifecycle. This command is also employed to list stopped and running containers, and view their details.

Exercise 23-3: Run, Interact with, and Remove a Named Container

This exercise should be done on `server20` as `user1` with `sudo` where required. In this exercise, you will run a container based on the latest version of a *universal base image* (`ubi`) for RHEL 8 available in the Red Hat Container Registry. This image provides the base operating system layer for the deployment of containerized applications. You will assign this container a name and run a few native Linux commands in a terminal window interactively. You will exit out of the container and invoke a command inside the container from `server20`. Finally, you will reconnect to the container, exit out, and delete it to mark the completion of the exercise.

1. Delete the directory `/etc/docker` (if it exists) to avoid permission issues:

```
[user1@server20 ~]$ sudo rm -rf /etc/docker
```

2. Launch a container using `ubi8` (RHEL 8 image). Name this container `rhel8-base-os` and open a terminal session (`-t`) for interaction (`-i`). Use the *podman run* command.

```
[user1@server20 ~]$ podman run -ti --name rhel8-base-os ubi8
Trying to pull registry.redhat.io/ubi8:latest...Getting image source signatures
Copying blob ec1681b6a383: 70.44 MiB / 70.44 MiB [=====] 14s
Copying blob c4d668e229cd: 1.75 KiB / 1.75 KiB [=====] 14s
Copying config ecbc6f53bba0: 4.26 KiB / 4.26 KiB [=====] 0s
Writing manifest to image destination
Storing signatures
[root@2d77b04238b8 ~]#
```

The above command downloaded the latest version of the specified image automatically even though no FQIN was provided. This is because it searched through the registries listed in the `/etc/containers/registries.conf` file and retrieved the image from wherever it found it first (`registry.redhat.io`). It also opened a terminal session inside the container as the `root` user to interact with the containerized RHEL 8 OS. The container ID is reflected as the hostname in the container's command prompt (last line in the output). This is an auto-generated ID.

3. Run a few basic commands such as `pwd`, `ls`, `cat`, and `date` inside the container for verification:

```
[root@2d77b04238b8 /]# pwd  
/  
[root@2d77b04238b8 /]# ls  
bin dev home lib64 media opt root sbin sys usr  
boot etc lib lost+found mnt proc run srv tmp var  
[root@2d77b04238b8 /]# cat /etc/redhat-release  
Red Hat Enterprise Linux release 8.2 (Ootpa)  
[root@2d77b04238b8 /]#  
[root@2d77b04238b8 /]# date  
Thu Oct 15 14:39:15 UTC 2020
```

4. Close the terminal session when done:

```
[root@2d77b04238b8 /]# exit  
exit  
[user1@server20 ~]$
```

5. Execute a command inside the container directly from the host using *podman exec*:

```
[user1@server20 ~]$ podman exec rhel8-base-os cat /etc/redhat-release  
Red Hat Enterprise Linux release 8.2 (Ootpa)
```

6. Reconnect to the container using *podman attach*:

```
[user1@server20 ~]$ podman attach rhel8-base-os  
[root@2d77b04238b8 /]#  
[root@2d77b04238b8 /]# pwd  
/
```

7. Close the terminal session to return to the host system's command prompt:

```
[root@2d77b04238b8 /]# exit  
exit  
[user1@server20 ~]$
```

8. Delete the container using *podman rm*:

```
[user1@server20 ~]$ podman rm rhel8-base-os  
2d77b04238b8869588092c850d755771574e6e23e3ac81f24fc87a2b8a925cd3
```

Confirm the removal with **podman ps**.

This concludes the exercise.

Exercise 23-4: Run a Nameless Container and Auto-Remove it After Entry Point Command Execution

This exercise should be done on *server20* as *user1*.

In this exercise, you will launch a container based on the latest version of a *universal base image* (ubi) for RHEL 7 available in the Red Hat Container Registry. This image provides the base operating system layer to deploy containerized applications. You will enter a Linux command at the command line for execution inside the container as an entry point command and ensure that the container is deleted right after that.

1. Start a container using ubi7 (RHEL 7 image) and run *ls* as an entry point command. Use *podman run* with the --rm option to remove the container as soon as the entry point command has finished running.

```
[user1@server20 ~]$ podman run --rm ubi7 ls
Trying to pull registry.access.redhat.com/ubi7...
Getting image source signatures
Copying blob 2bd25ca12457 done
Copying blob 1323a241cc06 done
Copying config fdef99b341 done
Writing manifest to image destination
Storing signatures
bin
boot
dev
etc
.......
```

The *ls* command was executed successfully and the container was terminated thereafter.

2. Confirm the container removal with *podman ps*:

```
[user1@server20 ~]$ podman ps
CONTAINER ID  IMAGE   COMMAND   CREATED   STATUS    PORTS   NAMES
[user1@server20 ~]$
```

The container no longer exists.

This concludes the exercise.

Advanced Container Management

Containers run a variety of applications and databases that may need to communicate with applications or databases running in other containers on the same or different host system over certain network ports. Preset environment variables may be passed when launching containers or new variables may be set for containerized applications to consume for proper operation. Information stored during an application execution is lost when a container is restarted or erased. This behavior can be overridden by making a directory on the host available inside the container for saving data persistently. Containers may be configured to start and stop with the transitioning of the

host system via the systemd service. These advanced tasks are also performed with the *podman* command. Let's take a closer look.

Containers and Port Mapping

Applications running in different containers often need to exchange data for proper operation. For instance, a containerized Apache web server may need to talk to a MySQL database instance running in a different container. It may also need to talk to the outside world over a port such as 80 or 8080. To support this traffic flow, appropriate port mappings are established between the host system and each container.

EXAM TIP: As a normal user, you cannot map a host port below 1024 to a container port.

Exercise 23-5: Configure Port Mapping

This exercise should be done on *server20* as *root*.

In this exercise, you will launch a container called *rhel7-port-map* in detached mode (as a daemon) with host port 10000 mapped to port 8000 inside the container. You will use a version of the RHEL 7 image with Apache web server software pre-installed. This image is available in the Red Hat Container Registry. You will list the running container and confirm the port mapping.

1. Search for an Apache web server image for RHEL 7 using *podman search*:

```
[root@server20 ~]# podman search registry.redhat.io/rhel7/httpd
INDEX      NAME                                     DESCRIPTION
          STARS   OFFICIAL   AUTOMATED
redhat.io  registry.redhat.io/rhscl/httpd-24-rhel7  Apache HTTP 2.4 Server
          0
....
```

The Red Hat Container Registry has an Apache web server image called *httpd-24-rhel7* available as evident from the output.

2. Log in to *registry.redhat.io* using the Red Hat credentials to access the image:

```
[root@server20 ~]# podman login registry.redhat.io
Username:
Password:
Login Succeeded!
[root@server20 ~]#
```

3. Download the latest version of the Apache image using *podman pull*:

```
[root@server20 ~]# podman pull registry.redhat.io/rhscl/httpd-24-rhel7
Trying to pull registry.redhat.io/rhscl/httpd-24-rhel7...
Getting image source signatures
Copying blob 2bd25ca12457 done
Copying blob 1323a241cc06 done
Copying blob 5d011ac93e74 done
Copying blob 18a6e61d9d3e done
Copying config 610c5665a9 done
Writing manifest to image destination
Storing signatures
610c5665a9e62ba1f5f7a0a1af9f4715d82c007d3d1a5d38fcefa7328ae93bca
```

4. Verify the download using *podman images*:

```
[root@server20 ~]# podman images
REPOSITORY          TAG      IMAGE ID      CREATED
SIZE
registry.redhat.io/rhscl/httpd-24-rhel7    latest   610c5665a9e6  6 weeks ago
328 MB
```

5. Launch a container named (*--name*) *rhel7-port-map* in detached mode (*-d*) to run the containerized Apache web server with host port 10000 mapped to container port 8000 (*-p*). Use the *podman run* command with appropriate flags.

```
[root@server20 ~]# podman run -dp 10000:8000 --name rhel7-port-map httpd-24-rhel7
203d673fb08a246c4787a284af986c531ee597d1d7e6bd0e614cb767de0cd25d
```

6. Verify that the container was launched successfully using *podman ps*:

```
[root@server20 ~]# podman ps
CONTAINER ID  IMAGE                                COMMAND
CREATED      STATUS      PORTS                NAMES
203d673fb08a  registry.redhat.io/rhscl/httpd-24-rhel7:latest  /usr/bin/run-http...
51 seconds ago  Up 50 seconds ago  0.0.0.0:10000->8000/tcp  rhel7-port-map
```

The container is running (up for 50 seconds and created 51 seconds ago). The output also indicates the port mapping.

7. You can also use *podman port* to view the mapping:

```
[root@server20 ~]# podman port rhel7-port-map
8000/tcp -> 0.0.0.0:10000
```

Now any inbound web traffic on host port 10000 will be redirected to the container.

This concludes the exercise.

Exercise 23-6: Stop, Restart, and Remove a Container

This exercise should be done on *server20* as *root*.

This exercise is a continuation of [Exercise 23-5](#) in which a container named *rhel7-port-map* was launched.

In this exercise, you will stop the container, restart it, stop it again, and then erase it. You will use appropriate *podman* subcommands and verify each transition.

1. Verify the current operational state of the container *rhel7-port-map*:

```
[root@server20 ~]# podman ps
CONTAINER ID IMAGE COMMAND
CREATED STATUS PORTS NAMES
203d673fb08a registry.redhat.io/rhscl/httpd-24-rhel7:latest /usr/bin/run-http...
8 minutes ago Up 8 minutes ago 0.0.0.0:10000->8000/tcp rhel7-port-map
```

The container has been up and running for 8 minutes. It was created 8 minutes ago.

2. Stop the container and confirm using the *stop* and *ps* subcommands (the *-a* option with *ps* also includes the stopped containers in the output):

```
[root@server20 ~]# podman stop rhel7-port-map
203d673fb08a246c4787a284af986c531ee597d1d7e6bd0e614cb767de0cd25d
[root@server20 ~]#
[root@server20 ~]# podman ps -a
CONTAINER ID IMAGE COMMAND
CREATED STATUS PORTS NAMES
203d673fb08a registry.redhat.io/rhscl/httpd-24-rhel7:latest /usr/bin/run-http...
10 minutes ago Exited (0) 5 seconds ago 0.0.0.0:10000->8000/tcp rhel7-port-ma
p
```

The container is in stopped (Exited) state for the past 5 seconds. It was created 10 minutes ago.

3. Start the container and confirm with the *start* and *ps* subcommands:

```
[root@server20 ~]# podman start rhel7-port-map
rhel7-port-map
[root@server20 ~]# podman ps -a
CONTAINER ID IMAGE COMMAND
CREATED STATUS PORTS NAMES
203d673fb08a registry.redhat.io/rhscl/httpd-24-rhel7:latest /usr/bin/run-http...
11 minutes ago Up 1 second ago 0.0.0.0:10000->8000/tcp rhel7-port-map
```

Observe the creation and running times.

4. Stop the container and remove it using the *stop* and *rm* subcommands:

```
[root@server20 ~]# podman stop rhel7-port-map
203d673fb08a246c4787a284af986c531ee597d1d7e6bd0e614cb767de0cd25d
[root@server20 ~]# podman rm rhel7-port-map
203d673fb08a246c4787a284af986c531ee597d1d7e6bd0e614cb767de0cd25d
```

5. Confirm the removal using the *-a* option with *podman ps* to also include any stopped containers:

```
[root@server20 ~]# podman ps -a
CONTAINER ID  IMAGE   COMMAND   CREATED   STATUS    PORTS   NAMES
```

The container no longer exists.

This concludes the exercise.

Containers and Environment Variables

Many a times, it is necessary to pass a host's pre-defined environment variable, such as PATH, to a containerized application for consumption. Moreover, it may also be necessary at times to set new variables to inject debugging flags or sensitive information such as passwords, access keys, or other secrets for use inside containers. Passing host environment variables and setting new environment variables is done at the time of launching a container. The *podman* command allows multiple variables to be passed or set with the *-e* option.

EXAM TIP: Use the *-e* option with each variable that you want to pass or set.

Refer to [Chapter 07](#) "The Bash Shell" for more information on variables, their types, and how to define and view them.

Exercise 23-7: Pass and Set Environment Variables

This exercise should be done on *server20* as *user1*.

In this exercise, you will launch a container using the latest version of a ubi for RHEL 8 available in the Red Hat Container Registry. You will inject the HISTSIZE environment variable and a variable called SECRET with the value "secret123". You will name this container *rhel8-env-vars* and have a shell terminal opened to check the variable settings. Finally, you will remove this container.

1. Launch a container with an interactive terminal session (*-it*) and inject (*-e*) variables HISTSIZE and SECRET as directed. Use the specified container image.

```
[user1@server20 ~]$ podman run -it -e HISTSIZE -e SECRET="secret123" --name rhel8-
env-vars ubi8
Trying to pull registry.access.redhat.com/ubi8...
Getting image source signatures
Copying blob c4d668e229cd done
Copying blob ec1681b6a383 done
Copying config ecbc6f53bb done
Writing manifest to image destination
Storing signatures
[root@b8535edb130d /]#
```

The container is launched with a terminal opened for interaction.

2. Verify both variables using the `echo` command:

```
[root@b8535edb138d /]# echo $HISTSIZE $SECRET  
1000 secret123
```

Both variables are set and show their respective values.

3. Disconnect from the container using the `exit` command, and stop and remove it using the `stop` and `rm` subcommands:

```
[root@b8535edb138d /]# exit  
exit  
[user1@server20 ~]$ podman stop rhel8-env-vars  
b8535edb138d0680f21f764e424eb4bff1ba103f56292c9bc09a9fd54c5e96ac  
[user1@server20 ~]$ podman rm rhel8-env-vars  
b8535edb138d0680f21f764e424eb4bff1ba103f56292c9bc09a9fd54c5e96ac
```

At this point, you may want to confirm the deletion by running `podman ps -a`.

This concludes the exercise.

Containers and Persistent Storage

Containers are normally launched for a period of time to run an application and are then stopped or deleted when their job is finished. Any data that is produced during runtime is lost on their restart, failure, or termination. This data may be saved for persistence on a host directory by attaching it to a container. The containerized application will see the attached directory just like any other local directory and will use it to store data if it is configured to do so. Any data that is saved on the directory will be available even after the container is rebooted or removed. Later, this directory can be re-attached to other containers to give them access to the stored data or to save their own data. The source directory on the host may itself exist on any local or remote file system.

EXAM TIP: Proper ownership, permissions, and SELinux file type must be set to ensure persistent storage is accessed and allows data writes without issues.

There are a few simple steps that should be performed to configure a directory before it can be attached to a container. These steps include the correct ownership, permissions, and SELinux type (`container_file_t`). The special SELinux file type is applied to prevent containerized applications (especially those running in root containers) from gaining undesired privileged access to host files and processes, or other running containers on the host if compromised.

Exercise 23-8: Attach Persistent Storage and Access Data Across Containers

This exercise should be done on `server20` as `user1` with `sudo` where required.

In this exercise, you will set up a directory on `server20` and attach it to a new container. You will write some data to the directory while in the container. You will delete the container and launch another container with the same directory attached. You will observe that the saved data is available in the new container and is accessible. You will remove the container to mark the completion of this exercise.

1. Create a directory called `/host_data`, set full permissions on it, and confirm:

```
[user1@server20 ~]$ sudo mkdir /host_data
[user1@server20 ~]$ sudo chmod 777 /host_data
[user1@server20 ~]$ ll -d /host_data
drwxrwxrwx. 2 root root 6 Oct 21 10:33 /host_data
```

2. Launch a root container called `rhel7-persistent-data` (`--name`) in interactive mode (`-it`) using the latest ubi7 image. Specify the attachment point (`/container_data`) to be used inside the container for the host directory (`/host_data`) with the `-v` (volume) option. Add `:Z` as shown to ensure the SELinux type `container_file_t` is automatically set on the directory and files within.

```
[user1@server20 ~]$ sudo podman run --name rhel7-persistent-data -v /host_data:/container_data:Z -it ubi7
Trying to pull registry.access.redhat.com/ubi7...
Getting image source signatures
Copying blob 2bd25ca12457 done
Copying blob 1323a241cc06 done
Copying config fdef99b341 done
Writing manifest to image destination
Storing signatures
[root@41fa789bca4f ~]#
```

3. Confirm the presence of the directory inside the container using the `ls` command on `/container_host`:

```
[root@41fa789bca4f ~]# ls -ldz /container_data --color=never
drwxrwxrwx. root root system_u:object_r:container_file_t:s0:c579,c977 /container_data
```

The host directory is available as `/container_data` inside the container with the correct SELinux type.

4. Create a file called `testfile` with the `echo` command under `/container_host`:

```
[root@41fa789bca4f ~]# echo "This is persistent storage" > /container_data/testfile
```

5. Verify the file creation and the SELinux type on it:

```
[root@41fa789bca4f /]# ls -lZ /container_data/
-rw-r--r--. root root system_u:object_r:container_file_t:s0:c579,c977 testfile
```

The file inherits the SELinux type from the parent directory.

6. Exit out of the container and check the presence of the file in the host directory:

```
[root@41fa789bca4f /]# exit
exit
[user1@server20 ~]$ ls -lZ /host_data/
total 4
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c579,c977 27 Nov  3 12
:24 testfile
```

The output indicates the exact same attributes on the file.

7. Stop and remove the container using the *stop* and *rm* subcommands:

```
[user1@server20 ~]$ sudo podman stop rhel7-persistent-data
41fa789bca4f7fac752af5b378ee0bcfa297d8e1958a1a4852657420c6c2cf2a
[user1@server20 ~]$ sudo podman rm rhel7-persistent-data
41fa789bca4f7fac752af5b378ee0bcfa297d8e1958a1a4852657420c6c2cf2a
```

8. Launch a new root container called *rhel8-persistent-data* (*--name*) in interactive mode (*-it*) using the latest ubi8 image from any of the defined registries. Specify the attachment point (*/container_data2*) to be used inside the container for the host directory (*/host_data*) with the *-v* (volume) option. Add *:Z* as shown to ensure the SELinux type *container_file_t* is automatically set on the directory and files within.

```
[user1@server20 ~]$ sudo podman run -it --name rhel8-persistent-data -v /host_data:/container_data2:Z ubi8
Trying to pull registry.access.redhat.com/ubi8...
Getting image source signatures
Copying blob c4d668e229cd done
Copying blob ec1681b6a383 done
Copying config ecbbc6f53bb done
Writing manifest to image destination
Storing signatures
[root@778b119612e0 /]#
```

9. Confirm the presence of the directory inside the container using the *ls* command on */container_host2*:

```
[root@778b119612e0 /]# ls -ldZ /container_data2/
drwxrwxrwx. 2 root root system_u:object_r:container_file_t:s0:c107,c538 22 Nov  3 17
:24 /container_data2/
[root@778b119612e0 /]#
[root@778b119612e0 /]# ls -lZ /container_data2/
total 4
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c107,c538 27 Nov  3 17
:24 testfile
[root@778b119612e0 /]#
[root@778b119612e0 /]# cat /container_data2/testfile
This is persistent storage
[root@778b119612e0 /]#
```

The host directory is available as `/container_data2` inside this new container with the correct SELinux type. The output also confirms the persistence of the `testfile` data that was written in the previous container.

10. Create a file called `testfile2` with the `echo` command under `/container_data2`:

```
[root@778b119612e0 /]# echo "This is persistent storage2" > /container_data2/testfile2
[root@778b119612e0 /]#
[root@778b119612e0 /]# ls -lZ /container_data2/
total 8
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c107,c538 27 Nov  3 17
:24 testfile
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c107,c538 28 Nov  3 17
:38 testfile2
```

11. Exit out of the container and confirm the existence of both files in the host directory:

```
[root@778b119612e0 /]# exit
exit
[user1@server20 ~]$ ls -lZ /host_data/
total 8
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c107,c538 27 Nov  3 12
:24 testfile
-rw-r--r--. 1 root root system_u:object_r:container_file_t:s0:c107,c538 28 Nov  3 12
:38 testfile2
```

12. Stop and remove the container using the `stop` and `rm` subcommands:

```
[user1@server20 ~]$ sudo podman stop rhel8-persistent-data
778b119612e040006c77a7c580c7d9f7c5e3299da3cf70d2c770daa6e2f161f1f
[user1@server20 ~]$ sudo podman rm rhel8-persistent-data
778b119612e040006c77a7c580c7d9f7c5e3299da3cf70d2c770daa6e2f161f1f
```

13. Re-check the presence of the files in the host directory:

```
[user1@server20 ~]$ ll /host_data/
total 8
-rw-r--r--. 1 root root 27 Nov  3 12:24 testfile
-rw-r--r--. 1 root root 28 Nov  3 12:38 testfile2
```

Both files still exist and they can be shared with other containers if required.

This concludes the exercise.

Container State Management with `systemd`

So far you have seen how containers are started, stopped, and deleted by hand using the management command. In real life environments multiple containers run on a single host and it becomes a challenging task to change their operational state or delete them manually. In RHEL 8, these administrative functions can be automated via the `systemd` service (refer to

[Chapter 12](#) “System Initialization, Message Logging, and System Tuning” for details on `systemd`).

There are several steps that need to be completed to configure container state management via `systemd`. These steps vary for root and rootless container setups and include the creation of service unit files and their storage in appropriate directory locations (`~/.config/systemd/user` for rootless containers and `/etc/systemd/system` for root containers). Once setup and enabled, the containers will start and stop automatically as a `systemd` service with the host state transition or manually with the `systemctl` command.



The `podman` command to start and stop containers is no longer needed if the `systemd` setup is in place. You may experience issues if you continue to use `podman` for container state transitioning.

The start and stop behavior for rootless containers differs slightly from that of root containers. For the rootless setup, the containers are started when the relevant user logs in to the host and stopped when that user logs off from all their open terminal sessions; however, this default behavior can be altered by enabling lingering for that user with the `loginctl` command.



User lingering is a feature that, if enabled for a particular user, spawns a user manager for that user at system startup and keeps it running in the background to support long-running services configured for that user. The user need not log in.

EXAM TIP: Make sure that you use a normal user to launch rootless containers and the root user (or sudo) to launch root containers.

Rootless setup does not require elevated privileges of the `root` user.

Exercise 23-9: Configure a Root Container as a `systemd` Service

This exercise should be done on `server20` as `user1` with `sudo` where required.

In this exercise, you will create a `systemd` unit configuration file for managing the state of your root containers. You will launch a new container and use it as a template to generate a service unit file. You will stop and remove the launched container to avoid conflicts with new containers that will start. You will use the `systemctl` command to verify the automatic container start, stop, and deletion.

1. Launch a new container called `root-container` (`--name`) in detached mode (`-d`) using the latest `ubi8`:

```
[user1@server20 ~]$ sudo podman run -dt --name root-container ubi8
Trying to pull registry.access.redhat.com/ubi8...
Getting image source signatures
Copying blob ec1681b6a383 [=====] 70.4MiB / 70.4MiB
Copying blob c4d668e229cd done
Copying config ecbbc6f53bb done
Writing manifest to image destination
Storing signatures
c28c78c625ad2025d4734697194577de81e561531b019e7f4bf8a1d8fde7da5a
```

2. Confirm the new container using *podman ps*. Note the container ID.

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORNS NAMES		
c28c78c625ad	registry.access.redhat.com/ubi8:latest	/bin/bash	12 seconds ago
Up 10 seconds ago	root-container		

3. Create (generate) a service unit file called *root-container.service* under */etc/systemd/system* while ensuring that the next new container that will be launched based on this configuration file will not require the source container (*--new*) to work. The *tee* command will show the generated file content on the screen as well as store it in the specified file.

```
[user1@server20 ~]$ sudo podman generate systemd --new --name root-container |  
sudo tee /etc/systemd/system/root-container.service  
# container-root-container.service  
# autogenerated by Podman 1.9.3  
# Wed Oct 28 08:35:56 EDT 2020  
  
[Unit]  
Description=Podman container-root-container.service  
Documentation=man:podman-generate-systemd(1)  
Wants=network.target  
After=network-online.target  
  
[Service]  
Environment=PODMAN_SYSTEMD_UNIT=%n  
Restart=on-failure  
ExecStartPre=/usr/bin/rm -f %t/%n-pid %t/%n-cid  
ExecStart=/usr/bin/podman run --common-pidfile %t/%n-pid --cidfile %t/%n-cid --  
cgroupons=no-common -d -dt --name root-container ubi8  
ExecStop=/usr/bin/podman stop --ignore --cidfile %t/%n-cid -t 10  
ExecStopPost=/usr/bin/podman rm --ignore -f --cidfile %t/%n-cid  
PIDFile=%t/%n-pid  
KillMode=none  
Type=forking  
  
[Install]  
WantedBy=multi-user.target default.target
```

The unit file has the same syntax as any other systemd service configuration file. There are three sections—Unit, Service, and Install. (1) The unit section provides a short description of the service, the manual page location, and the dependencies (wants and after). (2) The service section highlights the full commands for starting (ExecStart) and stopping (ExecStop) containers. It also highlights the commands that will be executed before the container start (ExecStartPre) and after the container stop (ExecStopPost). There are a number of options and arguments with the commands to ensure a proper

transition. The restart on-failure stipulates that systemd will try to restart the container in the event of a failure. (3) The install section identifies the operational target the host needs to be running in before this container service can start. See [Chapter 12](#) “System Initialization, Message Logging, and System Tuning” for details on systemd units and targets. Also check out the manual pages for `systemd.unit`, `systemd.target`, and `systemd.service` if interested.

4. Stop and delete the source container (*root-container*) using the `stop` and `rm` subcommands:

```
[user1@server20 ~]$ sudo podman stop root-container  
c28c78c625ad2825d4734697194577de81e561531b019e7f4bf8a1d8fde7da5a  
[user1@server20 ~]$  
[user1@server20 ~]$ sudo podman rm root-container  
c28c78c625ad2825d4734697194577de81e561531b019e7f4bf8a1d8fde7da5a
```

Verify the removal by running `podman ps -a`.

5. Update systemd to bring the new service to its control (reboot the system if required):

```
[user1@server20 ~]$ sudo systemctl daemon-reload
```

6. Enable (enable) and start (--now) the container service using the `systemctl` command:

```
[user1@server20 ~]$ sudo systemctl enable --now root-container  
Created symlink /etc/systemd/system/multi-user.target.wants/root-container.service → /etc/systemd/system/root-container.service.  
Created symlink /etc/systemd/system/default.target.wants/root-container.service → /etc/systemd/system/root-container.service.
```

7. Check the running status of the new service with the `systemctl` command:

```
[user1@server20 ~]$ sudo systemctl status root-container  
● root-container.service - Podman container-root-container.service  
   Loaded: loaded (/etc/systemd/system/root-container.service; enabled; vendor>  
   Active: active (running) since Wed 2020-10-28 08:37:35 EDT; 14s ago  
     Docs: man:podman-generate-systemd(1)  
   Process: 4590 ExecStart=/usr/bin/podman run --common-pidfile /run/root-conta>  
   Process: 4588 ExecStartPre=/usr/bin/rm -f /run/root-container.service-pid /r>  
 Main PID: 4666 (common)  
    Tasks: 2 (limit: 11516)  
   Memory: 1.5M  
  CGroup: /system.slice/root-container.service  
          └─4666 /usr/bin/common --api-version 1 -s -c 2260bd01dc503681033853>  
  
Oct 28 08:37:33 server20.example.com systemd[1]: Starting Podman container-roo>  
Oct 28 08:37:35 server20.example.com podman[4590]: 2260bd01dc50368103385340472>  
Oct 28 08:37:35 server20.example.com systemd[1]: Started Podman container-root>
```

8. Verify the launch of a new container (compare the container ID with that of the source root container):

```
[user1@server20 ~]$ sudo podman ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
2260bd01dc50 registry.access.redhat.com/ubi8:latest /bin/bash 2 minutes ago
Up 2 minutes ago root-container
```

9. Restart the container service using the `systemctl` command:

```
[user1@server20 ~]$ sudo systemctl restart root-container
[user1@server20 ~]$
[user1@server20 ~]$ sudo systemctl status root-container
● root-container.service - Podman container-root-container.service
  Loaded: loaded (/etc/systemd/system/root-container.service; enabled; vendor)
    Active: active (running) since Wed 2020-10-28 08:40:28 EDT; 3s ago
      ...
      ...
```

10. Check the status of the container again. Observe the removal of the previous container and the launch of a new container (compare container IDs).

```
[user1@server20 ~]$ sudo podman ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
d06c0154dd61 registry.access.redhat.com/ubi8:latest /bin/bash 8 seconds ago
Up 7 seconds ago root-container
```

Each time the `root-container` service is restarted or `server20` is rebooted, a new container will be launched. You can verify this by comparing their container IDs.

This concludes the exercise.

Exercise 23-10: Configure a Rootless Container as a `systemd` Service

This exercise should be done on `server20` as `user1` with `sudo` where required. Log in as `conuser1` as directed.

In this exercise, you will create a `systemd` unit configuration file for managing the state of your rootless containers. You will launch a new container as `conuser1` (create this user) and use it as a template to generate a service unit file. You will stop and remove the launched container to avoid conflicts with new containers that will start. You will use the `systemctl` command as `conuser1` to verify the automatic container start, stop, and deletion.

1. Create a user account called `conuser1` and assign a simple password:

```
[user1@server20 ~]$ sudo useradd conuser1
[user1@server20 ~]$ echo conuser1 | sudo passwd --stdin conuser1
Changing password for user conuser1.
passwd: all authentication tokens updated successfully.
```

2. Open a new terminal window on *server20* and log in as *conuser1*. Create directory *~/.config/systemd/user* to store a service unit file:

```
[conuser1@server20 ~]$ mkdir ~/.config/systemd/user -p
```

3. Launch a new container called *rootless-container* (--name) in detached mode (-d) using the latest ubi8:

```
[conuser1@server20 ~]$ podman run -dt --name rootless-container ubi8
Trying to pull registry.access.redhat.com/ubi8...
Getting image source signatures
Copying blob c4d668e229cd done
Copying blob ec1681b6a383 done
Copying config ecbc6f53bb done
Writing manifest to image destination
Storing signatures
e84ae12edd17365261ffd97ba29504ca56e16aca63d40ae1205c5f275ba244d1
```

4. Confirm the new container using *podman ps*. Note the container ID.

```
[conuser1@server20 ~]$ podman ps
CONTAINER ID  IMAGE                                COMMAND      CREATED
           STATUS     PORTS NAMES
e84ae12edd17  registry.access.redhat.com/ubi8:latest /bin/bash  35 seconds ago
              Up 34 seconds ago          rootless-container
```

5. Create (*generate*) a service unit file called *rootless-container.service* under *~/.config/systemd/user* while ensuring that the next new container that will be launched based on this configuration file will not require the source container (*--new*) to work:

```
[conuser1@server20 ~]$ podman generate systemd --new --name rootless-container >
~/.config/systemd/user/rootless-container.service
```

6. Display the content of the unit file:

```
[conuser1@server20 ~]$ cat ~/.config/systemd/user/rootless-container.service
# container-rootless-container.service
# autogenerated by Podman 1.9.3
# Tue Oct 27 23:29:06 EDT 2020

[Unit]
Description=Podman container-rootless-container.service
Documentation=man:podman-generate-systemd(1)
Wants=network.target
After=network-online.target

[Service]
Environment=PODMAN_SYSTEMD_UNIT=%n
Restart=on-failure
ExecStartPre=/usr/bin/rm -f %t/%n-pid %t/%n-cid
ExecStart=/usr/bin/podman run --common-pidfile %t/%n-pid --cidfile %t/%n-cid --c
groups=no-common -d --name rootless-container ubi8
ExecStop=/usr/bin/podman stop --ignore --cidfile %t/%n-cid -t 10
ExecStopPost=/usr/bin/podman rm --ignore -f --cidfile %t/%n-cid
PIDFile=%t/%n-pid
KillMode=none
Type=forking

[Install]
WantedBy=multi-user.target default.target
```

See [Exercise 23-9](#) earlier for an analysis of the file content.

7. Stop and delete the source container *rootless-container* using the *stop* and *rm* subcommands:

```
[conuser1@server20 ~]$ podman stop rootless-container
e84ae12edd17365261ffd97ba29504ca56e16aca63d40ae1205c5f275ba244d1
[conuser1@server20 ~]$
[conuser1@server20 ~]$ podman rm rootless-container
e84ae12edd17365261ffd97ba29504ca56e16aca63d40ae1205c5f275ba244d1
```

Verify the removal by running **podman ps -a**.

8. Update systemd to bring the new service to its control (reboot the system if required). Use the *--user* option with the *systemctl* command.

```
[conuser1@server20 ~]$ systemctl --user daemon-reload
```

9. Enable (enable) and start (--now) the container service using the *systemctl* command:

```
[conuser1@server20 ~]$ systemctl --user enable --now rootless-container.service
Created symlink /home/conuser1/.config/systemd/user/multi-user.target.wants/root
less-container.service → /home/conuser1/.config/systemd/user/rootless-container.
service.
Created symlink /home/conuser1/.config/systemd/user/default.target.wants/rootles
s-container.service → /home/conuser1/.config/systemd/user/rootless-container.ser
vice.
```

10. Check the running status of the new service using the *systemctl* command:

```
[conuser1@server20 ~]$ systemctl --user status rootless-container
● rootless-container.service - Podman container-rootless-container.service
   Loaded: loaded (/home/conuser1/.config/systemd/user/rootless-container.servi>
   Active: active (running) since Wed 2020-10-28 09:12:30 EDT; 43s ago
     Docs: man:podman-generate-systemd(1)
 Process: 3636 ExecStart=/usr/bin/podman run --common-pidfile /run/user/3001/>
 Process: 3635 ExecStartPre=/usr/bin/rm -f /run/user/3001/rootless-container.>
Main PID: 3654 (common)
   CGroup: /user.slice/user-3001.slice/user@3001.service/rootless-container.se>
           └─3646 /usr/bin/fuse-overlayfs -o lowerdir=/home/conuser1/.local/sh>
             ├─3650 /usr/bin/slirp4netns --disable-host-loopback --mtu 65520 --e>
             ├─3654 /usr/bin/common --api-version 1 -c 49139787c70f9ea41b12433da>
             ├─49139787c70f9ea41b12433da53379b43d0012d197a15fd180fc64f53b3f07fe
             └─3665 /bin/bash
```

- Verify the launch of a new container (compare the container ID with that of the source rootless container):

```
[conuser1@server20 ~]$ podman ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
49139787c70f registry.access.redhat.com/ubi8:latest /bin/bash 2 minutes ago
Up 2 minutes ago rootless-container
```

- Enable the container service to start and stop with host transition using the *logind* command (systemd login manager) and confirm:

```
[conuser1@server20 ~]$ logctl enable-linger
[conuser1@server20 ~]$ logctl show-user conuser1 | grep -i linger
Linger=yes
```

- Restart the container service using the *systemctl* command:

```
[conuser1@server20 ~]$ systemctl --user restart rootless-container
[conuser1@server20 ~]$ systemctl --user status rootless-container
● rootless-container.service - Podman container-rootless-container.service
   Loaded: loaded (/home/conuser1/.config/systemd/user/rootless-container.servi>
   Active: active (running) since Wed 2020-10-28 09:17:09 EDT; 3s ago
     ....
```

- Check the status of the container again. Observe the removal of the previous container and the launch of a new container (compare container IDs).

```
[conuser1@server20 ~]$ podman ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
aa3114703427 registry.access.redhat.com/ubi8:latest /bin/bash 9 seconds ago
Up 8 seconds ago rootless-container
```

Each time the *rootless-container* service is restarted or *server20* is rebooted, a new container will be launched. You can verify this by comparing their container IDs.

This concludes the exercise.

Chapter Summary

In this last chapter, we discussed the container technology that has gained tremendous popularity and been deployed in massive numbers over the recent years. It offers benefits that are unmatched with any previous virtualization technology.

We analyzed the core components of the technology: images, registries, and containers. We interacted with images located in remote registries and local storage. We looked at a variety of ways to launch containers, with and without a name. We examined the benefits and use cases for mapping network ports, injecting environment variables, and making host storage available inside containers for persistent data storage. The last topic expounded on controlling the operational states of root and rootless containers via `systemd` and ensuring that they are auto-started and auto-stopped with the host startup and shutdown. The chapter presented several exercises to reinforce the learning.

Review Questions

1. How would you ensure that data stored inside a container will not be deleted automatically when the container is restarted?
2. What is the name of the primary tool that is used for container management?
3. Which option must be included with the `systemctl` command to manage the state of a rootless container?
4. Rootless containers are less secure than root containers. True or False?
5. What is one thing that you do not get with rootless containers? Select from: security, ability to map privileged ports, ability to pass environment variables, or can be managed via `systemd`.
6. What is the significance of a tag in a container image?
7. Which command would you use to inspect a container image sitting in a remote registry?
8. What would you run to remove all locally stored images in one shot?
9. Containers can be run on a bare metal server or in a virtual machine. True or False?
10. Which feature would you use to allow traffic flow for your Apache web server running inside a container?
11. It is mandatory to download an image prior to running a container based off it. True or False?
12. How would you connect to a container running in the background?
13. By default, any data saved inside a container is lost when the container is restarted. True or False?

14. Can a single host directory be attached to multiple containers at the same time?
15. Which of these—isolation, security, high transition time, less overhead—is not a benefit of the container technology?
16. What is the name and location of the system-wide file that stores a list of insecure registries?
17. How would you ensure that a rootless container for a specific user is automatically started with the host startup without the need for the user to log in?
18. What would you run to remove all stopped containers?
19. What is the default tag used with container images?
20. The per-user `systemd` service configuration file is stored under `/etc/systemd/user` directory. True or False?

Answers to Review Questions

1. Attach a host directory to the container and use it for storing data that requires persistence.
2. The primary container management tool is called *podman*.
3. The `--user` option is required with the `systemctl` command to control the state of a rootless container.
4. False.
5. The ability to map privileged ports is not directly supported with rootless containers.
6. A tag is typically used to identify a version of the image.
7. The `skopeo` command can be used to inspect an image located in a remote registry.
8. Execute `podman rmi -a` to remove all locally stored images.
9. True.
10. Map a host port with a container port.
11. False. The specified image is automatically downloaded when starting a container.
12. Use the `podman attach` command to connect to a running container.
13. True.
14. Yes, a single host directory can be attached to multiple containers simultaneously.
15. High transition time is not a container benefit.
16. The file that stores insecure registry list is called `registries.conf` and it lives in the `/etc/containers` directory.
17. Use the `logind` command as that user and enable lingering.
18. Run `podman rm -a` to remove all stopped containers.
19. The default tag is ‘latest’ that identifies the latest version of an image.

20. False. The per-user systemd unit configuration file is stored under the user's home directory at `~/.config/systemd/user`.

DIY Challenge Labs

The following labs are useful to strengthen most of the concepts and topics learned in this chapter. It is expected that you perform these labs without any additional help. A step-by-step guide is not provided, as the implementation of these labs requires the knowledge that has been presented in this chapter. Use defaults or your own thinking for missing information.

Lab 23-1: Prepare to Launch Containers

Create a new user account called `conadm` on `server10` and give them full `sudo` rights. As `conadm`, register `server10` with RHSM using your Red Hat credentials. Install the module `container-tools` from RHSM and ensure that `podman` version is 1.9. (Hint: Chapters 05, 06, and 10).

Lab 23-2: Launch a Named Root Container with Port Mapping

As `conadm` with `sudo` (where required) on `server10`, inspect the latest version of RHEL 7 image and then download it to your computer. Launch a container called `root-cont-port` in attached terminal mode (`-it`) with host port 80 mapped to container port 8080. Run a few basic Linux commands such as `ls`, `pwd`, `df`, `cat /etc/redhat-release`, and `os-release` while in the container. Check to confirm the port mapping from `server10`. (Hint: use the `skopeo` and `podman` commands). **Note:** Do not remove the container yet.

Lab 23-3: Launch a Nameless Rootless Container with Two Variables

As `conadm` on `server10`, launch a container using the latest version of ubi8 image in detached mode (`-d`) with two environment variables `VAR1=lab1` and `VAR2=lab2` defined. Use the `exec` subcommand to check the settings for the variables directly from `server10`. Delete the container and the image when done. (Hint: use the `podman` command).

Lab 23-4: Launch a Named Rootless Container with Persistent Storage

As `conadm` with `sudo` (where required) on `server10`, create a directory called `/host_perm1` with full permissions, and a file called `str1` in it. Launch a

container called *rootless-cont-str* in attached terminal mode (-it) with the created directory mapped to */cont_perm1* inside the container. While in the container, check access to the directory and the presence of the file. Create a sub-directory and a file under */cont_perm1* and exit out of the container shell. List */host_perm1* on *server10* to verify the sub-directory and the file. Stop and delete the container. Remove */host_perm1*. (Hint: use the *podman* command).

Lab 23-5: Launch a Named Rootless Container with Port Mapping, Environment Variables, and Persistent Storage

As *conadm* with *sudo* (where required) on *server10*, launch a named rootless container called *rootless-cont-adv* in attached mode (-it) with two variables (HISTSIZE=100 and MYNAME=RedHat), host port 9000 mapped to container port 8080, and */host_perm2* mounted at */cont_perm2*. Check and confirm the settings while inside the container. Exit out of the container. **Note:** Do not remove the container yet. (Hint: use the *podman* command).

Lab 23-6: Control Rootless Container States via *systemd*

As *conadm* on *server10*, use the *rootless-cont-adv* container launched in Lab 23-5 as a template and generate a *systemd* service configuration file and store the file in the appropriate directory. Stop and remove the source container *rootless-cont-adv*. Add the support for the new service to *systemd* and enable the new service to auto-start at system reboots. Perform the required setup to ensure the container is launched without the need for the *conadm* user to log in. Reboot *server10* and confirm a successful start of the container service and the container. (Hint: use the *podman*, *systemctl*, and *loginctl* commands).

Lab 23-7: Control Root Container States via *systemd*

As *conadm* with *sudo* where required on *server10*, use the *root-cont-port* container launched in Lab 23-2 as a template and generate a *systemd* service configuration file and store the file in the appropriate directory. Stop and remove the source container *root-cont-port*. Add the support for the new service to *systemd* and enable the service to auto-start at system reboots. Reboot *server10* and confirm a successful start of the container service and the container. (Hint: use the *podman* and *systemctl* commands).

Appendix A: Sample RHCSA Exam 1

Time Duration: 3 hours

Passing Score: 70% (210 out of 300)

Instructions: The RHCSA exam, EX200, is offered electronically on a physical

computer running RHEL 8. The computer has two virtual machines with RHEL 8 running in each one of them. The exam presents two sets of tasks that are to be completed within the stipulated time in the identified virtual machine. Firewall and SELinux need to be considered. All settings performed in the virtual machines must survive reboots, or you will lose marks. Access to the Internet, printed and electronic material, and electronic devices is prohibited during the exam.

Setup for Sample Exam 1:

Build a virtual machine with RHEL 8 Server with GUI (Exercises 1-1 and 1-2). Use a 10GB disk for the OS with default partitioning. Add 2x300MB disks and a network interface. Do not configure the network interface or create a normal user account during installation.

Instructions:

- 01:** The following tasks are in addition to the exercises and labs presented in the book. No solutions are furnished, but hints to applicable exercises, chapters, or topics are provided in parentheses for reference.
- 02:** Please do not browse the Internet or seek help from other sources. However, you can refer to the manual pages, and the documentation under the /usr/share/doc directory. This rule does not apply to the kernel download task if included.
- 03:** The exam tasks should be completed in a terminal window using only the command line interface (no GUI).
- 04:** You can reboot the VM whenever you want during this exam, but retest the configuration after each reboot for verification.
- 05:** Use your knowledge and judgement for any missing configuration in task description.

Tasks:

Task 01: Assuming the root user password is lost, and your system is running in multi-user target with no current root session open. Reboot the system into an appropriate target level, and reset the root user password to root1234. ([Exercise 11-2](#)). After completing this task, log in as the root user and perform the remaining tasks presented below.

Task 02: Using a manual method (create/modify files by hand), configure a network connection on the primary network device with IP address 192.168.0.241/24, gateway 192.168.0.1, and nameserver 192.168.0.1. Use different IP assignments based on your lab setup. ([Exercise 16-3](#)).

Task 03: Using a manual method (modify file by hand), set the system hostname to [rhcsa1.example.com](#) and alias rhcsa1. Make sure that the new hostname is reflected in the command prompt. ([Exercises 16-1 and 16-5](#)).

Task 04: Set the default boot target to multi-user. ([Chapter 12](#), topic: Managing Target Units).

Task 05: Set SELinux to permissive mode. ([Chapter 21](#), topic: Viewing and Controlling SELinux Operational State).

Task 06: Perform a case-insensitive search for all lines in the /usr/share/dict/linux.words file that begin with the pattern “essential”. Redirect the output to /var/tmp/pattern.txt file. Make sure that empty lines are omitted. ([Chapter 07](#), topic: Regular Expressions).

Task 07: Change the primary command prompt for the root user to display the hostname, username, and current working directory information in that order. Update the per-user initialization file for permanence. ([Exercise 7-1](#)).

Task 08: Create user accounts called user10, user20, and user30. Set their passwords to Temp1234. Make user10 and user30 accounts to expire on December 31, 2021. ([Exercises 5-1, and 6-1 or 6-2](#)).

Task 09: Create a group called group10 and add user20 and user30 as secondary members. ([Exercise 6-4](#)).

Task 10: Create a user account called user40 with UID 2929. Set the password to user1234. ([Exercise 5-2](#)).

Task 11: Create a directory called dir1 under /var/tmp with ownership and owning group set to root. Configure default ACLs on the directory and give user10 read, write, and execute permissions. ([Exercise 4-8](#)).

Task 12: Attach the RHEL 8 ISO image to the VM and mount it persistently to /mnt/cdrom. Define access to both repositories and confirm. ([Exercise 10-1](#)).

Task 13: Create a logical volume called lvol1 of size 280MB in vgtest volume group. Mount the ext4 file system persistently to /mnt/mnt1.

([Exercises 14-1, 14-2, and 15-3](#)).

Task 14: Change group membership on /mnt/mnt1 to group10. Set read/write/execute permissions on /mnt/mnt1 for group members, and revoke all permissions for public. ([Exercises 6-4, 6-6, and either 4-1 or 4-2](#)).

Task 15: Create a logical volume called lvswap of size 280MB in vgtest volume group. Initialize the logical volume for swap use. Use the UUID and place an entry for persistence. ([Exercise 15-6](#)).

Task 16: Use the combination of tar and bzip2 commands to create a compressed archive of the /usr/lib directory. Store the archive under /var/tmp as usr.tar.bz2. ([Exercise 3-1](#)).

Task 17: Create a directory hierarchy /dir1/dir2/dir3/dir4 and apply SELinux contexts of /etc on it recursively. ([Chapter 03](#), topic: Creating Files and Directories, and [Exercise 21-2](#)).

Task 18: Enable access to the atd service for user20 and deny for user30. ([Chapter 08](#), topic: Controlling User Access).

Task 19: Add a custom message “This is RHCSA sample exam on \$(date) by \$LOGNAME” to the /var/log/messages file as the root user. Use regular expression to confirm the message entry to the log file. ([Chapter 07](#), topic: Regular Expressions, and [Chapter 12](#), topic: Logging Custom Messages).

Task 20: Allow user20 to use sudo without being prompted for their password. ([Chapter 06](#), topic: Doing as Superuser (or Doing as Substitute User)).

Task 21: Write a bash shell script to create three user accounts—user555, user666, and user777—with no login shell and passwords matching their usernames. The script should also extract the three usernames from the /etc/passwd file and redirect them into /var/tmp/newusers. ([Chapter 22: Script12](#) and [Chapter 07](#), topics: Regular Expressions, and Input, Output, and Error Redirections).

Task 22: Launch a simple container as user20 using the latest version of ubi7 image. Configure the container to auto-start at

system reboots without the need for user20 to log in. ([Exercise 23-10](#)).

Task 23: Launch another container as user20 using the latest version of ubi8 image with two environment variables SHELL and HOSTNAME. Configure the container to auto-start via systemd without the need for user20 to log in. Connect to the container and verify variable settings. ([Exercise 23-7](#) and [23-10](#)).

Reboot the system and validate the configuration.

Appendix B: Sample RHCSA Exam 2

Time Duration: 3 hours
Passing Score: 70% (210 out of 300)
Instructions: The RHCSA exam, EX200, is offered electronically on a physical computer running RHEL 8. The computer has two virtual machines with RHEL 8 running in each one of them. The exam presents two sets of tasks that are to be completed within the stipulated time in the identified virtual machine. Firewall and SELinux need to be considered. All settings performed in the virtual machines must survive reboots, or you will lose marks. Access to the Internet, printed and electronic material, and electronic devices is prohibited during the exam.

Setup for Sample Exam 2:

Build a virtual machine with RHEL 8 Server with GUI (Exercises 1-1 and 1-2). Use a 10GB disk for the OS with default partitioning. Add 1x400MB disk and a network interface. Do not configure the network interface or create a normal user account during installation.

Instructions:

01: The following tasks are in addition to the exercises and labs presented in the book. No solutions are furnished, but hints to applicable exercises, chapters, or topics are provided in parentheses for reference.

02: Please do not browse the Internet or seek help from other sources. However, you can refer to the manual pages, and the documentation under the /usr/share/doc directory. This rule does not apply to the kernel download task if included.

03: The exam tasks should be completed in a terminal window using only the command line interface (no GUI).

04: You can reboot the VM whenever you want during this exam, but retest the configuration after each reboot for verification.

05: Use your knowledge and judgement for any missing configuration in task description.

Tasks:

Task 01: Using the nmcli command, configure a network connection on the primary network device with IP address 192.168.0.242/24, gateway 192.168.0.1, and nameserver 192.168.0.1. Use different IP assignments based on your lab environment. ([Exercise 16-4](#)).

Task 02: Using the hostnamectl command, set the system hostname to [rhcsa2.example.com](#) and alias rhcsa2. Make sure that the new hostname is reflected in the command prompt. ([Exercises 16-1](#) and [16-5](#)).

Task 03: Create a user account called user70 with UID 7000 and comments “I am user70”. Set the maximum allowable inactivity for this user to 30 days. ([Exercises 5-2](#), and [6-1](#) or [6-2](#)).

Task 04: Create a user account called user50 with a non-interactive shell. ([Exercise 5-4](#)).

Task 05: Create a file called testfile1 under /var/tmp with ownership and owning group set to root. Configure access ACLs on the file and give user10 read and write access. Test access by logging in as user10 and editing the file. ([Chapter 03](#), topic: Creating Files and Directories, and [Exercise 4-7](#)).

Task 06: Attach the RHEL 8 ISO image to the VM and mount it persistently to /mnt/dvdrom. Define access to both repositories and confirm. ([Exercise 10-1](#)).

Task 07: Create a logical volume called lv1 of size equal to 10 LEs in vg1 volume group (create vg1 with PE size 8MB in a partition on the 400MB disk). Initialize the logical volume with XFS type and mount it on /mnt/lvfs1. Create a file called lv1file1 in the mount point. Set the file system to automatically mount at each system reboot. ([Exercises 14-1, 14-2](#), and [15-3](#)).

Task 08: Add a group called group20 and change group membership on /mnt/lvfs1 to group20. Set read/write/execute permissions on /mnt/lvfs1 for the owner, group members, and others. ([Exercises 6-4, 6-6](#), and [either 4-1 or 4-2](#)).

Task 09: Extend the file system in the logical volume lv1 by 64MB without unmounting it and without losing any data. Confirm the new size for the logical volume and the file system. ([Exercise 15-4](#)).

Task 10: Create a swap partition of size 85MB on the 400MB disk. Use its UUID and ensure it is activated after every system reboot. ([Exercise 15-6](#)).

Task 11: Create a disk partition of size 100MB on the 400MB disk and format it with Ext4 file system structures. Assign label stdlabel to the file system. Mount the file system on /mnt/stdfs1 persistently using the label. Create file stdfile1 in the mount point. ([Exercise 13-2](#) or [13-4](#), [Chapter 15](#), topic: Labeling a File System, and [Exercise 15-1](#)).

Task 12: Use the tar and gzip command combination to create a compressed archive of the /etc directory. Store the archive under

/var/tmp using a filename of your choice. ([Exercise 3-1](#)).

Task 13: Create a directory /direct01 and apply SELinux contexts for /root to it. ([Exercise 21-2](#)).

Task 14: Set up a cron job for user70 to search for files by the name “core” in the /var directory and copy them to the directory /var/tmp/coredir1. This job should run every Monday at 1:20 a.m. ([Chapter 04](#), topics: Using the find Command, and Using find with -exec and -ok Flags, and [Exercise 8-4](#)).

Task 15: Search for all files in the entire directory structure that have been modified in the past 30 days and save the file listing in the /var/tmp/modfiles.txt file. ([Chapter 04](#), topics: Using the find Command and Using find with -exec and -ok Flags).

Task 16: Modify the bootloader program and set the default autoboot timer value to 2 seconds. ([Exercise 11-1](#)).

Task 17: Determine the recommended tuning profile for the system and apply it. ([Exercise 12-2](#)).

Task 18: Configure Chrony to synchronize system time with the hardware clock. Remove all other NTP sources. ([Exercise 18-1](#)).

Task 19: Install package group called “Development Tools” and capture its information in /var/tmp/systemtools.out file. ([Chapter 03](#), topic: Regular Expressions, and [Exercise 10-3](#)).

Task 20: Lock user account user70. Use regular expressions to capture the line that shows the lock and store the output in file /var/tmp/user70.lock. ([Chapter 03](#), topic: Regular Expressions, and [Exercise 6-3](#)).

Task 21: Write a bash shell script so that it prints RHCSA when RHCE is passed as an argument, and vice versa. If no argument is provided, the script should print a usage message and quit with exit value 5. ([Chapter 22: Script10](#)).

Task 22: Launch a root container and configure it to auto-start via systemd. ([Exercise 23-9](#)).

Task 23: Launch a container as user80 with /data01 mapped to /data01 using the latest version of the ubi8 image. Configure a systemd service to auto-start the container on system reboots without the need for user80 to log in. Create files under the shared mount point and validate data persistence. ([Exercise 23-7](#) and [23-10](#)).

Reboot the system and validate the configuration.

Appendix C: Sample RHCSA Exam 3

Time Duration: 3 hours
Passing Score: 70% (210 out of 300)
Instructions: The RHCSA exam, EX200, is offered electronically on a physical computer running RHEL 8. The computer has two virtual machines with RHEL 8 running in each one of them. The exam presents two sets of tasks that are to be completed within the stipulated time in the identified virtual machine. Firewall and SELinux need to be considered. All settings performed in the virtual machines must survive reboots, or you will lose marks. Access to the Internet, printed and electronic material, and electronic devices is prohibited during the exam.

Setup for Sample Exam 3:

Build two virtual machines with RHEL 8 Server with GUI (Exercises 1-1 and 1-2). Use a 10GB disk for the OS with default partitioning. Add 1x4GB disk to VM1, 2x1GB disks to VM2, and a network interface to both virtual machines. Do not configure the network

interfaces or create a normal user account during installation.

Instructions:

- 01:** The following tasks are in addition to the exercises and labs presented in the book. No solutions are furnished, but hints to applicable exercises, chapters, or topics are provided in parentheses for reference.
- 02:** Please do not browse the Internet or seek help from other sources. However, you can refer to the manual pages, and the documentation under the /usr/share/doc directory. This rule does not apply to the kernel download task if included.
- 03:** The exam tasks should be completed in a terminal window using only the command line interface (no GUI).
- 04:** You can reboot the VM whenever you want during this exam, but retest the configuration after each reboot for verification.
- 05:** Use your knowledge and judgement for any missing configuration in task description.

Tasks:

- Task 01:** On VM1, set the system hostname to rhcsa3.example.com and alias rhcsa3 using the hostnamectl command. Make sure that the new hostname is reflected in the command prompt.
[\(Exercises 16-1 and 16-5\)](#).
- Task 02:** On rhcsa3, configure a network connection on the primary network device with IP address 192.168.0.243/24, gateway 192.168.0.1, and nameserver 192.168.0.1 using the nmcli command (use different IP assignments based on your lab environment).
[\(Exercise 16-4\)](#).

Task 03: On VM2, set the system hostname to [rhcsa4.example.com](#) and alias rhcsa4 using a manual method (modify file by hand). Make sure that the new hostname is reflected in the command prompt. ([Exercises 16-1 and 16-5](#)).

Task 04: On rhcsa4, configure a network connection on the primary network device with IP address 192.168.0.244/24, gateway 192.168.0.1, and nameserver 192.168.0.1 using a manual method (create/modify file by hand). Use different IP assignments based on your lab environment. ([Exercise 16-3](#)).

Task 05: Run “ping -c2 rhcsa4” on rhcsa3. Run “ping -c2 rhcsa3” on rhcsa4. You should see 0% loss in both outputs. ([Exercise 16-5](#)).

Task 06: On rhcsa3 and rhcsa4, attach the RHEL 8 ISO image to the VM and mount it persistently to /mnt/sr0. Define access to both repositories and confirm. ([Exercise 10-1](#)).

Task 07: On rhcsa3, add HTTP port 8300/tcp to the SELinux policy database persistently. ([Exercise 21-3](#)).

Task 08: On rhcsa3, create VDO volume called vdo1 on the 4GB disk with logical size 16GB and mounted with Ext4 structures on /mnt/vdo1. ([Exercises 13-6 and 13-7](#)).

Task 09: Configure NFS service on rhcsa3 and share /rh_share3 with rhcsa4. Configure AutoFS direct map on rhcsa4 to mount /rh_share3 on /mnt/rh_share4. User user80 (create on both systems) should be able to create files under the share on the NFS server as well as under the mount point on the NFS client. ([Exercises 5-1, 17-1, and 17-3](#)).

Task 10: Configure NFS service on rhcsa4 and share the home directory for user60 (create user60 on both systems) with rhcsa3. Configure AutoFS indirect map on rhcsa3 to automatically mount the home directory under /nfsdir when user60 logs on to rhcsa3. ([Exercises 5-1, 17-1, 17-4, and 17-5](#)).

Task 11: On rhcsa4, create Stratis pool pool1 and volume str1 on a 1GB disk and mount it to /mnt/str1. ([Exercise 15-5](#)).

Task 12: On rhcsa4, expand Stratis pool pool1 using the other 1GB disk. Confirm that /mnt/str1 sees the storage expansion. ([Exercise 15-5](#)).

Task 13: On rhcsa3, create a group called group30 with GID 3000, and add user60 and user80 to this group. Create a directory called /sdata, enable setgid bit on it, and add write permission bit for group members. Set ownership and owning group to root and group30. Create a file called file1 under /sdata as user60 and modify the file as user80 successfully. ([Exercises 4-5, 6-4](#), and [6-6](#)).

Task 14: On rhcsa3, create directory /var/dir1 with full permissions for everyone. Disallow non-owners to remove files. Test by creating file /var/dir1/stkfile1 as user60 and removing it as user80. ([Exercise 4-6](#)).

Task 15: On rhcsa3, search for all manual pages for the description containing the keyword “password” and redirect the output to file /var/tmp/man.out. ([Chapter 02](#), topic Searching by Keyword, and [Chapter 07](#), topic: Input, Output, and Error Redirections).

Task 16: On rhcsa3, create file Infile1 under /var/tmp and create one hard link /var/tmp/Infile2 and one soft link /boot/file1. Edit Infile1 using one link at a time and confirm. ([Exercises 3-2 and 3-3](#)).

Task 17: On rhcsa3, install module postgresql version 9.6 (select a different non-default version if 9.6 is not available). ([Exercise 10-5](#)).

Task 18: On rhcsa3, add the http service to the “external” firewalld zone persistently. ([Exercise 201](#)).

Task 19: On rhcsa3, set SELinux type shadow_t on a new file testfile1 in /usr and ensure that the context is not affected by a SELinux relabeling. ([Exercises 21-1 and 21-2](#)).

Task 20: Configure password-less ssh access for user60 from rhcsa3 to rhcsa4. ([Exercise 19-2](#)).

Task 21: Write a bash shell script that checks for the existence of files (not directories) under the /usr/bin directory that begin with the letters “ac” and display their statistics (the stat command). ([Chapter 22: Table 22-1](#) and [Script07](#)).

Task 22: On rhcsa3, launch a named container as user60 with host port 10000 mapped to container port 80. Employ the latest version of the ubi7 image. Configure a systemd service to auto-start the container without the need for user60 to log in. Validate port mapping using an appropriate podman subcommand. ([Exercises 23-5](#) and [23-10](#)).

Task 23: On rhcsa3, launch another named container as user60 with /host_data01 mapped to /container_data01, one variable ENVIRON=Exam, and host port 1050 mapped to container port 1050. Use the latest version of the ubi8 image. Configure a separate systemd service to auto-start the container without the need for user60 to log in. Create a file under the shared directory and validate data persistence. Verify port mapping and variable settings using appropriate podman subcommands. ([Exercises 23-5, 23-7, 23-8](#), and [23-10](#)).

Reboot the system and validate the configuration.

Appendix D: Sample RHCSA Exam 4

Time Duration: 3 hours
Passing Score: 70% (210 out of 300)
Instructions: The RHCSA exam, EX200, is offered electronically on a physical computer running RHEL 8. The computer has two virtual machines with RHEL 8 running in each one of them. The exam presents two sets of tasks that are to be completed within the stipulated time in the identified virtual machine. Firewall and SELinux need to be considered. All settings performed in the virtual machines must survive reboots, or you will lose marks. Access to the Internet, printed and electronic material, and electronic devices is prohibited during the exam.

Setup for Sample Exam 4:

Build two virtual machines with RHEL 8 Server with GUI (Exercises 1-1 and 1-2). Use a 10GB disk for the OS with default partitioning. Add 1x4GB disk to VM2 and a network interface to both virtual machines. Do not configure the network interfaces or create a normal user account during installation.

Instructions:

01: The following tasks are in addition to the exercises and labs presented in the book. No solutions are furnished, but hints to applicable exercises, chapters, or topics are provided in parentheses for reference.

02: Please do not browse the Internet or seek help from other sources. However, you can refer to the manual pages, and the documentation under the /usr/share/doc directory. This rule does not apply to the kernel download task if included.

03: The exam tasks should be completed in a terminal window using only the command line interface (no GUI).

04: You can reboot the VM whenever you want during this exam, but retest the configuration after each reboot for verification.

05: Use your knowledge and judgement for any missing configuration in task description.

Tasks:

Task 01: On VM1, set the system hostname to [rhcsa5.example.com](#) and alias rhcsa5 using the hostnamectl command. Make sure that the new hostname is reflected in the command prompt.
[\(Exercises 16-1 and 16-5\)](#).

Task 02: On rhcsa5, configure a network connection on the primary network device with IP address 192.168.0.245/24, gateway 192.168.0.1, and nameserver 192.168.0.1 using the nmcli command. Use different IP assignments based on your lab environment.
[\(Exercise 16-4\)](#).

Task 03: On VM2, set the system hostname to [rhcsa6.example.com](#) and alias rhcsa6 using a manual method (modify file by hand). Make

sure that the new hostname is reflected in the command prompt. ([Exercises 16-1 and 16-5](#)).

Task 04: On rhcsa6, configure a network connection on the primary network device with IP address 192.168.0.246/24, gateway 192.168.0.1, and nameserver 192.168.0.1 using a manual method (create/modify files by hand). Use different IP assignments based on your lab environment. ([Exercise 16-3](#)).

Task 05: Run “ping -c2 rhcsa6” on rhcsa5. Run “ping -c2 rhcsa5” on rhcsa6. You should see 0% loss in both outputs. ([Exercise 16-5](#)).

Task 06: On rhcsa5 and rhcsa6, attach the RHEL 8 ISO image to the VM and mount it persistently to /mnt/sr0. Define access to both repositories and confirm. ([Exercise 10-1](#)).

Task 07: Export /share5 on rhcsa5 and mount it to /share6 persistently on rhcsa6. ([Exercises 17-1 and 17-2](#)).

Task 08: Use NFS to export home directories for all users (u1, u2, and u3) on rhcsa6 so that their home directories become available under /home1 when they log on to rhcsa5. Create u1, u2, and u3. ([Exercises 17-1 and 17-5](#)).

Task 09: On rhcsa5, add HTTP port 8400/udp to the public zone persistently. ([Exercise 21-3](#)).

Task 10: Configure password-less ssh access for u1 from rhcsa5 to rhcsa6. Copy the directory /etc/sysconfig from rhcsa5 to rhcsa6 under /var/tmp/remote securely. ([Exercise 19-2](#), and Chapter 19, topic: Copying Files Remotely Using scp).

Task 11: On rhcsa6, create VDO volume vdo2 on the 4GB disk with logical size 16GB and mounted persistently with XFS structures on /mnt/vdo2. ([Exercises 13-6 and 13-7](#)).

Task 12: On rhcsa6, install module “container-tools” stream rhel8. ([Exercise 10-5](#)).

Task 13: On rhcsa6, flip the value of the Boolean nfs_export_all_rw persistently. ([Exercise 21-5](#)).

Task 14: On rhcsa5 and rhcsa6, set the tuning profile to powersave. ([Exercise 12-2](#)).

Task 15: On rhcsa5, create file Infile1 under /var/tmp and create three hard links called hard1, hard2, and hard3 for it. Identify the inode number associated with all four files. Edit any of the files and observe the metadata for all the files for confirmation. ([Exercise 3-2](#)).

Task 16: On rhcsa5, members (user100 and user200) of group100 should be able to collaborate on files under /shared but cannot delete each other's files. ([Exercises 4-5 and 4-6](#)).

Task 17: Synchronize the entire /etc directory on rhcsa5 to /var/tmp/etc on rhcsa6. Use in-transit compression. Capture the output and any errors in the /var/tmp/etc.transfer file on rhcsa5 during the synchronization process. ([Chapter 19](#), topic: Synchronizing Files Remotely Using rsync, and [Chapter 07](#), topic: Regular Expressions).

Task 18: On rhcsa6, list all files that are part of the “setup” package, and use regular expressions and I/O redirection to send the output lines containing “hosts” to /var/tmp/setup.pkg. ([Exercise 9-2](#), and [Chapter 07](#), topics: Regular Expressions, and Input, Output, and Error Redirections).

Task 19: On rhcsa5, check the current version of the Linux kernel. Register rhcsa5 with RHSM and install the latest version of the kernel available. Ensure that the existing kernel and its configuration remain intact. Reboot the system and confirm the new version is loaded. ([Exercise 11-3](#), and [Chapter 02](#), topic: Viewing System Information).

Task 20: On rhcsa5, configure journald to store messages permanently under /var/log/journal and fall back to memory-only option if /var/log/journal directory does not exist or has permission/access issues. ([Exercise 12-1](#)).

Task 21: Write a bash shell script that defines an environment variable called ENV1=book1 and creates a user account that matches the value of the variable. ([Chapter 22](#): Script02 and Script03).

Task 22: On rhcsa5, launch a named root container with host port 443 mapped to container port 443. Employ the latest version of the ubi7 image. Configure a systemd service to auto-start the container at system reboots. Validate port mapping using an appropriate podman subcommand. ([Exercises 23-5 and 23-9](#)).

Task 23: On rhcsa5, launch a named container as user100 with /data01 mapped to /data01 and two variables KERN=\$(uname -r) and SHELL defined. Use the latest version of the ubi8 image. Configure a systemd service to auto-start the container at system reboots without the need for user100 to log in. Create a file under the shared mount point and validate data persistence. Verify port mapping using an appropriate podman subcommand. ([Exercises 23-7, 23-8, and 23-10](#)).

Reboot the system and validate the configuration.

Bibliography

The following websites, forums, and guides were referenced in writing this book:

1. www.virtualbox.org
2. docs.redhat.com/docs/en-US
3. developers.redhat.com
4. www.redhat.com
5. www.opensource.org
6. www.systemd.io
7. www.tldp.org
8. wiki.archlinux.org
9. www.ibm.com
10. www.centos.org
11. www.wikipedia.org
12. www.linux.org
13. www.firewalld.org
14. www.apache.org
15. www.gnome.org
16. www.ietf.org
17. www.isc.org
18. www.netfilter.org
19. www.nftables.org
20. www.nsa.gov/research/selinux
21. www.ntp.org
22. www.chrony.tuxfamily.org
23. www.openssh.org
24. www.pathname.com/fhs
25. www.docker.io
26. RHCSA Red Hat Enterprise Linux 8 book by Asghar Ghori

27. Red Hat Certified System Administrator & Engineer for RHEL 7 book by Asghar Ghori
28. Red Hat Certified System Administrator & Engineer for RHEL 6 book by Asghar Ghori

Glossary

. (single dot)	Represents current directory.
.. (double dots)	Represents parent directory of the current directory.
Absolute mode	A method of permission allocation to a file or directory.
Absolute path	A pathname that begins with a /.
Access ACLs	ACL settings applied to files.
Access Control List	A method of allocating file permissions to a specific user or group. See Named user and Named group.
Access mode	See Permission mode.
Access permission	See File permission.
Access right	See File permission.
Access Cache Vector	A special cache area that SELinux uses to store its decisions.
ACL	See Access Control List.
ACL mask	Controls the maximum permissions a named user or named group can have.
Address Resolution Protocol	A protocol used to determine a system's Ethernet address when its IP address is known.
Address space	Memory location that a process can refer.
Administrator	See Superuser.
Algorithm	A set of well-defined but complex mathematical instructions used for data encryption and decryption.
Alias	A short name to refer to a lengthy command.
Alias substitution	See Alias.
Anaconda	RHEL's installation program.
Anacron	A service that runs missing cron and at jobs after a system reboot.
Apache	A popular HTTP web server software.
Application module	A complete set of packages to install a software application.
Application stream	A method of making multiple versions of a software application available for installation from the same repository.

AppStream	One of the yum repositories in RHEL 8 that provides a number of add-on software applications along with some core operating system components.
Archive	A file that contains one or more files.
Argument	A value passed to a command or program.
ARP	See Address Resolution Protocol.
ASCII	An acronym for American Standard Code for Information Interchange.
Asymmetric encryption technique	A technique that uses a combination of public/private keys to allow two network entities to communicate privately.
Auditing	System and user activity record and analysis.
Authentication	The process of identifying a user to a system.
AutoFS	The NFS client-side service that automatically mounts and unmounts an NFS share on an as-needed basis.
AutoFS maps	Configuration files to define the directory location to automount a remote share.
Automounter	See AutoFS.
AVC	See Access Vector Cache.
Background process	A process that runs in the background.
Backup	Process of saving data on an alternative media such as a tape or another disk.
BaseOS	One of the yum repositories in RHEL 8 that includes the foundational RHEL components.
Bash shell	A feature-rich default shell available in Red Hat Enterprise Linux.
Berkeley Internet	A University of California at Berkeley implementation of DNS for Linux and
Name Domain	UNIX platforms. See also Domain Name System.
Binary package	A software package available in a format that yum/dnf/rpm can recognize and install.
BIND	See Berkeley Internet Name Domain.
BIOS	Basic I/O System. Software code that sits in the computer's non-volatile memory and is executed when the system is booted. Also see Firmware.
Block	A collection of bytes of data transmitted as a single unit.
Block device file	A file associated with devices that transfer data randomly in blocks. Common examples are disk, CD, and DVD.
Bluetooth	A wireless technology for communication.

Boolean	The on/off switch to permit or deny an SELinux rule for a service.
Boot	See Boot process.
Bootloader	A small program that loads the operating system in memory.
Boot order	The sequence in which to try devices to boot the system.
Boot process	The process of starting up a system to a usable state.
Bourne Again Shell	See Bash shell.
Bus	Data communication path among devices in a computer system.
Cache	A temporary storage area on the system where frequently accessed information is duplicated for quick future access.
Calling process	See Parent process.
CentOS	Community Enterprise Operating System. A 100% unsponsored rebuild of Red Hat Enterprise Linux OS available for free.
Cgroup	See Control group.
Challenge-response authentication	An authentication method that presents one or more arbitrary challenge questions to the user.
Character special file	A file associated with devices that transfer data serially, one character at a time. Common examples are disk, tape, and mouse.
Child directory	A directory one level below the current directory.
Child process	A sub-process started by a process.
Child shell	A child shell is spawned by the current shell as needed.
Chrony	An implementation of Network Time Protocol for time synchronization on network devices.
CIDR	See Classless Inter-Domain Routing.
Classless Inter-Domain Routing	A technique for better use of IP addresses. It also results in smaller and less cluttered routing tables.
Command	An instruction given to the system to perform a task.
Command aliasing	See Alias.
Command history	See History substitution.
Command interpreter	See Shell.
Command argument line	See Positional parameter.

Command completion line	See Tab completion.
Command editing line	A shell feature that allows editing at the command line.
Command prompt	The OS prompt where you type commands.
Command substitution	A shell feature that allows the assignment of the output of an executed command to a variable.
Compression	The process of compressing data.
Container	A set of processes that runs in complete isolation from rest of the processes on the system.
Containerized application	An application packaged to run inside a container.
Container image	A file that contains all necessary components required by an application to run smoothly and securely.
Container registry	A public or private storage location for container images.
Context (SELinux)	A set of SELinux attributes applied to SELinux subjects and objects.
Contiguous blocks data	A series of data blocks.
Control group	A process management technique.
Core	A core is a processor that shares the chip with other cores. Multi-core processor chips are common.
CPU-intensive	A program or application that heavily uses system processors.
Crash	An abnormal system shutdown caused by electrical outage or kernel malfunction, etc.
Crontable	A table of cron jobs scheduled for a user. Commonly abbreviated as crontab.
Current directory	The present working directory.
Current shell	The shell where a program is launched. Compare with Child shell.
DAC (SELinux)	See Discretionary Access Control.
Daemon	A server process that runs in the background and responds to client requests.
Database	A collection of data.
D-bus	Desktop Bus. Another communication method that allows multiple services running in parallel on a system to talk to one another on the same or remote system. Compare with Socket.

De-duplication	A technique to remove redundant data blocks from storage to conserve space and improve performance.
De-encapsulation	The reverse of encapsulation. See Encapsulation.
Default	Predefined values or settings that are automatically accepted by commands or programs.
Default ACLs	ACL settings applied to directories.
Default permissions	Permissions assigned to a file and directory at creation.
Defunct process	See Zombie process.
Desktop bus	See D-bus.
Desktop environment	Software such as GNOME that provides graphical environment for users to interact with the system.
Device	A peripheral such as a printer, disk drive, or a CD/DVD device.
Device driver	The software that controls a device.
Device file	See Special file.
DHCP	See Dynamic Host Configuration Protocol.
Directory structure	Inverted tree-like Linux/UNIX directory structure.
Discretionary Access Control	A rich set of traditional access controls in Linux.
Disk-system based file	A file system created on a non-volatile storage device.
Disk partitioning	Creation of partitions on a given storage device so as to access them as distinct, independent logical containers for data storage.
Display manager	Application that is responsible for the presentation of graphical login screen.
Dnf	An upcoming major enhancement to yum.
DNS	See Domain Name System.
DNS name space	See Name space.
Domain	A group of computers configured to use a service such as DNS or NIS.
Domain Name	The de facto hostname resolution service used on the Internet and corporate networks.
System	
Domain (SELinux)	It ascertains the type of access that a process has.
Domain transitioning	The ability of a process running in one SELinux domain to enter another domain to execute a task in that domain.
Driver	See Device driver.
Dynamic Host Configuration	A networking service that provides IP assignments to

	devices.
Protocol	
Encapsulation	The process of forming a packet through the seven OSI layers.
Encryption	A method of scrambling information for privacy. See asymmetric encryption technique and symmetric encryption technique.
Encryption keys	A single secret key or a pair of private/public keys that is used to encrypt and decrypt data for private communication between two network entities.
Environment variable	A variable whose value is inherited by programs in sub-shells.
EOF	Marks the End OF File.
Error redirection	A shell feature that allows forwarding error messages generated during a command execution to an alternative destination (file, printer, etc.).
Ethernet	A family of networking technologies designed for LANs.
Ethernet address	See MAC address.
Exit code	A value returned by a command when it finishes execution.
Exit value	See Exit code.
Export	See Share.
Exporting	The process of making a directory or file system available over the network for sharing.
Extended file system	A type of file system that has been around in Linux for decades and currently has the fourth generation included and widely used in recent Linux distributions.
Extent	The smallest unit of space allocation in LVM. It is always contiguous. See Logical extent and Physical extent.
External command	A command external to the shell.
Fedora	Red Hat sponsored community project for collaborative enhancement of Red Hat Enterprise Linux OS.
Fibre channel	A family of networking technologies designed for storage networking.
File descriptor	A unique, per-process integer value used to refer to an open file.
File globbing	See Filename expansion.
Filename expansion	A series of characters used in matching filenames. Also see Metacharacters and Wildcard characters.
File permission	Read, write, execute or no permission assigned to a file

	or directory at the user, group, or public level.
File system	A grouping of files stored in special data structures.
File Protocol Transfer	A widely used protocol for file exchange.
Filter	A command that performs data transformation on the given input.
Firewall	A software or hardware appliance used for blocking inbound unauthorized access.
Firewalld	A dynamic firewall manager.
Firewalld zone	A method of segregating incoming network traffic.
Firmware	The BIOS or the UEFI code in x86-based systems.
FQIN	See Fully Qualified Image Name.
FTP	See File Transfer Protocol.
Full path	See Absolute path.
Fully Image Qualified Name	A container image name that includes all the necessary information to access it.
Gateway	A device that connects two networks.
Gateway address	An IP address that allows a system to communicate with computers on a different network.
GECOS	General Electric Comprehensive Operating System. The comments field in the /etc/passwd file.
GID	See Group ID.
Globally Identifier Unique	See Universally Unique Identifier.
Globbing	See Regular expression.
GNOME	GNU Object Model Environment. An intuitive graphical user environment.
GNU	GNU Not Unix. A project initiated to develop a completely free Unix-like operating system.
GPG	Gnu Privacy Guard. An open source implementation of PGP. See PGP.
GPL	General Public License that allows the use of software developed under GNU project to be available for free to the general public.
GPT	See GUID Partition Table.
Graphical User Interface	An interface that allows users to interact with the operating system or application graphically.
Group	A collection of users that requires same permissions on files and directories.
Group collaboration	A collection of users from different groups with identical

	rights on files for the purpose of sharing.
Group ID	A numeric identifier assigned to a group.
GRUB2	Grand Unified Bootloader version 2. The second generation of the GRUB bootloader program that loads the operating system in memory.
GSSAPI-based authentication	An authentication method that provides a standard interface for security mechanisms to be plugged in.
Guest	An operating system instance that runs in a virtual machine.
GUI	See Graphical User Interface.
GUID	See Universally Unique Identifier.
GUID Table Partition	A small disk partition on a UEFI system that stores disk partition information.
Hard link	A mapping between a filename and its inode number.
Hardware address	See MAC address.
Hardware clock	See Real-Time Clock.
Hashing	See Password hashing.
History expansion	See History substitution.
History substitution	A shell feature that enables the storage of previously executed commands.
Home directory	A directory where a user lands when he logs into the system.
Host-based firewall	A firewall service that runs on the Linux system.
Host-based authentication	An authentication method that allows a single user, a group of users, or all users on the client to be authenticated on the server.
Hostname	A unique name assigned to a network node.
Hostname resolution	See Name resolution.
Host table	A file that maintains IP and hostname mappings.
HTTP	See HyperText Transfer Protocol.
HTTPS	See HyperText Transfer Protocol Secure.
HyperText Protocol Transfer	HyperText Transfer Protocol. Allows access to web pages.
HyperText Protocol Secure Transfer	Secure cousin of HTTP. Allows access to secure web pages.
Hypervisor	Software loaded on a computer to virtualize its hardware.
ICMP	See Internet Control Message Protocol.
Index node	An index node number holds a file's properties including

	permissions, size and creation/modification time as well as contains a pointer to the data blocks that actually store the file data.
Init	An older method of system initialization. It has been replaced by systemd in newer Linux versions.
Initialization files	See Shell startup files.
Initial permissions	Predefined permission settings that are used to calculate default permissions for new files and directories.
Initial Setup	Program that starts at first system reboot after a system has been installed to customize authentication, firewall, network, time zone and other services.
Inode	See Index node.
Inode table	A table in a file system that keeps a record of inode numbers.
Input redirection	A shell feature that allows supplying input to a command from an alternative source (file, etc.).
Installable package	See Binary package.
Installer program	A program that is launched to install an operating system or application.
Interface card	See Network device.
Internet	A complex network of computers and routers.
Internet Control	A well-known networking protocol that is primarily used for testing and debugging.
Message Protocol	See Internet Protocol.
Internet Protocol	A protocol that is responsible for relaying traffic between network entities.
Inter-Process Communication	Allows processes to communicate directly with each other by sharing parts of their virtual memory address space, and then reading and writing data stored in that shared virtual memory.
I/O redirection	A shell feature that allows getting input from a non-default location and sending output and error messages to non-default locations.
IP	See Internet Protocol.
IP address	A unique 32- or 128-bit software address assigned to a network node.
IPC	See Inter-Process Communication.
ISO9660	A file system type used to mount optical devices.
Job	A process started in the background.
Job control	The management of jobs running in the background

	and foreground.
Job scheduling	Execution of commands, programs, or scripts in future.
Journald	A systemd-based logging service for collecting and storing logging data.
Journaled file system	A file system that uses the journaling mechanism for swift recovery after a system crash.
Journaling	A file system feature that allows it to maintain a journal (log) of its metadata changes to be used to fix any potential anomalies that may arise due to an abnormal system shutdown.
Kerberos	A networking protocol used for user authentication over unsecure networks.
Kernel	Software that controls the entire system including all hardware and software.
Kernel-Virtual based Machine	An open source hypervisor software used for host virtualization.
Kvdo	A kernel module to support the Virtual Data Optimizer feature.
KVM	See Kernel-based Virtual Machine.
Label (storage)	A unique partition identifier that may be used instead of a UUID or device file.
Label (SELinux)	See Context.
Labeling	The process of mapping files with their stored SELinux contexts.
Latency	The time it takes for a data packet to travel between two network entities.
Link	An object that associates a filename to any type of file.
Link count	Number of links that refers to a file.
Link layer address	See MAC address.
Linux	A UNIX-like, open source operating system.
Load balancing	A technique whereby more than one server serve client requests to share the load.
Localhost	A reserved, non-networked hostname assigned to every device. It represents the device itself.
Local variable	A variable whose value is private to the shell (current shell) it is defined in.
Logical extent	A unit of space allocation for logical volumes in LVM.
Logical volume	A logical container in LVM that holds a file system or

	swap.
Login Manager	See Display manager.
Logging	A process of capturing desired alerts and forwarding them to preconfigured locations.
Logical construct	Controls the flow of a script via test conditions.
Logical Manager	A widely used disk partitioning solution.
Volume	
Login	A process that begins when a user enters a username and password at the login prompt.
Login directory	See Home directory.
Loopback	A reserved IP address assigned to a device for testing and troubleshooting local issues.
Looping construct	Performs an action on a list of elements or repeatedly until a condition becomes true or false.
LVM	See Logical Volume Manager.
MAC address	A unique 48-bit hardware address of a network interface. Also called physical address, Ethernet address, and hardware address.
MAC (SELinux)	See Mandatory Access Control.
Machine	A computer, system, workstation, desktop, or server.
Major number	A number that points to a device driver.
Mandatory Control Access	A rich set of policies for granular access controls.
Map	See AutoFS map.
Masquerading	A variant of NAT.
Master Boot Record	A small region on the disk that stores disk partition information.
MBR	See Master Boot Record.
Memory-based file	A kernel-managed virtual file system created in memory at system boot and destroyed at system shutdown.
system	
Memory-intensive	A program or application that heavily uses memory.
Metacharacters	A series of characters that have special meaning to the shell and are used in pattern matching and filename globbing. Also see Wildcard characters.
Minor number	A unique number that points to an individual device controlled by a specific device driver.
MLS	See Multi-Level Security.
Module (kernel)	Device drivers used to control hardware devices and software components.

Module (package)	See Application module.
Mounting	Attaching a device (a file system, a CD/DVD) to the directory structure.
Multi-Security Level	One of the two standard SELinux policies that controls access at deeper levels.
Named group	A specific group that receives ACLs.
Named pipe	Allows two unrelated processes running on the same system or on two different systems to communicate with each other and exchange data.
Named user	A specific user that receives ACLs.
Name resolution	A technique to determine IP address by providing hostname.
Namespace	A layer of isolation between process groups and the rest of the system.
Name space	A hierarchical organization of DNS domains on the Internet.
NAT	See Network Address Translation.
NDP	See Neighbor Discovery Protocol.
Neighbor Discovery	A networking protocol that is used to discover Ipv6 devices and troubleshoot networking issues.
Protocol	
Netfilter	A framework that provides a set of hooks within the kernel to enable it to intercept and manipulate data packets.
Netmask	See Subnet mask.
Network	Two or more computers joined together to share resources.
Network Address	Allows systems on an internal network to access external networks using a single IP address.
Translation	
Network classes	Ranges of IP addresses classified into five distinct categories.
Network connection	A connection profile attached to a network device (interface).
Network device	A physical or virtual network interface assigned to a system for network connectivity.
Network File	A networking protocol that allows Linux systems to share resources (files, directories, and file systems) on the network.
System	
Network interface	See Network device.

Network card interface	See Network device.
NetworkManager	A Linux service that is used to configure, administer, and monitor network devices and connections.
Network mask	See Subnet mask.
Network Time	A networking protocol that is used to synchronize the system clock with a reliable time source.
Protocol	See Network device.
NIC	See Network File System.
NFS	A system that mounts an exported Linux resource.
NFS client	A system that exports (shares) a resource for mounting by an NFS client.
NFS server	A packet classification framework to monitor network traffic.
Nftables	It determines the priority of a process.
Niceness	See Niceness.
Nice value	A network device with a hostname and IP address.
Node	A unique name assigned to a node.
Node name	A user without the ability to log in to the system.
Nologin account (user)	A user account with limited privileges on the system.
Normal account (user)	See Network Time Protocol.
NTP	A system that receives time from a primary or secondary NTP server for its clock adjustments.
NTP client	Two or more time servers that operate at the same stratum level.
NTP peer	A pool of time servers.
NTP pool	See Primary NTP server and Secondary NTP server.
NTP server	A file, directory, file system, device, network connection, network interface, network socket, network port, etc.
Object (SELinux)	A method for setting permissions on a file or directory using octal numbering system.
Octal mode	A 3 digit numbering system that represents values from 0 to 7.
Octal system numbering	A systemd way of activating a service when needed.
On-activation demand	Any software whose source code is published and is accessible at no cost to the public under GNU GPL for copy, modification and redistribution.
Open source	A free implementation of secure shell services and
OpenSSH	

	utilities.
Orphan process	An alive child process of a terminated parent process.
Output redirection	A shell feature that allows forwarding a command output to an alternative destination (file, printer, etc.).
Owner	A user who has ownership rights on a file, directory, or process.
Owning user	The owner of a file or directory.
Owning group	The group of a file or directory.
Package	A set of necessary files and metadata that makes up a software application.
Package credibility	The authenticity or originality of a package.
Package database	A directory location that stores metadata for installed packages.
Package dependency	Additional required packages for a successful installation or functioning of another package.
Package group	A group of similar applications that can be managed as a single entity.
Package integrity	A state of being complete and error-free.
Package module	See Application module.
Paging	The process of transferring data between memory and swap space.
PAM	See Pluggable Authentication Module.
Parent directory	A directory one level above the current directory.
Parent process	A process with one or more child processes spawned.
Parent process ID	The ID of a process that starts a child process.
Parallelism	A systemd way of starting multiple services concurrently at system boot.
Partition	A partition created on a storage device.
Password aging	A mechanism that provides enhanced control on user passwords.
Password-based authentication	An authentication method that prompts users to enter their passwords to be signed in.
Password hashing	A one-way process of converting a legible text string into a random but unique string of characters using one of the several available password hashing algorithms.
Pattern matching	See Regular expression.
Peer	See NTP peer.
Per-user startup files	A set of initialization files that defines custom settings for an individual user upon logging in.
Performance-based	Hands-on implementation.

Performance monitoring	The process of acquiring data from system components for analysis and decision-making purposes.
Permission	Right to read, write, or execute.
Permission class	Access rights on files and directories based on an individual user, a group of users, or everyone else on the system.
Permission type	Read, write, or execute permission bits set on files or directories.
Permission mode	Add, revoke, or assign a permission type to a permission class.
Persistent storage	A host directory mounted inside a container to store application-generated data for persistence.
PGP	Pretty Good Privacy. An encryption program to ensure data privacy and secrecy.
Physical address	See MAC address.
Physical extent	A unit of space allocation on physical volumes in LVM.
Physical volume	A disk or a partition logically brought under LVM control.
PID	See Process ID.
Pipe	Sends output of one command as input to the second command.
Pipeline	A command construction with the pipe character used multiple times.
Pluggable Authentication	A set of library routines that allows using any authentication service available on a system for user authentication, password modification and user account validation purposes.
Module	
Policy (SELinux)	A set of rules enforced system-wide for analysis of security attributes on subjects and objects.
Pool	See Storage pool and Thin pool.
Pool (NTP)	See NTP pool.
Port	A number appended to an IP address. This number could be associated with a well-known service or is randomly generated.
Port forwarding	A method of directing incoming network traffic to an alternative network port.
Port mapping	Allows containerized applications to communicate with one another and with the container host.
Positional parameter	An argument supplied to a script at the time of its

	invocation, and its position is determined by the shell based on location with reference to the calling script.
POST	Power-On-Self-Test that runs at system boot to test hardware. See BIOS, Firmware, and UEFI.
Postfix	A mail transfer application used for sending and receiving mail.
PPID	See Parent process ID.
Primary DNS	A system that acts as the primary provider of DNS zones.
Primary NTP server	A system that gets time from a more reliable source and provides time to secondary servers or clients.
Primary prompt	The symbol where commands and programs are typed for execution.
Priority	See Process priority.
Private key	A randomly generated portion of the private/public key combination that is used to decode the messages encrypted with the paired public key.
Privilege	An extra right to accomplish something.
Process	Any command, program, or daemon that runs on a system.
Process ID	A numeric identifier assigned by kernel to each process spawned.
Process niceness	See Niceness.
Process priority	The value at which a process is running. This value is determined based on the current niceness setting.
Process state	One of multiple states in which a process is held during its lifecycle.
Processor	A CPU. It may contain more than one cores.
Profile (module)	A list of recommended packages that are organized for purpose-built convenient installations.
Prompt	See Primary prompt and Secondary prompt.
Protocol	A common language that communicating nodes understand.
Proxy	A system that acts on behalf of other systems to access network services.
Public key	A randomly generated portion of the private/public key combination that is used to encode messages destined for a specific user.
Public key-based authentication	An authentication method that uses a public/private key pair for user authentication.

Public encryption key	See Asymmetric encryption technique.
Quoting	Treats the specified special character as a regular character by disabling their special meaning.
Real-Time Clock	A battery-backed hardware clock on the system.
Recovery	A function that recovers a crashed system to its previous normal state. It may require restoring lost data files.
Redhat Manager	A file format used for packaging software for RHEL and its clones.
Package	
Red Hat	
Subscription	A comprehensive management service provided by Red Hat to its clients.
Management	
Redirection	Getting input from and sending output to non-default destinations.
Regex	See Regular expression.
Regexp	See Regular expression.
Registry	See Container registry.
Regular expression	A string of characters commonly used for pattern matching and filename globbing.
Relative path	A path to a file relative to the current user location in the file system hierarchy.
Renicing	Changing the niceness of a running process.
Repository	A URL location that provides access to software packages for installation.
Rescue mode	A special boot mode for fixing and recovering an unbootable system.
Resolver	The client-side of DNS.
Return code	See Exit code.
RHCE	Red Hat Certified Engineer. A designation that may be earned by passing a performance based RHCE exam.
RHCSA	Red Hat Certified System Administrator. A designation that may be earned by passing a performance based RHCSA exam.
RHEL	Red Hat Enterprise Linux.
RHSM	See Red Hat Subscription Management.
Role (SELinux)	It controls who (SELinux subject) is allowed to access what (SELinux domains or types).
Root (user) account	See Superuser.
Router	A device that routes data packets from one network to

	another.
Routing	The process of choosing a path over which to send a data packet.
Root container	A container launched by the root user or with root privileges.
Rootless container	A container launched by a normal, unprivileged Linux user.
Root servers	The thirteen most accurate root DNS servers.
RPM	See RedHat Package Manager.
Rsyslog	Essential Linux service for capturing system messages and forwarded them to various destinations for storage.
RTC	See Real-Time Clock.
Runtime	The operational state of an operating system.
SAS	Serial Attached SCSI. See Small Computer System Interface.
SATA	Serial Advanced Technology Attachment. This disk technology is a successor to the PATA drives.
Script	A text program written to perform a series of tasks.
SCSI	See Small Computer System Interface.
Search path	A list of directories where the system looks for the specified command.
Seccomp	See Secure Computing Mode.
Secondary DNS	A system that acts as an alternate provider of DNS zones.
Secondary NTP	A system that gets time from a primary NTP server and provides time to NTP clients.
server	
Secondary prompt	A prompt indicating that the entered command needs more input.
Secret encryption key	See Symmetric encryption technique.
Secure Mode	A Linux feature that impose security constraints to protect processes.
Computing	
Secure shell	A set of tools that gives secure access to a system.
Security context	SELinux security attributes set on files, processes, users, ports, etc.
Security Enhanced	An implementation of Mandatory Access Control architecture for enhanced
Linux	and granular control on files, processes, users, ports, etc.
SELinux	See Security Enhanced Linux.

Server (hardware)	Typically, a larger and more powerful system that offers services to network users.
Server (software)	A process or daemon that runs on the system to serve client requests.
Service account (user)	A user account that is used to control an installed application or service.
Set Group ID	Sets effective group ID.
Set User ID	Sets effective user ID.
Setgid	See Set group ID.
Setuid	See Set user ID.
Shadow password	A mechanism to store passwords and password aging data in a secure file.
Share	A directory or file system shared over the network.
Shared memory	A portion in physical memory created by a process to share it with other processes that communicate with that process.
Sharing	See Exporting.
Shell	The Linux command interpreter that sits between a user and kernel.
Shell parameter	An entity that holds a value such as a name, special character, or number.
Shell program	See Script.
Shell script	See Script.
Shell scripting	Programming in a Linux shell to automate one or a series of tasks.
Shell startup files	A set of files that are used to define the environment for a user upon logging in.
Shell variable	See Local variable.
Signal	A software interrupt sent to a process.
Simple Transfer Mail Protocol	A networking protocol used for email transfer over the Internet.
Single user mode	An operating system state in which the system cannot be accessed over the network.
Skeleton directory	A directory location where user default configuration templates are stored.
Small System Computer Interface	A parallel interface used to connect peripheral devices to the system.
SMTP	See Simple Mail Transfer Protocol.
Snapshot	The state of a system at a certain point in time.
Socket	A communication method that allows a process to talk

	to another process on the same or remote system.
Soft link	See Symbolic link.
Source package	A software package that can be modified and repackaged for a specific purpose.
Special characters	See Metacharacters.
Special file	A file that points to a specific device.
Special file permissions	Additional access permission bits that may be set on files and directories, where applicable, to give extra rights to (or limit rights for) normal users on executable files and shared directories. Also see Set user ID, Set group ID, and Sticky bit.
SSH	See Secure Shell.
Standard error	A standard location to forward error messages to. Also see Error redirection.
Standard input	A standard location to receive input from. Also see Input redirection.
Standard output	A standard location to forward output to. Also see Output redirection.
Startup files	See Shell startup files.
Stderr	See Standard error.
Stdin	See Standard input.
Stdout	See Standard output.
Sticky bit	Disallows non-owners to delete files located in a directory.
Storage pool	A logical storage space created with one or more disks or partitions.
Stratis	A simplified storage management solution.
Stratum level	The categorization of NTP time sources based on reliability and accuracy.
Stream (module)	Represents a collection of packages that are organized by version.
String	A series of characters.
Subject (SELinux)	A process or user.
Subnet	One of the smaller networks formed using the process of subnetting. See Subnetting.
Subnet mask	Segregates the network bits from the node bits in an IP address.
Subnetting	The process of dividing an IP address into several smaller subnetworks.
Sub-shell	See Child shell. Compare with Current shell.

Substituting users	See Switching users.
Sudo	A method of delegating a portion of superuser privileges to normal users.
Superblock	A small portion in a file system that holds the file system's critical information.
Superuser	A user with full powers on the system.
Swap	Alternative disk or file system location for paging.
Switch	A network device that looks at the MAC address and switches the packet to the correct destination port based on the MAC address.
Switching users	The ability to switch into a different user account provided the target user's password is known.
Symbolic link	A shortcut that points to a file or directory located somewhere in the directory hierarchy. Compare with hard link.
Symbolic mode	A method of setting permissions on a file using non-decimal values.
Symlink	See Symbolic link.
Symmetric encryption technique	A technique that employs a secret key for private communication between two network entities.
Syslog	See rsyslog.
System	A computer or a logical partition in a computer that runs an operating system.
System Administrator	Person responsible for installing, configuring and managing a RHEL system.
System call	A mechanism that applications use to request service from the kernel.
System console	A display terminal that acts as the system console.
Systemd	System daemon. The default method of system initialization and service management in newer Linux distributions including RHEL 7 and RHEL 8.
System recovery	The process of recovering an unbootable system.
System tuning	A service in RHEL 8 to monitor connected devices and dynamically adjust their parameters for performance improvement.
System-wide startup files	A set of initialization files that defines common settings for all users upon logging in.
Tab completion	A shell feature that allows completing a file or command name by typing a partial name at the command line and then hitting the

	Tab key twice for additional matching possibilities.
Target	A logical collection of systemd units. All units within a target are treated as a single entity.
Targeted policy	An SELinux policy.
TCP	See Transmission Control Protocol.
TCP/IP	Transmission Control Protocol / Internet Protocol. A stacked, standard suite of protocols for computer communication.
Terminal	A window where commands are executed.
Test condition	Used in logical constructs to decide what to do next.
Thin pool	A pool of storage that uses the thin provisioning technology to allow the creation of volumes much larger than their actual physical size.
Thin provisioning	An economical technique of storage allocation and utilization.
Thrashing	Excessive amount of paging.
Throughput	The amount of data transferred between two network entities within a specified period of time.
Tilde expansion	See Tilde substitution.
Tilde substitution	A shell feature that uses the tilde character as a shortcut to navigate within the directory tree.
Time source	A reference device that provides time to other devices.
Transmission Control Protocol	A stateful and reliable transport protocol. Compare with UDP.
Tty	Refers to a terminal.
Tuning profile	A set of attributes that can be applied to a system for improving performance of certain components.
Type enforcement	It controls the ability of an SELinux subject to access domains and types.
Udevd	Dynamic device management service.
UDP	See User Datagram Protocol.
UDS	See Universal De-duplication Service.
UEFI	See Unified Extensible Firmware Interface.
UID	See User ID.
Umask	See User mask.
Unified Extensible Firmware Interface	Software code used in computers for pre-boot system management. Also see Firmware.
Universal duplication	A kernel module to support data de-duplication.

De- Service	
Universally Unique Identifier	A unique alphanumeric software identifier used to identify an object, such as a disk or disk partition.
Unmounting	Detaching a mounted file system or a CD/DVD from the directory structure.
Unit	A systemd object used to organize service startups, socket creation, etc.
Universal Time	The reference time used around the world to determine the local time and time zone.
Coordinated USB	Universal Serial Bus. A bus standard to connect peripheral devices.
User Protocol Datagram	A stateless and unreliable transport protocol. Compare with TCP.
User ID	A numeric identifier assigned to a user.
User mask	A value used in calculating default access rights on new files and directories.
User Private Group	Referred to the GID that matches with the user's UID for safeguarding the user's private data from other users.
UTC	See Universal Time Coordinated.
UUID	See Universally Unique Identifier.
Variable	A temporary storage of data in memory.
Variable substitution	A shell feature that allows the value of a variable to be used in a command.
VDO	See Virtual Data Optimizer.
VFAT	See Virtual File Allocation Table.
VirtualBox	A type II hypervisor to virtualize an operating system.
VirtualBox Manager	The management interface for VirtualBox.
Virtual console	One of several console screens available for system access.
Virtual Optimizer Data	A feature to conserve disk space, improve data throughput, and save cost.
Virtual Allocation File Table	An MSDOS-compatible file system type.
Virtual file system	See memory-based file system.
Virtual host	An approach to host more than one website on a single system using unique or shared IP addresses.
	A technology that allows a single physical computer to run several independent

Virtualization	logical computers (called virtual machines) with complete isolation from one another.
Virtual machine	A logical computer running within a virtualized environment.
Volume group	A logical container in LVM that holds physical volumes, logical volumes, file systems, and swap.
Volume-managing file system	A storage management solution that dynamically and transparently manages the underlying logical volume layer for file systems.
Wayland	An innovative, superior, faster networking protocol that has replaced the X Window System protocol in RHEL 8. See X Window System protocol.
Web	A system of interlinked hypertext documents accessed over a network or the Internet via a web browser.
Web server	A system or service that provides web clients access to website pages.
Wildcard characters	A subset of metacharacters used for character matching in strings. See also Metacharacters.
Workload	Any application, database, program, or a combination that runs on the system.
XFS	eXtended File System. A high-performance journaling file system type.
X Window System	A networking protocol that lays the foundation to run graphical applications.
protocol	See Wayland.
Yum repository	See Repository.
Zero-elimination block	A technique to remove empty (zero-byte) data blocks from storage.
Zombie process	A child process that terminated abnormally and whose parent process still waits for it.
Zone (DNS)	A delegated portion of a DNS name space.
Zone (Firewalld)	A firewalld zone for traffic management.

Index

.bash_history file, [157](#)
.bash_profile file, [172](#)
.bashrc file, [172](#)

A

Absolute path, [49](#)
Access Control List, [105](#)
 Access ACLs, [105](#)
 Default ACLs, [109](#)
 Defined, [105](#)
 mask, [106](#)
 Named groups, [105](#)
 Named users, [105](#)
Access permissions (See File permissions)
Address Resolution Protocol (See Networking)
alias command, [160](#)
Alias substitution (See Shell)
anacron command, [192](#)
anacrontab file, [191](#)
Application stream (See Package)
apropos command, [55](#)
Archiving, [64](#)
Archiving tools, [64](#)
ARP (See Networking)
at command, [187](#)
at.allow file, [186](#)
at.deny file, [186](#)
atd service, [186, 187](#)
audit.log file, [476](#)

Auto File System (See AutoFS)

AutoFS, [404](#)

 Automounting user home directories, [410](#)

 Benefits, [405](#)

 Configuration file, [405](#)

 Defined, [404](#)

 How AutoFS works, [405](#)

 Maps, [406](#)

 Direct, [406](#)

 Indirect, [408](#)

 Master, [406](#)

 autofs service, [404](#)

 autofs.conf file, [405](#)

B

bashrc file, [170](#)

Berkeley Internet Name Domain (See Domain Name System)

bg command, [170](#)

BIND (See Domain Name System)

blkid command, [350](#)

Block device file (See File type)

Boot process, [250](#)

 Bootloader phase, [251](#)

 Firmware phase, [250](#)

 BIOS, [250](#)

 UEFI, [250](#)

 Initialization phase, [251](#)

 Kernel phase, [251](#)

boot.log file, [289](#)

btmp file, [119](#)

bunzip2 command, [66](#)

bzip2 command, [65](#)

bzip2 vs gzip, [66](#)

C

Calling process (See Process)

cat command, [74, 75](#)

cd command, [49](#)

chage command, [136](#)

Character device file (See File type)

chcon command, [469](#)
chgrp command, [146](#)
Child process (See Process)
chmod command, [91](#)
chown command, [146](#)
Chrony (See Network Time Protocol)
chrony.conf file, [420](#)
chronyc command, [421](#)
chronyd service, [421](#)
CIDR (See Networking)
Classless Inter-Domain Routing (See Networking)
clear command, [51](#)
Command aliasing (See Shell)
Command construction, [45](#)
Command history (See Shell)
Command interpreter (See Shell)
Command line completion (See Shell)
Command line editing (See Shell)
Compression (file), [64](#)
Compression tools, [64](#)
config file, [469](#)
Containerized applications, [502](#)
Containers
 Bare-Metal or Virtual Machine, [504](#)
 Benefits, [503](#)
 Cgroups, [503](#)
 Configuration file (system-wide), [509](#)
 Defined, [502](#)
 Environment variables, [520](#)
 FQIN, [505](#)
 Images, [505](#)
 Managing
 Containers (advanced), [517](#)
 Containers (basic), [514](#)
 Images, [512](#)
 Operational state, [525](#)
 Persistent storage, [521](#)
 Port mappings, [517](#)
 Variables, [520](#)
 Namespaces, [503](#)
 Persistent storage, [521](#)
 Port mapping, [517](#)
 Registry, [505](#)

Root, 506
Rootless, 506
Secure Computing Mode, 503
State management with systemd, 525
Counting words, lines, and characters, 78
cp command, 78
cron log file, 187, 189
cron.allow file, 186
crond service, 186, 189
crontab command, 189
crontab file, 189
Crontable, 189

D

DAC (See SELinux)
Daemon (See Process)
date command, 425
D-bus (See Initialization)
De-duplication, 314
Desktop environment, 37
Desktop manager, 36
Device driver (See Kernel)
df command, 350, 354
dig command, 428
Display/Login manager, 36
dnf command, 217
dnf.conf file, 216
DNS (See Domain Name System)
Documentation /usr/share/doc, 58
Domain Name System, 425
 BIND, 425
 Domain, 425
 FQDN, 426
 Managing
 Lookup tools, 428
 Name resolution, 425
 Name space, 425
 Resolver configuration file, 426
 Resolver sources and order, 427
 Roles, 426
 Client, 426

Primary (Master), 426
Secondary (Slave), 426
du command, 350, 355

E

e2label command, 350
echo command, 153, 483
Editing files, 69
Encapsulation, 450
env command, 153
Ethernet address (See Networking)
export command, 153
exportfs command, 401

F

fdisk command, 303
fg command, 169
FHS (See Filesystem Hierarchy Standard)
File and directory
 Copying directory, 79
 Copying file, 78
 Creating directory, 75
 Creating file, 74
 Displaying content, 75
 Moving directory, 80
 Moving file, 79
 Removing file, 80
 Renaming directory, 80
 Renaming file, 79
file command, 62
File permissions, 90
 Calculating default, 94
 Classes, 90
 Default, 93
 Initial, 93
 Modes, 91
 Modifying, 91
 Using octal notation, 91
 Using symbolic notation, 91
 Special, 95

- setgid on directories, 98
- setgid on files, 96
- setuid (suid), 95
- Sticky bit, 100
- Types, 90
- umask, 93

File system

- /, 40
- /boot, 41
- /dev, 42
- /home, 41
- /opt, 41
- /proc, 42
- /run, 43
- /sys, 43
- /tmp, 42
- /usr, 41
- /var, 41

Benefits, 346

Categories, 40, 346

- Disk-based, 40
- Memory-based, 40
- Network-based, 40

Defined, 346

Extended file system with journal, 348

fstab file, 353

Managing, 349

- Commands, 349
- Determining UUID, 351
- Labeling, 352
- Mount options, 351
- Mounting, 350
- Mounting automatically, 353
- Unmounting, 351

Monitoring, 354

Top-level, 39

Types, 346

- Extended, 347
- ISO9660, 349
- VFAT, 348
- XFS, 348

UUID, 351

File type

Block device, 42, 63
Character device, 63
Directory, 63
Raw device, 42
Regular, 62
Symbolic link, 64
Symlink (See Symbolic link)
Filesystem Hierarchy Standard, 39
find command, 102
Finding files, 102
Firewall
 Defined, 450
 Firewalld, 450
 Managing, 453
 Service, 452
 Service files, 452
 Zone, 450
 Zone files, 451
 Host-based, 450
firewall-cmd command, 454
Firewalld (See Firewall)
firewalld service, 450
FQDN (See Domain Name System)
FQIN (See Containers), 505
free command, 367
fstab file, 351, 353, 401
Fully Qualified Domain Name (See Domain Name System)
Fully Qualified Image Name (See Containers), 505

G

gdisk command, 310
getenforce command, 469, 470
getent command, 430
getfacl command, 105
getsebool command, 469
Getting help, 52
GNU, 2
gpasswd command, 126
GPL, 2
grep command, 166
Group

- Creating, [141](#)
- Deleting, [141](#)
- Identifying, [120](#)
- Modifying, [141](#)
 - Owning group, [146](#)
- group file, [124](#)
- groupadd command, [141](#)
- groupdel command, [141](#)
- groupmod command, [141](#)
- groups command, [120](#)
- grub file, [254](#)
- grub.cfg file, [253](#), [254](#)
- GRUB², [251](#)
 - Configuration files
 - grub, [254](#)
 - grub.cfg, [253](#)
 - grubenv, [255](#)
 - Managing, [251](#)
 - Specific targets, [256](#)
- grub²-mkconfig command, [254](#), [255](#)
- grubenv file, [255](#)
- gshadow file, [125](#)
- gunzip command, [65](#)
- gzip command, [65](#)
- gzip vs bzip², [66](#)

H

- halt command, [284](#)
- Hardware address (See Networking)
- Hardware clock, [424](#)
- head command, [77](#)
- history command, [158](#)
- History expansion (See Shell)
- History substitution (See Shell)
- host command, [429](#)
- Hostname, [376](#)
- hostname command, [376](#)
- hostname file, [376](#)
- hostnamectl command, [376](#)
- hosts file, [393](#), [428](#)

ICMP (See Networking)
id command, 120
ifdown command, 388
ifup command, 388
Index node number (See Inode), 81
info command, 57
Initialization
 systemd
 Control groups, 272
 D-bus, 273
 Listing previous system reboots, 117
 Managing, 276
 Setting default target, 283
 Switching target, 283
 Target, 282
 Unit, 277
 Viewing default target, 283
 Overview, 272
 Parallelism, 272
 Socket, 272
 Target
 Analyzing file, 276
 Defined, 275
 Types, 275
 Unit
 Analyzing file, 274
 Defined, 273
 State, 273
 Types, 274
Inode, 81
Installation
 LAB Setup, 4
 RHEL
 Adding keyboard and languages, 20
 Attaching image, 17
 Configuring install destination, 22
 Configuring network and hostname, 23
 Creating user account, 26
 Downloading, 14
 Finishing, 27
 Launching installer, 17

- Logs, 5
- Planning, 5
- Post-Installation, 28
- Selecting software, 21
- Selecting source, 20
- Setting root password, 26
- Virtual consoles, 5

VirtualBox

- Creating virtual machine, 10
- Downloading, 6

Internet Control Message Protocol (See Networking)

Internet Protocol (See Networking)

IP address (See Networking)

ip command, 378, 383

IPv6 address (See Networking)

J

- Job control, 169
- Job scheduling
 - Anacron, 191
 - Controlling access, 186
 - crontab file syntax, 189
 - Daemon atd, 187
 - crond, 189
 - Logging, 187
 - Overview, 186
 - Step values, 190
 - Using at, 187
 - Using crontab, 189
- jobs command, 169
- journalctl command, 291
- journald.conf file, 291

K

Kernel

- Analyzing version, 259
- Device driver, 258
- Directory structure, 260
- Installing, 264

Module, [262](#)
Modules, [258](#)
Overview, [258](#)
Packages, [258](#)
kill command, [184](#)
killall command, [185](#)
kvdo module, [314](#)

L

last command, [117](#)
lastb command, [118](#)
lastlog command, [119](#)
less command, [76](#)
let command, [496](#)
Linking files, [81](#)
 Copying vs. linking, [84](#)
 Hard link, [82](#)
 Link, [81](#)
 Symbolic link, [83](#)
Linux
 A quick look, [2](#)
 Defined, [2](#)
 Fedora Project, [3](#)
 RHEL history, [3](#)
Linux directory structure, [39](#)
ln command, [83](#)
Log rotation, [286](#)
logger command, [290](#)
Logging in, [29](#)
Logging out, [31](#)
Logical Volume Manager (See Storage)
login.defs file, [127](#)
loginctl command, [525](#)
logrotate command, [287](#)
logrotate.conf file, [287](#)
Loopback address (See Networking)
ls command, [46](#)
lsblk command, [303, 326, 349](#)
lscpu command, [52](#)
lvcreate command, [325](#)
lvdisplay command, [325, 326](#)

lvm command, [325](#)
LVM (See Storage)
lvreduce command, [325](#)
lvremove command, [325](#)
lvrename command, [325](#)
lvresize command, [325](#)
lvs command, [324](#), [326](#)

M

MAC address (See Networking)
machine-id file, [294](#)
Major number, [64](#)
man command, [53](#)
mandb command, [55](#)
Manual page headings, [54](#)
Manual page sections, [54](#)
meminfo virtual file, [368](#)
messages log file, [289](#)
Metacharacters (See Shell)
Metadata (file), [81](#)
Minor number, [64](#)
mkdir command, [75](#)
mke2fs command, [350](#)
mkfs command, [350](#)
mkfs.ext3 command, [350](#)
mkfs.ext4 command, [350](#)
mkfs.vfat command, [350](#)
mkfs.xfs command, [350](#)
mkswap command, [368](#)
Module (See Package)
more command, [76](#)
mount command, [350](#), [401](#)
mounts virtual file, [351](#)
mv command, [79](#)

N

Name resolution (See Domain Name System)
NDP (See Networking)
Neighbor Discovery Protocol (See Networking)
Netfilter, [450](#)

netfilter kernel module, [450](#)
Netmask (See Networking)
Network adapter (See Networking)
Network classes (See Networking)
Network connection (See Networking)
Network connection profile (See Networking)
Network device (See Networking)
Network device naming (See Networking)
Network entity (See Networking)
Network File System
 Benefits, [400](#)
 Configuring, [401](#)
 Exporting, [400](#)
 Mounting, [400](#)
 Defined, [400](#)
 Share, [400](#)
 Versions, [401](#)
Network Interface Card (See Networking)
Network Time Protocol
 Chrony, [418](#)
 Configuration file, [420](#)
 Displaying time, [423](#)
 Modifying time, [424](#)
 Overview, [418](#)
 Roles, [419](#)
 Client, [419](#)
 Peer, [419](#)
 Primary server, [419](#)
 Secondary server, [419](#)
 Stratum levels, [419](#)
 Time server, [420](#)
 Time source
 Internet-based, [419](#)
 Overview, [418](#)
 Radio/Atomic clock, [419](#)
Networking
 Changing hostname, [377](#)
 Configuring
 Commands, [387](#)
 Fundamentals
 ARP, [383](#)
 CIDR notation, [380](#)
 Classes, [379](#)

Class A, [379](#)
Class B, [379](#)
Class C, [379](#)
Class D, [380](#)
Class E, [380](#)
Common protocols, [382](#)
Connection profile (anatomy), [385](#)
Consistent naming, [384](#)
Entity, [376](#)
Hostname, [376](#)
Hosts table, [393](#)
ICMP, [382](#)
IP address, [378](#)
IPv4 vs. IPv6, [384](#)
IPv6, [383](#)
Localhost, [378](#)
Loopback address, [378](#)
MAC address, [383](#)
NDP, [383](#)
Network connection, [385](#)
Network connection profile, [385](#)
Network Interface Card, [384](#)
NetworkManager, [389](#)
Node, [376](#)
Port, [382](#)
Protocol, [381](#)
Subnet mask, [380](#)
Subnetting, [380](#)
TCP, [381](#)
UDP, [381](#)
NetworkManager service, [389](#)
network-scripts directory, [385](#)
NFS (See Network File System)
Nftables, [450](#)
NIC (See Networking)
nice command, [181](#)
Nice value (See Process)
Niceness (See Process)
nmcli command, [376, 389](#)
nm-connection-editor command, [389](#)
nmtui command, [389](#)
Node (See Networking)
nologin.txt file, [132](#)

nslookup command, [429](#)
nsswitch.conf file, [427](#)
NTP (See Network Time Protocol)

O

Online help tools, [52](#)
OpenSSH
 Algorithms, [436](#)
 Authentication methods, [435](#)
 Challenge-Response, [435](#)
 GSSAPI-based, [435](#)
 Host-based, [435](#)
 Password-based, [435](#)
 Public/private key, [435](#)
 Copying files, [442](#)
 Defined, [434](#)
 Encryption techniques, [434](#)
 Asymmetric, [434](#)
 Symmetric, [434](#)
 Legacy unsecure commands, [434](#)
 Managing
 Client configuration file, [438](#)
 Server configuration file, [436](#)
 Packages, [436](#)
 Synchronizing files, [445](#)
 Transferring files, [444](#)
 Version, [436](#)

P

Package
 Application Stream, [214](#)
 Binary, [196](#)
 Database, [197](#)
 Dependency, [197](#)
 Groups, [214](#)
 Managing dnf/yum, [216](#)
 Configuration file, [216](#)
 Group
 Installing, [230](#)
 Listing, [229](#)

Removing, 231
Updating, 230
Individual package
Displaying, 223
Installing, 222
Listing, 220
Removing, 224
Searching metadata, 227
Searching provider, 228
Updating, 222
Module
Displaying, 238
Installing, 237
Listing, 236
Removing, 239
Switching stream, 243
Updating, 237
rpm
Extracting, 205
Freshening, 204
Installing, 203
Overwriting, 204
Querying, 200
Removing, 204
Upgrading, 203
Verifying attributes, 207
Verifying signatures, 205
Viewing GPG keys, 206
Metadata, 196
Module, 214
 Profile, 215
 Stream, 215
Naming convention, 197
Overview, 196
Repository
 AppStream, 215
 BaseOS, 214
 Overview, 215
Source, 196
Parent process (See Process)
parted command, 304, 308
partitions virtual file, 308
passwd command, 138, 139

passwd file, 121
Path, 48
Pathname, 48
Pattern matching, 166
Payload, 450
pgrep command, 180
Physical address (See Networking)
pidof command, 180
pinfo command, 57
ping command, 382, 393
Pipe (See Shell)
pkg-config command, 273
pkill command, 185
podman command, 508
Port (See Networking)
poweroff command, 284
printenv command, 153
Process
 Background job, 169
 Calling, 176
 Child, 176
 Daemon, 176
 Foreground job, 169
 Job control, 169
 Listing, 180
 Listing by ownership, 181
 Nice value, 181
 Niceness, 181
 Parent, 176
 Priority, 181
 Process ID, 176
 Signals, 184
 States, 176
 Viewing with ps, 177
 Viewing with top, 179
Process file system (See File system)
Process ID (See Process)
Process priority (See Process)
Process states (See Process)
profile file, 170
profile.d directory, 170
Protocol (See Networking)
protocols file, 381

ps command, [177](#)
Pseudo terminal, [50](#)
pvcreate command, [325](#)
pvdisplay command, [326](#)
pvremove command, [325](#)
pvs command, [323](#), [326](#)
pwd command, [48](#)

Q

Quoting mechanisms, [165](#)

R

Real-time clock (See Hardware clock)
reboot command, [284](#)
Red Hat Subscription Management, [217](#), [506](#)
redhat-release file, [156](#)
Redirection
 Defined, [155](#)
 Error, [156](#)
 Input, [156](#)
 Output, [156](#)
Regex (See Pattern matching)
Regexp (See Pattern matching)
registries.conf file, [509](#)
Registry (See Containers), [505](#)
Regular expressions (See Pattern matching)
Relative path, [49](#)
Repository (See Package)
resize2fs command, [350](#)
resolv.conf file, [386](#), [426](#)
restorecon command, [469](#)
RHSM (See Red Hat Subscription Management),
 [217](#), [506](#)
rm command, [80](#)
rmdir command, [81](#)
Root containers (See Containers), [506](#)
Rootless containers (See Containers), [506](#)
rpm command, [198](#)
rpm2cpio command, [205](#)
rsync command, [445](#)

`rsyslog.conf` file, 285
`rsyslogd` service, 284
`rsyslogd.pid` file, 285
RTC (See Hardware clock)

S

`scp` command, 434, 436, 442
`sealert` command, 469
secure log file, 146
Secure shell (See OpenSSH)
`seinfo` command, 465, 469
SELinux
 Activation modes, 469
 Booleans, 467
 Contexts for files, 466
 Contexts for ports, 466
 Contexts for processes, 465
 Contexts for users, 464
 Defined, 462
 Discretionary Access Control, 462
 File operations with SELinux context, 466
 Managing, 468
 Analyzing alerts, 476
 Commands, 468
 Operational state, 469
 Querying, 470
 Mandatory Access Control, 462
Terms
 Access, 463
 Context, 463
 Domain, 464
 Domain transitioning, 467
 Labeling, 463
 Level, 464
 Object, 463
 Policy, 463
 Role, 464
 SELinux user, 463
 Subject, 463
 Type, 464
 Type enforcement, 464

semanage command, 465, 466, 469
sesearch command, 469
sestatus command, 469, 470
sestatus.conf file, 471
setenforce command, 469
setfacl command, 105
Setgid bit, 96, 98
setroubleshootd service, 476
setsebool command, 469
Setuid bit, 95
sftp command, 434, 436, 444
shadow file, 122
Shadow mechanism, 122
Shell
 Alias substitution, 160
 Bash, 152
 Child shell, 152
 Command history, 157
 Command line completion, 159
 Command line editing, 158
 Command substitution, 155
 Current shell, 152
 Features, 152
 History expansion, 157
 History substitution, 157
 Metacharacters, 162
 Modifying command prompt, 154
 Overview, 152
 Pipe, 164
 Quoting mechanisms, 165
 Tab completion, 159
 Tilde substitution, 159
 Variable substitution, 155
Variables
 Displaying, 153
 Environment, 153
 Local, 152
 Setting, 153
 Shell, 152
 Unsetting, 153
Wildcards
 Asterisk, 162
 Exclamation point, 163

Question mark, 163
Square brackets, 163

Shell scripting

- Command line arguments, 486
- Debugging, 484
- Defined, 482
- Displaying system info, 482
- Executing, 483
- Exit codes, 488
- Logical statements, 488
- Looping construct, 495
- Parsing command output, 486
- Positional parameters, 486
- Shell parameter, 486
- Shifting command line arguments, 487
- Special parameter, 486
- Test conditions, 489, 496
- Using environment variables, 485
- Using for-do-done, 496, 497
- Using if-then-elif-fi, 493
- Using if-then-else-fi, 491
- Using if-then-fi, 490
- Using local variables, 485

shift command, 487

shutdown command, 284

Signal (See Process)

Socket, 272

ssh command, 434, 436, 442

ssh_config file, 438

ssh-copy-id command, 436

sshd service, 436

sshd_config file, 436

ssh-keygen command, 436

stat command, 62

Sticky bit, 100

Storage

- Benefits, 322
- LVM, 322
 - Concept, 322
 - Logical extent, 324
 - Logical volume, 324
- Managing
 - Commands, 325

- Physical extent, [324](#)
- Physical volume, [323](#)
- Volume group, [323](#)
- Managing
 - Tools, [304](#)
 - Using gdisk, [310](#)
 - Using parted, [308](#)
- Partition table
 - GPT, [303](#)
 - MBR, [302](#)
 - UEFI, [302](#)
- Stratis
 - Defined, [335](#)
 - Dynamic expansion, [336](#)
 - Managing, [337](#)
 - Pool, [336](#)
- Thin provisioning
 - Defined, [304](#)
 - pool, [304](#)
- VDO
 - Compression, [314](#)
 - De-duplication, [314](#)
 - Defined, [314](#)
 - How it works, [314](#)
 - Managing, [315](#)
 - Zero-block elimination, [314](#)
- Stratis (See Storage)
- stratis command, [337](#)
- stratisd service, [337](#)
- Stream (See Package)
- su command, [143](#)
- Subnet mask (See Networking)
- Subnetting (See Networking)
- Substituting users, [143](#)
- sudo command, [144](#)
- sudoers file, [144](#)
- Swap
 - Commands, [368](#)
 - Defined, [366](#)
 - Demand paging, [367](#)
 - Determining usage, [367](#)
 - Prioritizing, [368](#)
 - Thrashing, [367](#)

swapoff command, [368](#)
swapon command, [368](#)
System logging
 Configuration file, [285](#)
 Journal, [290](#)
 journald
 Configuration file, [291](#)
 Preserving, [293](#)
 Viewing, [291](#)
 Logging, [289](#)
 Logging custom messages, [290](#)
 Rotating log files, [286](#)
 rsyslogd, [284](#)
 systemd-journald daemon, [291](#)
System tuning (See Tuning)
systemctl command, [276](#)
systemd (See Initialization)
systemd-hostnamed service, [377](#)
systemd-journald service, [290](#)

T

tac command, [76](#)
tail command, [77](#)
Tape archive (See Archiving)
tar command, [66](#)
TCP (See Networking)
Thin provisioning (See Storage)
Tilde expansion (See Shell)
Tilde substitution (See Shell)
Time synchronization (See Network Time Protocol)
timedatectl command, [423](#)
top command, [179](#)
touch command, [74](#)
traceroute command, [383](#)
Transmission Control Protocol (See Networking)
tree command, [43](#)
tty command, [50](#)
tune2fs command, [350](#)
tuned service, [294](#)
tuned-admin command, [296](#)
Tuning, [294](#)

Daemon, [294](#)
Defined, [294](#)
Managing, [296](#)
Profile location, [295](#)
Profiles, [295](#)
type command, [51](#)

U

udevd service, [42, 385](#)
UDP (See Networking)
UDS (See Universal De-duplication Service)
UDS module, [314](#)
umask command, [93](#)
umount command, [350, 351](#)
unalias command, [162](#)
uname command, [51, 259](#)
Universal De-duplication Service, [314](#)
unset command, [153](#)
UPG (See User Private Group)
uptime command, [50](#)
User
 Authentication file
 group, [124](#)
 gshadow, [125](#)
 passwd, [121](#)
 shadow, [122](#)
 Configuring password aging, [136](#)
 Creating, [128](#)
 Deleting, [129](#)
 Doing as superuser, [144](#)
 Identifying, [120](#)
 Initialization file
 Per-user, [171](#)
 Sourcing sequence, [172](#)
 System-wide, [170](#)
 Locking and unlocking, [139](#)
 login.defs file, [127](#)
 Managing
 Listing logged-in users, [116](#)
 Listing previous failed logins, [118](#)
 Listing previous successful logins, [117](#)

- Listing recent logins, [119](#)
- Modifying, [129](#)
- Nologin account, [132](#)
- Owning user, [146](#)
- Password aging, [136](#)
- Skeleton directory, [129](#)
- Substituting, [143](#)
- Type
 - Normal, [120](#)
 - Service, [120](#)
 - Superuser, [120](#)
- User Private Group, [124](#)
- useradd file, [127](#)

User Datagram Protocol (See Networking)

User Private Group, [124](#)

useradd command, [128](#)

useradd file, [127](#)

usermod command, [129](#)

UUID (See File system)

V

- vdo command, [315](#)
- vdostats command, [315](#)
- vgcreate command, [325](#)
- vgdisplay command, [324](#)
- vgextend command, [325](#)
- vgreduce command, [325](#)
- vgremove command, [325](#)
- vgrename command, [325](#)
- vgs command, [323, 326](#)
- vi editor (See vim editor)
- vim editor, [69](#)
 - Changing, [72](#)
 - Copying, [72](#)
 - Deleting, [71](#)
 - Inserting, [70](#)
 - Modes, [69](#)
 - Command mode, [69](#)
 - Extended mode, [69](#)
 - Input mode, [69](#)
 - Last line mode, [69](#)

Moving, 72
Navigating, 70
Pasting, 72
Replacing, 72
Saving and quitting, 73
Searching, 71
Starting, 69
Undoing and repeating, 71
VirtualBox
 Changing boot order, 27
 Creating virtual machine, 10
 Downloading, 6
Volume-Managing file system, 336

W

w command, 117
Wayland, 36
 Desktop environment, 37
 Desktop manager, 36
 Display/Login manager, 36
wc command, 78
whatis command, 56
whereis command, 51
which command, 51
who command, 116
Wildcard characters (See Wildcards under Shell)
wtmp file, 118

X

xfs_admin command, 350
xfs_growfs command, 350
xfs_info command, 350
xfs_repair command, 348

Z

Zero-block elimination, 314

PERCEPTIONS OF A RENEGADE MIND

DAVID DICKIE

PERCEPTIONS OF A RENEGADE MIND



DAVID ICKE

**PERCEPTIONS
OF A
RENEGADE
MIND**

ickonic
publishing

First published in July 2021.



**New Enterprise House
St Helens Street
Derby
DE1 3GY
UK**

email: gareth.icke@davidicke.com

Copyright © 2021 David Icke

No part of this book may be reproduced in any form without permission from the Publisher, except for the quotation of brief passages in criticism

Cover Design: Gareth Icke
Book Design: Neil Hague

**British Library Cataloguing-in
Publication Data**
A catalogue record for this book is
available from the British Library

eISBN 978-18384153-1-0

PERCEPTIONS
OF A
RENEGADE
MIND



DAVID ICKE

Dedication:

To *Freeeeeedom!*

ICKONIC



THE ALTERNATIVE

NEW. DIFFERENT. REVOLUTIONARY

HUNDREDS OF CUTTING EDGE DOCUMENTARIES,
FEATURE FILMS, SERIES & PODCASTS.

SIGN UP NOW AT ICKONIC.COM

THE LIFE STORY OF DAVID ICKE
RENEGADE
THE FEATURE LENGTH FILM



AVAILABLE NOW AT DAVIDICKE.COM

Renegade:

Adjective

'Having rejected tradition: Unconventional.'

Merriam-Webster Dictionary

Acquiescence to tyranny is the death of the spirit

You may be 38 years old, as I happen to be. And one day, some great opportunity stands before you and calls you to stand up for some great principle, some great issue, some great cause. And you refuse to do it because you are afraid

... You refuse to do it because you want to live longer ...

You're afraid that you will lose your job, or you are afraid that you will be criticised or that you will lose your popularity, or you're afraid that somebody will stab you, or shoot at you or bomb your house; so you refuse to take the stand.

Well, you may go on and live until you are 90, but you're just as dead at 38 as you would be at 90. And the cessation of breathing in your life is but the belated announcement of an earlier death of the spirit.

Martin Luther King

**How the few control the many and always have – the many do
whatever they're told**

'Forward, the Light Brigade!'
Was there a man dismayed?
Not though the soldier knew
 Someone had blundered.
Theirs not to make reply,
Theirs not to reason why,
Theirs but to do and die.
 Into the valley of Death
 Rode the six hundred.

Cannon to right of them,
Cannon to left of them,
Cannon in front of them
 Volleyed and thundered;
Stormed at with shot and shell,
 Boldly they rode and well,
 Into the jaws of Death,
 Into the mouth of hell
 Rode the six hundred

Alfred Lord Tennyson (1809-1892)

The mist is lifting slowly
I can see the way ahead
And I've left behind the empty streets
That once inspired my life
And the strength of the emotion
Is like thunder in the air
'Cos the promise that we made each other
Haunts me to the end

The secret of your beauty
And the mystery of your soul
I've been searching for in everyone I meet
And the times I've been mistaken
It's impossible to say
And the grass is growing
Underneath our feet

The words that I remember
From my childhood still are true
That there's none so blind
As those who will not see
And to those who lack the courage
And say it's dangerous to try
Well they just don't know
That love eternal will not be denied

I know you're out there somewhere
Somewhere, somewhere
I know you're out there somewhere

Somewhere you can hear my voice
I know I'll find you somehow
Somehow, somehow
I know I'll find you somehow
And somehow I'll return again to you

The Moody Blues

Are you a gutless wonder - or a Renegade Mind?

Monuments put from pen to paper,
Turns me into a gutless wonder,
And if you tolerate this,
Then your children will be next.
Gravity keeps my head down,
Or is it maybe shame ...

Manic Street Preachers

Rise like lions after slumber
In unvanquishable number.
Shake your chains to earth like dew
Which in sleep have fallen on you.
Ye are many – they are few.

Percy Shelley

Contents

CHAPTER 1	'I'm thinking' – Oh, but <i>are you?</i>
CHAPTER 2	Renegade perception
CHAPTER 3	The Pushbacker sting
CHAPTER 4	'Covid': The calculated catastrophe
CHAPTER 5	There <i>is no</i> 'virus'
CHAPTER 6	Sequence of deceit
CHAPTER 7	War on your mind
CHAPTER 8	'Reframing' insanity
CHAPTER 9	We must have it? So what is it?
CHAPTER 10	Human 2.0
CHAPTER 11	Who controls the Cult?
CHAPTER 12	Escaping Wetiko
POSTSCRIPT	
APPENDIX	Cowan-Kaufman-Morell Statement on Virus Isolation
BIBLIOGRAPHY	
INDEX	

CHAPTER ONE

I'm thinking' – Oh, but *are* you?

Think for yourself and let others enjoy the privilege of doing so too
Voltaire

French-born philosopher, mathematician and scientist René Descartes became famous for his statement in Latin in the 17th century which translates into English as: 'I think, therefore I am.'

On the face of it that is true. Thought reflects perception and perception leads to both behaviour and self-identity. In that sense 'we' are what we think. But who or what is doing the thinking and is thinking the only route to perception? Clearly, as we shall see, 'we' are not always the source of 'our' perception, indeed with regard to humanity as a whole this is rarely the case; and thinking is far from the only means of perception. Thought is the village idiot compared with other expressions of consciousness that we all have the potential to access and tap into. This has to be true when we *are* those other expressions of consciousness which are infinite in nature. We have forgotten this, or, more to the point, been manipulated to forget.

These are not just the esoteric musings of the navel. The whole foundation of human control and oppression is control of perception. Once perception is hijacked then so is behaviour which is dictated by perception. Collective perception becomes collective behaviour and collective behaviour is what we call human society. Perception is all and those behind human control know that which is

why perception is the target 24/7 of the psychopathic manipulators that I call the Global Cult. They know that if they dictate perception they will dictate behaviour and collectively dictate the nature of human society. They are further aware that perception is formed from information received and if they control the circulation of information they will to a vast extent direct human behaviour.

Censorship of information and opinion has become globally Nazi-like in recent years and never more blatantly than since the illusory ‘virus pandemic’ was triggered out of China in 2019 and across the world in 2020. Why have billions submitted to house arrest and accepted fascistic societies in a way they would have never believed possible? Those controlling the information spewing from government, mainstream media and Silicon Valley (all controlled by the same Global Cult networks) told them they were in danger from a ‘deadly virus’ and only by submitting to house arrest and conceding their most basic of freedoms could they and their families be protected. This monumental and provable lie became the *perception* of the billions and therefore the *behaviour* of the billions. In those few words you have the whole structure and modus operandi of human control. Fear is a perception – False Emotion Appearing Real – and fear is the currency of control. In short ... get them by the balls (or give them the impression that you have) and their hearts and minds will follow. Nothing grips the dangly bits and freezes the rear-end more comprehensively than fear.

World number 1

There are two ‘worlds’ in what appears to be one ‘world’ and the prime difference between them is knowledge. First we have the mass of human society in which the population is maintained in coldly-calculated ignorance through control of information and the ‘education’ (indoctrination) system. That’s all you really need to control to enslave billions in a perceptual delusion in which what are perceived to be *their* thoughts and opinions are ever-repeated mantras that the system has been downloading all their lives through ‘education’, media, science, medicine, politics and academia

in which the personnel and advocates are themselves overwhelmingly the perceptual products of the same repetition. Teachers and academics in general are processed by the same programming machine as everyone else, but unlike the great majority they never leave the ‘education’ program. It gripped them as students and continues to grip them as programmers of subsequent generations of students. The programmed become the programmers – the programmed programmers. The same can largely be said for scientists, doctors and politicians and not least because as the American writer Upton Sinclair said: ‘It is difficult to get a man to understand something when his salary depends upon his not understanding it.’ If your career and income depend on thinking the way the system demands then you will – bar a few free-minded exceptions – concede your mind to the Perceptual Mainframe that I call the Postage Stamp Consensus. This is a tiny band of perceived knowledge and possibility ‘taught’ (downloaded) in the schools and universities, pounded out by the mainstream media and on which all government policy is founded. Try thinking, and especially speaking and acting, outside of the ‘box’ of consensus and see what that does for your career in the Mainstream Everything which bullies, harasses, intimidates and ridicules the population into compliance. Here we have the simple structure which enslaves most of humanity in a perceptual prison cell for an entire lifetime and I’ll go deeper into this process shortly. Most of what humanity is taught as fact is nothing more than programmed belief. American science fiction author Frank Herbert was right when he said: ‘Belief can be manipulated. Only knowledge is dangerous.’ In the ‘Covid’ age belief is promoted and knowledge is censored. It was always so, but never to the extreme of today.

World number 2

A ‘number 2’ is slang for ‘doing a poo’ and how appropriate that is when this other ‘world’ is doing just that on humanity every minute of every day. World number 2 is a global network of secret societies and semi-secret groups dictating the direction of society via

governments, corporations and authorities of every kind. I have spent more than 30 years uncovering and exposing this network that I call the Global Cult and knowing its agenda is what has made my books so accurate in predicting current and past events. Secret societies are secret for a reason. They want to keep their hoarded knowledge to themselves and their chosen initiates and to hide it from the population which they seek through ignorance to control and subdue. The whole foundation of the division between World 1 and World 2 is *knowledge*. What number 1 knows number 2 must not. Knowledge they have worked so hard to keep secret includes (a) the agenda to enslave humanity in a centrally-controlled global dictatorship, and (b) the nature of reality and life itself. The latter (b) must be suppressed to allow the former (a) to prevail as I shall be explaining. The way the Cult manipulates and interacts with the population can be likened to a spider's web. The 'spider' sits at the centre in the shadows and imposes its will through the web with each strand represented in World number 2 by a secret society, satanic or semi-secret group, and in World number 1 – the world of the seen – by governments, agencies of government, law enforcement, corporations, the banking system, media conglomerates and Silicon Valley ([Fig 1](#) overleaf). The spider and the web connect and coordinate all these organisations to pursue the same global outcome while the population sees them as individual entities working randomly and independently. At the level of the web governments *are* the banking system *are* the corporations *are* the media *are* Silicon Valley *are* the World Health Organization working from their inner cores as one unit. Apparently unconnected countries, corporations, institutions, organisations and people are on the *same team* pursuing the same global outcome. Strands in the web immediately around the spider are the most secretive and exclusive secret societies and their membership is emphatically restricted to the Cult inner-circle emerging through the generations from particular bloodlines for reasons I will come to. At the core of the core you would get them in a single room. That's how many people are dictating the direction of human society and its transformation

through the ‘Covid’ hoax and other means. As the web expands out from the spider we meet the secret societies that many people will be aware of – the Freemasons, Knights Templar, Knights of Malta, Opus Dei, the inner sanctum of the Jesuit Order, and such like. Note how many are connected to the Church of Rome and there is a reason for that. The Roman Church was established as a revamp, a rebranding, of the relocated ‘Church’ of Babylon and the Cult imposing global tyranny today can be tracked back to Babylon and Sumer in what is now Iraq.



Figure 1: The global web through which the few control the many. (Image Neil Hague.)

Inner levels of the web operate in the unseen away from the public eye and then we have what I call the cusp organisations located at the point where the hidden meets the seen. They include a series of satellite organisations answering to a secret society founded in London in the late 19th century called the Round Table and among them are the Royal Institute of International Affairs (UK, founded in 1920); Council on Foreign Relations (US, 1921); Bilderberg Group (worldwide, 1954); Trilateral Commission (US/worldwide, 1972); and the Club of Rome (worldwide, 1968) which was created to exploit environmental concerns to justify the centralisation of global power to ‘save the planet’. The Club of Rome instigated with others the human-caused climate change hoax which has led to all the ‘green

new deals' demanding that very centralisation of control. Cusp organisations, which include endless 'think tanks' all over the world, are designed to coordinate a single global policy between political and business leaders, intelligence personnel, media organisations and anyone who can influence the direction of policy in their own sphere of operation. Major players and regular attenders will know what is happening – or some of it – while others come and go and are kept overwhelmingly in the dark about the big picture. I refer to these cusp groupings as semi-secret in that they can be publicly identified, but what goes on at the inner-core is kept very much 'in house' even from most of their members and participants through a fiercely-imposed system of compartmentalisation. Only let them know what they need to know to serve your interests and no more. The structure of secret societies serves as a perfect example of this principle. Most Freemasons never get higher than the bottom three levels of 'degree' (degree of knowledge) when there are 33 official degrees of the Scottish Rite. Initiates only qualify for the next higher 'compartment' or degree if those at that level choose to allow them. Knowledge can be carefully assigned only to those considered 'safe'. I went to my local Freemason's lodge a few years ago when they were having an 'open day' to show how cuddly they were and when I chatted to some of them I was astonished at how little the rank and file knew even about the most ubiquitous symbols they use. The mushroom technique – keep them in the dark and feed them bullshit – applies to most people in the web as well as the population as a whole. Sub-divisions of the web mirror in theme and structure transnational corporations which have a headquarters somewhere in the world dictating to all their subsidiaries in different countries. Subsidiaries operate in their methodology and branding to the same centrally-dictated plan and policy in pursuit of particular ends. The Cult web functions in the same way. Each country has its own web as a subsidiary of the global one. They consist of networks of secret societies, semi-secret groups and bloodline families and their job is to impose the will of the spider and the global web in their particular country. Subsidiary networks control and manipulate the national political system, finance, corporations, media, medicine, etc. to

ensure that they follow the globally-dictated Cult agenda. These networks were the means through which the ‘Covid’ hoax could be played out with almost every country responding in the same way.

The ‘Yessir’ pyramid

Compartmentalisation is the key to understanding how a tiny few can dictate the lives of billions when combined with a top-down sequence of imposition and acquiescence. The inner core of the Cult sits at the peak of the pyramidal hierarchy of human society ([Fig 2](#) overleaf). It imposes its will – its agenda for the world – on the level immediately below which acquiesces to that imposition. This level then imposes the Cult will on the level below them which acquiesces and imposes on the next level. Very quickly we meet levels in the hierarchy that have no idea there even is a Cult, but the sequence of imposition and acquiescence continues down the pyramid in just the same way. ‘I don’t know why we are doing this but the order came from “on-high” and so we better just do it.’ Alfred Lord Tennyson said of the cannon fodder levels in his poem *The Charge of the Light Brigade*: ‘Theirs not to reason why; theirs but to do and die.’ The next line says that ‘into the valley of death rode the six hundred’ and they died because they obeyed without question what their perceived ‘superiors’ told them to do. In the same way the population capitulated to ‘Covid’. The whole hierarchical pyramid functions like this to allow the very few to direct the enormous many.

Eventually imposition-acquiescence-imposition-acquiescence comes down to the mass of the population at the foot of the pyramid. If they acquiesce to those levels of the hierarchy imposing on them (governments/law enforcement/doctors/media) a circuit is completed between the population and the handful of super-psychopaths in the Cult inner core at the top of the pyramid. Without a circuit-breaking refusal to obey, the sequence of imposition and acquiescence allows a staggeringly few people to impose their will upon the entirety of humankind. We are looking at the very sequence that has subjugated billions since the start of 2020. Our freedom has not been taken from us. Humanity has given it

away. Fascists do not impose fascism because there are not enough of them. Fascism is imposed by the population acquiescing to fascism. Put another way allowing their perceptions to be programmed to the extent that leads to the population giving their freedom away by giving their perceptions – their mind – away. If this circuit is not broken by humanity ceasing to cooperate with their own enslavement then nothing can change. For that to happen people have to critically think and see through the lies and window dressing and then summon the backbone to act upon what they see. The Cult spends its days working to stop either happening and its methodology is systematic and highly detailed, but it can be overcome and that is what this book is all about.

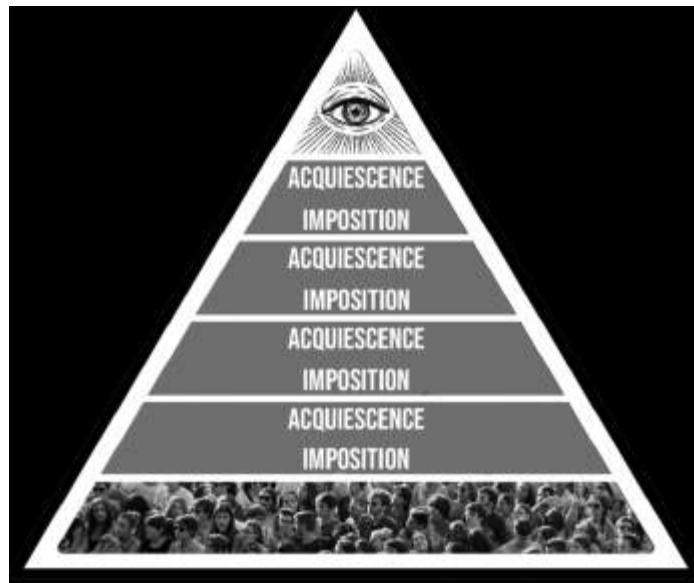


Figure 2: The simple sequence of imposition and compliance that allows a handful of people at the peak of the pyramid to dictate the lives of billions.

The Life Program

Okay, back to world number 1 or the world of the ‘masses’. Observe the process of what we call ‘life’ and it is a perceptual download from cradle to grave. The Cult has created a global structure in which perception can be programmed and the program continually topped-up with what appears to be constant confirmation that the program is indeed true reality. The important word here is ‘appears’.

This is the structure, the fly-trap, the Postage Stamp Consensus or Perceptual Mainframe, which represents that incredibly narrow band of perceived possibility delivered by the ‘education’ system, mainstream media, science and medicine. From the earliest age the download begins with parents who have themselves succumbed to the very programming their children are about to go through. Most parents don’t do this out of malevolence and mostly it is quite the opposite. They do what they believe is best for their children and that is what the program has told them is best. Within three or four years comes the major transition from parental programming to full-blown state (Cult) programming in school, college and university where perceptually-programmed teachers and academics pass on their programming to the next generations. Teachers who resist are soon marginalised and their careers ended while children who resist are called a problem child for whom Ritalin may need to be prescribed. A few years after entering the ‘world’ children are under the control of authority figures representing the state telling them when they have to be there, when they can leave and when they can speak, eat, even go to the toilet. This is calculated preparation for a lifetime of obeying authority in all its forms. Reflex-action fear of authority is instilled by authority from the start. Children soon learn the carrot and stick consequences of obeying or defying authority which is underpinned daily for the rest of their life. Fortunately I daydreamed through this crap and never obeyed authority simply because it told me to. This approach to my alleged ‘bettters’ continues to this day. There can be consequences of pursuing open-minded freedom in a world of closed-minded conformity. I spent a lot of time in school corridors after being ejected from the classroom for not taking some of it seriously and now I spend a lot of time being ejected from Facebook, YouTube and Twitter. But I can tell you that being true to yourself and not compromising your self-respect is far more exhilarating than bowing to authority for authority’s sake. You don’t have to be a sheep to the shepherd (authority) and the sheep dog (fear of not obeying authority).

The perceptual download continues throughout the formative years in school, college and university while script-reading ‘teachers’, ‘academics’ ‘scientists’, ‘doctors’ and ‘journalists’ insist that ongoing generations must be as programmed as they are. Accept the program or you will not pass your ‘exams’ which confirm your ‘degree’ of programming. It is tragic to think that many parents pressure their offspring to work hard at school to download the program and qualify for the next stage at college and university. The late, great, American comedian George Carlin said: ‘Here’s a bumper sticker I’d like to see: We are proud parents of a child who has resisted his teachers’ attempts to break his spirit and bend him to the will of his corporate masters.’ Well, the best of luck finding many of those, George. Then comes the moment to leave the formal programming years in academia and enter the ‘adult’ world of work. There you meet others in your chosen or prescribed arena who went through the same Postage Stamp Consensus program before you did. There is therefore overwhelming agreement between almost everyone on the basic foundations of Postage Stamp reality and the rejection, even contempt, of the few who have a mind of their own and are prepared to use it. This has two major effects. Firstly, the consensus confirms to the programmed that their download is really how things are. I mean, everyone knows that, right? Secondly, the arrogance and ignorance of Postage Stamp adherents ensure that anyone questioning the program will have unpleasant consequences for seeking their own truth and not picking their perceptions from the shelf marked: ‘Things you must believe without question and if you don’t you’re a dangerous lunatic conspiracy theorist and a harebrained nutter’.

Every government, agency and corporation is founded on the same Postage Stamp prison cell and you can see why so many people believe the same thing while calling it their own ‘opinion’. Fusion of governments and corporations in pursuit of the same agenda was the definition of fascism described by Italian dictator Benito Mussolini. The pressure to conform to perceptual norms downloaded for a lifetime is incessant and infiltrates society right

down to family groups that become censors and condemners of their own ‘black sheep’ for not, ironically, being sheep. We have seen an explosion of that in the ‘Covid’ era. Cult-owned global media unleashes its propaganda all day every day in support of the Postage Stamp and targets with abuse and ridicule anyone in the public eye who won’t bend their mind to the will of the tyranny. Any response to this is denied (certainly in my case). They don’t want to give a platform to expose official lies. Cult-owned-and-created Internet giants like Facebook, Google, YouTube and Twitter delete you for having an unapproved opinion. Facebook boasts that its AI censors delete 97-percent of ‘hate speech’ before anyone even reports it. Much of that ‘hate speech’ will simply be an opinion that Facebook and its masters don’t want people to see. Such perceptual oppression is widely known as fascism. Even Facebook executive Benny Thomas, a ‘CEO Global Planning Lead’, said in comments secretly recorded by investigative journalism operation Project Veritas that Facebook is ‘too powerful’ and should be broken up:

I mean, no king in history has been the ruler of two billion people, but Mark Zuckerberg is ... And he's 36. That's too much for a 36-year-old ... You should not have power over two billion people. I just think that's wrong.

Thomas said Facebook-owned platforms like Instagram, Oculus, and WhatsApp needed to be separate companies. ‘It’s too much power when they’re all one together’. That’s the way the Cult likes it, however. We have an executive of a Cult organisation in Benny Thomas that doesn’t know there is a Cult such is the compartmentalisation. Thomas said that Facebook and Google ‘are no longer companies, they’re countries’. Actually they are more powerful than countries on the basis that if you control information you control perception and control human society.

I love my oppressor

Another expression of this psychological trickery is for those who realise they are being pressured into compliance to eventually

convince themselves to believe the official narratives to protect their self-respect from accepting the truth that they have succumbed to meek and subservient compliance. Such people become some of the most vehement defenders of the system. You can see them everywhere screaming abuse at those who prefer to think for themselves and by doing so reminding the compliers of their own capitulation to conformity. ‘You are talking dangerous nonsense you Covidiot!!’ Are you trying to convince me or yourself? It is a potent form of Stockholm syndrome which is defined as: ‘A psychological condition that occurs when a victim of abuse identifies and attaches, or bonds, positively with their abuser.’ An example is hostages bonding and even ‘falling in love’ with their kidnappers. The syndrome has been observed in domestic violence, abused children, concentration camp inmates, prisoners of war and many and various Satanic cults. These are some traits of Stockholm syndrome listed at goodtherapy.org:

- Positive regard towards perpetrators of abuse or captor [see ‘Covid’].
- Failure to cooperate with police and other government authorities when it comes to holding perpetrators of abuse or kidnapping accountable [or in the case of ‘Covid’ cooperating with the police to enforce and defend their captors’ demands].
- Little or no effort to escape [see ‘Covid’].
- Belief in the goodness of the perpetrators or kidnappers [see ‘Covid’].
- Appeasement of captors. This is a manipulative strategy for maintaining one’s safety. As victims get rewarded – perhaps with less abuse or even with life itself – their appeasing behaviours are reinforced [see ‘Covid’].
- Learned helplessness. This can be akin to ‘if you can’t beat ‘em, join ‘em’. As the victims fail to escape the abuse or captivity, they may start giving up and soon realize it’s just easier for everyone if they acquiesce all their power to their captors [see ‘Covid’].

- Feelings of pity toward the abusers, believing they are actually victims themselves. Because of this, victims may go on a crusade or mission to 'save' [protect] their abuser [see the venom unleashed on those challenging the official 'Covid' narrative].
- Unwillingness to learn to detach from their perpetrators and heal. In essence, victims may tend to be less loyal to themselves than to their abuser [*definitely* see 'Covid'].

Ponder on those traits and compare them with the behaviour of great swathes of the global population who have defended governments and authorities which have spent every minute destroying their lives and livelihoods and those of their children and grandchildren since early 2020 with fascistic lockdowns, house arrest and employment deletion to 'protect' them from a 'deadly virus' that their abusers' perceptually created to bring about this very outcome. We are looking at mass Stockholm syndrome. All those that agree to concede their freedom will believe those perceptions are originating in their own independent 'mind' when in fact by conceding their reality to Stockholm syndrome they have by definition conceded any independence of mind. Listen to the 'opinions' of the acquiescing masses in this 'Covid' era and what gushes forth is the repetition of the official version of everything delivered unprocessed, unfiltered and unquestioned. The whole programming dynamic works this way. I must be free because I'm told that I am and so I think that I am.

You can see what I mean with the chapter theme of 'I'm thinking – Oh, but *are you?*' The great majority are not thinking, let alone for themselves. They are repeating what authority has told them to believe which allows them to be controlled. Weaving through this mentality is the fear that the 'conspiracy theorists' are right and this again explains the often hysterical abuse that ensues when you dare to contest the official narrative of anything. Denial is the mechanism of hiding from yourself what you don't want to be true. Telling people what they want to hear is easy, but it's an infinitely greater challenge to tell them what they would rather not be happening.

One is akin to pushing against an open door while the other is met with vehement resistance no matter what the scale of evidence. I don't want it to be true so I'll convince myself that it's not. Examples are everywhere from the denial that a partner is cheating despite all the signs to the reflex-action rejection of any idea that world events in which country after country act in exactly the same way are centrally coordinated. To accept the latter is to accept that a force of unspeakable evil is working to destroy your life and the lives of your children with nothing too horrific to achieve that end. Who the heck wants that to be true? But if we don't face reality the end is duly achieved and the consequences are far worse and ongoing than breaking through the walls of denial today with the courage to make a stand against tyranny.

Connect the dots – but how?

A crucial aspect of perceptual programming is to portray a world in which everything is random and almost nothing is connected to anything else. Randomness cannot be coordinated by its very nature and once you perceive events as random the idea they could be connected is waved away as the rantings of the tinfoil-hat brigade. You can't plan and coordinate random you idiot! No, you can't, but you can hide the coldly-calculated and long-planned behind the *illusion* of randomness. A foundation manifestation of the Renegade Mind is to scan reality for patterns that connect the apparently random and turn pixels and dots into pictures. This is the way I work and have done so for more than 30 years. You look for similarities in people, modus operandi and desired outcomes and slowly, then ever quicker, the picture forms. For instance: There would seem to be no connection between the 'Covid pandemic' hoax and the human-caused global-warming hoax and yet they are masks (appropriately) on the same face seeking the same outcome. Those pushing the global warming myth through the Club of Rome and other Cult agencies are driving the lies about 'Covid' – Bill Gates is an obvious one, but they are endless. Why would the same people be involved in both when they are clearly not connected? Oh, but they

are. Common themes with personnel are matched by common goals. The ‘solutions’ to both ‘problems’ are centralisation of global power to impose the will of the few on the many to ‘save’ humanity from ‘Covid’ and save the planet from an ‘existential threat’ (we need ‘zero Covid’ and ‘zero carbon emissions’). These, in turn, connect with the ‘dot’ of globalisation which was coined to describe the centralisation of global power in every area of life through incessant political and corporate expansion, trading blocks and superstates like the European Union. If you are the few and you want to control the many you have to centralise power and decision-making. The more you centralise power the more power the few at the centre will have over the many; and the more that power is centralised the more power those at the centre have to centralise even quicker. The momentum of centralisation gets faster and faster which is exactly the process we have witnessed. In this way the hoaxed ‘pandemic’ and the fakery of human-caused global warming serve the interests of globalisation and the seizure of global power in the hands of the Cult inner-circle which is behind ‘Covid’, ‘climate change’ and globalisation. At this point random ‘dots’ become a clear and obvious picture or pattern.

Klaus Schwab, the classic Bond villain who founded the Cult’s Gates-funded World Economic Forum, published a book in 2020, *The Great Reset*, in which he used the ‘problem’ of ‘Covid’ to justify a total transformation of human society to ‘save’ humanity from ‘climate change’. Schwab said: ‘The pandemic represents a rare but narrow window of opportunity to reflect, reimagine, and reset our world.’ What he didn’t mention is that the Cult he serves is behind both hoaxes as I show in my book *The Answer*. He and the Cult don’t have to reimagine the world. They know precisely what they want and that’s why they destroyed human society with ‘Covid’ to ‘build back better’ in their grand design. Their job is not to imagine, but to get humanity to imagine and agree with their plans while believing it’s all random. It must be pure coincidence that ‘The Great Reset’ has long been the Cult’s code name for the global imposition of fascism and replaced previous code-names of the ‘New World

'Order' used by Cult frontmen like Father George Bush and the 'New Order of the Ages' which emerged from Freemasonry and much older secret societies. New Order of the Ages appears on the reverse of the Great Seal of the United States as 'Novus ordo seclorum' underneath the Cult symbol used since way back of the pyramid and all seeing-eye ([Fig 3](#)). The pyramid is the hierarchy of human control headed by the illuminated eye that symbolises the force behind the Cult which I will expose in later chapters. The term 'Annuit Coeptis' translates as 'He favours our undertaking'. We are told the 'He' is the Christian god, but 'He' is not as I will be explaining.



Figure 3: The all-seeing eye of the Cult 'god' on the Freemason-designed Great Seal of the United States and also on the dollar bill.

Having you on

Two major Cult techniques of perceptual manipulation that relate to all this are what I have called since the 1990s Problem-Reaction-Solution (PRS) and the Totalitarian Tiptoe (TT). They can be uncovered by the inquiring mind with a simple question: Who benefits? The answer usually identifies the perpetrators of a given action or happening through the concept of 'he who most benefits from a crime is the one most likely to have committed it'. The Latin 'Cue bono?' – Who benefits? – is widely attributed to the Roman orator and statesman Marcus Tullius Cicero. No wonder it goes back so far when the concept has been relevant to human behaviour since

history was recorded. Problem-Reaction-Solution is the technique used to manipulate us every day by covertly creating a problem (or the illusion of one) and offering the solution to the problem (or the illusion of one). In the first phase you create the problem and blame someone or something else for why it has happened. This may relate to a financial collapse, terrorist attack, war, global warming or pandemic, anything in fact that will allow you to impose the ‘solution’ to change society in the way you desire at that time. The ‘problem’ doesn’t have to be real. PRS is manipulation of perception and all you need is the population to believe the problem is real. Human-caused global warming and the ‘Covid pandemic’ only have to be *perceived* to be real for the population to accept the ‘solutions’ of authority. I refer to this technique as NO-Problem-Reaction-Solution. Billions did not meekly accept house arrest from early 2020 because there was a real deadly ‘Covid pandemic’ but because they perceived – believed – that to be the case. The antidote to Problem-Reaction-Solution is to ask who benefits from the proposed solution. Invariably it will be anyone who wants to justify more control through deletion of freedom and centralisation of power and decision-making.

The two world wars were Problem-Reaction-Solutions that transformed and realigned global society. Both were manipulated into being by the Cult as I have detailed in books since the mid-1990s. They dramatically centralised global power, especially World War Two, which led to the United Nations and other global bodies thanks to the overt and covert manipulations of the Rockefeller family and other Cult bloodlines like the Rothschilds. The UN is a stalking horse for full-blown world government that I will come to shortly. The land on which the UN building stands in New York was donated by the Rockefellers and the same Cult family was behind Big Pharma scalpel and drug ‘medicine’ and the creation of the World Health Organization as part of the UN. They have been stalwarts of the eugenics movement and funded Hitler’s race-purity expert Ernst Rudin. The human-caused global warming hoax has been orchestrated by the Club of Rome through the UN which is

manufacturing both the ‘problem’ through its Intergovernmental Panel on Climate Change and imposing the ‘solution’ through its Agenda 21 and Agenda 2030 which demand the total centralisation of global power to ‘save the world’ from a climate hoax the United Nations is itself perpetrating. What a small world the Cult can be seen to be particularly among the inner circles. The bedfellow of Problem-Reaction-Solution is the Totalitarian Tiptoe which became the Totalitarian Sprint in 2020. The technique is fashioned to hide the carefully-coordinated behind the cover of apparently random events. You start the sequence at ‘A’ and you know you are heading for ‘Z’. You don’t want people to know that and each step on the journey is presented as a random happening while all the steps strung together lead in the same direction. The speed may have quickened dramatically in recent times, but you can still see the incremental approach of the Tiptoe in the case of ‘Covid’ as each new imposition takes us deeper into fascism. Tell people they have to do this or that to get back to ‘normal’, then this and this and this. With each new demand adding to the ones that went before the population’s freedom is deleted until it disappears. The spider wraps its web around the flies more comprehensively with each new diktat. I’ll highlight this in more detail when I get to the ‘Covid’ hoax and how it has been pulled off. Another prime example of the Totalitarian Tiptoe is how the Cult-created European Union went from a ‘free-trade zone’ to a centralised bureaucratic dictatorship through the Tiptoe of incremental centralisation of power until nations became mere administrative units for Cult-owned dark suits in Brussels.

The antidote to ignorance is knowledge which the Cult seeks vehemently to deny us, but despite the systematic censorship to that end the Renegade Mind can overcome this by vociferously seeking out the facts no matter the impediments put in the way. There is also a method of thinking and perceiving – *knowing* – that doesn’t even need names, dates, place-type facts to identify the patterns that reveal the story. I’ll get to that in the final chapter. All you need to know about the manipulation of human society and to what end is still out there – *at the time of writing* – in the form of books, videos

and websites for those that really want to breach the walls of programmed perception. To access this knowledge requires the abandonment of the mainstream media as a source of information in the awareness that this is owned and controlled by the Cult and therefore promotes mass perceptions that suit the Cult. Mainstream media lies all day, every day. That is its function and very reason for being. Where it does tell the truth, here and there, is only because the truth and the Cult agenda very occasionally coincide. If you look for fact and insight to the BBC, CNN and virtually all the rest of them you are asking to be conned and perceptually programmed.

Know the outcome and you'll see the journey

Events seem random when you have no idea where the world is being taken. Once you do the random becomes the carefully planned. Know the outcome and you'll see the journey is a phrase I have been using for a long time to give context to daily happenings that appear unconnected. Does a problem, or illusion of a problem, trigger a proposed 'solution' that further drives society in the direction of the outcome? Invariably the answer will be yes and the random – *abracadabra* – becomes the clearly coordinated. So what is this outcome that unlocks the door to a massively expanded understanding of daily events? I will summarise its major aspects – the fine detail is in my other books – and those new to this information will see that the world they thought they were living in is a very different place. The foundation of the Cult agenda is the incessant centralisation of power and all such centralisation is ultimately in pursuit of Cult control on a global level. I have described for a long time the planned world structure of top-down dictatorship as the Hunger Games Society. The term obviously comes from the movie series which portrayed a world in which a few living in military-protected hi-tech luxury were the overlords of a population condemned to abject poverty in isolated 'sectors' that were not allowed to interact. 'Covid' lockdowns and travel bans anyone? The 'Hunger Games' pyramid of structural control has the inner circle of the Cult at the top with pretty much the entire

population at the bottom under their control through dependency for survival on the Cult. The whole structure is planned to be protected and enforced by a military-police state ([Fig 4](#)).

Here you have the reason for the global lockdowns of the fake pandemic to coldly destroy independent incomes and livelihoods and make everyone dependent on the ‘state’ (the Cult that controls the ‘states’). I have warned in my books for many years about the plan to introduce a ‘guaranteed income’ – a barely survivable pittance – designed to impose dependency when employment was destroyed by AI technology and now even more comprehensively at great speed by the ‘Covid’ scam. Once the pandemic was played and lockdown consequences began to delete independent income the authorities began to talk right on cue about the need for a guaranteed income and a ‘Great Reset’. Guaranteed income will be presented as benevolent governments seeking to help a desperate people – desperate as a direct result of actions of the same governments. The truth is that such payments are a trap. You will only get them if you do exactly what the authorities demand including mass vaccination (genetic manipulation). We have seen this theme already in Australia where those dependent on government benefits have them reduced if parents don’t agree to have their children vaccinated according to an insane health-destroying government-dictated schedule. Calculated economic collapse applies to governments as well as people. The Cult wants rid of countries through the creation of a world state with countries broken up into regions ruled by a world government and super states like the European Union. Countries must be bankrupted, too, to this end and it’s being achieved by the trillions in ‘rescue packages’ and furlough payments, trillions in lost taxation, and money-no-object spending on ‘Covid’ including constant all-medium advertising (programming) which has made the media dependent on government for much of its income. The day of reckoning is coming – as planned – for government spending and given that it has been made possible by printing money and not by production/taxation there is inflation on the way that has the

potential to wipe out monetary value. In that case there will be no need for the Cult to steal your money. It just won't be worth anything (see the German Weimar Republic before the Nazis took over). Many have been okay with lockdowns while getting a percentage of their income from so-called furlough payments without having to work. Those payments are dependent, however, on people having at least a theoretical job with a business considered non-essential and ordered to close. As these business go under because they are closed by lockdown after lockdown the furlough stops and it will for everyone eventually. Then what? The 'then what?' is precisely the idea.



Figure 4: The Hunger Games Society structure I have long warned was planned and now the 'Covid' hoax has made it possible. This is the real reason for lockdowns.

Hired hands

Between the Hunger Games Cult elite and the dependent population is planned to be a vicious military-police state (a fusion of the two into one force). This has been in the making for a long time with police looking ever more like the military and carrying weapons to match. The pandemic scam has seen this process accelerate so fast as

lockdown house arrest is brutally enforced by carefully recruited fascist minds and gormless system-servers. The police and military are planned to merge into a centrally-directed world army in a global structure headed by a world government which wouldn't be elected even by the election fixes now in place. The world army is not planned even to be human and instead wars would be fought, primarily against the population, using robot technology controlled by artificial intelligence. I have been warning about this for decades and now militaries around the world are being transformed by this very AI technology. The global regime that I describe is a particular form of fascism known as a technocracy in which decisions are not made by clueless and co-opted politicians but by unelected technocrats – scientists, engineers, technologists and bureaucrats. Cult-owned-and-controlled Silicon Valley giants are examples of technocracy and they already have far more power to direct world events than governments. They are with their censorship *selecting* governments. I know that some are calling the 'Great Reset' a Marxist communist takeover, but fascism and Marxism are different labels for the same tyranny. Tell those who lived in fascist Germany and Stalinist Russia that there was a difference in the way their freedom was deleted and their lives controlled. I could call it a fascist technocracy or a Marxist technocracy and they would be equally accurate. The Hunger Games society with its world government structure would oversee a world army, world central bank and single world cashless currency imposing its will on a microchipped population ([Fig 5](#)). Scan its different elements and see how the illusory pandemic is forcing society in this very direction at great speed. Leaders of 23 countries and the World Health Organization (WHO) backed the idea in March, 2021, of a global treaty for 'international cooperation' in 'health emergencies' and nations should 'come together as a global community for peaceful cooperation that extends beyond this crisis'. Cut the Orwellian bullshit and this means another step towards global government. The plan includes a cashless digital money system that I first warned about in 1993. Right at the start of 'Covid' the deeply corrupt Tedros

Adhanom Ghebreyesus, the crooked and merely gofer ‘head’ of the World Health Organization, said it was possible to catch the ‘virus’ by touching cash and it was better to use cashless means. The claim was ridiculous nonsense and like the whole ‘Covid’ mind-trick it was nothing to do with ‘health’ and everything to do with pushing every aspect of the Cult agenda. As a result of the Tedros lie the use of cash has plummeted. The Cult script involves a single world digital currency that would eventually be technologically embedded in the body. China is a massive global centre for the Cult and if you watch what is happening there you will know what is planned for everywhere. The Chinese government is developing a digital currency which would allow fines to be deducted immediately via AI for anyone caught on camera breaking its fantastic list of laws and the money is going to be programmable with an expiry date to ensure that no one can accrue wealth except the Cult and its operatives.



Figure 5: The structure of global control the Cult has been working towards for so long and this has been enormously advanced by the ‘Covid’ illusion.

Serfdom is so smart

The Cult plan is far wider, extreme, and more comprehensive than even most conspiracy researchers appreciate and I will come to the true depths of deceit and control in the chapters ‘Who controls the

Cult?' and 'Escaping Wetiko'. Even the world that we know is crazy enough. We are being deluged with ever more sophisticated and controlling technology under the heading of 'smart'. We have smart televisions, smart meters, smart cards, smart cars, smart driving, smart roads, smart pills, smart patches, smart watches, smart skin, smart borders, smart pavements, smart streets, smart cities, smart communities, smart environments, smart growth, smart planet ... smart *everything* around us. Smart technologies and methods of operation are designed to interlock to create a global Smart Grid connecting the entirety of human society including human minds to create a centrally-dictated 'hive' mind. 'Smart cities' is code for densely-occupied megacities of total surveillance and control through AI. Ever more destructive frequency communication systems like 5G have been rolled out without any official testing for health and psychological effects (colossal). 5G/6G/7G systems are needed to run the Smart Grid and each one becomes more destructive of body and mind. Deleting independent income is crucial to forcing people into these AI-policed prisons by ending private property ownership (except for the Cult elite). The Cult's Great Reset now openly foresees a global society in which no one will own any possessions and everything will be rented while the Cult would own literally everything under the guise of government and corporations. The aim has been to use the lockdowns to destroy sources of income on a mass scale and when the people are destitute and in unrepayable amounts of debt (problem) Cult assets come forward with the pledge to write-off debt in return for handing over all property and possessions (solution). Everything – literally everything including people – would be connected to the Internet via AI. I was warning years ago about the coming Internet of Things (IoT) in which all devices and technology from your car to your fridge would be plugged into the Internet and controlled by AI. Now we are already there with much more to come. The next stage is the Internet of Everything (IoE) which is planned to include the connection of AI to the human brain and body to replace the human mind with a centrally-controlled AI mind. Instead of perceptions

being manipulated through control of information and censorship those perceptions would come direct from the Cult through AI. What do you think? You think whatever AI decides that you think. In human terms there would be no individual 'think' any longer. Too incredible? The ravings of a lunatic? Not at all. Cult-owned crazies in Silicon Valley have been telling us the plan for years without explaining the real motivation and calculated implications. These include Google executive and 'futurist' Ray Kurzweil who highlights the year 2030 for when this would be underway. He said:

Our thinking ... will be a hybrid of biological and non-biological thinking ... humans will be able to extend their limitations and 'think in the cloud' ... We're going to put gateways to the cloud in our brains ... We're going to gradually merge and enhance ourselves ... In my view, that's the nature of being human – we transcend our limitations.

As the technology becomes vastly superior to what we are then the small proportion that is still human gets smaller and smaller and smaller until it's just utterly negligible.

The sales-pitch of Kurzweil and Cult-owned Silicon Valley is that this would make us 'super-human' when the real aim is to make us post-human and no longer 'human' in the sense that we have come to know. The entire global population would be connected to AI and become the centrally-controlled 'hive-mind' of externally-delivered perceptions. The Smart Grid being installed to impose the Cult's will on the world is being constructed to allow particular locations – even one location – to control the whole global system. From these prime control centres, which absolutely include China and Israel, anything connected to the Internet would be switched on or off and manipulated at will. Energy systems could be cut, communication via the Internet taken down, computer-controlled driverless autonomous vehicles driven off the road, medical devices switched off, the potential is limitless given how much AI and Internet connections now run human society. We have seen nothing yet if we allow this to continue. Autonomous vehicle makers are working with law enforcement to produce cars designed to automatically pull over if they detect a police or emergency vehicle flashing from up to 100 feet away. At a police stop the car would be unlocked and the

window rolled down automatically. Vehicles would only take you where the computer (the state) allowed. The end of petrol vehicles and speed limiters on all new cars in the UK and EU from 2022 are steps leading to electric computerised transport over which ultimately you have no control. The picture is far bigger even than the Cult global network or web and that will become clear when I get to the nature of the ‘spider’. There is a connection between all these happenings and the instigation of DNA-manipulating ‘vaccines’ (which aren’t ‘vaccines’) justified by the ‘Covid’ hoax. That connection is the unfolding plan to transform the human body from a biological to a synthetic biological state and this is why synthetic biology is such a fast-emerging discipline of mainstream science. ‘Covid vaccines’ are infusing self-replicating synthetic genetic material into the cells to cumulatively take us on the Totalitarian Tiptoe from Human 1.0 to the synthetic biological Human 2.0 which will be physically and perceptually attached to the Smart Grid to one hundred percent control every thought, perception and deed.

Humanity needs to wake up and *fast*.

This is the barest explanation of where the ‘outcome’ is planned to go but it’s enough to see the journey happening all around us. Those new to this information will already see ‘Covid’ in a whole new context. I will add much more detail as we go along, but for the minutiae evidence see my mega-works, *The Answer*, *The Trigger* and *Everything You Need to Know But Have Never Been Told*.

Now – how does a Renegade Mind see the ‘world’?

CHAPTER TWO

Renegade Perception

It is one thing to be clever and another to be wise

George R.R. Martin

A simple definition of the difference between a programmed mind and a Renegade Mind would be that one sees only dots while the other connects them to see the picture. Reading reality with accuracy requires the observer to (a) know the planned outcome and (b) realise that everything, but *everything*, is connected.

The entirety of infinite reality is connected – that's its very nature – and with human society an expression of infinite reality the same must apply. Simple cause and effect is a connection. The effect is triggered by the cause and the effect then becomes the cause of another effect. Nothing happens in isolation because it *can't*. Life in whatever reality is simple choice and consequence. We make choices and these lead to consequences. If we don't like the consequences we can make different choices and get different consequences which lead to other choices and consequences. The choice and the consequence are not only connected they are indivisible. You can't have one without the other as an old song goes. A few cannot control the world unless those being controlled allow that to happen – cause and effect, choice and consequence. Control – who has it and who doesn't – is a two-way process, a symbiotic relationship, involving the controller and controlled. 'They took my freedom away!!' Well, yes, but you also gave it to them. Humanity is

subjected to mass control because humanity has acquiesced to that control. This is all cause and effect and literally a case of give and take. In the same way world events of every kind are connected and the Cult works incessantly to sell the illusion of the random and coincidental to maintain the essential (to them) perception of dots that hide the picture. Renegade Minds know this and constantly scan the world for patterns of connection. This is absolutely pivotal in understanding the happenings in the world and without that perspective clarity is impossible. First you know the planned outcome and then you identify the steps on the journey – the day-by-day apparently random which, when connected in relation to the outcome, no longer appear as individual events, but as the proverbial *chain* of events leading in the same direction. I'll give you some examples:

Political puppet show

We are told to believe that politics is 'adversarial' in that different parties with different beliefs engage in an endless tussle for power. There may have been some truth in that up to a point – and only a point – but today divisions between 'different' parties are rhetorical not ideological. Even the rhetorical is fusing into one-speak as the parties eject any remaining free thinkers while others succumb to the ever-gathering intimidation of anyone with the 'wrong' opinion. The Cult is not a new phenomenon and can be traced back thousands of years as my books have documented. Its intergenerational initiates have been manipulating events with increasing effect the more that global power has been centralised. In ancient times the Cult secured control through the system of monarchy in which 'special' bloodlines (of which more later) demanded the right to rule as kings and queens simply by birthright and by vanquishing others who claimed the same birthright. There came a time, however, when people had matured enough to see the unfairness of such tyranny and demanded a say in who governed them. Note the word – *governed* them. Not served them – *governed* them, hence government defined as 'the political direction and control exercised over the

actions of the members, citizens, or inhabitants of communities, societies, and states; direction of the affairs of a state, community, etc.' Governments exercise control over rather than serve just like the monarchies before them. Bizarrely there are still countries like the United Kingdom which are ruled by a monarch *and* a government that officially answers to the monarch. The UK head of state and that of Commonwealth countries such as Canada, Australia and New Zealand is 'selected' by who in a *single family* had unprotected sex with whom and in what order. Pinch me it can't be true. Ouch! Shit, it is. The demise of monarchies in most countries offered a potential vacuum in which some form of free and fair society could arise and the Cult had that base covered. Monarchies had served its interests but they couldn't continue in the face of such widespread opposition and, anyway, replacing a 'royal' dictatorship that people could see with a dictatorship 'of the people' hiding behind the concept of 'democracy' presented far greater manipulative possibilities and ways of hiding coordinated tyranny behind the illusion of 'freedom'.

Democracy is quite wrongly defined as government selected by the population. This is not the case at all. It is government selected by *some* of the population (and then only in theory). This 'some' doesn't even have to be the majority as we have seen so often in first-past-the-post elections in which the so-called majority party wins fewer votes than the 'losing' parties combined. Democracy can give total power to a party in government from a minority of the votes cast. It's a sleight of hand to sell tyranny as freedom. Seventy-four million Trump-supporting Americans didn't vote for the 'Democratic' Party of Joe Biden in the distinctly dodgy election in 2020 and yet far from acknowledging the wishes and feelings of that great percentage of American society the Cult-owned Biden government set out from day one to destroy them and their right to a voice and opinion. Empty shell Biden and his Cult handlers said they were doing this to 'protect democracy'. Such is the level of lunacy and sickness to which politics has descended. Connect the dots and relate them to the desired outcome – a world government run by self-appointed technocrats and no longer even elected

politicians. While operating through its political agents in government the Cult is at the same time encouraging public distain for politicians by putting idiots and incompetents in theoretical power on the road to deleting them. The idea is to instil a public reaction that says of the technocrats: 'Well, they couldn't do any worse than the pathetic politicians.' It's all about controlling perception and Renegade Minds can see through that while programmed minds cannot when they are ignorant of both the planned outcome and the manipulation techniques employed to secure that end. This knowledge can be learned, however, and fast if people choose to get informed.

Politics may at first sight appear very difficult to control from a central point. I mean look at the 'different' parties and how would you be able to oversee them all and their constituent parts? In truth, it's very straightforward because of their structure. We are back to the pyramid of imposition and acquiescence. Organisations are structured in the same way as the system as a whole. Political parties are not open forums of free expression. They are hierarchies. I was a national spokesman for the British Green Party which claimed to be a different kind of politics in which influence and power was devolved; but I can tell you from direct experience – and it's far worse now – that Green parties are run as hierarchies like all the others however much they may try to hide that fact or kid themselves that it's not true. A very few at the top of all political parties are directing policy and personnel. They decide if you are elevated in the party or serve as a government minister and to do that you have to be a yes man or woman. Look at all the maverick political thinkers who never ascended the greasy pole. If you want to progress within the party or reach 'high-office' you need to fall into line and conform. Exceptions to this are rare indeed. Should you want to run for parliament or Congress you have to persuade the local or state level of the party to select you and for that you need to play the game as dictated by the hierarchy. If you secure election and wish to progress within the greater structure you need to go on conforming to what is acceptable to those running the hierarchy

from the peak of the pyramid. Political parties are perceptual gulags and the very fact that there are party 'Whips' appointed to 'whip' politicians into voting the way the hierarchy demands exposes the ridiculous idea that politicians are elected to serve the people they are supposed to represent. Cult operatives and manipulation has long seized control of major parties that have any chance of forming a government and at least most of those that haven't. A new party forms and the Cult goes to work to infiltrate and direct. This has reached such a level today that you see video compilations of 'leaders' of all parties whether Democrats, Republicans, Conservative, Labour and Green parroting the same Cult mantra of 'Build Back Better' and the 'Great Reset' which are straight off the Cult song-sheet to describe the transformation of global society in response to the Cult-instigated hoaxes of the 'Covid pandemic' and human-caused 'climate change'. To see Caroline Lucas, the Green Party MP that I knew when I was in the party in the 1980s, speaking in support of plans proposed by Cult operative Klaus Schwab representing the billionaire global elite is a real head-shaker.

Many parties – one master

The party system is another mind-trick and was instigated to change the nature of the dictatorship by swapping 'royalty' for dark suits that people believed – though now ever less so – represented their interests. Understanding this trick is to realise that a single force (the Cult) controls all parties either directly in terms of the major ones or through manipulation of perception and ideology with others. You don't need to manipulate Green parties to demand your transformation of society in the name of 'climate change' when they are obsessed with the lie that this is essential to 'save the planet'. You just give them a platform and away they go serving your interests while believing they are being environmentally virtuous. America's political structure is a perfect blueprint for how the two or multi-party system is really a one-party state. The Republican Party is controlled from one step back in the shadows by a group made up of billionaires and their gofers known as neoconservatives or Neocons.

I have exposed them in fine detail in my books and they were the driving force behind the policies of the imbecilic presidency of Boy George Bush which included 9/11 (see *The Trigger* for a comprehensive demolition of the official story), the subsequent ‘war on terror’ (war of terror) and the invasions of Afghanistan and Iraq. The latter was a No-Problem-Reaction-Solution based on claims by Cult operatives, including Bush and British Prime Minister Tony Blair, about Saddam Hussein’s ‘weapons of mass destruction’ which did not exist as war criminals Bush and Blair well knew.

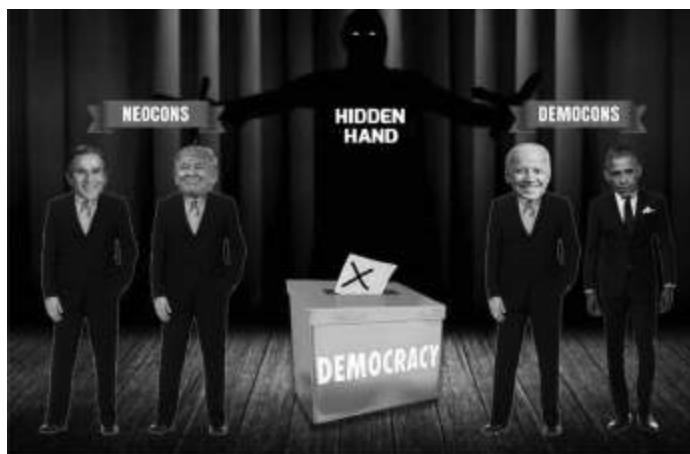


Figure 6: Different front people, different parties – same control system.

The Democratic Party has its own ‘Neocon’ group controlling from the background which I call the ‘Democons’ and here’s the penny-drop – the Neocons and Democons answer to the same masters one step further back into the shadows (Fig 6). At that level of the Cult the Republican and Democrat parties are controlled by the same people and no matter which is in power the Cult is in power. This is how it works in almost every country and certainly in Britain with Conservative, Labour, Liberal Democrat and Green parties now all on the same page whatever the rhetoric may be in their feeble attempts to appear different. Neocons operated at the time of Bush through a think tank called The Project for the New American Century which in September, 2000, published a document entitled *Rebuilding America’s Defenses: Strategies, Forces, and Resources*

For a New Century demanding that America fight ‘multiple, simultaneous major theatre wars’ as a ‘core mission’ to force regime-change in countries including Iraq, Libya and Syria. Neocons arranged for Bush (‘Republican’) and Blair (‘Labour Party’) to front-up the invasion of Iraq and when they departed the Democons orchestrated the targeting of Libya and Syria through Barack Obama (‘Democrat’) and British Prime Minister David Cameron (‘Conservative Party’). We have ‘different’ parties and ‘different’ people, but the same unfolding script. The more the Cult has seized the reigns of parties and personnel the more their policies have transparently pursued the same agenda to the point where the fascist ‘Covid’ impositions of the Conservative junta of Jackboot Johnson in Britain were opposed by the Labour Party because they were not fascist enough. The Labour Party is likened to the US Democrats while the Conservative Party is akin to a British version of the Republicans and on both sides of the Atlantic they all speak the same language and support the direction demanded by the Cult although some more enthusiastically than others. It’s a similar story in country after country because it’s all centrally controlled. Oh, but what about Trump? I’ll come to him shortly. Political ‘choice’ in the ‘party’ system goes like this: You vote for Party A and they get into government. You don’t like what they do so next time you vote for Party B and they get into government. You don’t like what they do when it’s pretty much the same as Party A and why wouldn’t that be with both controlled by the same force? Given that only two, sometimes three, parties have any chance of forming a government to get rid of Party B that you don’t like you have to vote again for Party A which ... you don’t like. This, ladies and gentlemen, is what they call ‘democracy’ which we are told – wrongly – is a term interchangeable with ‘freedom’.

The cult of cults

At this point I need to introduce a major expression of the Global Cult known as Sabbatian-Frankism. Sabbatian is also spelt as Sabbatean. I will summarise here. I have published major exposés

and detailed background in other works. Sabbatian-Frankism combines the names of two frauds posing as 'Jewish' men, Sabbatai Zevi (1626-1676), a rabbi, black magician and occultist who proclaimed he was the Jewish messiah; and Jacob Frank (1726-1791), the Polish 'Jew', black magician and occultist who said he was the reincarnation of 'messiah' Zevi and biblical patriarch Jacob. They worked across two centuries to establish the Sabbatian-Frankist cult that plays a major, indeed central, role in the manipulation of human society by the Global Cult which has its origins much further back in history than Sabbatai Zevi. I should emphasise two points here in response to the shrill voices that will scream 'anti-Semitism': (1) Sabbatian-Frankists are NOT Jewish and only pose as such to hide their cult behind a Jewish façade; and (2) my information about this cult has come from Jewish sources who have long realised that their society and community has been infiltrated and taken over by interloper Sabbatian-Frankists. Infiltration has been the foundation technique of Sabbatian-Frankism from its official origin in the 17th century. Zevi's Sabbatian sect attracted a massive following described as the biggest messianic movement in Jewish history, spreading as far as Africa and Asia, and he promised a return for the Jews to the 'Promised Land' of Israel. Sabbatianism was not Judaism but an inversion of everything that mainstream Judaism stood for. So much so that this sinister cult would have a feast day when Judaism had a fast day and whatever was forbidden in Judaism the Sabbatians were encouraged and even commanded to do. This included incest and what would be today called Satanism. Members were forbidden to marry outside the sect and there was a system of keeping their children ignorant of what they were part of until they were old enough to be trusted not to unknowingly reveal anything to outsiders. The same system is employed to this day by the Global Cult in general which Sabbatian-Frankism has enormously influenced and now largely controls.

Zevi and his Sabbatians suffered a setback with the intervention by the Sultan of the Islamic Ottoman Empire in the Middle East and what is now the Republic of Turkey where Zevi was located. The

Sultan gave him the choice of proving his ‘divinity’, converting to Islam or facing torture and death. Funnily enough Zevi chose to convert or at least appear to. Some of his supporters were disillusioned and drifted away, but many did not with 300 families also converting – only in theory – to Islam. They continued behind this Islamic smokescreen to follow the goals, rules and rituals of Sabbatianism and became known as ‘crypto-Jews’ or the ‘Dönmeh’ which means ‘to turn’. This is rather ironic because they didn’t ‘turn’ and instead hid behind a fake Islamic persona. The process of appearing to be one thing while being very much another would become the calling card of Sabbatianism especially after Zevi’s death and the arrival of the Satanist Jacob Frank in the 18th century when the cult became Sabbatian-Frankism and plumbbed still new depths of depravity and infiltration which included – still includes – human sacrifice and sex with children. Wherever Sabbatians go paedophilia and Satanism follow and is it really a surprise that Hollywood is so infested with child abuse and Satanism when it was established by Sabbatian-Frankists and is still controlled by them? Hollywood has been one of the prime vehicles for global perceptual programming and manipulation. How many believe the version of ‘history’ portrayed in movies when it is a travesty and inversion (again) of the truth? Rabbi Marvin Antelman describes Frankism in his book, *To Eliminate the Opiate*, as ‘a movement of complete evil’ while Jewish professor Gershom Scholem said of Frank in *The Messianic Idea in Judaism*: ‘In all his actions [he was] a truly corrupt and degenerate individual ... one of the most frightening phenomena in the whole of Jewish history.’ Frank was excommunicated by traditional rabbis, as was Zevi, but Frank was undeterred and enjoyed vital support from the House of Rothschild, the infamous banking dynasty whose inner-core are Sabbatian-Frankists and not Jews. Infiltration of the Roman Church and Vatican was instigated by Frank with many Dönmeh ‘turning’ again to convert to Roman Catholicism with a view to hijacking the reins of power. This was the ever-repeating modus operandi and continues to be so. Pose as an advocate of the religion, culture or country that you want to control and then

manipulate your people into the positions of authority and influence largely as advisers, administrators and Svengalis for those that appear to be in power. They did this with Judaism, Christianity (Christian Zionism is part of this), Islam and other religions and nations until Sabbatian-Frankism spanned the world as it does today.

Sabbatian Saudis and the terror network

One expression of the Sabbatian-Frankist Dönme within Islam is the ruling family of Saudi Arabia, the House of Saud, through which came the vile distortion of Islam known as Wahhabism. This is the violent creed followed by terrorist groups like Al-Qaeda and ISIS or Islamic State. Wahhabism is the hand-chopping, head-chopping ‘religion’ of Saudi Arabia which is used to keep the people in a constant state of fear so the interloper House of Saud can continue to rule. Al-Qaeda and Islamic State were lavishly funded by the House of Saud while being created and directed by the Sabbatian-Frankist network in the United States that operates through the Pentagon, CIA and the government in general of whichever ‘party’. The front man for the establishment of Wahhabism in the middle of the 18th century was a Sabbatian-Frankist ‘crypto-Jew’ posing as Islamic called Muhammad ibn Abd al-Wahhab. His daughter would marry the son of Muhammad bin Saud who established the first Saudi state before his death in 1765 with support from the British Empire. Bin Saud’s successors would establish modern Saudi Arabia in league with the British and Americans in 1932 which allowed them to seize control of Islam’s major shrines in Mecca and Medina. They have dictated the direction of Sunni Islam ever since while Iran is the major centre of the Shiite version and here we have the source of at least the public conflict between them. The Sabbatian network has used its Wahhabi extremists to carry out Problem-Reaction-Solution terrorist attacks in the name of ‘Al-Qaeda’ and ‘Islamic State’ to justify a devastating ‘war on terror’, ever-increasing surveillance of the population and to terrify people into compliance. Another insight of the Renegade Mind is the streetwise understanding that

just because a country, location or people are attacked doesn't mean that those apparently representing that country, location or people are not behind the attackers. Often they are *orchestrating* the attacks because of the societal changes that can be then justified in the name of 'saving the population from terrorists'.

I show in great detail in *The Trigger* how Sabbatian-Frankists were the real perpetrators of 9/11 and not '19 Arab hijackers' who were blamed for what happened. Observe what was justified in the name of 9/11 alone in terms of Middle East invasions, mass surveillance and control that fulfilled the demands of the Project for the New American Century document published by the Sabbatian Neocons. What appear to be enemies are on the deep inside players on the same Sabbatian team. Israel and Arab 'royal' dictatorships are all ruled by Sabbatians and the recent peace agreements between Israel and Saudi Arabia, the United Arab Emirates (UAE) and others are only making formal what has always been the case behind the scenes. Palestinians who have been subjected to grotesque tyranny since Israel was bombed and terrorised into existence in 1948 have never stood a chance. Sabbatian-Frankists have controlled Israel (so the constant theme of violence and war which Sabbatians love) and they have controlled the Arab countries that Palestinians have looked to for real support that never comes. 'Royal families' of the Arab world in Saudi Arabia, Bahrain, UAE, etc., are all Sabbatians with allegiance to the aims of the cult and not what is best for their Arabic populations. They have stolen the oil and financial resources from their people by false claims to be 'royal dynasties' with a genetic right to rule and by employing vicious militaries to impose their will.

Satanic 'illumination'

The Satanist Jacob Frank formed an alliance in 1773 with two other Sabbatians, Mayer Amschel Rothschild (1744-1812), founder of the Rothschild banking dynasty, and Jesuit-educated fraudulent Jew, Adam Weishaupt, and this led to the formation of the Bavarian Illuminati, firstly under another name, in 1776. The Illuminati would

be the manipulating force behind the French Revolution (1789-1799) and was also involved in the American Revolution (1775-1783) before and after the Illuminati's official creation. Weishaupt would later become (in public) a Protestant Christian in archetypal Sabbatian style. I read that his name can be decoded as Adam-Weishaupt or 'the first man to lead those who know'. He wasn't a leader in the sense that he was a subordinate, but he did lead those below him in a crusade of transforming human society that still continues today. The theme was confirmed as early as 1785 when a horseman courier called Lanz was reported to be struck by lightning and extensive Illuminati documents were found in his saddlebags. They made the link to Weishaupt and detailed the plan for world takeover. Current events with 'Covid' fascism have been in the making for a very long time. Jacob Frank was jailed for 13 years by the Catholic Inquisition after his arrest in 1760 and on his release he headed for Frankfurt, Germany, home city and headquarters of the House of Rothschild where the alliance was struck with Mayer Amschel Rothschild and Weishaupt. Rothschild arranged for Frank to be given the title of Baron and he became a wealthy nobleman with a big following of Jews in Germany, the Austro-Hungarian Empire and other European countries. Most of them would have believed he was on their side.

The name 'Illuminati' came from the Zohar which is a body of works in the Jewish mystical 'bible' called the Kabbalah. 'Zohar' is the foundation of Sabbatian-Frankist belief and in Hebrew 'Zohar' means 'splendour', 'radiance', 'illuminated', and so we have 'Illuminati'. They claim to be the 'Illuminated Ones' from their knowledge systematically hidden from the human population and passed on through generations of carefully-chosen initiates in the global secret society network or Cult. Hidden knowledge includes an awareness of the Cult agenda for the world and the nature of our collective reality that I will explore later. Cult 'illumination' is symbolised by the torch held by the Statue of Liberty which was gifted to New York by French Freemasons in Paris who knew exactly what it represents. 'Liberty' symbolises the goddess worshipped in

Babylon as Queen Semiramis or Ishtar. The significance of this will become clear. Notice again the ubiquitous theme of inversion with the Statue of 'Liberty' really symbolising mass control ([Fig 7](#)). A mirror-image statute stands on an island in the River Seine in Paris from where New York Liberty originated ([Fig 8](#)). A large replica of the Liberty flame stands on top of the Pont de l'Alma tunnel in Paris where Princess Diana died in a Cult ritual described in *The Biggest Secret*. Lucifer 'the light bringer' is related to all this (and much more as we'll see) and 'Lucifer' is a central figure in Sabbatian-Frankism and its associated Satanism. Sabbatians reject the Jewish Torah, or Pentateuch, the 'five books of Moses' in the Old Testament known as Genesis, Exodus, Leviticus, Numbers, and Deuteronomy which are claimed by Judaism and Christianity to have been dictated by 'God' to Moses on Mount Sinai. Sabbatians say these do not apply to them and they seek to replace them with the Zohar to absorb Judaism and its followers into their inversion which is an expression of a much greater global inversion. They want to delete all religions and force humanity to worship a one-world religion – Sabbatian Satanism that also includes worship of the Earth goddess. Satanic themes are being more and more introduced into mainstream society and while Christianity is currently the foremost target for destruction the others are planned to follow.



Figure 7: The Cult goddess of Babylon disguised as the Statue of Liberty holding the flame of Lucifer the 'light bringer'.



Figure 8: Liberty's mirror image in Paris where the New York version originated.

Marx brothers

Rabbi Marvin Antelman connects the Illuminati to the Jacobins in *To Eliminate the Opiate* and Jacobins were the force behind the French Revolution. He links both to the Bund der Gerechten, or League of the Just, which was the network that inflicted communism/Marxism on the world. Antelman wrote:

The original inner circle of the Bund der Gerechten consisted of born Catholics, Protestants and Jews [Sabbatian-Frankist infiltrators], and those representatives of respective subdivisions formulated schemes for the ultimate destruction of their faiths. The heretical Catholics laid plans which they felt would take a century or more for the ultimate destruction of the church; the apostate Jews for the ultimate destruction of the Jewish religion.

Sabbatian-created communism connects into this anti-religion agenda in that communism does not allow for the free practice of religion. The Sabbatian 'Bund' became the International Communist Party and Communist League and in 1848 'Marxism' was born with the Communist Manifesto of Sabbatian assets Karl Marx and Friedrich Engels. It is absolutely no coincidence that Marxism, just a different name for fascist and other centrally-controlled tyrannies, is being imposed worldwide as a result of the 'Covid' hoax and nor that Marxist/fascist China was the place where the hoax originated. The reason for this will become very clear in the chapter 'Covid: The calculated catastrophe'. The so-called 'Woke' mentality has hijacked

traditional beliefs of the political left and replaced them with far-right make-believe ‘social justice’ better known as Marxism. Woke will, however, be swallowed by its own perceived ‘revolution’ which is really the work of billionaires and billionaire corporations feigning being ‘Woke’. Marxism is being touted by Wokers as a replacement for ‘capitalism’ when we don’t have ‘capitalism’. We have cartelism in which the market is stitched up by the very Cult billionaires and corporations bankrolling Woke. Billionaires love Marxism which keeps the people in servitude while they control from the top.

Terminally naïve Wokers think they are ‘changing the world’ when it’s the Cult that is doing the changing and when they have played their vital part and become surplus to requirements they, too, will be targeted. The Illuminati-Jacobins were behind the period known as ‘The Terror’ in the French Revolution in 1793 and 1794 when Jacobin Maximillian de Robespierre and his Orwellian ‘Committee of Public Safety’ killed 17,000 ‘enemies of the Revolution’ who had once been ‘friends of the Revolution’. Karl Marx (1818-1883), whose Sabbatian creed of Marxism has cost the lives of at least 100 million people, is a hero once again to Wokers who have been systematically kept ignorant of real history by their ‘education’ programming. As a result they now promote a Sabbatian ‘Marxist’ abomination destined at some point to consume them. Rabbi Antelman, who spent decades researching the Sabbatian plot, said of the League of the Just and Karl Marx:

Contrary to popular opinion Karl Marx did not originate the Communist Manifesto. He was paid for his services by the League of the Just, which was known in its country of origin, Germany, as the Bund der Gaeachteten.

Antelman said the text attributed to Marx was the work of other people and Marx ‘was only repeating what others already said’. Marx was ‘a hired hack – lackey of the wealthy Illuminists’. Marx famously said that religion was the ‘opium of the people’ (part of the Sabbatian plan to demonise religion) and Antelman called his books, *To Eliminate the Opiate*. Marx was born Jewish, but his family converted to Christianity (Sabbatian modus operandi) and he

attacked Jews, not least in his book, *A World Without Jews*. In doing so he supported the Sabbatian plan to destroy traditional Jewishness and Judaism which we are clearly seeing today with the vindictive targeting of orthodox Jews by the Sabbatian government of Israel over 'Covid' laws. I don't follow any religion and it has done much damage to the world over centuries and acted as a perceptual straightjacket. Renegade Minds, however, are always asking *why* something is being done. It doesn't matter if they agree or disagree with what is happening – *why* is it happening is the question. The 'why?' can be answered with regard to religion in that religions create interacting communities of believers when the Cult wants to dismantle all discourse, unity and interaction (see 'Covid' lockdowns) and the ultimate goal is to delete all religions for a one-world religion of Cult Satanism worshipping their 'god' of which more later. We see the same 'why?' with gun control in America. I don't have guns and don't want them, but why is the Cult seeking to disarm the population at the same time that law enforcement agencies are armed to their molars and why has every tyrant in history sought to disarm people before launching the final takeover? They include Hitler, Stalin, Pol Pot and Mao who followed confiscation with violent seizing of power. You know it's a Cult agenda by the people who immediately race to the microphones to exploit dead people in multiple shootings. Ultra-Zionist Cult lackey Senator Chuck Schumer was straight on the case after ten people were killed in Boulder, Colorado in March, 2021. Simple rule ... if Schumer wants it the Cult wants it and the same with his ultra-Zionist mate the wild-eyed Senator Adam Schiff. At the same time they were calling for the disarmament of Americans, many of whom live a long way from a police response, Schumer, Schiff and the rest of these pampered clowns were sitting on Capitol Hill behind a razor-wired security fence protected by thousands of armed troops in addition to their own armed bodyguards. Mom and pop in an isolated home? They're just potential mass shooters.

Zion Mainframe

Sabbatian-Frankists and most importantly the Rothschilds were behind the creation of 'Zionism', a political movement that demanded a Jewish homeland in Israel as promised by Sabbatai Zevi. The very symbol of Israel comes from the German meaning of the name Rothschild. Dynasty founder Mayer Amschel Rothschild changed the family name from Bauer to Rothschild, or 'Red-Shield' in German, in deference to the six-pointed 'Star of David' hexagram displayed on the family's home in Frankfurt. The symbol later appeared on the flag of Israel after the Rothschilds were centrally involved in its creation. Hexagrams are not a uniquely Jewish symbol and are widely used in occult ('hidden') networks often as a symbol for Saturn (see my other books for why). Neither are Zionism and Jewishness interchangeable. Zionism is a political movement and philosophy and not a 'race' or a people. Many Jews oppose Zionism and many non-Jews, including US President Joe Biden, call themselves Zionists as does Israel-centric Donald Trump. America's support for the Israel government is pretty much a gimme with ultra-Zionist billionaires and corporations providing fantastic and dominant funding for both political parties. Former Congresswoman Cynthia McKinney has told how she was approached immediately she ran for office to 'sign the pledge' to Israel and confirm that she would always vote in that country's best interests. All American politicians are approached in this way. Anyone who refuses will get no support or funding from the enormous and all-powerful Zionist lobby that includes organisations like mega-lobby group AIPAC, the American Israel Public Affairs Committee. Trump's biggest funder was ultra-Zionist casino and media billionaire Sheldon Adelson while major funders of the Democratic Party include ultra-Zionist George Soros and ultra-Zionist financial and media mogul, Haim Saban. Some may reel back at the suggestion that Soros is an Israel-firster (Sabbatian-controlled Israel-firster), but Renegade Minds watch the actions not the words and everywhere Soros donates his billions the Sabbatian agenda benefits. In the spirit of Sabbatian inversion Soros pledged \$1 billion for a new university network to promote 'liberal values and tackle intolerance'. He made the announcement during his annual speech

at the Cult-owned World Economic Forum in Davos, Switzerland, in January, 2020, after his ‘harsh criticism’ of ‘authoritarian rulers’ around the world. You can only laugh at such brazen mendacity. How *he* doesn’t laugh is the mystery. Translated from the Orwellian ‘liberal values and tackle intolerance’ means teaching non-white people to hate white people and for white people to loathe themselves for being born white. The reason for that will become clear.

The ‘Anti-Semitism’ fraud

Zionists support the Jewish homeland in the land of Palestine which has been the Sabbatian-Rothschild goal for so long, but not for the benefit of Jews. Sabbatians and their global Anti-Semitism Industry have skewed public and political opinion to equate opposing the violent extremes of Zionism to be a blanket attack and condemnation of all Jewish people. Sabbatians and their global Anti-Semitism Industry have skewed public and political opinion to equate opposing the violent extremes of Zionism to be a blanket attack and condemnation of all Jewish people. This is nothing more than a Sabbatian protection racket to stop legitimate investigation and exposure of their agendas and activities. The official definition of ‘anti-Semitism’ has more recently been expanded to include criticism of Zionism – a *political movement* – and this was done to further stop exposure of Sabbatian infiltrators who created Zionism as we know it today in the 19th century. Renegade Minds will talk about these subjects when they know the shit that will come their way. People must decide if they want to know the truth or just cower in the corner in fear of what others will say. Sabbatians have been trying to label me as ‘anti-Semitic’ since the 1990s as I have uncovered more and more about their background and agendas. Useless, gutless, fraudulent ‘journalists’ then just repeat the smears without question and on the day I was writing this section a pair of unquestioning repeaters called Ben Quinn and Archie Bland (how appropriate) outright called me an ‘anti-Semite’ in the establishment propaganda sheet, the London *Guardian*, with no supporting evidence. The

Sabbatian Anti-Semitism Industry said so and who are they to question that? They wouldn't dare. Ironically 'Semitic' refers to a group of languages in the Middle East that are almost entirely Arabic. 'Anti-Semitism' becomes 'anti-Arab' which if the consequences of this misunderstanding were not so grave would be hilarious. Don't bother telling Quinn and Bland. I don't want to confuse them, bless 'em. One reason I am dubbed 'anti-Semitic' is that I wrote in the 1990s that Jewish operatives (Sabbatians) were heavily involved in the Russian Revolution when Sabbatians overthrew the Romanov dynasty. This apparently made me 'anti-Semitic'. Oh, really? Here is a section from *The Trigger*:

British journalist Robert Wilton confirmed these themes in his 1920 book *The Last Days of the Romanovs* when he studied official documents from the Russian government to identify the members of the Bolshevik ruling elite between 1917 and 1919. The Central Committee included 41 Jews among 62 members; the Council of the People's Commissars had 17 Jews out of 22 members; and 458 of the 556 most important Bolshevik positions between 1918 and 1919 were occupied by Jewish people. Only 17 were Russian. Then there were the 23 Jews among the 36 members of the vicious Cheka Soviet secret police established in 1917 who would soon appear all across the country.

Professor Robert Service of Oxford University, an expert on 20th century Russian history, found evidence that ['Jewish'] Leon Trotsky had sought to make sure that Jews were enrolled in the Red Army and were disproportionately represented in the Soviet civil bureaucracy that included the Cheka which performed mass arrests, imprisonment and executions of 'enemies of the people'. A US State Department Decimal File (861.00/5339) dated November 13th, 1918, names [Rothschild banking agent in America] Jacob Schiff and a list of ultra-Zionists as funders of the Russian Revolution leading to claims of a 'Jewish plot', but the key point missed by all is they were not 'Jews' – they were Sabbatian-Frankists.

Britain's Winston Churchill made the same error by mistake or otherwise. He wrote in a 1920 edition of the *Illustrated Sunday Herald* that those behind the Russian revolution were part of a 'worldwide conspiracy for the overthrow of civilisation and for the reconstitution of society on the basis of arrested development, of envious malevolence, and impossible equality' (see 'Woke' today because that has been created by the same network). Churchill said there was no need to exaggerate the part played in the creation of Bolshevism and in the actual bringing about of the Russian

Revolution 'by these international and for the most part atheistical Jews' ['atheistical Jews' = Sabbatians]. Churchill said it is certainly a very great one and probably outweighs all others: 'With the notable exception of Lenin, the majority of the leading figures are Jews.' He went on to describe, knowingly or not, the Sabbatian modus operandi of placing puppet leaders nominally in power while they control from the background:

Moreover, the principal inspiration and driving power comes from the Jewish leaders. Thus Tchitcherin, a pure Russian, is eclipsed by his nominal subordinate, Litvinoff, and the influence of Russians like Bukharin or Lunacharski cannot be compared with the power of Trotsky, or of Zinovieff, the Dictator of the Red Citadel (Petrograd), or of Krassin or Radek – all Jews. In the Soviet institutions the predominance of Jews is even more astonishing. And the prominent, if not indeed the principal, part in the system of terrorism applied by the Extraordinary Commissions for Combatting Counter-Revolution has been taken by Jews, and in some notable cases by Jewesses.

What I said about seriously disproportionate involvement in the Russian Revolution by Jewish 'revolutionaries' (Sabbatians) is provable fact, but truth is no defence against the Sabbatian Anti-Semitism Industry, its repeater parrots like Quinn and Bland, and the now breathtaking network of so-called 'Woke' 'anti-hate' groups with interlocking leaderships and funding which have the role of discrediting and silencing anyone who gets too close to exposing the Sabbatians. We have seen 'truth is no defence' confirmed in legal judgements with the Saskatchewan Human Rights Commission in Canada decreeing this: 'Truthful statements can be presented in a manner that would meet the definition of hate speech, and not all truthful statements must be free from restriction.' Most 'anti-hate' activists, who are themselves consumed by hatred, are too stupid and ignorant of the world to know how they are being used. They are far too far up their own virtue-signalling arses and it's far too dark for them to see anything.

The 'revolution' game

The background and methods of the 'Russian' Revolution are straight from the Sabbatian playbook seen in the French Revolution

and endless others around the world that appear to start as a revolution of the people against tyrannical rule and end up with a regime change to more tyrannical rule overtly or covertly. Wars, terror attacks and regime overthrows follow the Sabbatian cult through history with its agents creating them as Problem-Reaction-Solutions to remove opposition on the road to world domination. Sabbatian dots connect the Rothschilds with the Illuminati, Jacobins of the French Revolution, the 'Bund' or League of the Just, the International Communist Party, Communist League and the Communist Manifesto of Karl Marx and Friedrich Engels that would lead to the Rothschild-funded Russian Revolution. The sequence comes under the heading of 'creative destruction' when you advance to your global goal by continually destroying the status quo to install a new status quo which you then also destroy. The two world wars come to mind. With each new status quo you move closer to your planned outcome. Wars and mass murder are to Sabbatians a collective blood sacrifice ritual. They are obsessed with death for many reasons and one is that death is an inversion of life. Satanists and Sabbatians are obsessed with death and often target churches and churchyards for their rituals. Inversion-obsessed Sabbatians explain the use of inverted symbolism including the *inverted* pentagram and *inverted* cross. The inversion of the cross has been related to targeting Christianity, but the cross was a religious symbol long before Christianity and its inversion is a statement about the Sabbatian mentality and goals more than any single religion.

Sabbatians operating in Germany were behind the rise of the occult-obsessed Nazis and the subsequent Jewish exodus from Germany and Europe to Palestine and the United States after World War Two. The Rothschild dynasty was at the forefront of this both as political manipulators and by funding the operation. Why would Sabbatians help to orchestrate the horrors inflicted on Jews by the Nazis and by Stalin after they organised the Russian Revolution? Sabbatians hate Jews and their religion, that's why. They pose as Jews and secure positions of control within Jewish society and play the 'anti-Semitism' card to protect themselves from exposure

through a global network of organisations answering to the Sabbatian-created-and-controlled globe-spanning intelligence network that involves a stunning web of military-intelligence operatives and operations for a tiny country of just nine million. Among them are Jewish assets who are not Sabbatians but have been convinced by them that what they are doing is for the good of Israel and the Jewish community to protect them from what they have been programmed since childhood to believe is a Jew-hating hostile world. The Jewish community is just a highly convenient cover to hide the true nature of Sabbatians. Anyone getting close to exposing their game is accused by Sabbatian place-people and gofers of 'anti-Semitism' and claiming that all Jews are part of a plot to take over the world. I am not saying that. I am saying that Sabbatians – the *real* Jew-haters – have infiltrated the Jewish community to use them both as a cover and an 'anti-Semitic' defence against exposure. Thus we have the Anti-Semitism Industry targeted researchers in this way and most Jewish people think this is justified and genuine. They don't know that their 'Jewish' leaders and institutions of state, intelligence and military are not controlled by Jews at all, but cultists and stooges of Sabbatian-Frankism. I once added my name to a pro-Jewish freedom petition online and the next time I looked my name was gone and text had been added to the petition blurb to attack me as an 'anti-Semite' such is the scale of perceptual programming.

Moving on America

I tell the story in *The Trigger* and a chapter called 'Atlantic Crossing' how particularly after Israel was established the Sabbatians moved in on the United States and eventually grasped control of government administration, the political system via both Democrats and Republicans, the intelligence community like the CIA and National Security Agency (NSA), the Pentagon and mass media. Through this seriously compartmentalised network Sabbatians and their operatives in Mossad, Israeli Defense Forces (IDF) and US agencies pulled off 9/11 and blamed it on 19 'Al-Qaeda hijackers' dominated by men from, or connected to, Sabbatian-ruled Saudi

Arabia. The '19' were not even on the planes let alone flew those big passenger jets into buildings while being largely incompetent at piloting one-engine light aircraft. 'Hijacker' Hani Hanjour who is said to have flown American Airlines Flight 77 into the Pentagon with a turn and manoeuvre most professional pilots said they would have struggled to do was banned from renting a small plane by instructors at the Freeway Airport in Bowie, Maryland, just *six weeks* earlier on the grounds that he was an incompetent pilot. The Jewish population of the world is just 0.2 percent with even that almost entirely concentrated in Israel (75 percent Jewish) and the United States (around two percent). This two percent and globally 0.2 percent refers to *Jewish* people and not Sabbatian interlopers who are a fraction of that fraction. What a sobering thought when you think of the fantastic influence on world affairs of tiny Israel and that the Project for the New America Century (PNAC) which laid out the blueprint in September, 2000, for America's war on terror and regime change wars in Iraq, Libya and Syria was founded and dominated by Sabbatians known as 'Neocons'. The document conceded that this plan would not be supported politically or publicly without a major attack on American soil and a Problem-Reaction-Solution excuse to send troops to war across the Middle East. Sabbatian Neocons said:

... [The] process of transformation ... [war and regime change] ... is likely to be a long one, absent some catastrophic and catalysing event – like a new Pearl Harbor.

Four months later many of those who produced that document came to power with their inane puppet George Bush from the long-time Sabbatian Bush family. They included Sabbatian Dick Cheney who was officially vice-president, but really de-facto president for the entirety of the 'Bush' government. Nine months after the 'Bush' inauguration came what Bush called at the time 'the Pearl Harbor of the 21st century' and with typical Sabbatian timing and symbolism 2001 was the 60th anniversary of the attack in 1941 by the Japanese Air Force on Pearl Harbor, Hawaii, which allowed President Franklin Delano Roosevelt to take the United States into a Sabbatian-

instigated Second World War that he said in his election campaign that he never would. The evidence is overwhelming that Roosevelt and his military and intelligence networks knew the attack was coming and did nothing to stop it, but they did make sure that America's most essential naval ships were not in Hawaii at the time. Three thousand Americans died in the Pearl Harbor attacks as they did on September 11th. By the 9/11 year of 2001 Sabbatians had widely infiltrated the US government, military and intelligence operations and used their compartmentalised assets to pull off the 'Al-Qaeda' attacks. If you read *The Trigger* it will blow your mind to see the utterly staggering concentration of 'Jewish' operatives (Sabbatian infiltrators) in essential positions of political, security, legal, law enforcement, financial and business power before, during, and after the attacks to make them happen, carry them out, and then cover their tracks – and I do mean *staggering* when you think of that 0.2 percent of the world population and two percent of Americans which are Jewish while Sabbatian infiltrators are a fraction of that. A central foundation of the 9/11 conspiracy was the hijacking of government, military, Air Force and intelligence computer systems in real time through 'back-door' access made possible by Israeli (Sabbatian) 'cyber security' software. Sabbatian-controlled Israel is on the way to rivalling Silicon Valley for domination of cyberspace and is becoming the dominant force in cyber-security which gives them access to entire computer systems and their passcodes across the world. Then add to this that Zionists head (officially) Silicon Valley giants like Google (Larry Page and Sergey Brin), Google-owned YouTube (Susan Wojcicki), Facebook (Mark Zuckerberg and Sheryl Sandberg), and Apple (Chairman Arthur D. Levinson), and that ultra-Zionist hedge fund billionaire Paul Singer has a \$1 billion stake in Twitter which is only nominally headed by 'CEO' pothead Jack Dorsey. As cable news host Tucker Carlson said of Dorsey: 'There used to be debate in the medical community whether dropping a ton of acid had permanent effects and I think that debate has now ended.' Carlson made the comment after Dorsey told a hearing on Capitol Hill (if you cut through his bullshit) that he

believed in free speech so long as he got to decide what you can hear and see. These 'big names' of Silicon Valley are only front men and women for the Global Cult, not least the Sabbatians, who are the true controllers of these corporations. Does anyone still wonder why these same people and companies have been ferociously censoring and banning people (like me) for exposing any aspect of the Cult agenda and especially the truth about the 'Covid' hoax which Sabbatians have orchestrated?

The Jeffrey Epstein paedophile ring was a Sabbatian operation. He was officially 'Jewish' but he was a Sabbatian and women abused by the ring have told me about the high number of 'Jewish' people involved. The Epstein horror has Sabbatian written all over it and matches perfectly their modus operandi and obsession with sex and ritual. Epstein was running a Sabbatian blackmail ring in which famous people with political and other influence were provided with young girls for sex while everything was being filmed and recorded on hidden cameras and microphones at his New York house, Caribbean island and other properties. Epstein survivors have described this surveillance system to me and some have gone public. Once the famous politician or other figure knew he or she was on video they tended to do whatever they were told. Here we go again ...when you've got them by the balls their hearts and minds will follow. Sabbatians use this blackmail technique on a wide scale across the world to entrap politicians and others they need to act as demanded. Epstein's private plane, the infamous 'Lolita Express', had many well-known passengers including Bill Clinton while Bill Gates has flown on an Epstein plane and met with him four years after Epstein had been jailed for paedophilia. They subsequently met many times at Epstein's home in New York according to a witness who was there. Epstein's infamous side-kick was Ghislaine Maxwell, daughter of Mossad agent and ultra-Zionist mega-crooked British businessman, Bob Maxwell, who at one time owned the *Daily Mirror* newspaper. Maxwell was murdered at sea on his boat in 1991 by Sabbatian-controlled Mossad when he became a liability with his

business empire collapsing as a former Mossad operative has confirmed (see *The Trigger*).

Money, money, money, funny money ...

Before I come to the Sabbatian connection with the last three US presidents I will lay out the crucial importance to Sabbatians of controlling banking and finance. Sabbatian Mayer Amschel Rothschild set out to dominate this arena in his family's quest for total global control. What is freedom? It is, in effect, choice. The more choices you have the freer you are and the fewer your choices the more you are enslaved. In the global structure created over centuries by Sabbatians the biggest decider and restrictor of choice is ... money. Across the world if you ask people what they would like to do with their lives and why they are not doing that they will reply 'I don't have the money'. This is the idea. A global elite of multi-billionaires are described as 'greedy' and that is true on one level; but control of money – who has it and who doesn't – is not primarily about greed. It's about control. Sabbatians have seized ever more control of finance and sucked the wealth of the world out of the hands of the population. We talk now, after all, about the 'One-percent' and even then the wealthiest are a lot fewer even than that. This has been made possible by a money scam so outrageous and so vast it could rightly be called the scam of scams founded on creating 'money' out of nothing and 'loaning' that with interest to the population. Money out of nothing is called 'credit'. Sabbatians have asserted control over governments and banking ever more completely through the centuries and secured financial laws that allow banks to lend hugely more than they have on deposit in a confidence trick known as fractional reserve lending. Imagine if you could lend money that doesn't exist and charge the recipient interest for doing so. You would end up in jail. Bankers by contrast end up in mansions, private jets, Malibu and Monaco.

Banks are only required to keep a fraction of their deposits and wealth in their vaults and they are allowed to lend 'money' they don't have called 'credit'. Go into a bank for a loan and if you succeed

the banker will not move any real wealth into your account. They will type into your account the amount of the agreed 'loan' – say £100,000. This is not wealth that really exists; it is non-existent, fresh-air, created-out-of-nothing 'credit' which has never, does not, and will never exist except in theory. Credit is backed by nothing except wind and only has buying power because people think that it has buying power and accept it in return for property, goods and services. I have described this situation as like those cartoon characters you see chasing each other and when they run over the edge of a cliff they keep running forward on fresh air until one of them looks down, realises what's happened, and they all crash into the ravine. The whole foundation of the Sabbatian financial system is to stop people looking down except for periodic moments when they want to crash the system (as in 2008 and 2020 ongoing) and reap the rewards from all the property, businesses and wealth their borrowers had signed over as 'collateral' in return for a 'loan' of fresh air. Most people think that money is somehow created by governments when it comes into existence from the start as a debt through banks 'lending' illusory money called credit. Yes, the very currency of exchange is a *debt* from day one issued as an interest-bearing loan. Why don't governments create money interest-free and lend it to their people interest-free? Governments are controlled by Sabbatians and the financial system is controlled by Sabbatians for whom interest-free money would be a nightmare come true. Sabbatians underpin their financial domination through their global network of central banks, including the privately-owned US Federal Reserve and Britain's Bank of England, and this is orchestrated by a privately-owned central bank coordination body called the Bank for International Settlements in Basle, Switzerland, created by the usual suspects including the Rockefellers and Rothschilds. Central bank chiefs don't answer to governments or the people. They answer to the Bank for International Settlements or, in other words, the Global Cult which is dominated today by Sabbatians.

Built-in disaster

There are so many constituent scams within the overall banking scam. When you take out a loan of thin-air credit only the amount of that loan is theoretically brought into circulation to add to the amount in circulation; but you are paying back the principle plus interest. The additional interest is not created and this means that with every 'loan' there is a shortfall in the money in circulation between what is borrowed and what has to be paid back. There is never even close to enough money in circulation to repay all outstanding public and private debt including interest. Coldly weaved in the very fabric of the system is the certainty that some will lose their homes, businesses and possessions to the banking 'lender'. This is less obvious in times of 'boom' when the amount of money in circulation (and the debt) is expanding through more people wanting and getting loans. When a downturn comes and the money supply contracts it becomes painfully obvious that there is not enough money to service all debt and interest. This is less obvious in times of 'boom' when the amount of money in circulation (and the debt) is expanding through more people wanting and getting loans. When a downturn comes and the money supply contracts and it becomes painfully obvious – as in 2008 and currently – that there is not enough money to service all debt and interest.

Sabbatian banksters have been leading the human population through a calculated series of booms (more debt incurred) and busts (when the debt can't be repaid and the banks get the debtor's tangible wealth in exchange for non-existent 'credit'). With each 'bust' Sabbatian bankers have absorbed more of the world's tangible wealth and we end up with the One-percent. Governments are in bankruptcy levels of debt to the same system and are therefore owned by a system they do not control. The Federal Reserve, 'America's central bank', is privately-owned and American presidents only nominally appoint its chairman or woman to maintain the illusion that it's an arm of government. It's not. The 'Fed' is a cartel of private banks which handed billions to its associates and friends after the crash of 2008 and has been Sabbatian-controlled since it was manipulated into being in 1913 through the covert trickery of Rothschild banking agents Jacob Schiff and Paul

Warburg, and the Sabbatian Rockefeller family. Somehow from a Jewish population of two-percent and globally 0.2 percent (Sabbatian interlopers remember are far smaller) ultra-Zionists headed the Federal Reserve for 31 years between 1987 and 2018 in the form of Alan Greenspan, Bernard Bernanke and Janet Yellen (now Biden's Treasury Secretary) with Yellen's deputy chairman a Israeli-American dual citizen and ultra-Zionist Stanley Fischer, a former governor of the Bank of Israel. Ultra-Zionist Fed chiefs spanned the presidencies of Ronald Reagan ('Republican'), Father George Bush ('Republican'), Bill Clinton ('Democrat'), Boy George Bush ('Republican') and Barack Obama ('Democrat'). We should really add the pre-Greenspan chairman, Paul Adolph Volcker, 'appointed' by Jimmy Carter ('Democrat') who ran the Fed between 1979 and 1987 during the Carter and Reagan administrations before Greenspan took over. Volcker was a long-time associate and business partner of the Rothschilds. No matter what the 'party' officially in power the United States economy was directed by the same force. Here are members of the Obama, Trump and Biden administrations and see if you can make out a common theme.

Barack Obama ('Democrat')

Ultra-Zionists Robert Rubin, Larry Summers, and Timothy Geithner ran the US Treasury in the Clinton administration and two of them reappeared with Obama. Ultra-Zionist Fed chairman Alan Greenspan had manipulated the crash of 2008 through deregulation and jumped ship just before the disaster to make way for ultra-Zionist Bernard Bernanke to hand out trillions to Sabbatian 'too big to fail' banks and businesses, including the ubiquitous ultra-Zionist Goldman Sachs which has an ongoing staff revolving door operation between itself and major financial positions in government worldwide. Obama inherited the fallout of the crash when he took office in January, 2009, and fortunately he had the support of his ultra-Zionist White House Chief of Staff Rahm Emmanuel, son of a terrorist who helped to bomb Israel into being in 1948, and his ultra-Zionist senior adviser David Axelrod, chief strategist in Obama's two

successful presidential campaigns. Emmanuel, later mayor of Chicago and former senior fundraiser and strategist for Bill Clinton, is an example of the Sabbatian policy after Israel was established of migrating insider families to America so their children would be born American citizens. ‘Obama’ chose this financial team throughout his administration to respond to the Sabbatian-instigated crisis:

Timothy Geithner (ultra-Zionist) Treasury Secretary; Jacob J. Lew, Treasury Secretary; Larry Summers (ultra-Zionist), director of the White House National Economic Council; Paul Adolph Volcker (Rothschild business partner), chairman of the Economic Recovery Advisory Board; Peter Orszag (ultra-Zionist), director of the Office of Management and Budget overseeing all government spending; Penny Pritzker (ultra-Zionist), Commerce Secretary; Jared Bernstein (ultra-Zionist), chief economist and economic policy adviser to Vice President Joe Biden; Mary Schapiro (ultra-Zionist), chair of the Securities and Exchange Commission (SEC); Gary Gensler (ultra-Zionist), chairman of the Commodity Futures Trading Commission (CFTC); Sheila Bair (ultra-Zionist), chair of the Federal Deposit Insurance Corporation (FDIC); Karen Mills (ultra-Zionist), head of the Small Business Administration (SBA); Kenneth Feinberg (ultra-Zionist), Special Master for Executive [bail-out] Compensation. Feinberg would be appointed to oversee compensation (with strings) to 9/11 victims and families in a campaign to stop them having their day in court to question the official story. At the same time ultra-Zionist Bernard Bernanke was chairman of the Federal Reserve and these are only some of the ultra-Zionists with allegiance to Sabbatian-controlled Israel in the Obama government. Obama’s biggest corporate donor was ultra-Zionist Goldman Sachs which had employed many in his administration.

Donald Trump ('Republican')

Trump claimed to be an outsider (he wasn’t) who had come to ‘drain the swamp’. He embarked on this goal by immediately appointing ultra-Zionist Steve Mnuchin, a Goldman Sachs employee for 17

years, as his Treasury Secretary. Others included Gary Cohn (ultra-Zionist), chief operating officer of Goldman Sachs, his first Director of the National Economic Council and chief economic adviser, who was later replaced by Larry Kudlow (ultra-Zionist). Trump's senior adviser throughout his four years in the White House was his sinister son-in-law Jared Kushner, a life-long friend of Israel Prime Minister Benjamin Netanyahu. Kushner is the son of a convicted crook who was pardoned by Trump in his last days in office. Other ultra-Zionists in the Trump administration included: Stephen Miller, Senior Policy Adviser; Avrahm Berkowitz, Deputy Adviser to Trump and his Senior Adviser Jared Kushner; Ivanka Trump, Adviser to the President, who converted to Judaism when she married Jared Kushner; David Friedman, Trump lawyer and Ambassador to Israel; Jason Greenblatt, Trump Organization executive vice president and chief legal officer, who was made Special Representative for International Negotiations and the Israeli-Palestinian Conflict; Rod Rosenstein, Deputy Attorney General; Elliot Abrams, Special Representative for Venezuela, then Iran; John Eisenberg, National Security Council Legal Adviser and Deputy Council to the President for National Security Affairs; Anne Neuberger, Deputy National Manager, National Security Agency; Ezra Cohen-Watnick, Acting Under Secretary of Defense for Intelligence; Elan Carr, Special Envoy to monitor and combat anti-Semitism; Len Khodorkovsky, Deputy Special Envoy to monitor and combat anti-Semitism; Reed Cordish, Assistant to the President, Intragovernmental and Technology Initiatives. Trump Vice President Mike Pence and Secretary of State Mike Pompeo, both Christian Zionists, were also vehement supporters of Israel and its goals and ambitions.

Donald 'free-speech believer' Trump pardoned a number of financial and violent criminals while ignoring calls to pardon Julian Assange and Edward Snowden whose crimes are revealing highly relevant information about government manipulation and corruption and the widespread illegal surveillance of the American people by US 'security' agencies. It's so good to know that Trump is on the side of freedom and justice and not mega-criminals with

allegiance to Sabbatian-controlled Israel. These included a pardon for Israeli spy Jonathan Pollard who was jailed for life in 1987 under the Espionage Act. Aviem Sella, the Mossad agent who recruited Pollard, was also pardoned by Trump while Assange sat in jail and Snowden remained in exile in Russia. Sella had 'fled' (was helped to escape) to Israel in 1987 and was never extradited despite being charged under the Espionage Act. A Trump White House statement said that Sella's clemency had been 'supported by Benjamin Netanyahu, Ron Dermer, Israel's US Ambassador, David Friedman, US Ambassador to Israel and Miriam Adelson, wife of leading Trump donor Sheldon Adelson who died shortly before. Other friends of Jared Kushner were pardoned along with Sholom Weiss who was believed to be serving the longest-ever white-collar prison sentence of more than 800 years in 2000. The sentence was commuted of Ponzi-schemer Eliyahu Weinstein who defrauded Jews and others out of \$200 million. I did mention that Assange and Snowden were ignored, right? Trump gave Sabbatians almost everything they asked for in military and political support, moving the US Embassy from Tel Aviv to Jerusalem with its critical symbolic and literal implications for Palestinian statehood, and the 'deal of the Century' designed by Jared Kushner and David Friedman which gave the Sabbatian Israeli government the green light to substantially expand its already widespread program of building illegal Jewish-only settlements in the occupied land of the West Bank. This made a two-state 'solution' impossible by seizing all the land of a potential Palestinian homeland and that had been the plan since 1948 and then 1967 when the Arab-controlled Gaza Strip, West Bank, Sinai Peninsula and Syrian Golan Heights were occupied by Israel. All the talks about talks and road maps and delays have been buying time until the West Bank was physically occupied by Israeli real estate. Trump would have to be a monumentally ill-informed idiot not to see that this was the plan he was helping to complete. The Trump administration was in so many ways the Kushner administration which means the Netanyahu administration which means the Sabbatian administration. I understand why many opposing Cult fascism in all its forms gravitated to Trump, but he

was a crucial part of the Sabbatian plan and I will deal with this in the next chapter.

Joe Biden ('Democrat')

A barely cognitive Joe Biden took over the presidency in January, 2021, along with his fellow empty shell, Vice-President Kamala Harris, as the latest Sabbatian gofers to enter the White House. Names on the door may have changed and the 'party' – the force behind them remained the same as Zionists were appointed to a stream of pivotal areas relating to Sabbatian plans and policy. They included: Janet Yellen, Treasury Secretary, former head of the Federal Reserve, and still another ultra-Zionist running the US Treasury after Mnuchin (Trump), Lew and Geithner (Obama), and Summers and Rubin (Clinton); Anthony Blinken, Secretary of State; Wendy Sherman, Deputy Secretary of State (so that's 'Biden's' Sabbatian foreign policy sorted); Jeff Zients, White House coronavirus coordinator; Rochelle Walensky, head of the Centers for Disease Control; Rachel Levine, transgender deputy health secretary (that's 'Covid' hoax policy under control); Merrick Garland, Attorney General; Alejandro Mayorkas, Secretary of Homeland Security; Cass Sunstein, Homeland Security with responsibility for new immigration laws; Avril Haines, Director of National Intelligence; Anne Neuberger, National Security Agency cybersecurity director (note, cybersecurity); David Cohen, CIA Deputy Director; Ronald Klain, Biden's Chief of Staff (see Rahm Emanuel); Eric Lander, a 'leading geneticist', Office of Science and Technology Policy director (see Smart Grid, synthetic biology agenda); Jessica Rosenworcel, acting head of the Federal Communications Commission (FCC) which controls Smart Grid technology policy and electromagnetic communication systems including 5G. How can it be that so many pivotal positions are held by two-percent of the American population and 0.2 percent of the world population administration after administration no matter who is the president and what is the party? It's a coincidence? Of course it's not and this is why Sabbatians have built their colossal global web of interlocking 'anti-

hate' hate groups to condemn anyone who asks these glaring questions as an 'anti-Semite'. The way that Jewish people horrifically abused in Sabbatian-backed Nazi Germany are exploited to this end is stomach-turning and disgusting beyond words.

Political fusion

Sabbatian manipulation has reversed the roles of Republicans and Democrats and the same has happened in Britain with the Conservative and Labour Parties. Republicans and Conservatives were always labelled the 'right' and Democrats and Labour the 'left', but look at the policy positions now and the Democrat-Labour 'left' has moved further to the 'right' than Republicans and Conservatives under the banner of 'Woke', the Cult-created far-right tyranny. Where once the Democrat-Labour 'left' defended free speech and human rights they now seek to delete them and as I said earlier despite the 'Covid' fascism of the Jackboot Johnson Conservative government in the UK the Labour Party of leader Keir Starmer demanded even more extreme measures. The Labour Party has been very publicly absorbed by Sabbatians after a political and media onslaught against the previous leader, the weak and inept Jeremy Corbyn, over made-up allegations of 'anti-Semitism' both by him and his party. The plan was clear with this 'anti-Semite' propaganda and what was required in response was a swift and decisive 'fuck off' from Corbyn and a statement to expose the Anti-Semitism Industry (Sabbatian) attempt to silence Labour criticism of the Israeli government (Sabbatians) and purge the party of all dissent against the extremes of ultra-Zionism (Sabbatians). Instead Corbyn and his party fell to their knees and appeased the abusers which, by definition, is impossible. Appeasing one demand leads only to a new demand to be appeased until takeover is complete. Like I say – 'fuck off' would have been a much more effective policy and I have used it myself with great effect over the years when Sabbatians are on my case which is most of the time. I consider that fact a great compliment, by the way. The outcome of the Labour Party capitulation is that we now have a Sabbatian-controlled

Conservative Party ‘opposed’ by a Sabbatian-controlled Labour Party in a one-party Sabbatian state that hurtles towards the extremes of tyranny (the Sabbatian cult agenda). In America the situation is the same. Labour’s Keir Starmer spends his days on his knees with his tongue out pointing to Tel Aviv, or I guess now Jerusalem, while Boris Johnson has an ‘anti-Semitism czar’ in the form of former Labour MP John Mann who keeps Starmer company on his prayer mat.

Sabbatian influence can be seen in Jewish members of the Labour Party who have been ejected for criticism of Israel including those from families that suffered in Nazi Germany. Sabbatians despise real Jewish people and target them even more harshly because it is so much more difficult to dub them ‘anti-Semitic’ although in their desperation they do try.

CHAPTER THREE

The Pushbacker sting

Until you realize how easy it is for your mind to be manipulated, you remain the puppet of someone else's game

Evita Ochel

I will use the presidencies of Trump and Biden to show how the manipulation of the one-party state plays out behind the illusion of political choice across the world. No two presidencies could – on the face of it – be more different and apparently at odds in terms of direction and policy.

A Renegade Mind sees beyond the obvious and focuses on outcomes and consequences and not image, words and waffle. The Cult embarked on a campaign to divide America between those who blindly support its agenda (the mentality known as 'Woke') and those who are pushing back on where the Cult and its Sabbatians want to go. This presents infinite possibilities for dividing and ruling the population by setting them at war with each other and allows a perceptual ring fence of demonisation to encircle the Pushbackers in a modern version of the Little Big Horn in 1876 when American cavalry led by Lieutenant Colonel George Custer were drawn into a trap, surrounded and killed by Native American tribes defending their land of thousands of years from being seized by the government. In this modern version the roles are reversed and it's those defending themselves from the Sabbatian government who are surrounded and the government that's seeking to destroy them. This trap was set years ago and to explain how we must return to 2016

and the emergence of Donald Trump as a candidate to be President of the United States. He set out to overcome the best part of 20 other candidates in the Republican Party before and during the primaries and was not considered by many in those early stages to have a prayer of living in the White House. The Republican Party was said to have great reservations about Trump and yet somehow he won the nomination. When you know how American politics works – politics in general – there is no way that Trump could have become the party's candidate unless the Sabbatian-controlled 'Neocons' that run the Republican Party wanted that to happen. We saw the proof in emails and documents made public by WikiLeaks that the Democratic Party hierarchy, or Democons, systematically undermined the campaign of Bernie Sanders to make sure that Sabbatian gofer Hillary Clinton won the nomination to be their presidential candidate. If the Democons could do that then the Neocons in the Republican Party could have derailed Trump in the same way. But they didn't and at that stage I began to conclude that Trump could well be the one chosen to be president. If that was the case the 'why' was pretty clear to see – the goal of dividing America between Cult agenda-supporting Wokers and Pushbackers who gravitated to Trump because he was telling them what they wanted to hear. His constituency of support had been increasingly ignored and voiceless for decades and profoundly through the eight years of Sabbatian puppet Barack Obama. Now here was someone speaking their language of pulling back from the incessant globalisation of political and economic power, the exporting of American jobs to China and elsewhere by 'American' (Sabbatian) corporations, the deletion of free speech, and the mass immigration policies that had further devastated job opportunities for the urban working class of all races and the once American heartlands of the Midwest.

Beware the forked tongue

Those people collectively sighed with relief that at last a political leader was apparently on their side, but another trait of the Renegade Mind is that you look even harder at people telling you

what you want to hear than those who are telling you otherwise. Obviously as I said earlier people wish what they want to hear to be true and genuine and they are much more likely to believe that than someone saying what they don't want to here and don't want to be true. Sales people are taught to be skilled in eliciting by calculated questioning what their customers want to hear and repeating that back to them as their own opinion to get their targets to like and trust them. Assets of the Cult are also sales people in the sense of selling perception. To read Cult manipulation you have to play the long and expanded game and not fall for the Vaudeville show of party politics. Both American parties are vehicles for the Cult and they exploit them in different ways depending on what the agenda requires at that moment. Trump and the Republicans were used to be the focus of dividing America and isolating Pushbackers to open the way for a Biden presidency to become the most extreme in American history by advancing the full-blown Woke (Cult) agenda with the aim of destroying and silencing Pushbackers now labelled Nazi Trump supporters and white supremacists.

Sabbatians wanted Trump in office for the reasons described by ultra-Zionist Saul Alinsky (1909-1972) who was promoting the Woke philosophy through 'community organising' long before anyone had heard of it. In those days it still went by its traditional name of Marxism. The reason for the manipulated Trump phenomenon was laid out in Alinsky's 1971 book, *Rules for Radicals*, which was his blueprint for overthrowing democratic and other regimes and replacing them with Sabbatian Marxism. Not surprisingly his to-do list was evident in the Sabbatian French and Russian 'Revolutions' and that in China which will become very relevant in the next chapter about the 'Covid' hoax. Among Alinsky's followers have been the deeply corrupt Barack Obama, House Speaker Nancy Pelosi and Hillary Clinton who described him as a 'hero'. All three are Sabbatian stooges with Pelosi personifying the arrogant corrupt idiocy that so widely fronts up for the Cult inner core. Predictably as a Sabbatian advocate of the 'light-bringer' Alinsky features Lucifer on the dedication page of his book as the original radical who gained

his own kingdom ('Earth' as we shall see). One of Alinsky's golden radical rules was to pick an individual and focus all attention, hatred and blame on them and not to target faceless bureaucracies and corporations. *Rules for Radicals* is really a Sabbatian handbook with its contents repeatedly employed all over the world for centuries and why wouldn't Sabbatians bring to power their designer-villain to be used as the individual on which all attention, hatred and blame was bestowed? This is what they did and the only question for me is how much Trump knew that and how much he was manipulated. A bit of both, I suspect. This was Alinsky's Trump technique from a man who died in 1972. The technique has spanned history:

Pick the target, freeze it, personalize it, polarize it. Don't try to attack abstract corporations or bureaucracies. Identify a responsible individual. Ignore attempts to shift or spread the blame.

From the moment Trump came to illusory power everything was about him. It wasn't about Republican policy or opinion, but all about Trump. Everything he did was presented in negative, derogatory and abusive terms by the Sabbatian-dominated media led by Cult operations such as CNN, MSNBC, *The New York Times* and the Jeff Bezos-owned *Washington Post* – 'Pick the target, freeze it, personalize it, polarize it.' Trump was turned into a demon to be vilified by those who hated him and a demi-god loved by those who worshipped him. This, in turn, had his supporters, too, presented as equally demonic in preparation for the punchline later down the line when Biden was about to take office. It was here's a Trump, there's a Trump, everywhere a Trump, Trump. Virtually every news story or happening was filtered through the lens of 'The Donald'. You loved him or hated him and which one you chose was said to define you as Satan's spawn or a paragon of virtue. Even supporting some Trump policies or statements and not others was enough for an assault on your character. No shades of grey were or are allowed. Everything is black and white (literally and figuratively). A Californian I knew had her head utterly scrambled by her hatred for Trump while telling people they should love each other. She was so totally consumed by

Trump Derangement Syndrome as it became to be known that this glaring contradiction would never have occurred to her. By definition anyone who criticised Trump or praised his opponents was a hero and this lady described Joe Biden as 'a kind, honest gentleman' when he's a provable liar, mega-crook and vicious piece of work to boot. Sabbatians had indeed divided America using Trump as the fall-guy and all along the clock was ticking on the consequences for his supporters.

In hock to his masters

Trump gave Sabbatians via Israel almost everything they wanted in his four years. Ask and you shall receive was the dynamic between himself and Benjamin Netanyahu orchestrated by Trump's ultra-Zionist son-in-law Jared Kushner, his ultra-Zionist Ambassador to Israel, David Friedman, and ultra-Zionist 'Israel adviser', Jason Greenblatt. The last two were central to the running and protecting from collapse of his business empire, the Trump Organisation, and colossal business failures made him forever beholden to Sabbatian networks that bailed him out. By the start of the 1990s Trump owed \$4 billion to banks that he couldn't pay and almost \$1 billion of that was down to him personally and not his companies. This mega-disaster was the result of building two new casinos in Atlantic City and buying the enormous Taj Mahal operation which led to crippling debt payments. He had borrowed fantastic sums from 72 banks with major Sabbatian connections and although the scale of debt should have had him living in a tent alongside the highway they never foreclosed. A plan was devised to lift Trump from the mire by BT Securities Corporation and Rothschild Inc. and the case was handled by Wilber Ross who had worked for the Rothschilds for 27 years. Ross would be named US Commerce Secretary after Trump's election. Another crucial figure in saving Trump was ultra-Zionist 'investor' Carl Icahn who bought the Taj Mahal casino. Icahn was made special economic adviser on financial regulation in the Trump administration. He didn't stay long but still managed to find time to make a tidy sum of a reported \$31.3 million when he sold his

holdings affected by the price of steel three days before Trump imposed a 235 percent tariff on steel imports. What amazing bits of luck these people have. Trump and Sabbatian operatives have long had a close association and his mentor and legal adviser from the early 1970s until 1986 was the dark and genetically corrupt ultra-Zionist Roy Cohn who was chief counsel to Senator Joseph McCarthy's 'communist' witch-hunt in the 1950s. *Esquire* magazine published an article about Cohn with the headline 'Don't mess with Roy Cohn'. He was described as the most feared lawyer in New York and 'a ruthless master of dirty tricks ... [with] ... more than one Mafia Don on speed dial'. Cohn's influence, contacts, support and protection made Trump a front man for Sabbatians in New York with their connections to one of Cohn's many criminal employers, the 'Russian' Sabbatian Mafia. Israel-centric media mogul Rupert Murdoch was introduced to Trump by Cohn and they started a long friendship. Cohn died in 1986 weeks after being disbarred for unethical conduct by the Appellate Division of the New York State Supreme Court. The wheels of justice do indeed run slow given the length of Cohn's crooked career.

QAnon-sense

We are asked to believe that Donald Trump with his fundamental connections to Sabbatian networks and operatives has been leading the fight to stop the Sabbatian agenda for the fascistic control of America and the world. Sure he has. A man entrapped during his years in the White House by Sabbatian operatives and whose biggest financial donor was casino billionaire Sheldon Adelson who was Sabbatian to his DNA?? Oh, do come on. Trump has been used to divide America and isolate Pushbackers on the Cult agenda under the heading of 'Trump supporters', 'insurrectionists' and 'white supremacists'. The US Intelligence/Mossad Psyop or psychological operation known as QAnon emerged during the Trump years as a central pillar in the Sabbatian campaign to lead Pushbackers into the trap set by those that wished to destroy them. I knew from the start that QAnon was a scam because I had seen the same scenario many

times before over 30 years under different names and I had written about one in particular in the books. ‘Not again’ was my reaction when QAnon came to the fore. The same script is pulled out every few years and a new name added to the letterhead. The story always takes the same form: ‘Insiders’ or ‘the good guys’ in the government-intelligence-military ‘Deep State’ apparatus were going to instigate mass arrests of the ‘bad guys’ which would include the Rockefellers, Rothschilds, Barack Obama, Hillary Clinton, George Soros, etc., etc. Dates are given for when the ‘good guys’ are going to move in, but the dates pass without incident and new dates are given which pass without incident. The central message to Pushbackers in each case is that they don’t have to do anything because there is ‘a plan’ and it is all going to be sorted by the ‘good guys’ on the inside. ‘Trust the plan’ was a QAnon mantra when the only plan was to misdirect Pushbackers into putting their trust in a Psyop they believed to be real. Beware, beware, those who tell you what you want to hear and always check it out. Right up to Biden’s inauguration QAnon was still claiming that ‘the Storm’ was coming and Trump would stay on as president when Biden and his cronies were arrested and jailed. It was never going to happen and of course it didn’t, but what did happen as a result provided that punchline to the Sabbatian Trump/QAnon Psyop.

On January 6th, 2021, a very big crowd of Trump supporters gathered in the National Mall in Washington DC down from the Capitol Building to protest at what they believed to be widespread corruption and vote fraud that stopped Trump being re-elected for a second term as president in November, 2020. I say as someone that does not support Trump or Biden that the evidence is clear that major vote-fixing went on to favour Biden, a man with cognitive problems so advanced he can often hardly string a sentence together without reading the words written for him on the Teleprompter. Glaring ballot discrepancies included serious questions about electronic voting machines that make vote rigging a comparative cinch and hundreds of thousands of paper votes that suddenly appeared during already advanced vote counts and virtually all of

them for Biden. Early Trump leads in crucial swing states suddenly began to close and disappear. The pandemic hoax was used as the excuse to issue almost limitless numbers of mail-in ballots with no checks to establish that the recipients were still alive or lived at that address. They were sent to streams of people who had not even asked for them. Private organisations were employed to gather these ballots and who knows what they did with them before they turned up at the counts. The American election system has been manipulated over decades to become a sick joke with more holes than a Swiss cheese for the express purpose of dictating the results. Then there was the criminal manipulation of information by Sabbatian tech giants like Facebook, Twitter and Google-owned YouTube which deleted pro-Trump, anti-Biden accounts and posts while everything in support of Biden was left alone. Sabbatians wanted Biden to win because after the dividing of America it was time for full-on Woke and every aspect of the Cult agenda to be unleashed.

Hunter gatherer

Extreme Silicon Valley bias included blocking information by the *New York Post* exposing a Biden scandal that should have ended his bid for president in the final weeks of the campaign. Hunter Biden, his monumentally corrupt son, is reported to have sent a laptop to be repaired at a local store and failed to return for it. Time passed until the laptop became the property of the store for non-payment of the bill. When the owner saw what was on the hard drive he gave a copy to the FBI who did nothing even though it confirmed widespread corruption in which the Joe Biden family were using his political position, especially when he was vice president to Obama, to make multiple millions in countries around the world and most notably Ukraine and China. Hunter Biden's one-time business partner Tony Bobulinski went public when the story broke in the *New York Post* to confirm the corruption he saw and that Joe Biden not only knew what was going on he also profited from the spoils. Millions were handed over by a Chinese company with close

connections – like all major businesses in China – to the Chinese communist party of President Xi Jinping. Joe Biden even boasted at a meeting of the Cult's World Economic Forum that as vice president he had ordered the government of Ukraine to fire a prosecutor. What he didn't mention was that the same man just happened to be investigating an energy company which was part of Hunter Biden's corrupt portfolio. The company was paying him big bucks for no other reason than the influence his father had. Overnight Biden's presidential campaign should have been over given that he had lied publicly about not knowing what his son was doing. Instead almost the entire Sabbatian-owned mainstream media and Sabbatian-owned Silicon Valley suppressed circulation of the story. This alone went a mighty way to rigging the election of 2020. Cult assets like Mark Zuckerberg at Facebook also spent hundreds of millions to be used in support of Biden and vote 'administration'.

The Cult had used Trump as the focus to divide America and was now desperate to bring in moronic, pliable, corrupt Biden to complete the double-whammy. No way were they going to let little things like the will of the people thwart their plan. Silicon Valley widely censored claims that the election was rigged because it *was* rigged. For the same reason anyone claiming it was rigged was denounced as a 'white supremacist' including the pathetically few Republican politicians willing to say so. Right across the media where the claim was mentioned it was described as a 'false claim' even though these excuses for 'journalists' would have done no research into the subject whatsoever. Trump won seven million more votes than any sitting president had ever achieved while somehow a cognitively-challenged soon to be 78-year-old who was hidden away from the public for most of the campaign managed to win more votes than any presidential candidate in history. It makes no sense. You only had to see election rallies for both candidates to witness the enthusiasm for Trump and the apathy for Biden. Tens of thousands would attend Trump events while Biden was speaking in empty car parks with often only television crews attending and framing their shots to hide the fact that no one was there. It was pathetic to see

footage come to light of Biden standing at a podium making speeches only to TV crews and party fixers while reading the words written for him on massive Teleprompter screens. So, yes, those protestors on January 6th had a point about election rigging, but some were about to walk into a trap laid for them in Washington by the Cult Deep State and its QAnon Psyop. This was the Capitol Hill riot ludicrously dubbed an ‘insurrection’.

The spider and the fly

Renegade Minds know there are not two ‘sides’ in politics, only one side, the Cult, working through all ‘sides’. It’s a stage show, a puppet show, to direct the perceptions of the population into focusing on diversions like parties and candidates while missing the puppeteers with their hands holding all the strings. The Capitol Hill ‘insurrection’ brings us back to the Little Big Horn. Having created two distinct opposing groupings – Woke and Pushbackers – the trap was about to be sprung. Pushbackers were to be encircled and isolated by associating them all in the public mind with Trump and then labelling Trump as some sort of Confederate leader. I knew immediately that the Capitol riot was a set-up because of two things. One was how easy the rioters got into the building with virtually no credible resistance and secondly I could see – as with the ‘Covid’ hoax in the West at the start of 2020 – how the Cult could exploit the situation to move its agenda forward with great speed. My experience of Cult techniques and activities over more than 30 years has showed me that while they do exploit situations they haven’t themselves created this never happens with events of fundamental agenda significance. Every time major events giving cultists the excuse to rapidly advance their plan you find they are manipulated into being for the specific reason of providing that excuse – Problem-Reaction-Solution. Only a tiny minority of the huge crowd of Washington protestors sought to gain entry to the Capitol by smashing windows and breaching doors. That didn’t matter. The whole crowd and all Pushbackers, even if they did not support Trump, were going to be lumped together as dangerous

insurrectionists and conspiracy theorists. The latter term came into widespread use through a CIA memo in the 1960s aimed at discrediting those questioning the nonsensical official story of the Kennedy assassination and it subsequently became widely employed by the media. It's still being used by inept 'journalists' with no idea of its origin to discredit anyone questioning anything that authority claims to be true. When you are perpetrating a conspiracy you need to discredit the very word itself even though the dictionary definition of conspiracy is merely 'the activity of secretly planning with other people to do something bad or illegal' and 'a general agreement to keep silent about a subject for the purpose of keeping it secret'. On that basis there are conspiracies almost wherever you look. For obvious reasons the Cult and its lapdog media have to claim there are no conspiracies even though the word appears in state laws as with conspiracy to defraud, to murder, and to corrupt public morals.

Agent provocateurs are widely used by the Cult Deep State to manipulate genuine people into acting in ways that suit the desired outcome. By genuine in this case I mean protestors genuinely supporting Trump and claims that the election was stolen. In among them, however, were agents of the state wearing the garb of Trump supporters and QAnon to pump-prime the Capitol riot which some genuine Trump supporters naively fell for. I described the situation as 'Come into my parlour said the spider to the fly'. Leaflets appeared through the Woke paramilitary arm Antifa, the anti-fascist fascists, calling on supporters to turn up in Washington looking like Trump supporters even though they hated him. Some of those arrested for breaching the Capitol Building were sourced to Antifa and its stable mate Black Lives Matter. Both organisations are funded by Cult billionaires and corporations. One man charged for the riot was according to his lawyer a former FBI agent who had held top secret security clearance for 40 years. Attorney Thomas Plofchan said of his client, 66-year-old Thomas Edward Caldwell:

He has held a Top Secret Security Clearance since 1979 and has undergone multiple Special Background Investigations in support of his clearances. After retiring from the Navy, he

worked as a section chief for the Federal Bureau of Investigation from 2009-2010 as a GS-12 [mid-level employee].

He also formed and operated a consulting firm performing work, often classified, for U.S government customers including the US Drug Enforcement Agency, Department of Housing and Urban Development, the US Coast Guard, and the US Army Personnel Command.

A judge later released Caldwell pending trial in the absence of evidence about a conspiracy or that he tried to force his way into the building. *The New York Post* reported a 'law enforcement source' as saying that 'at least two known Antifa members were spotted' on camera among Trump supporters during the riot while one of the rioters arrested was John Earle Sullivan, a seriously extreme Black Lives Matter Trump-hater from Utah who was previously arrested and charged in July, 2020, over a BLM-Antifa riot in which drivers were threatened and one was shot. Sullivan is the founder of Utah-based Insurgence USA which is an affiliate of the Cult-created-and-funded Black Lives Matter movement. Footage appeared and was then deleted by Twitter of Trump supporters calling out Antifa infiltrators and a group was filmed changing into pro-Trump clothing before the riot. Security at the building was *pathetic* – as planned. Colonel Leroy Fletcher Prouty, a man with long experience in covert operations working with the US security apparatus, once described the tell-tale sign to identify who is involved in an assassination. He said:

No one has to direct an assassination – it happens. The active role is played secretly by permitting it to happen. This is the greatest single clue. Who has the power to call off or reduce the usual security precautions?

This principle applies to many other situations and certainly to the Capitol riot of January 6th, 2021.

The sting

With such a big and potentially angry crowd known to be gathering near the Capitol the security apparatus would have had a major police detail to defend the building with National Guard troops on

standby given the strength of feeling among people arriving from all over America encouraged by the QAnon Psyop and statements by Donald Trump. Instead Capitol Police ‘security’ was flimsy, weak, and easily breached. The same number of officers was deployed as on a regular day and that is a blatant red flag. They were not staffed or equipped for a possible riot that had been an obvious possibility in the circumstances. No protective and effective fencing worth the name was put in place and there were no contingency plans. The whole thing was basically a case of standing aside and waving people in. Once inside police mostly backed off apart from one Capitol police officer who ridiculously shot dead unarmed Air Force veteran protestor Ashli Babbitt without a warning as she climbed through a broken window. The ‘investigation’ refused to name or charge the officer after what must surely be considered a murder in the circumstances. They just lifted a carpet and swept. The story was endlessly repeated about five people dying in the ‘armed insurrection’ when there was no report of rioters using weapons. Apart from Babbitt the other four died from a heart attack, strokes and apparently a drug overdose. Capitol police officer Brian Sicknick was reported to have died after being bludgeoned with a fire extinguisher when he was alive after the riot was over and died later of what the Washington Medical Examiner’s Office said was a stroke. Sicknick had no external injuries. The lies were delivered like rapid fire. There was a narrative to build with incessant repetition of the lie until the lie became the accepted ‘everybody knows that’ truth. The ‘Big Lie’ technique of Nazi Propaganda Minister Joseph Goebbels is constantly used by the Cult which was behind the Nazis and is today behind the ‘Covid’ and ‘climate change’ hoaxes. Goebbels said:

If you tell a lie big enough and keep repeating it, people will eventually come to believe it. The lie can be maintained only for such time as the State can shield the people from the political, economic and/or military consequences of the lie. It thus becomes vitally important for the State to use all of its powers to repress dissent, for the truth is the mortal enemy of the lie, and thus by extension, the truth is the greatest enemy of the State.

Most protestors had a free run of the Capitol Building. This allowed pictures to be taken of rioters in iconic parts of the building including the Senate chamber which could be used as propaganda images against all Pushbackers. One Congresswoman described the scene as ‘the worst kind of non-security anybody could ever imagine’. Well, the first part was true, but someone obviously did imagine it and made sure it happened. Some photographs most widely circulated featured people wearing QAnon symbols and now the Psyop would be used to dub all QAnon followers with the ubiquitous fit-all label of ‘white supremacist’ and ‘insurrectionists’. When a Muslim extremist called Noah Green drove his car at two police officers at the Capitol Building killing one in April, 2021, there was no such political and media hysteria. They were just disappointed he wasn’t white.

The witch-hunt

Government prosecutor Michael Sherwin, an aggressive, dark-eyed, professional Rottweiler led the ‘investigation’ and to call it over the top would be to underestimate reality a thousand fold. Hundreds were tracked down and arrested for the crime of having the wrong political views and people were jailed who had done nothing more than walk in the building, committed no violence or damage to property, took a few pictures and left. They were labelled a ‘threat to the Republic’ while Biden sat in the White House signing executive orders written for him that were dismantling ‘the Republic’. Even when judges ruled that a mother and son should not be in jail the government kept them there. Some of those arrested have been badly beaten by prison guards in Washington and lawyers for one man said he suffered a fractured skull and was made blind in one eye. Meanwhile a woman is shot dead for no reason by a Capitol Police officer and we are not allowed to know who he is never mind what has happened to him although that will be *nothing*. The Cult’s QAnon/Trump sting to identify and isolate Pushbackers and then target them on the road to crushing and deleting them was a resounding success. You would have thought the Russians had

invaded the building at gunpoint and lined up senators for a firing squad to see the political and media reaction. Congresswoman Alexandria Ocasio-Cortez is a child in a woman's body, a terrible-twins, me, me, me, Woker narcissist of such proportions that words have no meaning. She said she thought she was going to die when 'insurrectionists' banged on her office door. It turned out she wasn't even in the Capitol Building when the riot was happening and the 'banging' was a Capitol Police officer. She referred to herself as a 'survivor' which is an insult to all those true survivors of violent and sexual abuse while she lives her pampered and privileged life talking drivel for a living. Her Woke colleague and fellow mega-narcissist Rashida Tlaib broke down describing the devastating effect on her, too, of *not being* in the building when the rioters were there. Ocasio-Cortez and Tlaib are members of a fully-Woke group of Congresswomen known as 'The Squad' along with Ilhan Omar and Ayanna Pressley. The Squad from what I can see can be identified by its vehement anti-white racism, anti-white men agenda, and, as always in these cases, the absence of brain cells on active duty.

The usual suspects were on the riot case immediately in the form of Democrat ultra-Zionist senators and operatives Chuck Schumer and Adam Schiff demanding that Trump be impeached for 'his part in the insurrection'. The same pair of prats had led the failed impeachment of Trump over the invented 'Russia collusion' nonsense which claimed Russia had helped Trump win the 2016 election. I didn't realise that Tel Aviv had been relocated just outside Moscow. I must find an up-to-date map. The Russia hoax was a Sabbatian operation to keep Trump occupied and impotent and to stop any rapport with Russia which the Cult wants to retain as a perceptual enemy to be pulled out at will. Puppet Biden began attacking Russia when he came to office as the Cult seeks more upheaval, division and war across the world. A two-year stage show 'Russia collusion inquiry' headed by the not-very-bright former 9/11 FBI chief Robert Mueller, with support from 19 lawyers, 40 FBI agents plus intelligence analysts, forensic accountants and other

staff, devoured tens of millions of dollars and found no evidence of Russia collusion which a ten-year-old could have told them on day one. Now the same moronic Schumer and Schiff wanted a second impeachment of Trump over the Capitol ‘insurrection’ (riot) which the arrested development of Schumer called another ‘Pearl Harbor’ while others compared it with 9/11 in which 3,000 died and, in the case of CNN, with the Rwandan genocide in the 1990s in which an estimated 500,000 to 600,000 were murdered, between 250, 000 and 500,000 women were raped, and populations of whole towns were hacked to death with machetes. To make those comparisons purely for Cult political reasons is beyond insulting to those that suffered and lost their lives and confirms yet again the callous inhumanity that we are dealing with. Schumer is a monumental idiot and so is Schiff, but they serve the Cult agenda and do whatever they’re told so they get looked after. Talking of idiots – another inane man who spanned the Russia and Capitol impeachment attempts was Senator Eric Swalwell who had the nerve to accuse Trump of collusion with the Russians while sleeping with a Chinese spy called Christine Fang or ‘Fang Fang’ which is straight out of a Bond film no doubt starring Klaus Schwab as the bloke living on a secret island and controlling laser weapons positioned in space and pointing at world capitals. Fang Fang plays the part of Bond’s infiltrator girlfriend which I’m sure she would enjoy rather more than sharing a bed with the brainless Swalwell, lying back and thinking of China. The FBI eventually warned Swalwell about Fang Fang which gave her time to escape back to the Chinese dictatorship. How very thoughtful of them. The second Trump impeachment also failed and hardly surprising when an impeachment is supposed to remove a sitting president and by the time it happened Trump was no longer president. These people are running your country America, well, officially anyway. Terrifying isn’t it?

Outcomes tell the story - always

The outcome of all this – and it’s the *outcome* on which Renegade Minds focus, not the words – was that a vicious, hysterical and

obviously pre-planned assault was launched on Pushbackers to censor, silence and discredit them and even targeted their right to earn a living. They have since been condemned as ‘domestic terrorists’ that need to be treated like Al-Qaeda and Islamic State. ‘Domestic terrorists’ is a label the Cult has been trying to make stick since the period of the Oklahoma bombing in 1995 which was blamed on ‘far-right domestic terrorists’. If you read *The Trigger* you will see that the bombing was clearly a Problem-Reaction-Solution carried out by the Deep State during a Bill Clinton administration so corrupt that no dictionary definition of the term would even nearly suffice. Nearly 30,000 troops were deployed from all over America to the empty streets of Washington for Biden’s inauguration. Ten thousand of them stayed on with the pretext of protecting the capital from insurrectionists when it was more psychological programming to normalise the use of the military in domestic law enforcement in support of the Cult plan for a police-military state. Biden’s fascist administration began a purge of ‘wrong-thinkers’ in the military which means anyone that is not on board with Woke. The Capitol Building was surrounded by a fence with razor wire and the Land of the Free was further symbolically and literally dismantled. The circle was completed with the installation of Biden and the exploitation of the QAnon Psyop.

America had never been so divided since the civil war of the 19th century, Pushbackers were isolated and dubbed terrorists and now, as was always going to happen, the Cult immediately set about deleting what little was left of freedom and transforming American society through a swish of the hand of the most controlled ‘president’ in American history leading (officially at least) the most extreme regime since the country was declared an independent state on July 4th, 1776. Biden issued undebated, dictatorial executive orders almost by the hour in his opening days in office across the whole spectrum of the Cult wish-list including diluting controls on the border with Mexico allowing thousands of migrants to illegally enter the United States to transform the demographics of America and import an election-changing number of perceived Democrat

voters. Then there were Biden deportation amnesties for the already illegally resident (estimated to be as high as 20 or even 30 million). A bill before Congress awarded American citizenship to anyone who could prove they had worked in agriculture for just 180 days in the previous two years as 'Big Ag' secured its slave labour long-term. There were the plans to add new states to the union such as Puerto Rico and making Washington DC a state. They are all parts of a plan to ensure that the Cult-owned Woke Democrats would be permanently in power.

Border – what border?

I have exposed in detail in other books how mass immigration into the United States and Europe is the work of Cult networks fuelled by the tens of billions spent to this and other ends by George Soros and his global Open Society (open borders) Foundations. The impact can be seen in America alone where the population has increased by *100 million* in little more than 30 years mostly through immigration. I wrote in *The Answer* that the plan was to have so many people crossing the southern border that the numbers become unstoppable and we are now there under Cult-owned Biden. El Salvador in Central America puts the scale of what is happening into context. A third of the population now lives in the United States, much of it illegally, and many more are on the way. The methodology is to crush Central and South American countries economically and spread violence through machete-wielding psychopathic gangs like MS-13 based in El Salvador and now operating in many American cities. Biden-imposed lax security at the southern border means that it is all but open. He said before his 'election' that he wanted to see a surge towards the border if he became president and that was the green light for people to do just that after election day to create the human disaster that followed for both America and the migrants. When that surge came the imbecilic Alexandria Ocasio-Cortez said it wasn't a 'surge' because they are 'children, not insurgents' and the term 'surge' (used by Biden) was a claim of 'white supremacists'.

This disingenuous lady may one day enter the realm of the most basic intelligence, but it won't be any time soon.

Sabbatians and the Cult are in the process of destroying America by importing violent people and gangs in among the genuine to terrorise American cities and by overwhelming services that cannot cope with the sheer volume of new arrivals. Something similar is happening in Europe as Western society in general is targeted for demographic and cultural transformation and upheaval. The plan demands violence and crime to create an environment of intimidation, fear and division and Soros has been funding the election of district attorneys across America who then stop prosecuting many crimes, reduce sentences for violent crimes and free as many violent criminals as they can. Sabbatians are creating the chaos from which order – their order – can respond in a classic Problem-Reaction-Solution. A Freemasonic moto says ‘Ordo Ab Chao’ (Order out of Chaos) and this is why the Cult is constantly creating chaos to impose a new ‘order’. Here you have the reason the Cult is constantly creating chaos. The ‘Covid’ hoax can be seen with those entering the United States by plane being forced to take a ‘Covid’ test while migrants flooding through southern border processing facilities do not. Nothing is put in the way of mass migration and if that means ignoring the government’s own ‘Covid’ rules then so be it. They know it’s all bullshit anyway. Any pushback on this is denounced as ‘racist’ by Wokers and Sabbatian fronts like the ultra-Zionist Anti-Defamation League headed by the appalling Jonathan Greenblatt which at the same time argues that Israel should not give citizenship and voting rights to more Palestinian Arabs or the ‘Jewish population’ (in truth the Sabbatian network) will lose control of the country.

Society-changing numbers

Biden’s masters have declared that countries like El Salvador are so dangerous that their people must be allowed into the United States for humanitarian reasons when there are fewer murders in large parts of many Central American countries than in US cities like

Baltimore. That is not to say Central America cannot be a dangerous place and Cult-controlled American governments have been making it so since way back, along with the dismantling of economies, in a long-term plan to drive people north into the United States. Parts of Central America are very dangerous, but in other areas the story is being greatly exaggerated to justify relaxing immigration criteria. Migrants are being offered free healthcare and education in the United States as another incentive to head for the border and there is no requirement to be financially independent before you can enter to prevent the resources of America being drained. You can't blame migrants for seeking what they believe will be a better life, but they are being played by the Cult for dark and nefarious ends. The numbers since Biden took office are huge. In February, 2021, more than 100,000 people were known to have tried to enter the US illegally through the southern border (it was 34,000 in the same month in 2020) and in March it was 170,000 – a 418 percent increase on March, 2020. These numbers are only known people, not the ones who get in unseen. The true figure for migrants illegally crossing the border in a single month was estimated by one congressman at 250,000 and that number will only rise under Biden's current policy. Gangs of murdering drug-running thugs that control the Mexican side of the border demand money – thousands of dollars – to let migrants cross the Rio Grande into America. At the same time gun battles are breaking out on the border several times a week between rival Mexican drug gangs (which now operate globally) who are equipped with sophisticated military-grade weapons, grenades and armoured vehicles. While the Capitol Building was being 'protected' from a non-existent 'threat' by thousands of troops, and others were still deployed at the time in the Cult Neocon war in Afghanistan, the southern border of America was left to its fate. This is not incompetence, it is cold calculation.

By March, 2021, there were 17,000 unaccompanied children held at border facilities and many of them are ensnared by people traffickers for paedophile rings and raped on their journey north to America. This is not conjecture – this is fact. Many of those designated

children are in reality teenage boys or older. Meanwhile Wokers posture their self-purity for encouraging poor and tragic people to come to America and face this nightmare both on the journey and at the border with the disgusting figure of House Speaker Nancy Pelosi giving disingenuous speeches about caring for migrants. The woman's evil. Wokers condemned Trump for having children in cages at the border (so did Obama, *Shhhh*), but now they are sleeping on the floor without access to a shower with one border facility 729 percent over capacity. The Biden insanity even proposed flying migrants from the southern border to the northern border with Canada for 'processing'. The whole shambles is being overseen by ultra-Zionist Secretary of Homeland Security, the moronic liar Alejandro Mayorkas, who banned news cameras at border facilities to stop Americans seeing what was happening. Mayorkas said there was not a ban on news crews; it was just that they were not allowed to film. Alongside him at Homeland Security is another ultra-Zionist Cass Sunstein appointed by Biden to oversee new immigration laws. Sunstein despises conspiracy researchers to the point where he suggests they should be banned or *taxed* for having such views. The man is not bonkers or anything. He's perfectly well-adjusted, but adjusted to what is the question. Criticise what is happening and you are a 'white supremacist' when earlier non-white immigrants also oppose the numbers which effect their lives and opportunities. Black people in poor areas are particularly damaged by uncontrolled immigration and the increased competition for work opportunities with those who will work for less. They are also losing voting power as Hispanics become more dominant in former black areas. It's a downward spiral for them while the billionaires behind the policy drone on about how much they care about black people and 'racism'. None of this is about compassion for migrants or black people – that's just wind and air. Migrants are instead being mercilessly exploited to transform America while the countries they leave are losing their future and the same is true in Europe. Mass immigration may now be the work of Woke Democrats, but it can be traced back to the 1986 Immigration Reform and Control Act (it

wasn't) signed into law by Republican hero President Ronald Reagan which gave amnesty to millions living in the United States illegally and other incentives for people to head for the southern border. Here we have the one-party state at work again.

Save me syndrome

Almost every aspect of what I have been exposing as the Cult agenda was on display in even the first days of 'Biden' with silencing of Pushbackers at the forefront of everything. A Renegade Mind will view the Trump years and QAnon in a very different light to their supporters and advocates as the dots are connected. The QAnon/Trump Psyop has given the Cult all it was looking for. We may not know how much, or little, that Trump realised he was being used, but that's a side issue. This pincer movement produced the desired outcome of dividing America and having Pushbackers isolated. To turn this around we have to look at new routes to empowerment which do not include handing our power to other people and groups through what I will call the 'Save Me Syndrome' – 'I want someone else to do it so that I don't have to'. We have seen this at work throughout human history and the QAnon/Trump Psyop is only the latest incarnation alongside all the others. Religion is an obvious expression of this when people look to a 'god' or priest to save them or tell them how to be saved and then there are 'save me' politicians like Trump. Politics is a diversion and not a 'saviour'. It is a means to block positive change, not make it possible.

Save Me Syndrome always comes with the same repeating theme of handing your power to whom or what you believe will save you while your real 'saviour' stares back from the mirror every morning. Renegade Minds are constantly vigilant in this regard and always asking the question 'What can I do?' rather than 'What can someone else do for me?' Gandhi was right when he said: 'You must be the change you want to see in the world.' We are indeed the people we have been waiting for. We are presented with a constant raft of reasons to concede that power to others and forget where the real power is. Humanity has the numbers and the Cult does not. It has to

use diversion and division to target the unstoppable power that comes from unity. Religions, governments, politicians, corporations, media, QAnon, are all different manifestations of this power-diversion and dilution. Refusing to give your power to governments and instead handing it to Trump and QAnon is not to take a new direction, but merely to recycle the old one with new names on the posters. I will explore this phenomenon as we proceed and how to break the cycles and recycles that got us here through the mists of repeating perception and so repeating history.

For now we shall turn to the most potent example in the entire human story of the consequences that follow when you give your power away. I am talking, of course, of the 'Covid' hoax.

CHAPTER FOUR

'Covid': Calculated catastrophe

Facts are threatening to those invested in fraud
DaShanne Stokes

We can easily unravel the real reason for the 'Covid pandemic' hoax by employing the Renegade Mind methodology that I have outlined this far. We'll start by comparing the long-planned Cult outcome with the 'Covid pandemic' outcome. Know the outcome and you'll see the journey.

I have highlighted the plan for the Hunger Games Society which has been in my books for so many years with the very few controlling the very many through ongoing dependency. To create this dependency it is essential to destroy independent livelihoods, businesses and employment to make the population reliant on the state (the Cult) for even the basics of life through a guaranteed pittance income. While independence of income remained these Cult ambitions would be thwarted. With this knowledge it was easy to see where the 'pandemic' hoax was going once talk of 'lockdowns' began and the closing of all but perceived 'essential' businesses to 'save' us from an alleged 'deadly virus'. Cult corporations like Amazon and Walmart were naturally considered 'essential' while mom and pop shops and stores had their doors closed by fascist decree. As a result with every new lockdown and new regulation more small and medium, even large businesses not owned by the Cult, went to the wall while Cult giants and their frontmen and women grew financially fatter by the second. Mom and pop were

denied an income and the right to earn a living and the wealth of people like Jeff Bezos (Amazon), Mark Zuckerberg (Facebook) and Sergei Brin and Larry Page (Google/Alphabet) have reached record levels. The Cult was increasing its own power through further dramatic concentrations of wealth while the competition was being destroyed and brought into a state of dependency. Lockdowns have been instigated to secure that very end and were never anything to do with health. My brother Paul spent 45 years building up a bus repair business, but lockdowns meant buses were running at a fraction of normal levels for months on end. Similar stories can told in their hundreds of millions worldwide. Efforts of a lifetime coldly destroyed by Cult multi-billionaires and their lackeys in government and law enforcement who continued to earn their living from the taxation of the people while denying the right of the same people to earn theirs. How different it would have been if those making and enforcing these decisions had to face the same financial hardships of those they affected, but they never do.

Gates of Hell

Behind it all in the full knowledge of what he is doing and why is the psychopathic figure of Cult operative Bill Gates. His puppet Tedros at the World Health Organization declared 'Covid' a pandemic in March, 2020. The WHO had changed the definition of a 'pandemic' in 2009 just a month before declaring the 'swine flu pandemic' which would not have been so under the previous definition. The same applies to 'Covid'. The definition had included... 'an infection by an infectious agent, occurring simultaneously in different countries, with a significant mortality rate relative to the proportion of the population infected'. The new definition removed the need for 'significant mortality'. The 'pandemic' has been fraudulent even down to the definition, but Gates demanded economy-destroying lockdowns, school closures, social distancing, mandatory masks, a 'vaccination' for every man, woman and child on the planet and severe consequences and restrictions for those that refused. Who gave him this power? The

Cult did which he serves like a little boy in short trousers doing what his daddy tells him. He and his psychopathic missus even smiled when they said that much worse was to come (what they knew was planned to come). Gates responded in the matter-of-fact way of all psychopaths to a question about the effect on the world economy of what he was doing:

Well, it won't go to zero but it will shrink. Global GDP is probably going to take the biggest hit ever [Gates was smiling as he said this] ... in my lifetime this will be the greatest economic hit. But you don't have a choice. People act as if you have a choice. People don't feel like going to the stadium when they might get infected ... People are deeply affected by seeing these stats, by knowing they could be part of the transmission chain, old people, their parents and grandparents, could be affected by this, and so you don't get to say ignore what is going on here.

There will be the ability to open up, particularly in rich countries, if things are done well over the next few months, but for the world at large normalcy only returns when we have largely vaccinated the entire population.

The man has no compassion or empathy. How could he when he's a psychopath like all Cult players? My own view is that even beyond that he is very seriously mentally ill. Look in his eyes and you can see this along with his crazy flailing arms. You don't do what he has done to the world population since the start of 2020 unless you are mentally ill and at the most extreme end of psychopathic. You especially don't do it when to you know, as we shall see, that cases and deaths from 'Covid' are fakery and a product of monumental figure massaging. 'These stats' that Gates referred to are based on a 'test' that's not testing for the 'virus' as he has known all along. He made his fortune with big Cult support as an infamously ruthless software salesman and now buys global control of 'health' (death) policy without the population he affects having any say. It's a breathtaking outrage. Gates talked about people being deeply affected by fear of 'Covid' when that was because of *him* and his global network lying to them minute-by-minute supported by a lying media that he seriously influences and funds to the tune of hundreds of millions. He's handed big sums to media operations including the BBC, NBC, Al Jazeera, Univision, *PBS NewsHour*,

ProPublica, National Journal, The Guardian, The Financial Times, The Atlantic, Texas Tribune, USA Today publisher Gannett, Washington Monthly, Le Monde, Center for Investigative Reporting, Pulitzer Center on Crisis Reporting, National Press Foundation, International Center for Journalists, Solutions Journalism Network, the Poynter Institute for Media Studies, and many more. Gates is everywhere in the ‘Covid’ hoax and the man must go to prison – or a mental facility – for the rest of his life and his money distributed to those he has taken such enormous psychopathic pleasure in crushing.

The Muscle

The Hunger Games global structure demands a police-military state – a fusion of the two into one force – which viciously imposes the will of the Cult on the population and protects the Cult from public rebellion. In that regard, too, the ‘Covid’ hoax just keeps on giving. Often unlawful, ridiculous and contradictory ‘Covid’ rules and regulations have been policed across the world by moronic automatons and psychopaths made faceless by face-nappy masks and acting like the Nazi SS and fascist blackshirts and brownshirts of Hitler and Mussolini. The smallest departure from the rules decreed by the psychos in government and their clueless gofers were jumped upon by the face-nappy fascists. Brutality against public protestors soon became commonplace even on girls, women and old people as the brave men with the batons – the Face-Nappies as I call them – broke up peaceful protests and handed out fines like confetti to people who couldn’t earn a living let alone pay hundreds of pounds for what was once an accepted human right. Robot Face-Nappies of Nottingham police in the English East Midlands fined one group £11,000 for attending a child’s birthday party. For decades I charted the transformation of law enforcement as genuine, decent officers were replaced with psychopaths and the brain dead who would happily and brutally do whatever their masters told them. Now they were let loose on the public and I would emphasise the point that none of this just happened. The step-by-step change in the dynamic between police and public was orchestrated from the shadows by

those who knew where this was all going and the same with the perceptual reframing of those in all levels of authority and official administration through ‘training courses’ by organisations such as Common Purpose which was created in the late 1980s and given a massive boost in Blair era Britain until it became a global phenomenon. Supposed public ‘servants’ began to view the population as the enemy and the same was true of the police. This was the start of the explosion of behaviour manipulation organisations and networks preparing for the all-war on the human psyche unleashed with the dawn of 2020. I will go into more detail about this later in the book because it is a core part of what is happening.

Police desecrated beauty spots to deter people gathering and arrested women for walking in the countryside alone ‘too far’ from their homes. We had arrogant, clueless sergeants in the Isle of Wight police where I live posting on Facebook what they insisted the population must do or else. A schoolmaster sergeant called Radford looked young enough for me to ask if his mother knew he was out, but he was posting what he *expected* people to do while a Sergeant Wilkinson boasted about fining lads for meeting in a McDonald’s car park where they went to get a lockdown takeaway. Wilkinson added that he had even cancelled their order. What a pair of prats these people are and yet they have increasingly become the norm among Jackboot Johnson’s Yellowshirts once known as the British police. This was the theme all over the world with police savagery common during lockdown protests in the United States, the Netherlands, and the fascist state of Victoria in Australia under its tyrannical and again moronic premier Daniel Andrews. Amazing how tyrannical and moronic tend to work as a team and the same combination could be seen across America as arrogant, narcissistic Woke governors and mayors such as Gavin Newsom (California), Andrew Cuomo (New York), Gretchen Whitmer (Michigan), Lori Lightfoot (Chicago) and Eric Garcetti (Los Angeles) did their Nazi and Stalin impressions with the full support of the compliant brutality of their enforcers in uniform as they arrested small business owners defying

fascist shutdown orders and took them to jail in ankle shackles and handcuffs. This happened to bistro owner Marlena Pavlos-Hackney in Gretchen Whitmer's fascist state of Michigan when police arrived to enforce an order by a state-owned judge for 'putting the community at risk' at a time when other states like Texas were dropping restrictions and migrants were pouring across the southern border without any 'Covid' questions at all. I'm sure there are many officers appalled by what they are ordered to do, but not nearly enough of them. If they were truly appalled they would not do it. As the months passed every opportunity was taken to have the military involved to make their presence on the streets ever more familiar and 'normal' for the longer-term goal of police-military fusion.

Another crucial element to the Hunger Games enforcement network has been encouraging the public to report neighbours and others for 'breaking the lockdown rules'. The group faced with £11,000 in fines at the child's birthday party would have been dobbed-in by a neighbour with a brain the size of a pea. The technique was most famously employed by the Stasi secret police in communist East Germany who had public informants placed throughout the population. A police chief in the UK says his force doesn't need to carry out 'Covid' patrols when they are flooded with so many calls from the public reporting other people for visiting the beach. Dorset police chief James Vaughan said people were so enthusiastic about snitching on their fellow humans they were now operating as an auxiliary arm of the police: 'We are still getting around 400 reports a week from the public, so we will respond to reports ... We won't need to be doing hotspot patrols because people are very quick to pick the phone up and tell us.' Vaughan didn't say that this is a pillar of all tyrannies of whatever complexion and the means to hugely extend the reach of enforcement while spreading distrust among the people and making them wary of doing anything that might get them reported. Those narcissistic Isle of Wight sergeants Radford and Wilkinson never fail to add a link to their Facebook posts where the public can inform on their fellow slaves.

Neither would be self-aware enough to realise they were imitating the Stasi which they might well never have heard of. Government psychologists that I will expose later laid out a policy to turn communities against each other in the same way.

A coincidence? Yep, and I can knit fog

I knew from the start of the alleged pandemic that this was a Cult operation. It presented limitless potential to rapidly advance the Cult agenda and exploit manipulated fear to demand that every man, woman and child on the planet was ‘vaccinated’ in a process never used on humans before which infuses self-replicating *synthetic* material into human cells. Remember the plan to transform the human body from a biological to a synthetic biological state. I’ll deal with the ‘vaccine’ (that’s not actually a vaccine) when I focus on the genetic agenda. Enough to say here that mass global ‘vaccination’ justified by this ‘new virus’ set alarms ringing after 30 years of tracking these people and their methods. The ‘Covid’ hoax officially beginning in China was also a big red flag for reasons I will be explaining. The agenda potential was so enormous that I could dismiss any idea that the ‘virus’ appeared naturally. Major happenings with major agenda implications never occur without Cult involvement in making them happen. My questions were twofold in early 2020 as the media began its campaign to induce global fear and hysteria: Was this alleged infectious agent released on purpose by the Cult or did it even exist at all? I then did what I always do in these situations. I sat, observed and waited to see where the evidence and information would take me. By March and early April synchronicity was strongly – and ever more so since then – pointing me in the direction of *there is no ‘virus’*. I went public on that with derision even from swathes of the alternative media that voiced a scenario that the Chinese government released the ‘virus’ in league with Deep State elements in the United States from a top-level bio-lab in Wuhan where the ‘virus’ is said to have first appeared. I looked at that possibility, but I didn’t buy it for several reasons. Deaths from the ‘virus’ did not in any way match what they

would have been with a ‘deadly bioweapon’ and it is much more effective if you sell the *illusion* of an infectious agent rather than having a real one unless you can control through injection who has it and who doesn’t. Otherwise you lose control of events. A made-up ‘virus’ gives you a blank sheet of paper on which you can make it do whatever you like and have any symptoms or mutant ‘variants’ you choose to add while a real infectious agent would limit you to what it actually does. A phantom disease allows you to have endless ludicrous ‘studies’ on the ‘Covid’ dollar to widen the perceived impact by inventing ever more ‘at risk’ groups including one study which said those who walk slowly may be almost four times more likely to die from the ‘virus’. People are in psychiatric wards for less.

A real ‘deadly bioweapon’ can take out people in the hierarchy that are not part of the Cult, but essential to its operation. Obviously they don’t want that. Releasing a real disease means you immediately lose control of it. Releasing an illusory one means you don’t. Again it’s vital that people are extra careful when dealing with what they want to hear. A bioweapon unleashed from a Chinese laboratory in collusion with the American Deep State may fit a conspiracy narrative, but is it true? Would it not be far more effective to use the excuse of a ‘virus’ to justify the real bioweapon – the ‘vaccine’? That way your disease agent does not have to be transmitted and arrives directly through a syringe. I saw a French virologist Luc Montagnier quoted in the alternative media as saying he had discovered that the alleged ‘new’ severe acute respiratory syndrome coronavirus , or SARS-CoV-2, was made artificially and included elements of the human immunodeficiency ‘virus’ (HIV) and a parasite that causes malaria. SARS-CoV-2 is alleged to trigger an alleged illness called Covid-19. I remembered Montagnier’s name from my research years before into claims that an HIV ‘retrovirus’ causes AIDS – claims that were demolished by Berkeley virologist Peter Duesberg who showed that no one had ever proved that HIV causes acquired immunodeficiency syndrome or AIDS. Claims that become accepted as fact, publicly and medically, with no proof whatsoever are an ever-recurring story that profoundly applies to

'Covid'. Nevertheless, despite the lack of proof, Montagnier's team at the Pasteur Institute in Paris had a long dispute with American researcher Robert Gallo over which of them discovered and isolated the HIV 'virus' and with *no evidence* found it to cause AIDS. You will see later that there is also no evidence that any 'virus' causes any disease or that there is even such a thing as a 'virus' in the way it is said to exist. The claim to have 'isolated' the HIV 'virus' will be presented in its real context as we come to the shocking story – and it is a story – of SARS-CoV-2 and so will Montagnier's assertion that he identified the full SARS-CoV-2 genome.

Hoax in the making

We can pick up the 'Covid' story in 2010 and the publication by the Rockefeller Foundation of a document called 'Scenarios for the Future of Technology and International Development'. The inner circle of the Rockefeller family has been serving the Cult since John D. Rockefeller (1839-1937) made his fortune with Standard Oil. It is less well known that the same Rockefeller – the Bill Gates of his day – was responsible for establishing what is now referred to as 'Big Pharma', the global network of pharmaceutical companies that make outrageous profits dispensing scalpel and drug 'medicine' and are obsessed with pumping vaccines in ever-increasing number into as many human arms and backsides as possible. John D. Rockefeller was the driving force behind the creation of the 'education' system in the United States and elsewhere specifically designed to program the perceptions of generations thereafter. The Rockefeller family donated exceptionally valuable land in New York for the United Nations building and were central in establishing the World Health Organization in 1948 as an agency of the UN which was created from the start as a Trojan horse and stalking horse for world government. Now enter Bill Gates. His family and the Rockefellers have long been extremely close and I have seen genealogy which claims that if you go back far enough the two families fuse into the same bloodline. Gates has said that the Bill and Melinda Gates Foundation was inspired by the Rockefeller Foundation and why not

when both are serving the same Cult? Major tax-exempt foundations are overwhelmingly criminal enterprises in which Cult assets fund the Cult agenda in the guise of 'philanthropy' while avoiding tax in the process. Cult operatives can become mega-rich in their role of front men and women for the psychopaths at the inner core and they, too, have to be psychopaths to knowingly serve such evil. Part of the deal is that a big percentage of the wealth gleaned from representing the Cult has to be spent advancing the ambitions of the Cult and hence you have the Rockefeller Foundation, Bill and Melinda Gates Foundation (and so many more) and people like George Soros with his global Open Society Foundations spending their billions in pursuit of global Cult control. Gates is a global public face of the Cult with his interventions in world affairs including Big Tech influence; a central role in the 'Covid' and 'vaccine' scam; promotion of the climate change shakedown; manipulation of education; geoengineering of the skies; and his food-control agenda as the biggest owner of farmland in America, his GMO promotion and through other means. As one writer said: 'Gates monopolizes or wields disproportionate influence over the tech industry, global health and vaccines, agriculture and food policy (including biopiracy and fake food), weather modification and other climate technologies, surveillance, education and media.' The almost limitless wealth secured through Microsoft and other not-allowed-to-fail ventures (including vaccines) has been ploughed into a long, long list of Cult projects designed to enslave the entire human race. Gates and the Rockefellers have been working as one unit with the Rockefeller-established World Health Organization leading global 'Covid' policy controlled by Gates through his mouth-piece Tedros. Gates became the WHO's biggest funder when Trump announced that the American government would cease its donations, but Biden immediately said he would restore the money when he took office in January, 2021. The Gates Foundation (the Cult) owns through limitless funding the world health system and the major players across the globe in the 'Covid' hoax.

Okay, with that background we return to that Rockefeller Foundation document of 2010 headed ‘Scenarios for the Future of Technology and International Development’ and its ‘imaginary’ epidemic of a virulent and deadly influenza strain which infected 20 percent of the global population and killed eight million in seven months. The Rockefeller scenario was that the epidemic destroyed economies, closed shops, offices and other businesses and led to governments imposing fierce rules and restrictions that included mandatory wearing of face masks and body-temperature checks to enter communal spaces like railway stations and supermarkets. The document predicted that even after the height of the Rockefeller-envisioned epidemic the authoritarian rule would continue to deal with further pandemics, transnational terrorism, environmental crises and rising poverty. Now you may think that the Rockefellers are our modern-day seers or alternatively, and rather more likely, that they well knew what was planned a few years further on. Fascism had to be imposed, you see, to ‘protect citizens from risk and exposure’. The Rockefeller scenario document said:

During the pandemic, national leaders around the world flexed their authority and imposed airtight rules and restrictions, from the mandatory wearing of face masks to body-temperature checks at the entries to communal spaces like train stations and supermarkets. Even after the pandemic faded, this more authoritarian control and oversight of citizens and their activities stuck and even intensified. In order to protect themselves from the spread of increasingly global problems – from pandemics and transnational terrorism to environmental crises and rising poverty – leaders around the world took a firmer grip on power.

At first, the notion of a more controlled world gained wide acceptance and approval. Citizens willingly gave up some of their sovereignty – and their privacy – to more paternalistic states in exchange for greater safety and stability. Citizens were more tolerant, and even eager, for top-down direction and oversight, and national leaders had more latitude to impose order in the ways they saw fit.

In developed countries, this heightened oversight took many forms: biometric IDs for all citizens, for example, and tighter regulation of key industries whose stability was deemed vital to national interests. In many developed countries, enforced cooperation with a suite of new regulations and agreements slowly but steadily restored both order and, importantly, economic growth.

There we have the prophetic Rockefellers in 2010 and three years later came their paper for the Global Health Summit in Beijing, China, when government representatives, the private sector, international organisations and groups met to discuss the next 100 years of 'global health'. The Rockefeller Foundation-funded paper was called 'Dreaming the Future of Health for the Next 100 Years' and more prophecy ensued as it described a dystopian future: 'The abundance of data, digitally tracking and linking people may mean the 'death of privacy' and may replace physical interaction with transient, virtual connection, generating isolation and raising questions of how values are shaped in virtual networks.' Next in the 'Covid' hoax preparation sequence came a 'table top' simulation in 2018 for another 'imaginary' pandemic of a disease called Clade X which was said to kill 900 million people. The exercise was organised by the Gates-funded Johns Hopkins University's Center for Health Security in the United States and this is the very same university that has been compiling the disgustingly and systematically erroneous global figures for 'Covid' cases and deaths. Similar Johns Hopkins health crisis scenarios have included the Dark Winter exercise in 2001 and Atlantic Storm in 2005.

Nostradamus 201

For sheer predictive genius look no further prophecy-watchers than the Bill Gates-funded Event 201 held only six weeks before the 'coronavirus pandemic' is supposed to have broken out in China and Event 201 was based on a scenario of a global 'coronavirus pandemic'. Melinda Gates, the great man's missus, told the BBC that he had 'prepared for years' for a coronavirus pandemic which told us what we already knew. Nostradamugates had predicted in a TED talk in 2015 that a pandemic was coming that would kill a lot of people and demolish the world economy. My god, the man is a machine – possibly even literally. Now here he was only weeks before the real thing funding just such a simulated scenario and involving his friends and associates at Johns Hopkins, the World Economic Forum Cult-front of Klaus Schwab, the United Nations,

Johnson & Johnson, major banks, and officials from China and the Centers for Disease Control in the United States. What synchronicity – Johns Hopkins would go on to compile the fraudulent ‘Covid’ figures, the World Economic Forum and Schwab would push the ‘Great Reset’ in response to ‘Covid’, the Centers for Disease Control would be at the forefront of ‘Covid’ policy in the United States, Johnson & Johnson would produce a ‘Covid vaccine’, and everything would officially start just weeks later in China. Spooky, eh? They were even accurate in creating a simulation of a ‘virus’ pandemic because the ‘real thing’ would also be a simulation. Event 201 was not an exercise preparing for something that might happen; it was a rehearsal for what those in control knew was *going* to happen and very shortly. Hours of this simulation were posted on the Internet and the various themes and responses mirrored what would soon be imposed to transform human society. News stories were inserted and what they said would be commonplace a few weeks later with still more prophecy perfection. Much discussion focused on the need to deal with misinformation and the ‘anti-vax movement’ which is exactly what happened when the ‘virus’ arrived – was said to have arrived – in the West.

Cult-owned social media banned criticism and exposure of the official ‘virus’ narrative and when I said there *was* no ‘virus’ in early April, 2020, I was banned by one platform after another including YouTube, Facebook and later Twitter. The mainstream broadcast media in Britain was in effect banned from interviewing me by the Tony-Blair-created government broadcasting censor Ofcom headed by career government bureaucrat Melanie Dawes who was appointed just as the ‘virus’ hoax was about to play out in January, 2020. At the same time the Ickonic media platform was using Vimeo, another ultra-Zionist-owned operation, while our own player was being created and they deleted in an instant hundreds of videos, documentaries, series and shows to confirm their unbelievable vindictiveness. We had copies, of course, and they had to be restored one by one when our player was ready. These people have no class. Sabbatian Facebook promised free advertisements for the Gates-

controlled World Health Organization narrative while deleting ‘false claims and conspiracy theories’ to stop ‘misinformation’ about the alleged coronavirus. All these responses could be seen just a short while earlier in the scenarios of Event 201. Extreme censorship was absolutely crucial for the Cult because the official story was so ridiculous and unsupportable by the evidence that it could never survive open debate and the free-flow of information and opinion. If you can’t win a debate then don’t have one is the Cult’s approach throughout history. Facebook’s little boy front man – front boy – Mark Zuckerberg equated ‘credible and accurate information’ with official sources and exposing their lies with ‘misinformation’.

Silencing those that can see

The censorship dynamic of Event 201 is now the norm with an army of narrative-supporting ‘fact-checker’ organisations whose entire reason for being is to tell the public that official narratives are true and those exposing them are lying. One of the most appalling of these ‘fact-checkers’ is called NewsGuard founded by ultra-Zionist Americans Gordon Crovitz and Steven Brill. Crovitz is a former publisher of *The Wall Street Journal*, former Executive Vice President of Dow Jones, a member of the Council on Foreign Relations (CFR), and on the board of the American Association of Rhodes Scholars. The CFR and Rhodes Scholarships, named after Rothschild agent Cecil Rhodes who plundered the gold and diamonds of South Africa for his masters and the Cult, have featured widely in my books. NewsGuard don’t seem to like me for some reason – I really can’t think why – and they have done all they can to have me censored and discredited which is, to quote an old British politician, like being savaged by a dead sheep. They are, however, like all in the censorship network, very well connected and funded by organisations themselves funded by, or connected to, Bill Gates. As you would expect with anything associated with Gates NewsGuard has an offshoot called HealthGuard which ‘fights online health care hoaxes’. How very kind. Somehow the NewsGuard European Managing Director Anna-Sophie Harling, a remarkably young-

looking woman with no broadcasting experience and little hands-on work in journalism, has somehow secured a position on the ‘Content Board’ of UK government broadcast censor Ofcom. An executive of an organisation seeking to discredit dissidents of the official narratives is making decisions for the government broadcast ‘regulator’ about content?? Another appalling ‘fact-checker’ is Full Fact funded by George Soros and global censors Google and Facebook.

It’s amazing how many activists in the ‘fact-checking’, ‘anti-hate’, arena turn up in government-related positions – people like UK Labour Party activist Imran Ahmed who heads the Center for Countering Digital Hate founded by people like Morgan McSweeney, now chief of staff to the Labour Party’s hapless and useless ‘leader’ Keir Starmer. Digital Hate – which is what it really is – uses the American spelling of Center to betray its connection to a transatlantic network of similar organisations which in 2020 shapeshifted from attacking people for ‘hate’ to attacking them for questioning the ‘Covid’ hoax and the dangers of the ‘Covid vaccine’. It’s just a coincidence, you understand. This is one of Imran Ahmed’s hysterical statements: ‘I would go beyond calling anti-vaxxers conspiracy theorists to say they are an extremist group that pose a national security risk.’ No one could ever accuse this prat of understatement and he’s including in that those parents who are now against vaccines after their children were damaged for life or killed by them. He’s such a nice man. Ahmed does the rounds of the Woke media getting soft-ball questions from spineless ‘journalists’ who never ask what right he has to campaign to destroy the freedom of speech of others while he demands it for himself. There also seems to be an overrepresentation in Ofcom of people connected to the narrative-worshipping BBC. This incredible global network of narrative-support was super-vital when the ‘Covid’ hoax was played in the light of the mega-whopper lies that have to be defended from the spotlight cast by the most basic intelligence.

Setting the scene

The Cult plays the long game and proceeds step-by-step ensuring that everything is in place before major cards are played and they don't come any bigger than the 'Covid' hoax. The psychopaths can't handle events where the outcome isn't certain and as little as possible – preferably nothing – is left to chance. Politicians, government and medical officials who would follow direction were brought to illusory power in advance by the Cult web whether on the national stage or others like state governors and mayors of America. For decades the dynamic between officialdom, law enforcement and the public was changed from one of service to one of control and dictatorship. Behaviour manipulation networks established within government were waiting to impose the coming 'Covid' rules and regulations specifically designed to subdue and rewire the psyche of the people in the guise of protecting health. These included in the UK the Behavioural Insights Team part-owned by the British government Cabinet Office; the Scientific Pandemic Insights Group on Behaviours (SPI-B); and a whole web of intelligence and military groups seeking to direct the conversation on social media and control the narrative. Among them are the cyberwarfare (on the people) 77th Brigade of the British military which is also coordinated through the Cabinet Office as civilian and military leadership continues to combine in what they call the Fusion Doctrine. The 77th Brigade is a British equivalent of the infamous Israeli (Sabbatian) military cyberwarfare and Internet manipulation operation Unit 8200 which I expose at length in *The Trigger*. Also carefully in place were the medical and science advisers to government – many on the payroll past or present of Bill Gates – and a whole alternative structure of unelected government stood by to take control when elected parliaments were effectively closed down once the 'Covid' card was slammed on the table. The structure I have described here and so much more was installed in every major country through the Cult networks. The top-down control hierarchy looks like this: The Cult – Cult-owned Gates – the World Health Organization and Tedros – Gates-funded or controlled chief medical officers and science 'advisers' (dictators) in each country –

political ‘leaders’ – law enforcement – The People. Through this simple global communication and enforcement structure the policy of the Cult could be imposed on virtually the entire human population so long as they acquiesced to the fascism. With everything in place it was time for the button to be pressed in late 2019/early 2020.

These were the prime goals the Cult had to secure for its will to prevail:

- 1) Locking down economies, closing all but designated ‘essential’ businesses (Cult-owned corporations were ‘essential’), and putting the population under house arrest was an imperative to destroy independent income and employment and ensure dependency on the Cult-controlled state in the Hunger Games Society. Lockdowns had to be established as the global blueprint from the start to respond to the ‘virus’ and followed by pretty much the entire world.
- 2) The global population had to be terrified into believing in a deadly ‘virus’ that didn’t actually exist so they would unquestioningly obey authority in the belief that authority must know how best to protect them and their families. Software salesman Gates would suddenly morph into the world’s health expert and be promoted as such by the Cult-owned media.
- 3) A method of testing that wasn’t testing for the ‘virus’, but was only claimed to be, had to be in place to provide the illusion of ‘cases’ and subsequent ‘deaths’ that had a very different cause to the ‘Covid-19’ that would be scribbled on the death certificate.
- 4) Because there was no ‘virus’ and the great majority testing positive with a test not testing for the ‘virus’ would have no symptoms of anything the lie had to be sold that people without symptoms (without the ‘virus’) could still pass it on to others. This was crucial to justify for the first time quarantining – house arresting – healthy people. Without this the economy-destroying lockdown of *everybody* could not have been credibly sold.
- 5) The ‘saviour’ had to be seen as a vaccine which beyond evil drug companies were working like angels of mercy to develop as quickly as possible, with all corners cut, to save the day. The public must absolutely not know that the ‘vaccine’ had nothing to do with a ‘virus’ or that the contents were ready and waiting with a very different motive long before the ‘Covid’ card was even lifted from the pack.

I said in March, 2020, that the ‘vaccine’ would have been created way ahead of the ‘Covid’ hoax which justified its use and the following December an article in the New York *Intelligencer* magazine said the Moderna ‘vaccine’ had been ‘designed’ by

January, 2020. This was ‘before China had even acknowledged that the disease could be transmitted from human to human, more than a week before the first confirmed coronavirus case in the United States’. The article said that by the time the first American death was announced a month later ‘the vaccine had already been manufactured and shipped to the National Institutes of Health for the beginning of its Phase I clinical trial’. The ‘vaccine’ was actually ‘designed’ long before that although even with this timescale you would expect the article to ask how on earth it could have been done that quickly. Instead it asked why the ‘vaccine’ had not been rolled out then and not months later. Journalism in the mainstream is truly dead. I am going to detail in the next chapter why the ‘virus’ has never existed and how a hoax on that scale was possible, but first the foundation on which the Big Lie of ‘Covid’ was built.

The test that doesn’t test

Fraudulent ‘testing’ is the bottom line of the whole ‘Covid’ hoax and was the means by which a ‘virus’ that did not exist *appeared* to exist. They could only achieve this magic trick by using a test not testing for the ‘virus’. To use a test that *was* testing for the ‘virus’ would mean that every test would come back negative given there was no ‘virus’. They chose to exploit something called the RT-PCR test invented by American biochemist Kary Mullis in the 1980s who said publicly that his PCR test … *cannot detect infectious disease*. Yes, the ‘test’ used worldwide to detect infectious ‘Covid’ to produce all the illusory ‘cases’ and ‘deaths’ compiled by Johns Hopkins and others *cannot detect infectious disease*. This fact came from the mouth of the man who invented PCR and was awarded the Nobel Prize in Chemistry in 1993 for doing so. Sadly, and incredibly conveniently for the Cult, Mullis died in August, 2019, at the age of 74 just before his test would be fraudulently used to unleash fascism on the world. He was said to have died from pneumonia which was an irony in itself. A few months later he would have had ‘Covid-19’ on his death certificate. I say the timing of his death was convenient because had he lived Mullis, a brilliant, honest and decent man, would have been

vociferously speaking out against the use of his test to detect 'Covid' when it was never designed, or able, to do that. I know that to be true given that Mullis made the same point when his test was used to 'detect' – not detect – HIV. He had been seriously critical of the Gallo/Montagnier claim to have isolated the HIV 'virus' and shown it to cause AIDS for which Mullis said there was no evidence. AIDS is actually not a disease but a series of diseases from which people die all the time. When they die from those *same diseases* after a positive 'test' for HIV then AIDS goes on their death certificate. I think I've heard that before somewhere. Countries instigated a policy with 'Covid' that anyone who tested positive with a test not testing for the 'virus' and died of any other cause within 28 days and even longer 'Covid-19' had to go on the death certificate. Cases have come from the test that can't test for infectious disease and the deaths are those who have died of *anything* after testing positive with a test not testing for the 'virus'. I'll have much more later about the death certificate scandal.

Mullis was deeply dismissive of the now US 'Covid' star Anthony Fauci who he said was a liar who didn't know anything about anything – 'and I would say that to his face – nothing.' He said of Fauci: 'The man thinks he can take a blood sample, put it in an electron microscope and if it's got a virus in there you'll know it – he doesn't understand electron microscopy and he doesn't understand medicine and shouldn't be in a position like he's in.' That position, terrifyingly, has made him the decider of 'Covid' fascism policy on behalf of the Cult in his role as director since 1984 of the National Institute of Allergy and Infectious Diseases (NIAID) while his record of being wrong is laughable; but being wrong, so long as it's the *right kind* of wrong, is why the Cult loves him. He'll say anything the Cult tells him to say. Fauci was made Chief Medical Adviser to the President immediately Biden took office. Biden was installed in the White House by Cult manipulation and one of his first decisions was to elevate Fauci to a position of even more control. This is a coincidence? Yes, and I identify as a flamenco dancer called Lola. How does such an incompetent criminal like Fauci remain in that

pivotal position in American health since *the 1980s*? When you serve the Cult it looks after you until you are surplus to requirements. Kary Mullis said prophetically of Fauci and his like: ‘Those guys have an agenda and it’s not an agenda we would like them to have ... they make their own rules, they change them when they want to, and Tony Fauci does not mind going on television in front of the people who pay his salary and lie directly into the camera.’ Fauci has done that almost daily since the ‘Covid’ hoax began. Lying is in Fauci’s DNA. To make the situation crystal clear about the PCR test this is a direct quote from its inventor Kary Mullis:

It [the PCR test] doesn’t tell you that you’re sick and doesn’t tell you that the thing you ended up with was really going to hurt you ...’

Ask yourself why governments and medical systems the world over have been using this very test to decide who is ‘infected’ with the SARS-CoV-2 ‘virus’ and the alleged disease it allegedly causes, ‘Covid-19’. The answer to that question will tell you what has been going on. By the way, here’s a little show-stopper – the ‘new’ SARS-CoV-2 ‘virus’ was ‘identified’ as such right from the start using ... *the PCR test not testing for the ‘virus’*. If you are new to this and find that shocking then stick around. I have hardly started yet. Even worse, other ‘tests’, like the ‘Lateral Flow Device’ (LFD), are considered so useless that they have to be *confirmed* by the PCR test! Leaked emails written by Ben Dyson, adviser to UK ‘Health’ Secretary Matt Hancock, said they were ‘dangerously unreliable’. Dyson, executive director of strategy at the Department of Health, wrote: ‘As of today, someone who gets a positive LFD result in (say) London has at best a 25 per cent chance of it being a true positive, but if it is a self-reported test potentially as low as 10 per cent (on an optimistic assumption about specificity) or as low as 2 per cent (on a more pessimistic assumption).’ These are the ‘tests’ that schoolchildren and the public are being urged to have twice a week or more and have to isolate if they get a positive. Each fake positive goes in the statistics as a ‘case’ no matter how ludicrously inaccurate and the

'cases' drive lockdown, masks and the pressure to 'vaccinate'. The government said in response to the email leak that the 'tests' were accurate which confirmed yet again what shocking bloody liars they are. The real false positive rate is *100 percent* as we'll see. In another 'you couldn't make it up' the UK government agreed to pay £2.8 billion to California's Innova Medical Group to supply the irrelevant lateral flow tests. The company's primary test-making centre is in China. Innova Medical Group, established in March, 2020, is owned by Pasaca Capital Inc, chaired by Chinese-American millionaire Charles Huang who was born in Wuhan.

How it works – and how it doesn't

The RT-PCR test, known by its full title of Polymerase chain reaction, is used across the world to make millions, even billions, of copies of a DNA/RNA genetic information sample. The process is called 'amplification' and means that a tiny sample of genetic material is amplified to bring out the detailed content. I stress that it is not testing for an infectious disease. It is simply amplifying a sample of genetic material. In the words of Kary Mullis: 'PCR is ... just a process that's used to make a whole lot of something out of something.' To emphasise the point companies that make the PCR tests circulated around the world to 'test' for 'Covid' warn on the box that it can't be used to detect 'Covid' or infectious disease and is for research purposes only. It's okay, rest for a minute and you'll be fine. This is the test that produces the 'cases' and 'deaths' that have been used to destroy human society. All those global and national medical and scientific 'experts' demanding this destruction to 'save us' KNOW that the test is not testing for the 'virus' and the cases and deaths they claim to be real are an almost unimaginable fraud. Every one of them and so many others including politicians and psychopaths like Gates and Tedros must be brought before Nuremberg-type trials and jailed for the rest of their lives. The more the genetic sample is amplified by PCR the more elements of that material become sensitive to the test and by that I don't mean sensitive for a 'virus' but for elements of the genetic material which

is naturally in the body or relates to remnants of old conditions of various kinds lying dormant and causing no disease. Once the amplification of the PCR reaches a certain level *everyone* will test positive. So much of the material has been made sensitive to the test that everyone will have some part of it in their body. Even lying criminals like Fauci have said that once PCR amplifications pass 35 cycles everything will be a false positive that cannot be trusted for the reasons I have described. I say, like many proper doctors and scientists, that 100 percent of the 'positives' are false, but let's just go with Fauci for a moment.

He says that any amplification over 35 cycles will produce false positives and yet the US Centers for Disease Control (CDC) and Food and Drug Administration (FDA) have recommended up to 40 cycles and the National Health Service (NHS) in Britain admitted in an internal document for staff that it was using 45 cycles of amplification. A long list of other countries has been doing the same and at least one 'testing' laboratory has been using 50 cycles. Have you ever heard a doctor, medical 'expert' or the media ask what level of amplification has been used to claim a 'positive'. The 'test' comes back 'positive' and so you have the 'virus', end of story. Now we can see how the government in Tanzania could send off samples from a goat and a pawpaw fruit under human names and both came back positive for 'Covid-19'. Tanzania president John Magufuli mocked the 'Covid' hysteria, the PCR test and masks and refused to import the DNA-manipulating 'vaccine'. The Cult hated him and an article sponsored by the Bill Gates Foundation appeared in the London *Guardian* in February, 2021, headed 'It's time for Africa to rein in Tanzania's anti-vaxxer president'. Well, 'reined in' he shortly was. Magufuli appeared in good health, but then, in March, 2021, he was dead at 61 from 'heart failure'. He was replaced by Samia Hassan Suhulu who is connected to Klaus Schwab's World Economic Forum and she immediately reversed Magufuli's 'Covid' policy. A sample of cola tested positive for 'Covid' with the PCR test in Germany while American actress and singer-songwriter Erykah Badu tested positive in one nostril and negative in the other. Footballer Ronaldo called

the PCR test ‘bullshit’ after testing positive three times and being forced to quarantine and miss matches when there was nothing wrong with him. The mantra from Tedros at the World Health Organization and national governments (same thing) has been test, test, test. They know that the more tests they can generate the more fake ‘cases’ they have which go on to become ‘deaths’ in ways I am coming to. The UK government has its Operation Moonshot planned to test multiple millions every day in workplaces and schools with free tests for everyone to use twice a week at home in line with the Cult plan from the start to make testing part of life. A government advertisement for an ‘Interim Head of Asymptomatic Testing Communication’ said the job included responsibility for delivering a ‘communications strategy’ (propaganda) ‘to support the expansion of asymptomatic testing that *“normalises testing as part of everyday life”*. More tests means more fake ‘cases’, ‘deaths’ and fascism. I have heard of, and from, many people who booked a test, couldn’t turn up, and yet got a positive result through the post for a test they’d never even had. The whole thing is crazy, but for the Cult there’s method in the madness. Controlling and manipulating the level of amplification of the test means the authorities can control whenever they want the number of apparent ‘cases’ and ‘deaths’. If they want to justify more fascist lockdown and destruction of livelihoods they keep the amplification high. If they want to give the illusion that lockdowns and the ‘vaccine’ are working then they lower the amplification and ‘cases’ and ‘deaths’ will appear to fall. In January, 2021, the Cult-owned World Health Organization suddenly warned laboratories about over-amplification of the test and to lower the threshold. Suddenly headlines began appearing such as: ‘Why ARE “Covid” cases plummeting?’ This was just when the vaccine rollout was underway and I had predicted months before they would make cases appear to fall through amplification tampering when the ‘vaccine’ came. These people are so predictable.

Cow vaccines?

The question must be asked of what is on the test swabs being poked far up the nose of the population to the base of the brain? A nasal swab punctured one woman's brain and caused it to leak fluid. Most of these procedures are being done by people with little training or medical knowledge. Dr Lorraine Day, former orthopaedic trauma surgeon and Chief of Orthopaedic Surgery at San Francisco General Hospital, says the tests are really a '*vaccine*'. Cows have long been vaccinated this way. She points out that masks have to cover the nose and the mouth where it is claimed the 'virus' exists in saliva. Why then don't they take saliva from the mouth as they do with a DNA test instead of pushing a long swab up the nose towards the brain? The ethmoid bone separates the nasal cavity from the brain and within that bone is the cribriform plate. Dr Day says that when the swab is pushed up against this plate and twisted the procedure is 'depositing things back there'. She claims that among these 'things' are nanoparticles that can enter the brain. Researchers have noted that a team at the Gates-funded Johns Hopkins have designed tiny, star-shaped micro-devices that can latch onto intestinal mucosa and release drugs into the body. Mucosa is the thin skin that covers the inside surface of parts of the body such as *the nose* and mouth and produces mucus to protect them. The Johns Hopkins micro-devices are called 'theragrippers' and were 'inspired' by a parasitic worm that digs its sharp teeth into a host's intestines. Nasal swabs are also coated in the sterilisation agent ethylene oxide. The US National Cancer Institute posts this explanation on its website:

At room temperature, ethylene oxide is a flammable colorless gas with a sweet odor. It is used primarily to produce other chemicals, including antifreeze. In smaller amounts, ethylene oxide is used as a pesticide and a sterilizing agent. The ability of ethylene oxide to damage DNA makes it an effective sterilizing agent but also accounts for its cancer-causing activity.

The Institute mentions lymphoma and leukaemia as cancers most frequently reported to be associated with occupational exposure to ethylene oxide along with stomach and breast cancers. How does anyone think this is going to work out with the constant testing

regime being inflicted on adults and children at home and at school that will accumulate in the body anything that's on the swab?

Doctors know best

It is vital for people to realise that 'hero' doctors 'know' only what the Big Pharma-dominated medical authorities tell them to 'know' and if they refuse to 'know' what they are told to 'know' they are out the door. They are mostly not physicians or healers, but repeaters of the official narrative – or else. I have seen alleged professional doctors on British television make shocking statements that we are supposed to take seriously. One called 'Dr' Amir Khan, who is actually telling patients how to respond to illness, said that men could take the birth pill to 'help slow down the effects of Covid-19'. In March, 2021, another ridiculous 'Covid study' by an American doctor proposed injecting men with the female sex hormone progesterone as a 'Covid' treatment. British doctor Nighat Arif told the BBC that face coverings were now going to be part of ongoing normal. Yes, the vaccine protects you, she said (evidence?) ... but the way to deal with viruses in the community was always going to come down to hand washing, face covering and keeping a physical distance. That's not what we were told before the 'vaccine' was circulating. Arif said she couldn't imagine ever again going on the underground or in a lift without a mask. I was just thanking my good luck that she was not my doctor when she said – in March, 2021 – that if 'we are *behaving* and we are doing all the right things' she thought we could 'have our nearest and dearest around us at home ... around *Christmas* and *New Year!*' Her patronising delivery was the usual school teacher talking to six-year-olds as she repeated every government talking point and probably believed them all. If we have learned anything from the 'Covid' experience surely it must be that humanity's perception of doctors needs a fundamental rethink. NHS 'doctor' Sara Kayat told her television audience that the 'Covid vaccine' would '100 percent prevent hospitalisation and death'. Not even Big Pharma claimed that. We have to stop taking 'experts' at their word without question when so many of them are

clueless and only repeating the party line on which their careers depend. That is not to say there are not brilliant doctors – there are and I have spoken to many of them since all this began – but you won't see them in the mainstream media or quoted by the psychopaths and yes-people in government.

Remember the name – Christian Drosten

German virologist Christian Drosten, Director of Charité Institute of Virology in Berlin, became a national star after the pandemic hoax began. He was feted on television and advised the German government on 'Covid' policy. Most importantly to the wider world Drosten led a group that produced the 'Covid' testing protocol for the PCR test. What a remarkable feat given the PCR cannot test for infectious disease and even more so when you think that Drosten said that his method of testing for SARS-CoV-2 was developed 'without having virus material available'. *He developed a test for a 'virus' that he didn't have and had never seen.* Let that sink in as you survey the global devastation that came from what he did. The whole catastrophe of Drosten's 'test' was based on the alleged genetic sequence published by Chinese scientists on the Internet. We will see in the next chapter that this alleged 'genetic sequence' has never been produced by China or anyone and cannot be when there is no SARS-CoV-2. Drosten, however, doesn't seem to let little details like that get in the way. He was the lead author with Victor Corman from the same Charité Hospital of the paper 'Detection of 2019 novel coronavirus (2019-nCoV) by real-time PCR' published in a magazine called *Eurosurveillance*. This became known as the Corman-Drosten paper. In November, 2020, with human society devastated by the effects of the Corman-Drosten test baloney, the protocol was publicly challenged by 22 international scientists and independent researchers from Europe, the United States, and Japan. Among them were senior molecular geneticists, biochemists, immunologists, and microbiologists. They produced a document headed 'External peer review of the RTPCR test to detect SARS-Cov-2 Reveals 10 Major Flaws At The Molecular and Methodological Level: Consequences

For False-Positive Results'. The flaws in the Corman-Drosten test included the following:

- The test is non-specific because of erroneous design
- Results are enormously variable
- The test is unable to discriminate between the whole 'virus' and viral fragments
- It doesn't have positive or negative controls
- The test lacks a standard operating procedure
- It is unsupported by proper peer view

The scientists said the PCR 'Covid' testing protocol was not founded on science and they demanded the Corman-Drosten paper be retracted by *Eurosurveillance*. They said all present and previous Covid deaths, cases, and 'infection rates' should be subject to a massive retroactive inquiry. Lockdowns and travel restrictions should be reviewed and relaxed and those diagnosed through PCR to have 'Covid-19' should not be forced to isolate. Dr Kevin Corbett, a health researcher and nurse educator with a long academic career producing a stream of peer-reviewed publications at many UK universities, made the same point about the PCR test debacle. He said of the scientists' conclusions: 'Every scientific rationale for the development of that test has been totally destroyed by this paper. It's like Hiroshima/Nagasaki to the Covid test.' He said that China hadn't given them an isolated 'virus' when Drosten developed the test. Instead they had developed the test from *a sequence in a gene bank.*' Put another way ... *they made it up!* The scientists were supported in this contention by a Portuguese appeals court which ruled in November, 2020, that PCR tests are unreliable and it is unlawful to quarantine people based solely on a PCR test. The point about China not providing an isolated virus must be true when the 'virus' has never been isolated to this day and the consequences of that will become clear. Drosten and company produced this useless 'protocol' right on cue in January, 2020, just as the 'virus' was said to

be moving westward and it somehow managed to successfully pass a peer-review in 24 hours. In other words there was no peer-review for a test that would be used to decide who had 'Covid' and who didn't across the world. The Cult-created, Gates-controlled World Health Organization immediately recommended all its nearly 200 member countries to use the Drosten PCR protocol to detect 'cases' and 'deaths'. The sting was underway and it continues to this day.

So who is this Christian Drosten that produced the means through which death, destruction and economic catastrophe would be justified? His education background, including his doctoral thesis, would appear to be somewhat shrouded in mystery and his track record is dire as with another essential player in the 'Covid' hoax, the Gates-funded Professor Neil Ferguson at the Gates-funded Imperial College in London of whom more shortly. Drosten predicted in 2003 that the alleged original SARS 'virus' (SARS-1') was an epidemic that could have serious effects on economies and an effective vaccine would take at least two years to produce. Drosten's answer to every alleged 'outbreak' is a vaccine which you won't be shocked to know. What followed were just 774 official deaths worldwide and none in Germany where there were only nine cases. That is even if you believe there ever was a SARS 'virus' when the evidence is zilch and I will expand on this in the next chapter. Drosten claims to be co-discoverer of 'SARS-1' and developed a test for it in 2003. He was screaming warnings about 'swine flu' in 2009 and how it was a widespread infection far more severe than any dangers from a vaccine could be and people should get vaccinated. It would be helpful for Drosten's vocal chords if he simply recorded the words 'the virus is deadly and you need to get vaccinated' and copies could be handed out whenever the latest made-up threat comes along. Drosten's swine flu epidemic never happened, but Big Pharma didn't mind with governments spending hundreds of millions on vaccines that hardly anyone bothered to use and many who did wished they hadn't. A study in 2010 revealed that the risk of dying from swine flu, or H1N1, was no higher than that of the annual seasonal flu which is what at least most of 'it' really was as in

the case of ‘Covid-19’. A media investigation into Drosten asked how with such a record of inaccuracy he could be *the* government adviser on these issues. The answer to that question is the same with Drosten, Ferguson and Fauci – they keep on giving the authorities the ‘conclusions’ and ‘advice’ they want to hear. Drosten certainly produced the goods for them in January, 2020, with his PCR protocol garbage and provided the foundation of what German internal medicine specialist Dr Claus Köhnlein, co-author of *Virus Mania*, called the ‘test pandemic’. The 22 scientists in the *Eurosurveillance* challenge called out conflicts of interest within the Drosten ‘protocol’ group and with good reason. Olfert Landt, a regular co-author of Drosten ‘studies’, owns the biotech company TIB Molbiol Syntheselabor GmbH in Berlin which manufactures and sells the tests that Drosten and his mates come up with. They have done this with SARS, Enterotoxigenic E. coli (ETEC), MERS, Zika ‘virus’, yellow fever, and now ‘Covid’. Landt told the *Berliner Zeitung* newspaper:

The testing, design and development came from the Charité [Drosten and Corman]. We simply implemented it immediately in the form of a kit. And if we don’t have the virus, which originally only existed in Wuhan, we can make a synthetic gene to simulate the genome of the virus. That’s what we did very quickly.

This is more confirmation that the Drosten test was designed without access to the ‘virus’ and only a synthetic simulation which is what SARS-CoV-2 really is – a computer-generated synthetic fiction. It’s quite an enterprise they have going here. A Drosten team decides what the test for something should be and Landt’s biotech company flogs it to governments and medical systems across the world. His company must have made an absolute fortune since the ‘Covid’ hoax began. Dr Reiner Fuellmich, a prominent German consumer protection trial lawyer in Germany and California, is on Drosten’s case and that of Tedros at the World Health Organization for crimes against humanity with a class-action lawsuit being prepared in the United States and other legal action in Germany.

Why China?

Scamming the world with a ‘virus’ that doesn’t exist would seem impossible on the face of it, but not if you have control of the relatively few people that make policy decisions and the great majority of the global media. Remember it’s not about changing ‘real’ reality it’s about controlling *perception* of reality. You don’t have to make something happen you only have to make people *believe* that it’s happening. Renegade Minds understand this and are therefore much harder to swindle. ‘Covid-19’ is not a ‘real’ ‘virus’. It’s a mind virus, like a computer virus, which has infected the minds, not the bodies, of billions. It all started, publically at least, in China and that alone is of central significance. The Cult was behind the revolution led by its asset Mao Zedong, or Chairman Mao, which established the People’s Republic of China on October 1st, 1949. It should have been called The Cult’s Republic of China, but the name had to reflect the recurring illusion that vicious dictatorships are run by and for the people (see all the ‘Democratic Republics’ controlled by tyrants). In the same way we have the ‘Biden’ Democratic Republic of America officially ruled by a puppet tyrant (at least temporarily) on behalf of Cult tyrants. The creation of Mao’s merciless communist/fascist dictatorship was part of a frenzy of activity by the Cult at the conclusion of World War Two which, like the First World War, it had instigated through its assets in Germany, Britain, France, the United States and elsewhere. Israel was formed in 1948; the Soviet Union expanded its ‘Iron Curtain’ control, influence and military power with the Warsaw Pact communist alliance in 1955; the United Nations was formed in 1945 as a Cult precursor to world government; and a long list of world bodies would be established including the World Health Organization (1948), World Trade Organization (1948 under another name until 1995), International Monetary Fund (1945) and World Bank (1944). Human society was redrawn and hugely centralised in the global Problem-Reaction-Solution that was World War Two. All these changes were significant. Israel would become the headquarters of the Sabbatians

and the revolution in China would prepare the ground and control system for the events of 2019/2020.

Renegade Minds know there are no borders except for public consumption. The Cult is a seamless, borderless global entity and to understand the game we need to put aside labels like borders, nations, countries, communism, fascism and democracy. These delude the population into believing that countries are ruled within their borders by a government of whatever shade when these are mere agencies of a global power. America's illusion of democracy and China's communism/fascism are subsidiaries – vehicles – for the same agenda. We may hear about conflict and competition between America and China and on the lower levels that will be true; but at the Cult level they are branches of the same company in the way of the McDonald's example I gave earlier. I have tracked in the books over the years support by US governments of both parties for Chinese Communist Party infiltration of American society through allowing the sale of land, even military facilities, and the acquisition of American business and university influence. All this is underpinned by the infamous stealing of intellectual property and technological know-how. Cult-owned Silicon Valley corporations waive their fraudulent 'morality' to do business with human-rights-free China; Cult-controlled Disney has become China's PR department; and China in effect owns 'American' sports such as basketball which depends for much of its income on Chinese audiences. As a result any sports player, coach or official speaking out against China's horrific human rights record is immediately condemned or fired by the China-worshipping National Basketball Association. One of the first acts of China-controlled Biden was to issue an executive order telling federal agencies to stop making references to the 'virus' by the 'geographic location of its origin'. Long-time Congressman Jerry Nadler warned that criticising China, America's biggest rival, leads to hate crimes against Asian people in the United States. So shut up you bigot. China is fast closing in on Israel as a country that must not be criticised which is apt, really, given that Sabbatians control them both. The two countries have

developed close economic, military, technological and strategic ties which include involvement in China's 'Silk Road' transport and economic initiative to connect China with Europe. Israel was the first country in the Middle East to recognise the establishment of Mao's tyranny in 1950 months after it was established.

Project Wuhan – the 'Covid' Psyop

I emphasise again that the Cult plays the long game and what is happening to the world today is the result of centuries of calculated manipulation following a script to take control step-by-step of every aspect of human society. I will discuss later the common force behind all this that has spanned those centuries and thousands of years if the truth be told. Instigating the Mao revolution in China in 1949 with a 2020 'pandemic' in mind is not only how they work – the 71 years between them is really quite short by the Cult's standards of manipulation preparation. The reason for the Cult's Chinese revolution was to create a fiercely-controlled environment within which an extreme structure for human control could be incubated to eventually be unleashed across the world. We have seen this happen since the 'pandemic' emerged from China with the Chinese control-structure founded on AI technology and tyrannical enforcement sweep across the West. Until the moment when the Cult went for broke in the West and put its fascism on public display Western governments had to pay some lip-service to freedom and democracy to not alert too many people to the tyranny-in-the-making. Freedoms were more subtly eroded and power centralised with covert government structures put in place waiting for the arrival of 2020 when that smokescreen of 'freedom' could be dispensed with. The West was not able to move towards tyranny before 2020 anything like as fast as China which was created as a tyranny and had no limits on how fast it could construct the Cult's blueprint for global control. When the time came to impose that structure on the world it was the same Cult-owned Chinese communist/fascist government that provided the excuse – the 'Covid pandemic'. It was absolutely crucial to the Cult plan for the Chinese response to the 'pandemic' –

draconian lockdowns of the entire population – to become the blueprint that Western countries would follow to destroy the livelihoods and freedom of their people. This is why the Cult-owned, Gates-owned, WHO Director-General Tedros said early on:

The Chinese government is to be congratulated for the extraordinary measures it has taken to contain the outbreak. China is actually setting a new standard for outbreak response and it is not an exaggeration.

Forbes magazine said of China: ‘... those measures protected untold millions from getting the disease’. The Rockefeller Foundation ‘epidemic scenario’ document in 2010 said ‘prophetically’:

However, a few countries did fare better – China in particular. The Chinese government’s quick imposition and enforcement of mandatory quarantine for all citizens, as well as its instant and near-hermetic sealing off of all borders, saved millions of lives, stopping the spread of the virus far earlier than in other countries and enabling a swifter post-pandemic recovery.

Once again – *spooky*.

The first official story was the ‘bat theory’ or rather the bat diversion. The source of the ‘virus outbreak’ we were told was a “wet market” in Wuhan where bats and other animals are bought and eaten in horrifically unhygienic conditions. Then another story emerged through the alternative media that the ‘virus’ had been released on purpose or by accident from a BSL-4 (biosafety level 4) laboratory in Wuhan not far from the wet market. The lab was reported to create and work with lethal concoctions and bioweapons. Biosafety level 4 is the highest in the World Health Organization system of safety and containment. Renegade Minds are aware of what I call designer manipulation. The ideal for the Cult is for people to buy its prime narrative which in the opening salvos of the ‘pandemic’ was the wet market story. It knows, however, that there is now a considerable worldwide alternative media of researchers sceptical of anything governments say and they are often given a version of events in a form they can perceive as credible while misdirecting them from the real truth. In this case let them

think that the conspiracy involved is a ‘bioweapon virus’ released from the Wuhan lab to keep them from the real conspiracy – *there is no ‘virus’*. The WHO’s current position on the source of the outbreak at the time of writing appears to be: ‘We haven’t got a clue, mate.’ This is a good position to maintain mystery and bewilderment. The inner circle will know where the ‘virus’ came from – *nowhere*. The bottom line was to ensure the public believed there *was* a ‘virus’ and it didn’t much matter if they thought it was natural or had been released from a lab. The belief that there was a ‘deadly virus’ was all that was needed to trigger global panic and fear. The population was terrified into handing their power to authority and doing what they were told. They had to or they were ‘all gonna die’.

In March, 2020, information began to come my way from real doctors and scientists and my own additional research which had my intuition screaming: ‘Yes, that’s it! *There is no virus.*’ The ‘bioweapon’ was not the ‘virus’; it was the ‘vaccine’ already being talked about that would be the bioweapon. My conclusion was further enhanced by happenings in Wuhan. The ‘virus’ was said to be sweeping the city and news footage circulated of people collapsing in the street (which they’ve never done in the West with the same ‘virus’). The Chinese government was building ‘new hospitals’ in a matter of ten days to ‘cope with demand’ such was the virulent nature of the ‘virus’. Yet in what seemed like no time the ‘new hospitals’ closed – even if they even opened – and China declared itself ‘virus-free’. It was back to business as usual. This was more propaganda to promote the Chinese draconian lockdowns in the West as the way to ‘beat the virus’. Trouble was that we subsequently had lockdown after lockdown, but never business as usual. As the people of the West and most of the rest of the world were caught in an ever-worsening spiral of lockdown, social distancing, masks, isolated old people, families forced apart, and livelihood destruction, it was party-time in Wuhan. Pictures emerged of thousands of people enjoying pool parties and concerts. It made no sense until you realised there never was a ‘virus’ and the

whole thing was a Cult set-up to transform human society out of one its major global strongholds – China.

How is it possible to deceive virtually the entire world population into believing there is a deadly virus when there is not even a ‘virus’ let alone a deadly one? It’s nothing like as difficult as you would think and that’s clearly true because it happened.

Postscript: See end of book Postscript for more on the ‘Wuhan lab virus release’ story which the authorities and media were pushing heavily in the summer of 2021 to divert attention from the truth that the ‘Covid virus’ is pure invention.

CHAPTER FIVE

There is no ‘virus’

You can fool some of the people all of the time, and all of the people some of the time, but you cannot fool all of the people all of the time

Abraham Lincoln

The greatest form of mind control is repetition. The more you repeat the same mantra of alleged ‘facts’ the more will accept them to be true. It becomes an ‘everyone knows that, mate’. If you can also censor any other version or alternative to your alleged ‘facts’ you are pretty much home and cooking.

By the start of 2020 the Cult owned the global mainstream media almost in its entirety to spew out its ‘Covid’ propaganda and ignore or discredit any other information and view. Cult-owned social media platforms in Cult-owned Silicon Valley were poised and ready to unleash a campaign of ferocious censorship to obliterate all but the official narrative. To complete the circle many demands for censorship by Silicon Valley were led by the mainstream media as ‘journalists’ became full-out enforcers for the Cult both as propagandists and censors. Part of this has been the influx of young people straight out of university who have become ‘journalists’ in significant positions. They have no experience and a headful of programmed perceptions from their years at school and university at a time when today’s young are the most perceptually-targeted generations in known human history given the insidious impact of technology. They enter the media perceptually prepared and ready to repeat the narratives of the system that programmed them to

repeat its narratives. The BBC has a truly pathetic ‘specialist disinformation reporter’ called Marianna Spring who fits this bill perfectly. She is clueless about the world, how it works and what is really going on. Her role is to discredit anyone doing the job that a proper journalist would do and system-serving hacks like Spring wouldn’t dare to do or even see the need to do. They are too busy licking the arse of authority which can never be wrong and, in the case of the BBC propaganda programme, *Panorama*, contacting payments systems such as PayPal to have a donations page taken down for a film company making documentaries questioning vaccines. Even the BBC soap opera *EastEnders* included a disgracefully biased scene in which an inarticulate white working class woman was made to look foolish for questioning the ‘vaccine’ while a well-spoken black man and Asian woman promoted the government narrative. It ticked every BBC box and the fact that the black and minority community was resisting the ‘vaccine’ had nothing to do with the way the scene was written. The BBC has become a disgusting tyrannical propaganda and censorship operation that should be defunded and disbanded and a free media take its place with a brief to stop censorship instead of demanding it. A BBC ‘interview’ with Gates goes something like: ‘Mr Gates, sir, if I can call you sir, would you like to tell our audience why you are such a great man, a wonderful humanitarian philanthropist, and why you should absolutely be allowed as a software salesman to decide health policy for approaching eight billion people? Thank you, sir, please sir.’ Propaganda programming has been incessant and merciless and when all you hear is the same story from the media, repeated by those around you who have only heard the same story, is it any wonder that people on a grand scale believe absolute mendacious garbage to be true? You are about to see, too, why this level of information control is necessary when the official ‘Covid’ narrative is so nonsensical and unsupportable by the evidence.

Structure of Deceit

The pyramid structure through which the ‘Covid’ hoax has been manifested is very simple and has to be to work. As few people as possible have to be involved with full knowledge of what they are doing – and why – or the real story would get out. At the top of the pyramid are the inner core of the Cult which controls Bill Gates who, in turn, controls the World Health Organization through his pivotal funding and his puppet Director-General mouthpiece, Tedros.

Before he was appointed Tedros was chair of the Gates-founded Global Fund to ‘fight against AIDS, tuberculosis and malaria’, a board member of the Gates-funded ‘vaccine alliance’ GAVI, and on the board of another Gates-funded organisation. Gates owns him and picked him for a specific reason – Tedros is a crook and worse. ‘Dr’ Tedros (he’s not a medical doctor, the first WHO chief not to be) was a member of the tyrannical Marxist government of Ethiopia for decades with all its human rights abuses. He has faced allegations of corruption and misappropriation of funds and was exposed three times for covering up cholera epidemics while Ethiopia’s health minister. Tedros appointed the mass-murdering genocidal Zimbabwe dictator Robert Mugabe as a WHO goodwill ambassador for public health which, as with Tedros, is like appointing a psychopath to run a peace and love campaign. The move was so ridiculous that he had to drop Mugabe in the face of widespread condemnation. American economist David Steinman, a Nobel peace prize nominee, lodged a complaint with the International Criminal Court in The Hague over alleged genocide by Tedros when he was Ethiopia’s foreign minister. Steinman says Tedros was a ‘crucial decision maker’ who directed the actions of Ethiopia’s security forces from 2013 to 2015 and one of three officials in charge when those security services embarked on the ‘killing’ and ‘torturing’ of Ethiopians. You can see where Tedros is coming from and it’s sobering to think that he has been the vehicle for Gates and the Cult to direct the global response to ‘Covid’. Think about that. A psychopathic Cult dictates to psychopath Gates who dictates to psychopath Tedros who dictates how countries of the world must respond to a ‘Covid virus’ never scientifically shown to exist. At the same time psychopathic Cult-owned Silicon Valley information

giants like Google, YouTube, Facebook and Twitter announced very early on that they would give the Cult/Gates/Tedros/WHO version of the narrative free advertising and censor those who challenged their intelligence-insulting, mendacious story.

The next layer in the global ‘medical’ structure below the Cult, Gates and Tedros are the chief medical officers and science ‘advisers’ in each of the WHO member countries which means virtually all of them. Medical officers and arbiters of science (they’re not) then take the WHO policy and recommended responses and impose them on their country’s population while the political ‘leaders’ say they are deciding policy (they’re clearly not) by ‘following the science’ on the advice of the ‘experts’ – the same medical officers and science ‘advisers’ (dictators). In this way with the rarest of exceptions the entire world followed the same policy of lockdown, people distancing, masks and ‘vaccines’ dictated by the psychopathic Cult, psychopathic Gates and psychopathic Tedros who we are supposed to believe give a damn about the health of the world population they are seeking to enslave. That, amazingly, is all there is to it in terms of crucial decision-making. Medical staff in each country then follow like sheep the dictates of the shepherds at the top of the national medical hierarchies – chief medical officers and science ‘advisers’ who themselves follow like sheep the shepherds of the World Health Organization and the Cult. Shepherds at the national level often have major funding and other connections to Gates and his Bill and Melinda Gates Foundation which carefully hands out money like confetti at a wedding to control the entire global medical system from the WHO down.

Follow the money

Christopher Whitty, Chief Medical Adviser to the UK Government at the centre of ‘virus’ policy, a senior adviser to the government’s Scientific Advisory Group for Emergencies (SAGE), and Executive Board member of the World Health Organization, was gifted a grant of \$40 million by the Bill and Melinda Gates Foundation for malaria research in Africa. The BBC described the unelected Whitty as ‘the

official who will probably have the greatest impact on our everyday lives of any individual policymaker in modern times' and so it turned out. What Gates and Tedros have said Whitty has done like his equivalents around the world. Patrick Vallance, co-chair of SAGE and the government's Chief Scientific Adviser, is a former executive of Big Pharma giant GlaxoSmithKline with its fundamental financial and business connections to Bill Gates. In September, 2020, it was revealed that Vallance owned a deferred bonus of shares in GlaxoSmithKline worth £600,000 while the company was 'developing' a 'Covid vaccine'. Move along now – nothing to see here – what could possibly be wrong with that? Imperial College in London, a major player in 'Covid' policy in Britain and elsewhere with its 'Covid-19' Response Team, is funded by Gates and has big connections to China while the now infamous Professor Neil Ferguson, the useless 'computer modeller' at Imperial College is also funded by Gates. Ferguson delivered the dramatically inaccurate excuse for the first lockdowns (much more in the next chapter). The Institute for Health Metrics and Evaluation (IHME) in the United States, another source of outrageously false 'Covid' computer models to justify lockdowns, is bankrolled by Gates who is a vehement promotor of lockdowns. America's version of Whitty and Vallance, the again now infamous Anthony Fauci, has connections to 'Covid vaccine' maker Moderna as does Bill Gates through funding from the Bill and Melinda Gates Foundation. Fauci is director of the National Institute of Allergy and Infectious Diseases (NIAID), a major recipient of Gates money, and they are very close. Deborah Birx who was appointed White House Coronavirus Response Coordinator in February, 2020, is yet another with ties to Gates. Everywhere you look at the different elements around the world behind the coordination and decision making of the 'Covid' hoax there is Bill Gates and his money. They include the World Health Organization; Centers for Disease Control (CDC) in the United States; National Institutes of Health (NIH) of Anthony Fauci; Imperial College and Neil Ferguson; the London School of Hygiene where Chris Whitty worked; Regulatory agencies like the UK Medicines & Healthcare products Regulatory Agency (MHRA)

which gave emergency approval for ‘Covid vaccines’; Wellcome Trust; GAVI, the Vaccine Alliance; the Coalition for Epidemic Preparedness Innovations (CEPI); Johns Hopkins University which has compiled the false ‘Covid’ figures; and the World Economic Forum. A [Nationalfile.com](#) article said:

Gates has a lot of pull in the medical world, he has a multi-million dollar relationship with Dr. Fauci, and Fauci originally took the Gates line supporting vaccines and casting doubt on [the drug hydroxychloroquine]. Coronavirus response team member Dr. Deborah Birx, appointed by former president Obama to serve as United States Global AIDS Coordinator, also sits on the board of a group that has received billions from Gates’ foundation, and Birx reportedly used a disputed Bill Gates-funded model for the White House’s Coronavirus effort. Gates is a big proponent for a population lockdown scenario for the Coronavirus outbreak.

Another funder of Moderna is the Defense Advanced Research Projects Agency (DARPA), the technology-development arm of the Pentagon and one of the most sinister organisations on earth. DARPA had a major role with the CIA covert technology-funding operation In-Q-Tel in the development of Google and social media which is now at the centre of global censorship. Fauci and Gates are extremely close and openly admit to talking regularly about ‘Covid’ policy, but then why wouldn’t Gates have a seat at every national ‘Covid’ table after his Foundation committed \$1.75 billion to the ‘fight against Covid-19’. When passed through our Orwellian Translation Unit this means that he has bought and paid for the Cult-driven ‘Covid’ response worldwide. Research the major ‘Covid’ response personnel in your own country and you will find the same Gates funding and other connections again and again. Medical and science chiefs following World Health Organization ‘policy’ sit atop a medical hierarchy in their country of administrators, doctors and nursing staff. These ‘subordinates’ are told they must work and behave in accordance with the policy delivered from the ‘top’ of the national ‘health’ pyramid which is largely the policy delivered by the WHO which is the policy delivered by Gates and the Cult. The whole ‘Covid’ narrative has been imposed on medical staff by a climate of fear although great numbers don’t even need that to comply. They do so through breathtaking levels of ignorance and

include doctors who go through life simply repeating what Big Pharma and their hierarchical masters tell them to say and believe. No wonder Big Pharma ‘medicine’ is one of the biggest killers on Planet Earth.

The same top-down system of intimidation operates with regard to the Cult Big Pharma cartel which also dictates policy through national and global medical systems in this way. The Cult and Big Pharma agendas are the same because the former controls and owns the latter. ‘Health’ administrators, doctors, and nursing staff are told to support and parrot the dictated policy or they will face consequences which can include being fired. How sad it’s been to see medical staff meekly repeating and imposing Cult policy without question and most of those who can see through the deceit are only willing to speak anonymously off the record. They know what will happen if their identity is known. This has left the courageous few to expose the lies about the ‘virus’, face masks, overwhelmed hospitals that aren’t, and the dangers of the ‘vaccine’ that isn’t a vaccine. When these medical professionals and scientists, some renowned in their field, have taken to the Internet to expose the truth their articles, comments and videos have been deleted by Cult-owned Facebook, Twitter and YouTube. What a real head-shaker to see YouTube videos with leading world scientists and highly qualified medical specialists with an added link underneath to the notorious Cult propaganda website *Wikipedia* to find the ‘facts’ about the same subject.

HIV – the ‘Covid’ trial-run

I’ll give you an example of the consequences for health and truth that come from censorship and unquestioning belief in official narratives. The story was told by PCR inventor Kary Mullis in his book *Dancing Naked in the Mind Field*. He said that in 1984 he accepted as just another scientific fact that Luc Montagnier of France’s Pasteur Institute and Robert Gallo of America’s National Institutes of Health had independently discovered that a ‘retrovirus’ dubbed HIV (human immunodeficiency virus) caused AIDS. They

were, after all, Mullis writes, specialists in retroviruses. This is how the medical and science pyramids work. Something is announced or *assumed* and then becomes an everybody-knows-that purely through repetition of the assumption as if it is fact. Complete crap becomes accepted truth with no supporting evidence and only repetition of the crap. This is how a 'virus' that doesn't exist became the 'virus' that changed the world. The HIV-AIDS fairy story became a multi-billion pound industry and the media poured out propaganda terrifying the world about the deadly HIV 'virus' that caused the lethal AIDS. By then Mullis was working at a lab in Santa Monica, California, to detect retroviruses with his PCR test in blood donations received by the Red Cross. In doing so he asked a virologist where he could find a reference for HIV being the cause of AIDS. 'You don't need a reference,' the virologist said ... '*Everybody knows it.*' Mullis said he wanted to quote a reference in the report he was doing and he said he felt a little funny about not knowing the source of such an important discovery when everyone else seemed to. The virologist suggested he cite a report by the Centers for Disease Control and Prevention (CDC) on morbidity and mortality. Mullis read the report, but it only said that an organism had been identified and did not say how. The report did not identify the original scientific work. Physicians, however, *assumed* (key recurring theme) that if the CDC was convinced that HIV caused AIDS then proof must exist. Mullis continues:

I did computer searches. Neither Montagnier, Gallo, nor anyone else had published papers describing experiments which led to the conclusion that HIV probably caused AIDS. I read the papers in Science for which they had become well known as AIDS doctors, but all they had said there was that they had found evidence of a past infection by something which was probably HIV in some AIDS patients.

They found antibodies. Antibodies to viruses had always been considered evidence of past disease, not present disease. Antibodies signaled that the virus had been defeated. The patient had saved himself. There was no indication in these papers that this virus caused a disease. They didn't show that everybody with the antibodies had the disease. In fact they found some healthy people with antibodies.

Mullis asked why their work had been published if Montagnier and Gallo hadn't really found this evidence, and why had they been fighting so hard to get credit for the discovery? He says he was hesitant to write 'HIV is the probable cause of AIDS' until he found published evidence to support that. 'Tens of thousands of scientists and researchers were spending billions of dollars a year doing research based on this idea,' Mullis writes. 'The reason had to be there somewhere; otherwise these people would not have allowed their research to settle into one narrow channel of investigation.' He said he lectured about PCR at numerous meetings where people were always talking about HIV and he asked them how they knew that HIV was the cause of AIDS:

Everyone said something. Everyone had the answer at home, in the office, in some drawer. They all knew, and they would send me the papers as soon as they got back. But I never got any papers. Nobody ever sent me the news about how AIDS was caused by HIV.

Eventually Mullis was able to ask Montagnier himself about the reference proof when he lectured in San Diego at the grand opening of the University of California AIDS Research Center. Mullis says this was the last time he would ask his question without showing anger. Montagnier said he should reference the CDC report. 'I read it', Mullis said, and it didn't answer the question. 'If Montagnier didn't know the answer who the hell did?' Then one night Mullis was driving when an interview came on National Public Radio with Peter Duesberg, a prominent virologist at Berkeley and a California Scientist of the Year. Mullis says he finally understood why he could not find references that connected HIV to AIDS – *there weren't any!* No one had ever proved that HIV causes AIDS even though it had spawned a multi-billion pound global industry and the media was repeating this as fact every day in their articles and broadcasts terrifying the shit out of people about AIDS and giving the impression that a positive test for HIV (see 'Covid') was a death sentence. Duesberg was a threat to the AIDS gravy train and the agenda that underpinned it. He was therefore abused and castigated after he told the Proceedings of the National Academy of Sciences

there was no good evidence implicating the new ‘virus’. Editors rejected his manuscripts and his research funds were deleted. Mullis points out that the CDC has defined AIDS as one of more than 30 diseases *if accompanied* by a positive result on a test that detects antibodies to HIV; but those same diseases are not defined as AIDS cases when antibodies are not detected:

If an HIV-positive woman develops uterine cancer, for example, she is considered to have AIDS. If she is not HIV positive, she simply has uterine cancer. An HIV-positive man with tuberculosis has AIDS; if he tests negative he simply has tuberculosis. If he lives in Kenya or Colombia, where the test for HIV antibodies is too expensive, he is simply presumed to have the antibodies and therefore AIDS, and therefore he can be treated in the World Health Organization’s clinic. It’s the only medical help available in some places. And it’s free, because the countries that support WHO are worried about AIDS.

Mullis accuses the CDC of continually adding new diseases (see ever more ‘Covid symptoms’) to the grand AIDS definition and of virtually doctoring the books to make it appear as if the disease continued to spread. He cites how in 1993 the CDC enormously broadened its AIDS definition and county health authorities were delighted because they received \$2,500 per year from the Federal government for every reported AIDS case. Ladies and gentlemen, I have just described, via Kary Mullis, the ‘Covid pandemic’ of 2020 and beyond. Every element is the same and it’s been pulled off in the same way by the same networks.

The ‘Covid virus’ exists? Okay – prove it. Er ... still waiting

What Kary Mullis described with regard to ‘HIV’ has been repeated with ‘Covid’. A claim is made that a new, or ‘novel’, infection has been found and the entire medical system of the world repeats that as fact exactly as they did with HIV and AIDS. No one in the mainstream asks rather relevant questions such as ‘How do you know?’ and ‘Where is your proof?’ The SARS-CoV-2 ‘virus’ and the ‘Covid-19 disease’ became an overnight ‘everybody-knows-that’. The origin could be debated and mulled over, but what you could not suggest was that ‘SARS-CoV-2’ didn’t exist. That would be

ridiculous. ‘Everybody knows’ the ‘virus’ exists. Well, I didn’t for one along with American proper doctors like Andrew Kaufman and Tom Cowan and long-time American proper journalist Jon Rappaport. We dared to pursue the obvious and simple question: ‘Where’s the evidence?’ The overwhelming majority in medicine, journalism and the general public did not think to ask that. After all, *everyone knew* there was a new ‘virus’. Everyone was saying so and I heard it on the BBC. Some would eventually argue that the ‘deadly virus’ was nothing like as deadly as claimed, but few would venture into the realms of its very existence. Had they done so they would have found that the evidence for that claim had gone AWOL as with HIV causes AIDS. In fact, not even that. For something to go AWOL it has to exist in the first place and scientific proof for a ‘SARS-Cov-2’ can be filed under nothing, nowhere and zilch.

Dr Andrew Kaufman is a board-certified forensic psychiatrist in New York State, a Doctor of Medicine and former Assistant Professor and Medical Director of Psychiatry at SUNY Upstate Medical University, and Medical Instructor of Hematology and Oncology at the Medical School of South Carolina. He also studied biology at the Massachusetts Institute of Technology (MIT) and trained in Psychiatry at Duke University. Kaufman is retired from allopathic medicine, but remains a consultant and educator on natural healing, I saw a video of his very early on in the ‘Covid’ hoax in which he questioned claims about the ‘virus’ in the absence of any supporting evidence and with plenty pointing the other way. I did everything I could to circulate his work which I felt was asking the pivotal questions that needed an answer. I can recommend an excellent pull-together interview he did with the website The Last Vagabond entitled *Dr Andrew Kaufman: Virus Isolation, Terrain Theory and Covid-19* and his website is andrewkaufmanmd.com. Kaufman is not only a forensic psychiatrist; he is forensic in all that he does. He always reads original scientific papers, experiments and studies instead of second-third-fourth-hand reports about the ‘virus’ in the media which are repeating the repeated repetition of the narrative. When he did so with the original Chinese ‘virus’ papers Kaufman

realised that there was no evidence of a ‘SARS-Cov-2’. They had never – from the start – shown it to exist and every repeat of this claim worldwide was based on the accepted existence of proof that was nowhere to be found – see Kary Mullis and HIV. Here we go again.

Let's postulate

Kaufman discovered that the Chinese authorities immediately concluded that the cause of an illness that broke out among about 200 initial patients in Wuhan was a ‘new virus’ when there were no grounds to make that conclusion. The alleged ‘virus’ was not isolated from other genetic material in their samples and then shown through a system known as Koch’s postulates to be the causative agent of the illness. The world was told that the SARS-Cov-2 ‘virus’ caused a disease they called ‘Covid-19’ which had ‘flu-like’ symptoms and could lead to respiratory problems and pneumonia. If it wasn’t so tragic it would almost be funny. *‘Flu-like’ symptoms?* *Pneumonia? Respiratory disease?* What in CHINA and particularly in Wuhan, one of the most polluted cities in the world with a resulting epidemic of respiratory disease?? Three hundred thousand people get pneumonia in China every year and there are nearly a billion cases worldwide of ‘flu-like symptoms’. These have a whole range of causes – including pollution in Wuhan – but no other possibility was credibly considered in late 2019 when the world was told there was a new and deadly ‘virus’. The global prevalence of pneumonia and ‘flu-like systems’ gave the Cult networks unlimited potential to re-diagnose these other causes as the mythical ‘Covid-19’ and that is what they did from the very start. Kaufman revealed how Chinese medical and science authorities (all subordinates to the Cult-owned communist government) took genetic material from the lungs of only a few of the first patients. The material contained their own cells, bacteria, fungi and other microorganisms living in their bodies. The only way you could prove the existence of the ‘virus’ and its responsibility for the alleged ‘Covid-19’ was to isolate the virus from all the other material – a process also known as ‘purification’ – and

then follow the postulates sequence developed in the late 19th century by German physician and bacteriologist Robert Koch which became the ‘gold standard’ for connecting an alleged causation agent to a disease:

1. The microorganism (bacteria, fungus, virus, etc.) must be present in every case of the disease and all patients must have the same symptoms. It must also *not be present in healthy individuals*.
2. The microorganism must be isolated from the host with the disease. If the microorganism is a bacteria or fungus it must be grown in a pure culture. If it is a virus, it must be purified (i.e. containing no other material except the virus particles) from a clinical sample.
3. The specific disease, with all of its characteristics, must be reproduced when the infectious agent (the purified virus or a pure culture of bacteria or fungi) is inoculated into a healthy, susceptible host.
4. The microorganism must be recoverable from the experimentally infected host as in step 2.

Not one of these criteria has been met in the case of ‘SARS-Cov-2’ and ‘Covid-19’. Not ONE. EVER. Robert Koch refers to bacteria and not viruses. What are called ‘viral particles’ are so minute (hence masks are useless by any definition) that they could only be seen after the invention of the electron microscope in the 1930s and can still only be observed through that means. American bacteriologist and virologist Thomas Milton Rivers, the so-called ‘Father of Modern Virology’ who was very significantly director of the Rockefeller Institute for Medical Research in the 1930s, developed a less stringent version of Koch’s postulates to identify ‘virus’ causation known as ‘Rivers criteria’. ‘Covid’ did not pass that process either. Some even doubt whether any ‘virus’ can be isolated from other particles containing genetic material in the Koch method. Freedom of Information requests in many countries asking for scientific proof that the ‘Covid virus’ has been purified and isolated and shown to exist have all come back with a ‘we don’t have that’ and when this happened with a request to the UK Department of Health they added this comment:

However, outside of the scope of the [Freedom of Information Act] and on a discretionary basis, the following information has been advised to us, which may be of interest. Most infectious diseases are caused by viruses, bacteria or fungi. Some bacteria or fungi have the capacity to grow on their own in isolation, for example in colonies on a petri dish. Viruses are different in that they are what we call ‘obligate pathogens’ – that is, they cannot survive or reproduce without infecting a host ...

... For some diseases, it is possible to establish causation between a microorganism and a disease by isolating the pathogen from a patient, growing it in pure culture and reintroducing it to a healthy organism. These are known as ‘Koch’s postulates’ and were developed in 1882. However, as our understanding of disease and different disease-causing agents has advanced, these are no longer the method for determining causation [Andrew Kaufman asks why in that case are there two published articles falsely claiming to satisfy Koch’s postulates].

It has long been known that viral diseases cannot be identified in this way as viruses cannot be grown in ‘pure culture’. When a patient is tested for a viral illness, this is normally done by looking for the presence of antigens, or viral genetic code in a host with molecular biology techniques [Kaufman asks how you could know the origin of these chemicals without having a pure culture for comparison].

For the record ‘antigens’ are defined so:

Invading microorganisms have antigens on their surface that the human body can recognise as being foreign – meaning not belonging to it. When the body recognises a foreign antigen, lymphocytes (white blood cells) produce antibodies, which are complementary in shape to the antigen.

Notwithstanding that this is open to question in relation to ‘SARS-CoV-2’ the presence of ‘antibodies’ can have many causes and they are found in people that are perfectly well. Kary Mullis said: ‘Antibodies ... had always been considered evidence of past disease, not present disease.’

‘Covid’ really is a computer ‘virus’

Where the UK Department of Health statement says ‘viruses’ are now ‘diagnosed’ through a ‘viral genetic code in a host with molecular biology techniques’, they mean ... *the PCR test* which its inventor said cannot test for infectious disease. They have no credible method of connecting a ‘virus’ to a disease and we will see that there is no scientific proof that any ‘virus’ causes any disease or there is any such thing as a ‘virus’ in the way that it is described. Tenacious Canadian researcher Christine Massey and her team made

some 40 Freedom of Information requests to national public health agencies in different countries asking for proof that SARS-CoV-2 has been isolated and not one of them could supply that information. Massey said of her request in Canada: 'Freedom of Information reveals Public Health Agency of Canada has no record of 'SARS-CoV-2' isolation performed by anyone, anywhere, ever.' If you accept the comment from the UK Department of Health it's because they can't isolate a 'virus'. Even so many 'science' papers claimed to have isolated the 'Covid virus' until they were questioned and had to admit they hadn't. A reply from the Robert Koch Institute in Germany was typical: 'I am not aware of a paper which purified isolated SARS-CoV-2.' So what the hell was Christian Drosten and his gang using to design the 'Covid' testing protocol that has produced all the illusory Covid' cases and 'Covid' deaths when the head of the Chinese version of the CDC admitted there was a problem right from the start in that the 'virus' had never been isolated/purified? Breathe deeply: What they are calling 'Covid' is actually created by a *computer program* i.e. *they made it up* – er, that's it. They took lung fluid, with many sources of genetic material, from one single person alleged to be infected with Covid-19 by a PCR test which they *claimed*, without clear evidence, contained a 'virus'. They used several computer programs to create a model of a theoretical virus genome sequence from more than fifty-six million small sequences of RNA, each of an unknown source, assembling them like a puzzle with no known solution. The computer filled in the gaps with sequences from bits in the gene bank to make it look like a bat SARS-like coronavirus! A wave of the magic wand and poof, an *in silico* (computer-generated) genome, a scientific fantasy, was created. UK health researcher Dr Kevin Corbett made the same point with this analogy:

... It's like giving you a few bones and saying that's your fish. It could be any fish. Not even a skeleton. Here's a few fragments of bones. That's your fish ... It's all from gene bank and the bits of the virus sequence that weren't there they made up.

They synthetically created them to fill in the blanks. That's what genetics is; it's a code. So it's ABBBCCDDDD and you're missing some what you think is EEE so you put it in. It's all

synthetic. You just manufacture the bits that are missing. This is the end result of the geneticization of virology. This is basically a computer virus.

Further confirmation came in an email exchange between British citizen journalist Frances Leader and the government's Medicines & Healthcare Products Regulatory Agency (the Gates-funded MHRA) which gave emergency permission for untested 'Covid vaccines' to be used. The agency admitted that the 'vaccine' is not based on an isolated 'virus', but comes from a *computer-generated model*. Frances Leader was naturally banned from Cult-owned fascist Twitter for making this exchange public. The process of creating computer-generated alleged 'viruses' is called 'in silico' or 'in silicon' – computer chips – and the term 'in silico' is believed to originate with biological experiments using only a computer in 1989. 'Vaccines' involved with 'Covid' are also produced 'in silico' or by computer not a natural process. If the original 'virus' is nothing more than a made-up computer model how can there be 'new variants' of something that never existed in the first place? They are not new 'variants'; they are new *computer models* only minutely different to the original program and designed to further terrify the population into having the 'vaccine' and submitting to fascism. You want a 'new variant'? Click, click, enter – there you go. Tell the medical profession that you have discovered a 'South African variant', 'UK variants' or a 'Brazilian variant' and in the usual HIV-causes-AIDS manner they will unquestioningly repeat it with no evidence whatsoever to support these claims. They will go on television and warn about the dangers of 'new variants' while doing nothing more than repeating what they have been told to be true and knowing that any deviation from that would be career suicide. Big-time insiders will know it's a hoax, but much of the medical community is clueless about the way they are being played and themselves play the public without even being aware they are doing so. What an interesting 'coincidence' that AstraZeneca and Oxford University were conducting 'Covid vaccine trials' in the three countries – the UK, South Africa and Brazil – where the first three 'variants' were claimed to have 'broken out'.

Here's your 'virus' – it's a unicorn

Dr Andrew Kaufman presented a brilliant analysis describing how the 'virus' was imagined into fake existence when he dissected an article published by *Nature* and written by 19 authors detailing *alleged* 'sequencing of a complete viral genome' of the 'new SARS-CoV-2 virus'. This computer-modelled *in silico* genome was used as a template for all subsequent genome sequencing experiments that resulted in the so-called variants which he said now number more than 6,000. The fake genome was constructed from more than 56 million individual short strands of RNA. Those little pieces were assembled into longer pieces by finding areas of overlapping sequences. The computer programs created over two million possible combinations from which the authors simply chose the longest one. They then compared this to a 'bat virus' and the computer 'alignment' rearranged the sequence and filled in the gaps! They called this computer-generated abomination the 'complete genome'. Dr Tom Cowan, a fellow medical author and collaborator with Kaufman, said such computer-generation constitutes scientific fraud and he makes this superb analogy:

Here is an equivalency: A group of researchers claim to have found a unicorn because they found a piece of a hoof, a hair from a tail, and a snippet of a horn. They then add that information into a computer and program it to re-create the unicorn, and they then claim this computer re-creation is the real unicorn. Of course, they had never actually seen a unicorn so could not possibly have examined its genetic makeup to compare their samples with the actual unicorn's hair, hooves and horn.

The researchers claim they decided which is the real genome of SARS-CoV-2 by 'consensus', sort of like a vote. Again, different computer programs will come up with different versions of the imaginary 'unicorn', so they come together as a group and decide which is the real imaginary unicorn.

This is how the 'virus' that has transformed the world was brought into fraudulent 'existence'. Extraordinary, yes, but as the Nazis said the bigger the lie the more will believe it. Cowan, however, wasn't finished and he went on to identify what he called the real blockbuster in the paper. He quotes this section from a paper written

by virologists and published by the CDC and then explains what it means:

Therefore, we examined the capacity of SARS-CoV-2 to infect and replicate in several common primate and human cell lines, including human adenocarcinoma cells (A549), human liver cells (HUH 7.0), and human embryonic kidney cells (HEK-293T). In addition to Vero E6 and Vero CCL81 cells. ... Each cell line was inoculated at high multiplicity of infection and examined 24h post-infection.

No CPE was observed in any of the cell lines except in Vero cells, which grew to greater than 10 to the 7th power at 24 h post-infection. In contrast, HUH 7.0 and 293T showed only modest viral replication, and A549 cells were incompatible with SARS CoV-2 infection.

Cowan explains that when virologists attempt to prove infection they have three possible 'hosts' or models on which they can test. The first was humans. Exposure to humans was generally not done for ethical reasons and has never been done with SARS-CoV-2 or any coronavirus. The second possible host was animals. Cowan said that forgetting for a moment that they never actually use purified virus when exposing animals they do use solutions that they *claim* contain the virus. Exposure to animals has been done with SARS-CoV-2 in an experiment involving mice and this is what they found: *None of the wild (normal) mice got sick*. In a group of genetically-modified mice, a statistically insignificant number lost weight and had slightly bristled fur, but they experienced nothing like the illness called 'Covid-19'. Cowan said the third method – the one they mostly rely on – is to inoculate solutions they *say* contain the virus onto a variety of tissue cultures. This process had never been shown to kill tissue *unless* the sample material was starved of nutrients and poisoned as *part of the process*. Yes, incredibly, in tissue experiments designed to show the 'virus' is responsible for killing the tissue they starve the tissue of nutrients and add toxic drugs including antibiotics and they do not have control studies to see if it's the starvation and poisoning that is degrading the tissue rather than the 'virus' they allege to be in there somewhere. You want me to pinch you? Yep, I understand. Tom Cowan said this about the whole nonsensical farce as he explains what that quote from the CDC paper really means:

The shocking thing about the above quote is that using their own methods, the virologists found that solutions containing SARS-CoV-2 – even in high amounts – were NOT, I repeat NOT, infective to any of the three human tissue cultures they tested. In plain English, this means they proved, on their terms, that this ‘new coronavirus’ is not infectious to human beings. It is ONLY infective to monkey kidney cells, and only then when you add two potent drugs (gentamicin and amphotericin), known to be toxic to kidneys, to the mix.

My friends, read this again and again. These virologists, published by the CDC, performed a clear proof, on their terms, showing that the SARS-CoV-2 virus is harmless to human beings. That is the only possible conclusion, but, unfortunately, this result is not even mentioned in their conclusion. They simply say they can provide virus stocks cultured only on monkey Vero cells, thanks for coming.

Cowan concluded: ‘If people really understood how this “science” was done, I would hope they would storm the gates and demand honesty, transparency and truth.’ Dr Michael Yeadon, former Vice President and Chief Scientific Adviser at drug giant Pfizer has been a vocal critic of the ‘Covid vaccine’ and its potential for multiple harm. He said in an interview in April, 2021, that ‘not one [vaccine] has the virus. He was asked why vaccines normally using a ‘dead’ version of a disease to activate the immune system were not used for ‘Covid’ and instead we had the synthetic methods of the ‘mRNA Covid vaccine’. Yeadon said that to do the former ‘you’d have to have some of [the virus] wouldn’t you?’ He added: ‘No-one’s got any – seriously.’ Yeadon said that surely they couldn’t have fooled the whole world for a year without having a virus, ‘but oddly enough ask around – no one’s got it’. He didn’t know why with all the ‘great labs’ around the world that the virus had not been isolated – ‘Maybe they’ve been too busy running bad PCR tests and vaccines that people don’t need.’ What is today called ‘science’ is not ‘science’ at all. Science is no longer what is, but whatever people can be manipulated to *believe* that it is. Real science has been hijacked by the Cult to dispense and produce the ‘expert scientists’ and contentions that suit the agenda of the Cult. How big-time this has happened with the ‘Covid’ hoax which is entirely based on fake science delivered by fake ‘scientists’ and fake ‘doctors’. The human-caused climate change hoax is also entirely based on fake science delivered by fake ‘scientists’ and fake ‘climate experts’. In both cases real

scientists, climate experts and doctors have their views suppressed and deleted by the Cult-owned science establishment, media and Silicon Valley. This is the ‘science’ that politicians claim to be ‘following’ and a common denominator of ‘Covid’ and climate are Cult psychopaths Bill Gates and his mate Klaus Schwab at the Gates-funded World Economic Forum. But, don’t worry, it’s all just a coincidence and absolutely nothing to worry about. Zzzzzzzz.

What is a ‘virus’ REALLY?

Dr Tom Cowan is one of many contesting the very existence of viruses let alone that they cause disease. This is understandable when there is no scientific evidence for a disease-causing ‘virus’. German virologist Dr Stefan Lanka won a landmark case in 2017 in the German Supreme Court over his contention that there is no such thing as a measles virus. He had offered a big prize for anyone who could prove there is and Lanka won his case when someone sought to claim the money. There is currently a prize of more than 225,000 euros on offer from an Isolate Truth Fund for anyone who can prove the isolation of SARS-CoV-2 and its genetic substance. Lanka wrote in an article headed ‘The Misconception Called Virus’ that scientists think a ‘virus’ is causing tissue to become diseased and degraded when in fact it is the *processes they are using* which do that – not a ‘virus’. Lanka has done an important job in making this point clear as Cowan did in his analysis of the CDC paper. Lanka says that all claims about viruses as disease-causing pathogens are wrong and based on ‘easily recognisable, understandable and verifiable misinterpretations.’ Scientists believed they were working with ‘viruses’ in their laboratories when they were really working with ‘typical particles of specific dying tissues or cells ...’ Lanka said that the tissue decaying process claimed to be caused by a ‘virus’ still happens when no alleged ‘virus’ is involved. It’s the *process* that does the damage and not a ‘virus’. The genetic sample is deprived of nutrients, removed from its energy supply through removal from the body and then doused in toxic antibiotics to remove any bacteria. He confirms again that establishment scientists do not (pinch me)

conduct control experiments to see if this is the case and if they did they would see the claims that 'viruses' are doing the damage is nonsense. He adds that during the measles 'virus' court case he commissioned an independent laboratory to perform just such a control experiment and the result was that the tissues and cells died in the exact same way as with alleged 'infected' material. This is supported by a gathering number of scientists, doctors and researchers who reject what is called 'germ theory' or the belief in the body being infected by contagious sources emitted by other people. Researchers Dawn Lester and David Parker take the same stance in their highly-detailed and sourced book *What Really Makes You Ill – Why everything you thought you knew about disease is wrong* which was recommended to me by a number of medical professionals genuinely seeking the truth. Lester and Parker say there is no provable scientific evidence to show that a 'virus' can be transmitted between people or people and animals or animals and people:

The definition also claims that viruses are the cause of many diseases, as if this has been definitively proven. But this is not the case; there is no original scientific evidence that definitively demonstrates that any virus is the cause of any disease. The burden of proof for any theory lies with those who proposed it; but none of the existing documents provides 'proof' that supports the claim that 'viruses' are pathogens.

Dr Tom Cowan employs one of his clever analogies to describe the process by which a 'virus' is named as the culprit for a disease when what is called a 'virus' is only material released by cells detoxing themselves from infiltration by chemical or radiation poisoning. The tidal wave of technologically-generated radiation in the 'smart' modern world plus all the toxic food and drink are causing this to happen more than ever. Deluded 'scientists' misread this as a gathering impact of what they wrongly label 'viruses'.

Paper can infect houses

Cowan said in an article for davidicke.com – with his tongue only mildly in his cheek – that he believed he had made a tremendous

discovery that may revolutionise science. He had discovered that small bits of paper are alive, ‘well alive-ish’, can ‘infect’ houses, and then reproduce themselves inside the house. The result was that this explosion of growth in the paper inside the house causes the house to explode, blowing it to smithereens. His evidence for this new theory is that in the past months he had carefully examined many of the houses in his neighbourhood and found almost no scraps of paper on the lawns and surrounds of the house. There was an occasional stray label, but nothing more. Then he would return to these same houses a week or so later and with a few, not all of them, particularly the old and decrepit ones, he found to his shock and surprise they were littered with stray bits of paper. He knew then that the paper had infected these houses, made copies of itself, and blew up the house. A young boy on a bicycle at one of the sites told him he had seen a demolition crew using dynamite to explode the house the previous week, but Cowan dismissed this as the idle thoughts of silly boys because ‘I was on to something big’. He was on to how ‘scientists’ mistake genetic material in the detoxifying process for something they call a ‘virus’. Cowan said of his house and paper story:

If this sounds crazy to you, it’s because it should. This scenario is obviously nuts. But consider this admittedly embellished, for effect, current viral theory that all scientists, medical doctors and virologists currently believe.

He takes the example of the ‘novel SARS-Cov2’ virus to prove the point. First they take someone with an undefined illness called ‘Covid-19’ and don’t even attempt to find any virus in their sputum. Never mind the scientists still describe how this ‘virus’, which they have not located attaches to a cell receptor, injects its genetic material, in ‘Covid’s’ case, RNA, into the cell. The RNA once inserted exploits the cell to reproduce itself and makes ‘thousands, nay millions, of copies of itself ... Then it emerges victorious to claim its next victim’:

If you were to look in the scientific literature for proof, actual scientific proof, that uniform SARS-CoV2 viruses have been properly isolated from the sputum of a sick person, that actual spike proteins could be seen protruding from the virus (which has not been found), you would find that such evidence doesn't exist.

If you go looking in the published scientific literature for actual pictures, proof, that these spike proteins or any viral proteins are ever attached to any receptor embedded in any cell membrane, you would also find that no such evidence exists. If you were to look for a video or documented evidence of the intact virus injecting its genetic material into the body of the cell, reproducing itself and then emerging victorious by budding off the cell membrane, you would find that no such evidence exists.

The closest thing you would find is electron micrograph pictures of cellular particles, possibly attached to cell debris, both of which to be seen were stained by heavy metals, a process that completely distorts their architecture within the living organism. This is like finding bits of paper stuck to the blown-up bricks, thereby proving the paper emerged by taking pieces of the bricks on its way out.

The Enders baloney

Cowan describes the 'Covid' story as being just as make-believe as his paper story and he charts back this fantasy to a Nobel Prize winner called John Enders (1897-1985), an American biomedical scientist who has been dubbed 'The Father of Modern Vaccines'. Enders is claimed to have 'discovered' the process of the viral culture which 'proved' that a 'virus' caused measles. Cowan explains how Enders did this 'by using the EXACT same procedure that has been followed by every virologist to find and characterize every new virus since 1954'. Enders took throat swabs from children with measles and immersed them in 2ml of milk. Penicillin (100u/ml) and the antibiotic streptomycin (50,g/ml) were added and the whole mix was centrifuged – rotated at high speed to separate large cellular debris from small particles and molecules as with milk and cream, for example. Cowan says that if the aim is to find little particles of genetic material ('viruses') in the snot from children with measles it would seem that the last thing you would do is mix the snot with other material – milk –that also has genetic material. 'How are you ever going to know whether whatever you found came from the snot or the milk?' He points out that streptomycin is a 'nephrotoxic' or poisonous-to-the-kidney drug. You will see the relevance of that

shortly. Cowan says that it gets worse, much worse, when Enders describes the culture medium upon which the virus 'grows': 'The culture medium consisted of bovine amniotic fluid (90%), beef embryo extract (5%), horse serum (5%), antibiotics and phenol red as an indicator of cell metabolism.' Cowan asks incredulously: 'Did he just say that the culture medium also contained fluids and tissues that are themselves rich sources of genetic material?' The genetic cocktail, or 'medium', is inoculated onto tissue and cells from rhesus monkey *kidney* tissue. This is where the importance of streptomycin comes in and currently-used antimicrobials and other drugs that are *poisonous to kidneys* and used in ALL modern viral cultures (e.g. gentamicin, streptomycin, and amphotericin). Cowan asks: 'How are you ever going to know from this witch's brew where any genetic material comes from as we now have five different sources of rich genetic material in our mix?' Remember, he says, that all genetic material, whether from monkey kidney tissues, bovine serum, milk, etc., is made from the exact same components. The same central question returns: 'How are you possibly going to know that it was the virus that killed the kidney tissue and not the toxic antibiotic and starvation rations on which you are growing the tissue?' John Enders answered the question himself – *you can't*:

A second agent was obtained from an uninoculated culture of monkey kidney cells. The cytopathic changes [death of the cells] it induced in the unstained preparations could not be distinguished with confidence from the viruses isolated from measles.

The death of the cells ('cytopathic changes') happened in exactly the same manner, whether they inoculated the kidney tissue with the measles snot or not, Cowan says. 'This is evidence that the destruction of the tissue, the very proof of viral causation of illness, was not caused by anything in the snot because they saw the same destructive effect when the snot was not even used ... the cytopathic, i.e., cell-killing, changes come from the process of the culture itself, not from any virus in any snot, period.' Enders quotes in his 1957 paper a virologist called Ruckle as reporting similar findings 'and in addition has isolated an agent from monkey kidney tissue that is so

far indistinguishable from human measles virus'. In other words, Cowan says, these particles called 'measles viruses' are simply and clearly breakdown products of the starved and poisoned tissue. For measles 'virus' see all 'viruses' including the so-called 'Covid virus'. Enders, the 'Father of Modern Vaccines', also said:

There is a potential risk in employing cultures of primate cells for the production of vaccines composed of attenuated virus, since the presence of other agents possibly latent in primate tissues cannot be definitely excluded by any known method.

Cowan further quotes from a paper published in the journal *Viruses* in May, 2020, while the 'Covid pandemic' was well underway in the media if not in reality. 'EVs' here refers to particles of genetic debris from our own tissues, such as exosomes of which more in a moment: 'The remarkable resemblance between EVs and viruses has caused quite a few problems in the studies focused on the analysis of EVs released during viral infections.' Later the paper adds that to date a reliable method that can actually guarantee a complete separation (of EVs from viruses) DOES NOT EXIST. This was published at a time when a fairy tale 'virus' was claimed in total certainty to be causing a fairy tale 'viral disease' called 'Covid-19' – a fairy tale that was already well on the way to transforming human society in the image that the Cult has worked to achieve for so long. Cowan concludes his article:

To summarize, there is no scientific evidence that pathogenic viruses exist. What we think of as 'viruses' are simply the normal breakdown products of dead and dying tissues and cells. When we are well, we make fewer of these particles; when we are starved, poisoned, suffocated by wearing masks, or afraid, we make more.

There is no engineered virus circulating and making people sick. People in laboratories all over the world are making genetically modified products to make people sick. These are called vaccines. There is no virome, no 'ecosystem' of viruses, viruses are not 8%, 50% or 100 % of our genetic material. These are all simply erroneous ideas based on the misconception called a virus.

What is 'Covid'? Load of bollocks

The background described here by Cowan and Lanka was emphasised in the first video presentation that I saw by Dr Andrew Kaufman when he asked whether the ‘Covid virus’ was in truth a natural defence mechanism of the body called ‘exosomes’. These are released by cells when in states of toxicity – see the same themes returning over and over. They are released ever more profusely as chemical and radiation toxicity increases and think of the potential effect therefore of 5G alone as its destructive frequencies infest the human energetic information field with a gathering pace (5G went online in Wuhan in 2019 as the ‘virus’ emerged). I’ll have more about this later. Exosomes transmit a warning to the rest of the body that ‘Houston, we have a problem’. Kaufman presented images of exosomes and compared them with ‘Covid’ under an electron microscope and the similarity was remarkable. They both attach to the same cell receptors (*claimed* in the case of ‘Covid’), contain the same genetic material in the form of RNA or ribonucleic acid, and both are found in ‘viral cell cultures’ with damaged or dying cells. James Hildreth MD, President and Chief Executive Officer of the Meharry Medical College at Johns Hopkins, said: ‘The virus is fully an exosome in every sense of the word.’ Kaufman’s conclusion was that there is no ‘virus’: ‘This entire pandemic is a completely manufactured crisis … there is no evidence of anyone dying from [this] illness.’ Dr Tom Cowan and Sally Fallon Morell, authors of *The Contagion Myth*, published a statement with Dr Kaufman in February, 2021, explaining why the ‘virus’ does not exist and you can read it that in full in the Appendix.

‘Virus’ theory can be traced to the ‘cell theory’ in 1858 of German physician Rudolf Virchow (1821-1920) who contended that disease originates from a single cell infiltrated by a ‘virus’. Dr Stefan Lanka said that findings and insights with respect to the structure, function and central importance of tissues in the creation of life, which were already known in 1858, comprehensively refute the cell theory. Virchow ignored them. We have seen the part later played by John Enders in the 1950s and Lanka notes that infection theories were only established as a global dogma through the policies and

eugenics of the Third Reich in Nazi Germany (creation of the same Sabbatian cult behind the ‘Covid’ hoax). Lanka said: ‘Before 1933, scientists dared to contradict this theory; after 1933, these critical scientists were silenced’. Dr Tom Cowan’s view is that ill-health is caused by too much of something, too little of something, or toxification from chemicals and radiation – not contagion. We must also highlight as a major source of the ‘virus’ theology a man still called the ‘Father of Modern Virology’ – Thomas Milton Rivers (1888-1962). There is no way given the Cult’s long game policy that it was a coincidence for the ‘Father of Modern Virology’ to be director of the Rockefeller Institute for Medical Research from 1937 to 1956 when he is credited with making the Rockefeller Institute a leader in ‘viral research’. Cult Rockefellers were the force behind the creation of Big Pharma ‘medicine’, established the World Health Organisation in 1948, and have long and close associations with the Gates family that now runs the WHO during the pandemic hoax through mega-rich Cult gofer and psychopath Bill Gates.

Only a Renegade Mind can see through all this bullshit by asking the questions that need to be answered, not taking ‘no’ or prevarication for an answer, and certainly not hiding from the truth in fear of speaking it. Renegade Minds have always changed the world for the better and they will change this one no matter how bleak it may currently appear to be.

CHAPTER SIX

Sequence of deceit

If you tell the truth, you don't have to remember anything

Mark Twain

Against the background that I have laid out this far the sequence that took us from an invented 'virus' in Cult-owned China in late 2019 to the fascist transformation of human society can be seen and understood in a whole new context.

We were told that a deadly disease had broken out in Wuhan and the world media began its campaign (coordinated by behavioural psychologists as we shall see) to terrify the population into unquestioning compliance. We were shown images of Chinese people collapsing in the street which never happened in the West with what was supposed to be the same condition. In the earliest days when alleged cases and deaths were few the fear register was hysterical in many areas of the media and this would expand into the common media narrative across the world. The real story was rather different, but we were never told that. The Chinese government, one of the Cult's biggest centres of global operation, said they had discovered a new illness with flu-like and pneumonia-type symptoms in a city with such toxic air that it is overwhelmed with flu-like symptoms, pneumonia and respiratory disease. Chinese scientists said it was a new – 'novel' – coronavirus which they called Sars-Cov-2 and that it caused a disease they labelled 'Covid-19'. There was no evidence for this and the 'virus' has never to this day been isolated, purified and its genetic code established from that. It

was from the beginning a computer-generated fiction. Stories of Chinese whistleblowers saying the number of deaths was being suppressed or that the ‘new disease’ was related to the Wuhan bio-lab misdirected mainstream and alternative media into cul-de-sacs to obscure the real truth – there was no ‘virus’.

Chinese scientists took genetic material from the lung fluid of just a few people and said they had found a ‘new’ disease when this material had a wide range of content. There was no evidence for a ‘virus’ for the very reasons explained in the last two chapters. The ‘virus’ has never been shown to (a) exist and (b) cause any disease. People were diagnosed on symptoms that are so widespread in Wuhan and polluted China and with a PCR test that can’t detect infectious disease. On this farce the whole global scam was sold to the rest of the world which would also diagnose respiratory disease as ‘Covid-19’ from symptoms alone or with a PCR test not testing for a ‘virus’. Flu miraculously disappeared *worldwide* in 2020 and into 2021 as it was redesignated ‘Covid-19’. It was really the same old flu with its ‘flu-like’ symptoms attributed to ‘flu-like’ ‘Covid-19’. At the same time with very few exceptions the Chinese response of draconian lockdown and fascism was the chosen weapon to respond across the West as recommended by the Cult-owned Tedros at the Cult-owned World Health Organization run by the Cult-owned Gates. All was going according to plan. Chinese scientists – everything in China is controlled by the Cult-owned government – compared their contaminated RNA lung-fluid material with other RNA sequences and said it appeared to be just under 80 percent identical to the SARS-CoV-1 ‘virus’ claimed to be the cause of the SARS (severe acute respiratory syndrome) ‘outbreak’ in 2003. They decreed that because of this the ‘new virus’ had to be related and they called it SARS-CoV-2. There are some serious problems with this assumption and *assumption* was all it was. Most ‘factual’ science turns out to be assumptions repeated into everyone-knows-that. A match of under 80-percent is meaningless. Dr Kaufman makes the point that there’s a 96 percent genetic correlation between humans and chimpanzees, but ‘no one would say our genetic material is part

of the chimpanzee family'. Yet the Chinese authorities were claiming that a much lower percentage, less than 80 percent, proved the existence of a new 'coronavirus'. For goodness sake human DNA is 60 percent similar to a *banana*.

You are feeling sleepy

The entire 'Covid' hoax is a global Psyop, a psychological operation to program the human mind into believing and fearing a complete fantasy. A crucial aspect of this was what *appeared* to happen in Italy. It was all very well streaming out daily images of an alleged catastrophe in Wuhan, but to the Western mind it was still on the other side of the world in a very different culture and setting. A reaction of 'this could happen to me and my family' was still nothing like as intense enough for the mind-doctors. The Cult needed a Western example to push people over that edge and it chose Italy, one of its major global locations going back to the Roman Empire. An Italian 'Covid' crisis was manufactured in a particular area called Lombardy which just happens to be notorious for its toxic air and therefore respiratory disease. Wuhan, China, *déjà vu*. An hysterical media told horror stories of Italians dying from 'Covid' in their droves and how Lombardy hospitals were being overrun by a tidal wave of desperately ill people needing treatment after being struck down by the 'deadly virus'. Here was the psychological turning point the Cult had planned. Wow, if this is happening in Italy, the Western mind concluded, this indeed could happen to me and my family. Another point is that Italian authorities responded by following the Chinese blueprint so vehemently recommended by the Cult-owned World Health Organization. They imposed fascistic lockdowns on the whole country viciously policed with the help of surveillance drones sweeping through the streets seeking out anyone who escaped from mass house arrest. Livelihoods were destroyed and psychology unravelled in the way we have witnessed since in all lockdown countries. Crucial to the plan was that Italy responded in this way to set the precedent of suspending freedom and imposing fascism in a 'Western liberal democracy'. I emphasised in an

animated video explanation on davidicke.com posted in the summer of 2020 how important it was to the Cult to expand the Chinese lockdown model across the West. Without this, and the bare-faced lie that non-symptomatic people could still transmit a ‘disease’ they didn’t have, there was no way locking down the whole population, sick and not sick, could be pulled off. At just the right time and with no evidence Cult operatives and gofers claimed that people without symptoms could pass on the ‘disease’. In the name of protecting the ‘vulnerable’ like elderly people, who lockdowns would kill by the tens of thousands, we had for the first time healthy people told to isolate as well as the sick. The great majority of people who tested positive had no symptoms because there was nothing wrong with them. It was just a trick made possible by a test not testing for the ‘virus’.

Months after my animated video the Gates-funded Professor Neil Ferguson at the Gates-funded Imperial College confirmed that I was right. He didn’t say it in those terms, naturally, but he did say it. Ferguson will enter the story shortly for his outrageously crazy ‘computer models’ that led to Britain, the United States and many other countries following the Chinese and now Italian methods of response. Put another way, following the Cult script. Ferguson said that SAGE, the UK government’s scientific advisory group which has controlled ‘Covid’ policy from the start, wanted to follow the Chinese lockdown model (while they all continued to work and be paid), but they wondered if they could possibly, in Ferguson’s words, ‘get away with it in Europe’. ‘Get away with it’? Who the hell do these moronic, arrogant people think they are? This appalling man Ferguson said that once Italy went into national lockdown they realised they, too, could mimic China:

It’s a communist one-party state, we said. We couldn’t get away with it in Europe, we thought ... and then Italy did it. And we realised we could. Behind this garbage from Ferguson is a simple fact: Doing the same as China in every country was the plan from the start and Ferguson’s ‘models’ would play a central role in achieving that. It’s just a coincidence, of course, and absolutely nothing to worry your little head about.

Oops, sorry, our mistake

Once the Italian segment of the Psyop had done the job it was designed to do a very different story emerged. Italian authorities revealed that 99 percent of those who had 'died from Covid-19' in Italy had one, two, three, or more 'co-morbidities' or illnesses and health problems that could have ended their life. The US Centers for Disease Control and Prevention (CDC) published a figure of 94 percent for Americans dying of 'Covid' while having other serious medical conditions – on average two to three (some five or six) other potential causes of death. In terms of death from an unproven 'virus' I say it is 100 percent. The other one percent in Italy and six percent in the US would presumably have died from 'Covid's' flu-like symptoms with a range of other possible causes in conjunction with a test not testing for the 'virus'. Fox News reported that even more startling figures had emerged in one US county in which 410 of 422 deaths attributed to 'Covid-19' had other potentially deadly health conditions. The Italian National Health Institute said later that the average age of people dying with a 'Covid-19' diagnosis in Italy was about 81. Ninety percent were over 70 with ten percent over 90. In terms of other reasons to die some 80 percent had two or more chronic diseases with half having three or more including cardiovascular problems, diabetes, respiratory problems and cancer. Why is the phantom 'Covid-19' said to kill overwhelmingly old people and hardly affect the young? Old people continually die of many causes and especially respiratory disease which you can re-diagnose 'Covid-19' while young people die in tiny numbers by comparison and rarely of respiratory disease. Old people 'die of Covid' because they die of other things that can be redesignated 'Covid' and it really is that simple.

Flu has flown

The blueprint was in place. Get your illusory 'cases' from a test not testing for the 'virus' and redesignate other causes of death as 'Covid-19'. You have an instant 'pandemic' from something that is nothing more than a computer-generated fiction. With near-on a

billion people having ‘flu-like’ symptoms every year the potential was limitless and we can see why flu quickly and apparently miraculously disappeared *worldwide* by being diagnosed ‘Covid-19’. The painfully bloody obvious was explained away by the childlike media in headlines like this in the UK *‘Independent’*: ‘Not a single case of flu detected by Public Health England this year as Covid restrictions suppress virus’. I kid you not. The masking, social distancing and house arrest that did not make the ‘Covid virus’ disappear somehow did so with the ‘flu virus’. Even worse the article, by a bloke called Samuel Lovett, suggested that maybe the masking, sanitising and other ‘Covid’ measures should continue to keep the flu away. With a ridiculousness that disturbs your breathing (it’s ‘Covid-19’) the said Lovett wrote: ‘With widespread social distancing and mask-wearing measures in place throughout the UK, the usual routes of transmission for influenza have been blocked.’ He had absolutely no evidence to support that statement, but look at the consequences of him acknowledging the obvious. With flu not disappearing at all and only being relabelled ‘Covid-19’ he would have to contemplate that ‘Covid’ was a hoax on a scale that is hard to imagine. You need guts and commitment to truth to even go there and that’s clearly something Samuel Lovett does not have in abundance. He would never have got it through the editors anyway.

Tens of thousands die in the United States alone every winter from flu including many with pneumonia complications. CDC figures record *45 million* Americans diagnosed with flu in 2017-2018 of which 61,000 died and some reports claim 80,000. Where was the same hysteria then that we have seen with ‘Covid-19’? Some 250,000 Americans are admitted to hospital with pneumonia every year with about 50,000 cases proving fatal. About 65 million suffer respiratory disease every year and three million deaths makes this the third biggest cause of death worldwide. You only have to redesignate a portion of all these people ‘Covid-19’ and you have an instant global pandemic or the *appearance* of one. Why would doctors do this? They are told to do this and all but a few dare not refuse those who must be obeyed. Doctors in general are not researching their own

knowledge and instead take it direct and unquestioned from the authorities that own them and their careers. The authorities say they must now diagnose these symptoms ‘Covid-19’ and not flu, or whatever, and they do it. Dark suits say put ‘Covid-19’ on death certificates no matter what the cause of death and the doctors do it. Renegade Minds don’t fall for the illusion that doctors and medical staff are all highly-intelligent, highly-principled, seekers of medical truth. *Some are*, but not the majority. They are repeaters, gofers, and yes sir, no sir, purveyors of what the system demands they purvey. The ‘Covid’ con is not merely confined to diseases of the lungs. Instructions to doctors to put ‘Covid-19’ on death certificates for anyone dying of *anything* within 28 days (or much more) of a positive test not testing for the ‘virus’ opened the floodgates. The term dying *with* ‘Covid’ and not *of* ‘Covid’ was coined to cover the truth. Whether it was a *with* or an *of* they were all added to the death numbers attributed to the ‘deadly virus’ compiled by national governments and globally by the Gates-funded Johns Hopkins operation in the United States that was so involved in those ‘pandemic’ simulations. Fraudulent deaths were added to the ever-growing list of fraudulent ‘cases’ from false positives from a false test. No wonder Professor Walter Ricciardi, scientific advisor to the Italian minister of health, said after the Lombardy hysteria had done its job that ‘Covid’ death rates were due to Italy having the second oldest population in the world and to *how hospitals record deaths*:

The way in which we code deaths in our country is very generous in the sense that all the people who die in hospitals with the coronavirus are deemed to be dying of the coronavirus. On re-evaluation by the National Institute of Health, only 12 per cent of death certificates have shown a direct causality from coronavirus, while 88 per cent of patients who have died have at least one pre-morbidity – many had two or three.

This is extraordinary enough when you consider the propaganda campaign to use Italy to terrify the world, but how can they even say twelve percent were genuine when the ‘virus’ has not been shown to exist, its ‘code’ is a computer program, and diagnosis comes from a test not testing for it? As in China, and soon the world, ‘Covid-19’ in

Italy was a redesignation of diagnosis. Lies and corruption were to become the real ‘pandemic’ fuelled by a pathetically-compliant medical system taking its orders from the tiny few at the top of their national hierarchy who answered to the World Health Organization which answers to Gates and the Cult. Doctors were told – ordered – to diagnose a particular set of symptoms ‘Covid-19’ and put that on the death certificate for any cause of death if the patient had tested positive with a test not testing for the virus or had ‘Covid’ symptoms like the flu. The United States even introduced big financial incentives to manipulate the figures with hospitals receiving £4,600 from the Medicare system for diagnosing someone with regular pneumonia, \$13,000 if they made the diagnosis from the same symptoms ‘Covid-19’ pneumonia, and \$39, 000 if they put a ‘Covid’ diagnosed patient on a ventilator that would almost certainly kill them. A few – painfully and pathetically few – medical whistleblowers revealed (before Cult-owned YouTube deleted their videos) that they had been instructed to ‘let the patient crash’ and put them straight on a ventilator instead of going through a series of far less intrusive and dangerous methods as they would have done before the pandemic hoax began and the financial incentives kicked in. We are talking cold-blooded murder given that ventilators are so damaging to respiratory systems they are usually the last step before heaven awaits. Renegade Minds never fall for the belief that people in white coats are all angels of mercy and cannot be full-on psychopaths. I have explained in detail in *The Answer* how what I am describing here played out across the world coordinated by the World Health Organization through the medical hierarchies in almost every country.

Medical scientist calls it

Information about the non-existence of the ‘virus’ began to emerge for me in late March, 2020, and mushroomed after that. I was sent an email by Sir Julian Rose, a writer, researcher, and organic farming promotor, from a medical scientist friend of his in the United States. Even at that early stage in March the scientist was able to explain

how the ‘Covid’ hoax was being manipulated. He said there were no reliable tests for a specific ‘Covid-19 virus’ and nor were there any reliable agencies or media outlets for reporting numbers of actual ‘Covid-19’ cases. We have seen in the long period since then that he was absolutely right. ‘Every action and reaction to Covid-19 is based on totally flawed data and we simply cannot make accurate assessments,’ he said. Most people diagnosed with ‘Covid-19’ were showing nothing more than cold and flu-like symptoms ‘because most coronavirus strains *are* nothing more than cold/flu-like symptoms’. We had farcical situations like an 84-year-old German man testing positive for ‘Covid-19’ and his nursing home ordered to quarantine only for him to be found to have a common cold. The scientist described back then why PCR tests and what he called the ‘Mickey Mouse test kits’ were useless for what they were claimed to be identifying. ‘The idea these kits can isolate a specific virus like Covid-19 is nonsense,’ he said. Significantly, he pointed out that ‘if you want to create a totally false panic about a totally false pandemic – pick a coronavirus’. This is exactly what the Cult-owned Gates, World Economic Forum and Johns Hopkins University did with their Event 201 ‘simulation’ followed by their real-life simulation called the ‘pandemic’. The scientist said that all you had to do was select the sickest of people with respiratory-type diseases in a single location – ‘say Wuhan’ – and administer PCR tests to them. You can then claim that anyone showing ‘viral sequences’ similar to a coronavirus ‘which will inevitably be quite a few’ is suffering from a ‘new’ disease:

Since you already selected the sickest flu cases a fairly high proportion of your sample will go on to die. You can then say this ‘new’ virus has a CFR [case fatality rate] higher than the flu and use this to infuse more concern and do more tests which will of course produce more ‘cases’, which expands the testing, which produces yet more ‘cases’ and so on and so on. Before long you have your ‘pandemic’, and all you have done is use a simple test kit trick to convert the worst flu and pneumonia cases into something new that doesn’t ACTUALLY EXIST [my emphasis].

He said that you then ‘just run the same scam in other countries’ and make sure to keep the fear message running high ‘so that people

will feel panicky and less able to think critically'. The only problem to overcome was the fact *there is no* actual new deadly pathogen and only regular sick people. This meant that deaths from the 'new deadly pathogen' were going to be way too low for a real new deadly virus pandemic, but he said this could be overcome in the following ways – all of which would go on to happen:

1. You can claim this is just the beginning and more deaths are imminent [you underpin this with fantasy 'computer projections']. Use this as an excuse to quarantine everyone and then claim the quarantine prevented the expected millions of dead.
2. You can [say that people] 'minimizing' the dangers are irresponsible and bully them into not talking about numbers.
3. You can talk crap about made up numbers hoping to blind people with pseudoscience.
4. You can start testing well people (who, of course, will also likely have shreds of coronavirus [RNA] in them) and thus inflate your 'case figures' with 'asymptomatic carriers' (you will of course have to spin that to sound deadly even though any virologist knows the more symptom-less cases you have the less deadly is your pathogen).

The scientist said that if you take these simple steps 'you can have your own entirely manufactured pandemic up and running in weeks'. His analysis made so early in the hoax was brilliantly prophetic of what would actually unfold. Pulling all the information together in these recent chapters we have this is simple 1, 2, 3, of how you can delude virtually the entire human population into believing in a 'virus' that doesn't exist:

- A 'Covid case' is someone who tests positive with a test not testing for the 'virus'.
- A 'Covid death' is someone who dies of *any cause* within 28 days (or much longer) of testing positive with a test not testing for the 'virus'.
- Asymptomatic means there is nothing wrong with you, but they claim you can pass on what you don't have to justify locking

down (quarantining) healthy people in totality.

The foundations of the hoax are that simple. A study involving ten million people in Wuhan, published in November, 2020, demolished the whole lie about those without symptoms passing on the ‘virus’. They found ‘300 asymptomatic cases’ and traced their contacts to find that not one of them was detected with the ‘virus’.

‘Asymptomatic’ patients and their contacts were isolated for no less than two weeks and nothing changed. I know it’s all crap, but if you are going to claim that those without symptoms can transmit ‘the virus’ then you must produce evidence for that and they never have. Even World Health Organization official Dr Maria Van Kerkhove, head of the emerging diseases and zoonosis unit, said as early as June, 2020, that she doubted the validity of asymptomatic transmission. She said that ‘from the data we have, it still seems to be rare that an asymptomatic person actually transmits onward to a secondary individual’ and by ‘rare’ she meant that she couldn’t cite any case of asymptomatic transmission.

The Ferguson factor

The problem for the Cult as it headed into March, 2020, when the script had lockdown due to start, was that despite all the manipulation of the case and death figures they still did not have enough people alleged to have died from ‘Covid’ to justify mass house arrest. This was overcome in the way the scientist described: ‘You can claim this is just the beginning and more deaths are imminent ... Use this as an excuse to quarantine everyone and then claim the quarantine prevented the expected millions of dead.’ Enter one Professor Neil Ferguson, the Gates-funded ‘epidemiologist’ at the Gates-funded Imperial College in London. Ferguson is Britain’s Christian Drosten in that he has a dire record of predicting health outcomes, but is still called upon to advise government on the next health outcome when another ‘crisis’ comes along. This may seem to be a strange and ridiculous thing to do. Why would you keep turning for policy guidance to people who have a history of being

monumentally wrong? Ah, but it makes sense from the Cult point of view. These ‘experts’ keep on producing predictions that suit the Cult agenda for societal transformation and so it was with Neil Ferguson as he revealed his horrific (and clearly insane) computer model predictions that allowed lockdowns to be imposed in Britain, the United States and many other countries. Ferguson does not have even an A-level in biology and would appear to have no formal training in computer modelling, medicine or epidemiology, according to Derek Winton, an MSc in Computational Intelligence. He wrote an article somewhat aghast at what Ferguson did which included taking no account of respiratory disease ‘seasonality’ which means it is far worse in the winter months. Who would have thought that respiratory disease could be worse in the winter? Well, certainly not Ferguson.

The massively China-connected Imperial College and its bizarre professor provided the excuse for the long-incubated Chinese model of human control to travel westward at lightning speed. Imperial College confirms on its website that it collaborates with the Chinese Research Institute; publishes more than 600 research papers every year with Chinese research institutions; has 225 Chinese staff; 2,600 Chinese students – the biggest international group; 7,000 former students living in China which is the largest group outside the UK; and was selected for a tour by China’s President Xi Jinping during his state visit to the UK in 2015. The college takes major donations from China and describes itself as the UK’s number one university collaborator with Chinese research institutions. The China communist/fascist government did not appear phased by the woeful predictions of Ferguson and Imperial when during the lockdown that Ferguson induced the college signed a five-year collaboration deal with China tech giant Huawei that will have Huawei’s indoor 5G network equipment installed at the college’s West London tech campus along with an ‘AI cloud platform’. The deal includes Chinese sponsorship of Imperial’s Venture Catalyst entrepreneurship competition. Imperial is an example of the enormous influence the Chinese government has within British and North American

universities and research centres – and further afield. Up to 200 academics from more than a dozen UK universities are being investigated on suspicion of ‘unintentionally’ helping the Chinese government build weapons of mass destruction by ‘transferring world-leading research in advanced military technology such as aircraft, missile designs and cyberweapons’. Similar scandals have broken in the United States, but it’s all a coincidence. Imperial College serves the agenda in many other ways including the promotion of every aspect of the United Nations Agenda 21/2030 (the Great Reset) and produced computer models to show that human-caused ‘climate change’ is happening when in the real world it isn’t. Imperial College is driving the climate agenda as it drives the ‘Covid’ agenda (both Cult hoaxes) while Patrick Vallance, the UK government’s Chief Scientific Adviser on ‘Covid’, was named Chief Scientific Adviser to the UN ‘climate change’ conference known as COP26 hosted by the government in Glasgow, Scotland. ‘Covid’ and ‘climate’ are fundamentally connected.

Professor Woeful

From Imperial’s bosom came Neil Ferguson still advising government despite his previous disasters and it was announced early on that he and other key people like UK Chief Medical Adviser Chris Whitty had caught the ‘virus’ as the propaganda story was being sold. Somehow they managed to survive and we had Prime Minister Boris Johnson admitted to hospital with what was said to be a severe version of the ‘virus’ in this same period. His whole policy and demeanour changed when he returned to Downing Street. It’s a small world with these government advisors – especially in their communal connections to Gates – and Ferguson had partnered with Whitty to write a paper called ‘Infectious disease: Tough choices to reduce Ebola transmission’ which involved another scare-story that didn’t happen. Ferguson’s ‘models’ predicted that up to 150, 000 could die from ‘mad cow disease’, or BSE, and its version in sheep if it was transmitted to humans. BSE was not transmitted and instead triggered by an organophosphate pesticide used to treat a pest on

cows. Fewer than 200 deaths followed from the human form. Models by Ferguson and his fellow incompetents led to the unnecessary culling of millions of pigs, cattle and sheep in the foot and mouth outbreak in 2001 which destroyed the lives and livelihoods of farmers and their families who had often spent decades building their herds and flocks. Vast numbers of these animals did not have foot and mouth and had no contact with the infection. Another ‘expert’ behind the cull was Professor Roy Anderson, a computer modeller at Imperial College specialising in the epidemiology of *human*, not animal, disease. Anderson has served on the Bill and Melinda Gates Grand Challenges in Global Health advisory board and chairs another Gates-funded organisation. Gates is everywhere.

In a precursor to the ‘Covid’ script Ferguson backed closing schools ‘for prolonged periods’ over the swine flu ‘pandemic’ in 2009 and said it would affect a third of the world population if it continued to spread at the speed he claimed to be happening. His mates at Imperial College said much the same and a news report said: ‘One of the authors, the epidemiologist and disease modeller Neil Ferguson, who sits on the World Health Organisation’s emergency committee for the outbreak, said the virus had “full pandemic potential”.’ Professor Liam Donaldson, the Chris Whitty of his day as Chief Medical Officer, said the worst case could see 30 percent of the British people infected by swine flu with 65,000 dying. Ferguson and Donaldson were indeed proved correct when at the end of the year the number of deaths attributed to swine flu was 392. The term ‘expert’ is rather liberally applied unfortunately, not least to complete idiots. Swine flu ‘projections’ were great for GlaxoSmithKline (GSK) as millions rolled in for its Pandemrix influenza vaccine which led to brain damage with children most affected. The British government (taxpayers) paid out more than £60 million in compensation after GSK was given immunity from prosecution. Yet another ‘Covid’ déjà vu. Swine flu was supposed to have broken out in Mexico, but Dr Wolfgang Wodarg, a German doctor, former member of parliament and critic of the ‘Covid’ hoax, observed ‘the spread of swine flu’ in Mexico City at the time. He

said: 'What we experienced in Mexico City was a very mild flu which did not kill more than usual – which killed even fewer people than usual.' Hyping the fear against all the facts is not unique to 'Covid' and has happened many times before. Ferguson is reported to have over-estimated the projected death toll of bird flu (H5N1) by some three million-fold, but bird flu vaccine makers again made a killing from the scare. This is some of the background to the Neil Ferguson who produced the perfectly-timed computer models in early 2020 predicting that half a million people would die in Britain without draconian lockdown and 2.2 million in the United States. Politicians panicked, people panicked, and lockdowns of alleged short duration were instigated to 'flatten the curve' of cases gleaned from a test not testing for the 'virus'. I said at the time that the public could forget the 'short duration' bit. This was an agenda to destroy the livelihoods of the population and force them into mass control through dependency and there was going to be nothing 'short' about it. American researcher Daniel Horowitz described the consequences of the 'models' spewed out by Gates-funded Ferguson and Imperial College:

What led our government and the governments of many other countries into panic was a single Imperial College of UK study, funded by global warming activists, that predicted 2.2 million deaths if we didn't lock down the country. In addition, the reported 8-9% death rate in Italy scared us into thinking there was some other mutation of this virus that they got, which might have come here.

Together with the fact that we were finally testing and had the ability to actually report new cases, we thought we were headed for a death spiral. But again ... we can't flatten a curve if we don't know when the curve started.

How about it *never* started?

Giving them what they want

An investigation by German news outlet *Welt Am Sonntag* (*World on Sunday*) revealed how in March, 2020, the German government gathered together 'leading scientists from several research institutes and universities' and 'together, they were to produce a [modelling]

paper that would serve as legitimization for further tough political measures'. The Cult agenda was justified by computer modelling not based on evidence or reality; it was specifically constructed to justify the Cult demand for lockdowns all over the world to destroy the independent livelihoods of the global population. All these modellers and everyone responsible for the 'Covid' hoax have a date with a trial like those in Nuremberg after World War Two when Nazis faced the consequences of their war crimes. These corrupt-beyond-belief 'modellers' wrote the paper according to government instructions and it said that if lockdown measures were lifted then up to one million Germans would die from 'Covid-19' adding that some would die 'agonizingly at home, gasping for breath' unable to be treated by hospitals that couldn't cope. All lies. No matter – it gave the Cult all that it wanted. What did long-time government 'modeller' Neil Ferguson say? If the UK and the United States didn't lockdown half a million would die in Britain and 2.2 million Americans. Anyone see a theme here? 'Modellers' are such a crucial part of the lockdown strategy that we should look into their background and follow the money. Researcher Rosemary Frei produced an excellent article headlined 'The Modelling-paper Mafiosi'. She highlights a guy called John Edmunds, a British epidemiologist, and professor in the Faculty of Epidemiology and Population Health at the London School of Hygiene & Tropical Medicine. He studied at Imperial College. Edmunds is a member of government 'Covid' advisory bodies which have been dictating policy, the New and Emerging Respiratory Virus Threats Advisory Group (NERVTAG) and the Scientific Advisory Group for Emergencies (SAGE).

Ferguson, another member of NERVTAG and SAGE, led the way with the original 'virus' and Edmunds has followed in the 'variant' stage and especially the so-called UK or Kent variant known as the 'Variant of Concern' (VOC) B.1.1.7. He said in a co-written report for the Centre for Mathematical modelling of Infectious Diseases at the London School of Hygiene and Tropical Medicine, with input from the Centre's 'Covid-19' Working Group, that there was 'a realistic

possibility that VOC B.1.1.7 is associated with an increased risk of death compared to non-VOC viruses'. Fear, fear, fear, get the vaccine, fear, fear, fear, get the vaccine. Rosemary Frei reveals that almost all the paper's authors and members of the modelling centre's 'Covid-19' Working Group receive funding from the Bill and Melinda Gates Foundation and/or the associated Gates-funded Wellcome Trust. The paper was published by e-journal *Medr* ^{xiv} which only publishes papers not peer-reviewed and the journal was established by an organisation headed by Facebook's Mark Zuckerberg and his missus. What a small world it is. Frei discovered that Edmunds is on the Scientific Advisory Board of the Coalition for Epidemic Preparedness Innovations (CEPI) which was established by the Bill and Melinda Gates Foundation, Klaus Schwab's Davos World Economic Forum and Big Pharma giant Wellcome. CEPI was 'launched in Davos [in 2017] to develop vaccines to stop future epidemics', according to its website. 'Our mission is to accelerate the development of vaccines against emerging infectious diseases and enable equitable access to these vaccines for people during outbreaks.' What kind people they are. Rosemary Frei reveals that Public Health England (PHE) director Susan Hopkins is an author of her organisation's non-peer-reviewed reports on 'new variants'. Hopkins is a professor of infectious diseases at London's Imperial College which is gifted tens of millions of dollars a year by the Bill and Melinda Gates Foundation. Gates-funded modelling disaster Neil Ferguson also co-authors Public Health England reports and he spoke in December, 2020, about the potential danger of the B.1.1.7. 'UK variant' promoted by Gates-funded modeller John Edmunds. When I come to the 'Covid vaccines' the 'new variants' will be shown for what they are – bollocks.

Connections, connections

All these people and modellers are lockdown-obsessed or, put another way, they demand what the Cult demands. Edmunds said in January, 2021, that to ease lockdowns too soon would be a disaster and they had to 'vaccinate much, much, much more widely than the

elderly'. Rosemary Frei highlights that Edmunds is married to Jeanne Pimenta who is described in a LinkedIn profile as director of epidemiology at GlaxoSmithKline (GSK) and she held shares in the company. Patrick Vallance, co-chair of SAGE and the government's Chief Scientific Adviser, is a former executive of GSK and has a deferred bonus of shares in the company worth £600,000. GSK has serious business connections with Bill Gates and is collaborating with mRNA-'vaccine' company CureVac to make 'vaccines' for the new variants that Edmunds is talking about. GSK is planning a 'Covid vaccine' with drug giant Sanofi. Puppet Prime Minister Boris Johnson announced in the spring of 2021 that up to 60 million vaccine doses were to be made at the GSK facility at Barnard Castle in the English North East. Barnard Castle, with a population of just 6,000, was famously visited in breach of lockdown rules in April, 2020, by Johnson aide Dominic Cummings who said that he drove there 'to test his eyesight' before driving back to London. Cummings would be better advised to test his integrity – not that it would take long. The GSK facility had nothing to do with his visit then although I'm sure Patrick Vallance would have been happy to arrange an introduction and some tea and biscuits. Ruthless psychopath Gates has made yet another fortune from vaccines in collaboration with Big Pharma companies and gushes at the phenomenal profits to be made from vaccines – more than a 20-to-1 return as he told one interviewer. Gates also tweeted in December, 2019, with the foreknowledge of what was coming: 'What's next for our foundation? I'm particularly excited about what the next year could mean for one of the best buys in global health: vaccines.'

Modeller John Edmunds is a big promotor of vaccines as all these people appear to be. He's the dean of the London School of Hygiene & Tropical Medicine's Faculty of Epidemiology and Population Health which is primarily funded by the Bill and Melinda Gates Foundation and the Gates-established and funded GAVI vaccine alliance which is the Gates vehicle to vaccinate the world. The organisation Doctors Without Borders has described GAVI as being 'aimed more at supporting drug-industry desires to promote new

products than at finding the most efficient and sustainable means for fighting the diseases of poverty'. But then that's why the psychopath Gates created it. John Edmunds said in a video that the London School of Hygiene & Tropical Medicine is involved in every aspect of vaccine development including large-scale clinical trials. He contends that mathematical modelling can show that vaccines protect individuals and society. That's on the basis of shit in and shit out, I take it. Edmunds serves on the UK Vaccine Network as does Ferguson and the government's foremost 'Covid' adviser, the grim-faced, dark-eyed Chris Whitty. The Vaccine Network says it works 'to support the government to identify and shortlist targeted investment opportunities for the most promising vaccines and vaccine technologies that will help combat infectious diseases with epidemic potential, and to address structural issues related to the UK's broader vaccine infrastructure'. Ferguson is acting Director of the Imperial College Vaccine Impact Modelling Consortium which has funding from the Bill and Melina Gates Foundation and the Gates-created GAVI 'vaccine alliance'. Anyone wonder why these characters see vaccines as the answer to every problem? Ferguson is wildly enthusiastic in his support for GAVI's campaign to vaccine children en masse in poor countries. You would expect someone like Gates who has constantly talked about the need to reduce the population to want to fund vaccines to keep more people alive. I'm sure that's why he does it. The John Edmunds London School of Hygiene & Tropical Medicine (LSHTM) has a Vaccines Manufacturing Innovation Centre which develops, tests and commercialises vaccines. Rosemary Frei writes:

The vaccines centre also performs affiliated activities like combating 'vaccine hesitancy'. The latter includes the Vaccine Confidence Project. The project's stated purpose is, among other things, 'to provide analysis and guidance for early response and engagement with the public to ensure sustained confidence in vaccines and immunisation'. The Vaccine Confidence Project's director is LSHTM professor Heidi Larson. For more than a decade she's been researching how to combat vaccine hesitancy.

How the bloody hell can blokes like John Edmunds and Neil Ferguson with those connections and financial ties model 'virus' case

and death projections for the government and especially in a way that gives their paymasters like Gates exactly what they want? It's insane, but this is what you find throughout the world.

'Covid' is not dangerous, oops, wait, yes it is

Only days before Ferguson's nightmare scenario made Jackboot Johnson take Britain into a China-style lockdown to save us from a deadly 'virus' the UK government website gov.uk was reporting something very different to Ferguson on a page of official government guidance for 'high consequence infectious diseases (HCID)'. It said this about 'Covid-19':

As of 19 March 2020, COVID-19 *is no longer considered to be a high consequence infectious diseases (HCID) in the UK* [my emphasis]. The 4 nations public health HCID group made an interim recommendation in January 2020 to classify COVID-19 as an HCID. This was based on consideration of the UK HCID criteria about the virus and the disease with information available during the early stages of the outbreak.

Now that more is known about COVID-19, the public health bodies in the UK have reviewed the most up to date information about COVID-19 against the UK HCID criteria. They have determined that several features have now changed; in particular, more information is available about mortality rates (low overall), and there is now greater clinical awareness and a specific and sensitive laboratory test, the availability of which continues to increase. The Advisory Committee on Dangerous Pathogens (ACDP) is also of the opinion that COVID-19 should no longer be classified as an HCID.

Soon after the government had been exposed for downgrading the risk they upgraded it again and everyone was back to singing from the same Cult hymn book. Ferguson and his fellow Gates clones indicated that lockdowns and restrictions would have to continue until a Gates-funded vaccine was developed. Gates said the same because Ferguson and his like were repeating the Gates script which is the Cult script. 'Flatten the curve' became an ongoing nightmare of continuing lockdowns with periods in between of severe restrictions in pursuit of destroying independent incomes and had nothing to do with protecting health about which the Cult gives not a shit. Why wouldn't Ferguson be pushing a vaccine 'solution' when he's owned by vaccine-obsessive Gates who makes a fortune from them and

when Ferguson heads the Vaccine Impact Modelling Consortium at Imperial College funded by the Gates Foundation and GAVI, the ‘vaccine alliance’, created by Gates as his personal vaccine promotion operation? To compound the human catastrophe that Ferguson’s ‘models’ did so much to create he was later exposed for breaking his own lockdown rules by having sexual liaisons with his married girlfriend Antonia Staats at his home while she was living at another location with her husband and children. Staats was a ‘climate’ activist and senior campaigner at the Soros-funded Avaaz which I wouldn’t trust to tell me that grass is green. Ferguson had to resign as a government advisor over this hypocrisy in May, 2020, but after a period of quiet he was back being quoted by the ridiculous media on the need for more lockdowns and a vaccine rollout. Other government-advising ‘scientists’ from Imperial College held the fort in his absence and said lockdown could be indefinite until a vaccine was found. The Cult script was being sung by the payrolled choir. I said there was no intention of going back to ‘normal’ when the ‘vaccine’ came because the ‘vaccine’ is part of a very different agenda that I will discuss in Human 2.0. Why would the Cult want to let the world go back to normal when destroying that normal forever was the whole point of what was happening? House arrest, closing businesses and schools through lockdown, (un)social distancing and masks all followed the Ferguson fantasy models. Again as I predicted (these people are so predictable) when the ‘vaccine’ arrived we were told that house arrest, lockdown, (un)social distancing and masks would still have to continue. I will deal with the masks in the next chapter because they are of fundamental importance.

Where's the 'pandemic'?

Any mildly in-depth assessment of the figures revealed what was really going on. Cult-funded and controlled organisations still have genuine people working within them such is the number involved. So it is with Genevieve Briand, assistant program director of the Applied Economics master’s degree program at Johns Hopkins

University. She analysed the impact that 'Covid-19' had on deaths from *all* causes in the United States using official data from the CDC for the period from early February to early September, 2020. She found that allegedly 'Covid' *related*-deaths exceeded those from heart disease which she found strange with heart disease always the biggest cause of fatalities. Her research became even more significant when she noted the sudden decline in 2020 of *all* non-'Covid' deaths: 'This trend is completely contrary to the pattern observed in all previous years ... the total decrease in deaths by other causes almost exactly equals the increase in deaths by Covid-19.' This was such a game, set and match in terms of what was happening that Johns Hopkins University deleted the article on the grounds that it 'was being used to support false and dangerous inaccuracies about the impact of the pandemic'. No – because it exposed the scam from official CDC figures and this was confirmed when those figures were published in January, 2021. Here we can see the effect of people dying from heart attacks, cancer, road accidents and gunshot wounds – *anything* – having 'Covid-19' on the death certificate along with those diagnosed from 'symptoms' who had even not tested positive with a test not testing for the 'virus'. I am not kidding with the gunshot wounds, by the way. Brenda Bock, coroner in Grand County, Colorado, revealed that two gunshot victims tested positive for the 'virus' within the previous 30 days and were therefore classified as 'Covid deaths'. Bock said: 'These two people had tested positive for Covid, but that's not what killed them. A gunshot wound is what killed them.' She said she had not even finished her investigation when the state listed the gunshot victims as deaths due to the 'virus'. The death and case figures for 'Covid-19' are an absolute joke and yet they are repeated like parrots by the media, politicians and alleged medical 'experts'. The official Cult narrative is the only show in town.

Genevieve Briand found that deaths from all causes were not exceptional in 2020 compared with previous years and a Spanish magazine published figures that said the same about Spain which was a 'Covid' propaganda hotspot at one point. *Discovery Salud*, a

health and medicine magazine, quoted government figures which showed how 17,000 *fewer* people died in Spain in 2020 than in 2019 and more than 26,000 fewer than in 2018. The age-standardised mortality rate for England and Wales when age distribution is taken into account was significantly lower in 2020 than the 1970s, 80s and 90s, and was only the ninth highest since 2000. Where is the ‘pandemic’?

Post mortems and autopsies virtually disappeared for ‘Covid’ deaths amid claims that ‘virus-infected’ bodily fluids posed a risk to those carrying out the autopsy. This was rejected by renowned German pathologist and forensic doctor Klaus Püschel who said that he and his staff had by then done 150 autopsies on ‘Covid’ patients with no problems at all. He said they were needed to know why some ‘Covid’ patients suffered blood clots and not severe respiratory infections. The ‘virus’ is, after all, called SARS or ‘severe acute respiratory syndrome’. I highlighted in the spring of 2020 this phenomenon and quoted New York intensive care doctor Cameron Kyle-Sidell who posted a soon deleted YouTube video to say that they had been told to prepare to treat an infectious disease called ‘Covid-19’, but that was not what they were dealing with. Instead he likened the lung condition of the most severely ill patients to what you would expect with cabin depressurisation in a plane at 30,000 feet or someone dropped on the top of Everest without oxygen or acclimatisation. I have never said this is not happening to a small minority of alleged ‘Covid’ patients – I am saying this is not caused by a phantom ‘contagious virus’. Indeed Kyle-Sidell said that ‘Covid-19’ was not the disease they were told was coming their way. ‘We are operating under a medical paradigm that is untrue,’ he said, and he believed they were treating the wrong disease: ‘These people are being slowly starved of oxygen.’ Patients would take off their oxygen masks in a state of fear and stress and while they were blue in the face on the brink of death. They did not look like patients dying of pneumonia. You can see why they don’t want autopsies when their virus doesn’t exist and there is another condition in some people that they don’t wish to be uncovered. I should add here that

the 5G system of millimetre waves was being rapidly introduced around the world in 2020 and even more so now as they fire 5G at the Earth from satellites. At 60 gigahertz within the 5G range that frequency interacts with the oxygen molecule and stops people breathing in sufficient oxygen to be absorbed into the bloodstream. They are installing 5G in schools and hospitals. The world is not mad or anything. 5G can cause major changes to the lungs and blood as I detail in *The Answer* and these consequences are labelled 'Covid-19', the alleged symptoms of which can be caused by 5G and other electromagnetic frequencies as cells respond to radiation poisoning.

The 'Covid death' scam

Dr Scott Jensen, a Minnesota state senator and medical doctor, exposed 'Covid' Medicare payment incentives to hospitals and death certificate manipulation. He said he was sent a seven-page document by the US Department of Health 'coaching' him on how to fill out death certificates which had never happened before. The document said that he didn't need to have a laboratory test for 'Covid-19' to put that on the death certificate and that shocked him when death certificates are supposed to be about facts. Jensen described how doctors had been 'encouraged, if not pressured' to make a diagnosis of 'Covid-19' if they thought it was probable or '*presumed*'. No positive test was necessary – not that this would have mattered anyway. He said doctors were told to diagnose 'Covid' by symptoms when these were the same as colds, allergies, other respiratory problems, and certainly with influenza which 'disappeared' in the 'Covid' era. A common sniffle was enough to get the dreaded verdict. Ontario authorities decreed that a single care home resident with *one* symptom from a long list must lead to the isolation of the entire home. Other courageous doctors like Jensen made the same point about death figure manipulation and how deaths by other causes were falling while 'Covid-19 deaths' were rising at the same rate due to re-diagnosis. Their videos rarely survive long on YouTube with its Cult-supporting algorithms courtesy of CEO Susan Wojcicki and her bosses at Google. Figure-tampering was so glaring

and ubiquitous that even officials were letting it slip or outright saying it. UK chief scientific adviser Patrick Vallance said on one occasion that ‘Covid’ on the death certificate doesn’t mean ‘Covid’ was the cause of death (so why the hell is it there?) and we had the rare sight of a BBC reporter telling the truth when she said: ‘Someone could be successfully treated for Covid, in say April, discharged, and then in June, get run over by a bus and die ... That person would still be counted as a Covid death in England.’ Yet the BBC and the rest of the world media went on repeating the case and death figures as if they were real. Illinois Public Health Director Dr Ngozi Ezike revealed the deceit while her bosses must have been clenching their buttocks:

If you were in a hospice and given a few weeks to live and you were then found to have Covid that would be counted as a Covid death. [There might be] a clear alternate cause, but it is still listed as a Covid death. So everyone listed as a Covid death doesn’t mean that was the cause of the death, but that they had Covid at the time of death.

Yes, a ‘Covid virus’ never shown to exist and tested for with a test not testing for the ‘virus’. In the first period of the pandemic hoax through the spring of 2020 the process began of designating almost everything a ‘Covid’ death and this has continued ever since. I sat in a restaurant one night listening to a loud conversation on the next table where a family was discussing in bewilderment how a relative who had no symptoms of ‘Covid’, and had died of a long-term problem, could have been diagnosed a death by the ‘virus’. I could understand their bewilderment. If they read this book they will know why this medical fraud has been perpetrated the world over.

Some media truth shock

The media ignored the evidence of death certificate fraud until eventually one columnist did speak out when she saw it first-hand. Bel Mooney is a long-time national newspaper journalist in Britain currently working for the *Daily Mail*. Her article on February 19th, 2021, carried this headline: ‘My dad Ted passed three Covid tests

and died of a chronic illness yet he's officially one of Britain's 120,000 victims of the virus and is far from alone ... so how many more are there?' She told how her 99-year-old father was in a care home with a long-standing chronic obstructive pulmonary disease and vascular dementia. Maybe, but he was still aware enough to tell her from the start that there was no 'virus' and he refused the 'vaccine' for that reason. His death was not unexpected given his chronic health problems and Mooney said she was shocked to find that 'Covid-19' was declared the cause of death on his death certificate. She said this was a 'bizarre and unacceptable untruth' for a man with long-time health problems who had tested negative twice at the home for the 'virus'. I was also shocked by this story although not by what she said. I had been highlighting the death certificate manipulation for ten months. It was the confirmation that a professional full-time journalist only realised this was going on when it affected her directly and neither did she know that whether her dad tested positive or negative was irrelevant with the test not testing for the 'virus'. Where had she been? She said she did not believe in 'conspiracy theories' without knowing I'm sure that this and 'conspiracy theorists' were terms put into widespread circulation by the CIA in the 1960s to discredit those who did not accept the ridiculous official story of the Kennedy assassination. A blanket statement of 'I don't believe in conspiracy theories' is always bizarre. The dictionary definition of the term alone means the world is drowning in conspiracies. What she said was even more daft when her dad had just been affected by the 'Covid' conspiracy. Why else does she think that 'Covid-19' was going on the death certificates of people who died of something else?

To be fair once she saw from personal experience what was happening she didn't mince words. Mooney was called by the care home on the morning of February 9th to be told her father had died in his sleep. When she asked for the official cause of death what came back was 'Covid-19'. Mooney challenged this and was told there had been deaths from Covid on the dementia floor (confirmed by a test not testing for the 'virus') so they considered it 'reasonable

to assume'. 'But doctor,' Mooney rightly protested, 'an assumption isn't a diagnosis.' She said she didn't blame the perfectly decent and sympathetic doctor – 'he was just doing his job'. Sorry, but that's *bullshit*. He wasn't doing his job at all. He was putting a false cause of death on the death certificate and that is a criminal offence for which he should be brought to account and the same with the millions of doctors worldwide who have done the same. They were not doing their job they were following orders and that must not wash at new Nuremberg trials any more than it did at the first ones. Mooney's doctor was 'assuming' (presuming) as he was told to, but 'just following orders' makes no difference to his actions. A doctor's job is to serve the patient and the truth, not follow orders, but that's what they have done all over the world and played a central part in making the 'Covid' hoax possible with all its catastrophic consequences for humanity. Shame on them and they must answer for their actions. Mooney said her disquiet worsened when she registered her father's death by telephone and was told by the registrar there had been very many other cases like hers where 'the deceased' had not tested positive for 'Covid' yet it was recorded as the cause of death. The test may not matter, but those involved at their level *think* it matters and it shows a callous disregard for accurate diagnosis. The pressure to do this is coming from the top of the national 'health' pyramids which in turn obey the World Health Organization which obeys Gates and the Cult. Mooney said the registrar agreed that this must distort the national figures adding that 'the strangest thing is that every winter we record countless deaths from flu, and this winter there have been none. Not one!' She asked if the registrar thought deaths from flu were being misdiagnosed and lumped together with 'Covid' deaths. The answer was a 'puzzled yes'. Mooney said that the funeral director said the same about 'Covid' deaths which had nothing to do with 'Covid'. They had lost count of the number of families upset by this and other funeral companies in different countries have had the same experience. Mooney wrote:

The nightly shroud-waving and shocking close-ups of pain imposed on us by the TV news bewildered and terrified the population into eager compliance with lockdowns. We were invited to ‘save the NHS’ and to grieve for strangers – the real-life loved ones behind those shocking death counts. Why would the public imagine what I now fear, namely that the way Covid-19 death statistics are compiled might make the numbers seem greater than they are?

Oh, just a little bit – like 100 percent.

Do the maths

Mooney asked why a country would wish to skew its mortality figures by wrongly certifying deaths? What had been going on? Well, if you don’t believe in conspiracies you will never find the answer which is that *it’s a conspiracy*. She did, however, describe what she had discovered as a ‘national scandal’. In reality it’s a global scandal and happening everywhere. Pillars of this conspiracy were all put into place before the button was pressed with the Drosten PCR protocol and high amplifications to produce the cases and death certificate changes to secure illusory ‘Covid’ deaths.

Mooney notes that normally two doctors were needed to certify a death, with one having to know the patient, and how the rules were changed in the spring of 2020 to allow one doctor to do this. In the same period ‘Covid deaths’ were decreed to be all cases where Covid-19 was put on the death certificate even without a positive test or any symptoms. Mooney asked: ‘How many of the 30,851 (as of January 15) care home resident deaths with Covid-19 on the certificate (32.4 per cent of all deaths so far) were based on an assumption, like that of my father? And what has that done to our national psyche?’ All of them is the answer to the first question and it has devastated and dismantled the national psyche, actually the global psyche, on a colossal scale. In the UK case and death data is compiled by organisations like Public Health England (PHE) and the Office for National Statistics (ONS). Mooney highlights the insane policy of counting a death from any cause as ‘Covid-19’ if this happens within 28 days of a positive test (with a test not testing for the ‘virus’) and she points out that ONS statistics reflect deaths ‘involving Covid’ ‘or due to Covid’ which meant in practice any

death where ‘Covid-19’ was mentioned on the death certificate. She described the consequences of this fraud:

Most people will accept the narrative they are fed, so panicky governments here and in Europe witnessed the harsh measures enacted in totalitarian China and jumped into lockdown. Headlines about Covid deaths tolled like the knell that would bring doomsday to us all. Fear stalked our empty streets. Politicians parroted the frankly ridiculous aim of ‘zero Covid’ and shut down the economy, while most British people agreed that lockdown was essential and (astonishingly to me, as a patriotic Brit) even wanted more restrictions.

For what? Lies on death certificates? Never mind the grim toll of lives ruined, suicides, schools closed, rising inequality, depression, cancelled hospital treatments, cancer patients in a torture of waiting, poverty, economic devastation, loneliness, families kept apart, and so on. How many lives have been lost as a direct result of lockdown?

She said that we could join in a national chorus of shock and horror at reaching the 120,000 death toll which was surely certain to have been totally skewed all along, but what about the human cost of lockdown justified by these ‘death figures’? *The British Medical Journal* had reported a 1,493 percent increase in cases of children taken to Great Ormond Street Hospital with abusive head injuries alone and then there was the effect on families:

Perhaps the most shocking thing about all this is that families have been kept apart – and obeyed the most irrational, changing rules at the whim of government – because they believed in the statistics. They succumbed to fear, which his generation rejected in that war fought for freedom. Dad (God rest his soul) would be angry. And so am I.

Another theme to watch is that in the winter months when there are more deaths from all causes they focus on ‘Covid’ deaths and in the summer when the British Lung Foundation says respiratory disease plummets by 80 percent they rage on about ‘cases’. Either way fascism on population is always the answer.

Nazi eugenics in the 21st century

Elderly people in care homes have been isolated from their families month after lonely month with no contact with relatives and grandchildren who were banned from seeing them. We were told

that lockdown fascism was to ‘protect the vulnerable’ like elderly people. At the same time Do Not Resuscitate (DNR) orders were placed on their medical files so that if they needed resuscitation it wasn’t done and ‘Covid-19’ went on their death certificates. Old people were not being ‘protected’ they were being culled – murdered in truth. DNR orders were being decreed for disabled and young people with learning difficulties or psychological problems. The UK Care Quality Commission, a non-departmental body of the Department of Health and Social Care, found that 34 percent of those working in health and social care were pressured into placing ‘do not attempt cardiopulmonary resuscitation’ orders on ‘Covid’ patients who suffered from disabilities and learning difficulties without involving the patient or their families in the decision. UK judges ruled that an elderly woman with dementia should have the DNA-manipulating ‘Covid vaccine’ against her son’s wishes and that a man with severe learning difficulties should have the jab despite his family’s objections. Never mind that many had already died. The judiciary always supports doctors and government in fascist dictatorships. They wouldn’t dare do otherwise. A horrific video was posted showing fascist officers from Los Angeles police forcibly giving the ‘Covid’ shot to women with special needs who were screaming that they didn’t want it. The same fascists are seen giving the jab to a sleeping elderly woman in a care home. This is straight out of the Nazi playbook. Hitler’s Nazis committed mass murder of the mentally ill and physically disabled throughout Germany and occupied territories in the programme that became known as Aktion T4, or just T4. Sabbatian-controlled Hitler and his grotesque crazies set out to kill those they considered useless and unnecessary. The Reich Committee for the Scientific Registering of Hereditary and Congenital Illnesses registered the births of babies identified by physicians to have ‘defects’. By 1941 alone more than 5,000 children were murdered by the state and it is estimated that in total the number of innocent people killed in Aktion T4 was between 275,000 and 300,000. Parents were told their children had been sent away for ‘special treatment’ never to return. It is rather pathetic to see claims about plans for new extermination camps being dismissed today

when the same force behind current events did precisely that 80 years ago. Margaret Sanger was a Cult operative who used 'birth control' to sanitise her programme of eugenics. Organisations she founded became what is now Planned Parenthood. Sanger proposed that 'the whole dysgenic population would have its choice of segregation or sterilization'. These included epileptics, 'feeble-minded', and prostitutes. Sanger opposed charity because it perpetuated 'human waste'. She reveals the Cult mentality and if anyone thinks that extermination camps are a 'conspiracy theory' their naivety is touching if breathtakingly stupid.

If you don't believe that doctors can act with callous disregard for their patients it is worth considering that doctors and medical staff agreed to put government-decreed DNR orders on medical files and do nothing when resuscitation is called for. I don't know what you call such people in your house. In mine they are Nazis from the Josef Mengele School of Medicine. Phenomenal numbers of old people have died worldwide from the effects of lockdown, depression, lack of treatment, the 'vaccine' (more later) and losing the will to live. A common response at the start of the manufactured pandemic was to remove old people from hospital beds and transfer them to nursing homes. The decision would result in a mass cull of elderly people in those homes through lack of treatment – *not* 'Covid'. Care home whistleblowers have told how once the 'Covid' era began doctors would not come to their homes to treat patients and they were begging for drugs like antibiotics that often never came. The most infamous example was ordered by New York governor Andrew Cuomo, brother of a moronic CNN host, who amazingly was given an Emmy Award for his handling of the 'Covid crisis' by the ridiculous Wokers that hand them out. Just how ridiculous could be seen in February, 2021, when a Department of Justice and FBI investigation began into how thousands of old people in New York died in nursing homes after being discharged from hospital to make way for 'Covid' patients on Cuomo's say-so – and how he and his staff covered up these facts. This couldn't have happened to a nicer psychopath. Even then there was a 'Covid' spin. Reports said that

thousands of old people who tested positive for ‘Covid’ in hospital were transferred to nursing homes to both die of ‘Covid’ and transmit it to others. No – they were in hospital because they were ill and the fact that they tested positive with a test not testing for the ‘virus’ is irrelevant. They were ill often with respiratory diseases ubiquitous in old people near the end of their lives. Their transfer out of hospital meant that their treatment stopped and many would go on to die.

They're old. Who gives a damn?

I have exposed in the books for decades the Cult plan to cull the world’s old people and even to introduce at some point what they call a ‘demise pill’ which at a certain age everyone would take and be out of here by law. In March, 2021, Spain legalised euthanasia and assisted suicide following the Netherlands, Belgium, Luxembourg and Canada on the Tiptoe to the demise pill. Treatment of old people by many ‘care’ homes has been a disgrace in the ‘Covid’ era. There are many, many, caring staff – I know some. There have, however, been legions of stories about callous treatment of old people and their families. Police were called when families came to take their loved ones home in the light of isolation that was killing them. They became prisoners of the state. Care home residents in insane, fascist Ontario, Canada, were not allowed to leave their *room* once the ‘Covid’ hoax began. UK staff have even wheeled elderly people away from windows where family members were talking with them. Oriana Criscuolo from Stockport in the English North West dropped off some things for her 80-year-old father who has Parkinson’s disease and dementia and she wanted to wave to him through a ground-floor window. She was told that was ‘illegal’. When she went anyway they closed the curtains in the middle of the day. Oriana said:

It’s just unbelievable. I cannot understand how care home staff – people who are being paid to care – have become so uncaring. Their behaviour is inhumane and cruel. It’s beyond belief.

She was right and this was not a one-off. What a way to end your life in such loveless circumstances. UK registered nurse Nicky Millen, a proper old school nurse for 40 years, said that when she started her career care was based on dignity, choice, compassion and empathy. Now she said ‘the things that are important to me have gone out of the window.’ She was appalled that people were dying without their loved ones and saying goodbye on iPads. Nicky described how a distressed 89-year-old lady stroked her face and asked her ‘how many paracetamol would it take to finish me off’. Life was no longer worth living while not seeing her family. Nicky said she was humiliated in front of the ward staff and patients for letting the lady stroke her face and giving her a cuddle. Such is the dehumanisation that the ‘Covid’ hoax has brought to the surface. Nicky worked in care homes where patients told her they were being held prisoner. ‘I want to live until I die’, one said to her. ‘I had a lady in tears because she hadn’t seen her great-grandson.’ Nicky was compassionate old school meeting psychopathic New Normal. She also said she had worked on a ‘Covid’ ward with no ‘Covid’ patients. Jewish writer Shai Held wrote an article in March, 2020, which was headlined ‘The Staggering, Heartless Cruelty Toward the Elderly’. What he described was happening from the earliest days of lockdown. He said ‘the elderly’ were considered a group and not unique individuals (the way of the Woke). Shai Held said:

Notice how the all-too-familiar rhetoric of dehumanization works: ‘The elderly’ are bunched together as a faceless mass, all of them considered culprits and thus effectively deserving of the suffering the pandemic will inflict upon them. Lost entirely is the fact that the elderly are individual human beings, each with a distinctive face and voice, each with hopes and dreams, memories and regrets, friendships and marriages, loves lost and loves sustained.

‘The elderly’ have become another dehumanised group for which anything goes and for many that has resulted in cold disregard for their rights and their life. The distinctive face that Held talks about is designed to be deleted by masks until everyone is part of a faceless mass.

'War-zone' hospitals myth

Again and again medical professionals have told me what was really going on and how hospitals 'overrun like war zones' according to the media were virtually empty. The mantra from medical whistleblowers was please don't use my name or my career is over. Citizen journalists around the world sneaked into hospitals to film evidence exposing the 'war-zone' lie. They really *were* largely empty with closed wards and operating theatres. I met a hospital worker in my town on the Isle of Wight during the first lockdown in 2020 who said the only island hospital had never been so quiet. Lockdown was justified by the psychopaths to stop hospitals being overrun. At the same time that the island hospital was near-empty the military arrived here to provide *extra beds*. It was all propaganda to ramp up the fear to ensure compliance with fascism as were never-used temporary hospitals with thousands of beds known as Nightingales and never-used make-shift mortuaries opened by the criminal UK government. A man who helped to install those extra island beds attributed to the army said they were never used and the hospital was empty. Doctors and nurses 'stood around talking or on their phones, wandering down to us to see what we were doing'. There were no masks or social distancing. He accused the useless local island paper, the *County Press*, of 'pumping the fear as if our hospital was overrun and we only have one so it should have been'. He described ambulances parked up with crews outside in deck chairs. When his brother called an ambulance he was told there was a two-hour backlog which he called 'bullshit'. An old lady on the island fell 'and was in a bad way', but a caller who rang for an ambulance was told the situation wasn't urgent enough. Ambulance stations were working under capacity while people would hear ambulances with sirens blaring driving through the streets. When those living near the stations realised what was going on they would follow them as they left, circulated around an urban area with the sirens going, and then came back without stopping. All this was to increase levels of fear and the same goes for the 'ventilator shortage crisis' that cost tens of millions for hastily produced ventilators never to be used.

Ambulance crews that agreed to be exploited in this way for fear propaganda might find themselves a mirror. I wish them well with that. Empty hospitals were the obvious consequence of treatment and diagnoses of non-'Covid' conditions cancelled and those involved handed a death sentence. People have been dying at home from undiagnosed and untreated cancer, heart disease and other life-threatening conditions to allow empty hospitals to deal with a 'pandemic' that wasn't happening.

Death of the innocent

'War-zones' have been laying off nursing staff, even doctors where they can. There was no work for them. Lockdown was justified by saving lives and protecting the vulnerable they were actually killing with DNR orders and preventing empty hospitals being 'overrun'. In Britain the mantra of stay at home to 'save the NHS' was everywhere and across the world the same story was being sold when it was all lies. Two California doctors, Dan Erickson and Artin Massihi at Accelerated Urgent Care in Bakersfield, held a news conference in April, 2020, to say that intensive care units in California were 'empty, essentially', with hospitals shutting floors, not treating patients and laying off doctors. The California health system was working at minimum capacity 'getting rid of doctors because we just don't have the volume'. They said that people with conditions such as heart disease and cancer were not coming to hospital out of fear of 'Covid-19'. Their video was deleted by Susan Wojcicki's Cult-owned YouTube after reaching five million views. Florida governor Ron Desantis, who rejected the severe lockdowns of other states and is being targeted for doing so, said that in March, 2020, every US governor was given models claiming they would run out of hospital beds in days. That was never going to happen and the 'modellers' knew it. Deceit can be found at every level of the system. Urgent children's operations were cancelled including fracture repairs and biopsies to spot cancer. Eric Nicholls, a consultant paediatrician, said 'this is obviously concerning and we need to return to normal operating and to increase capacity as soon as possible'. Psychopaths

in power were rather less concerned *because* they are psychopaths. Deletion of urgent care and diagnosis has been happening all over the world and how many kids and others have died as a result of the actions of these cold and heartless lunatics dictating ‘health’ policy? The number must be stratospheric. Richard Sullivan, professor of cancer and global health at King’s College London, said people feared ‘Covid’ more than cancer such was the campaign of fear. ‘Years of lost life will be quite dramatic’, Sullivan said, with ‘a huge amount of avoidable mortality’. Sarah Woolnough, executive director for policy at Cancer Research UK, said there had been a 75 percent drop in urgent referrals to hospitals by family doctors of people with suspected cancer. Sullivan said that ‘a lot of services have had to scale back – we’ve seen a dramatic decrease in the amount of elective cancer surgery’. Lockdown deaths worldwide has been absolutely fantastic with the *New York Post* reporting how data confirmed that ‘lockdowns end more lives than they save’:

There was a sharp decline in visits to emergency rooms and an increase in fatal heart attacks because patients didn’t receive prompt treatment. Many fewer people were screened for cancer. Social isolation contributed to excess deaths from dementia and Alzheimer’s.

Researchers predicted that the social and economic upheaval would lead to tens of thousands of “deaths of despair” from drug overdoses, alcoholism and suicide. As unemployment surged and mental-health and substance-abuse treatment programs were interrupted, the reported levels of anxiety, depression and suicidal thoughts increased dramatically, as did alcohol sales and fatal drug overdoses.

This has been happening while nurses and other staff had so much time on their hands in the ‘war-zones’ that Tic-Tok dancing videos began appearing across the Internet with medical staff dancing around in empty wards and corridors as people died at home from causes that would normally have been treated in hospital.

Mentions in dispatches

One brave and truth-committed whistleblower was Louise Hampton, a call handler with the UK NHS who made a viral Internet video saying she had done ‘fuck all’ during the ‘pandemic’

which was ‘a load of bollocks’. She said that ‘Covid-19’ was rebranded flu and of course she lost her job. This is what happens in the medical and endless other professions now when you tell the truth. Louise filmed inside ‘war-zone’ accident and emergency departments to show they were empty and I mean *empty* as in no one there. The mainstream media could have done the same and blown the gaff on the whole conspiracy. They haven’t to their eternal shame. Not that most ‘journalists’ seem capable of manifesting shame as with the psychopaths they slavishly repeat without question. The relative few who were admitted with serious health problems were left to die alone with no loved ones allowed to see them because of ‘Covid’ rules and they included kids dying without the comfort of mum and dad at their bedside while the evil behind this couldn’t give a damn. It was all good fun to them. A Scottish NHS staff nurse publicly quit in the spring of 2021 saying: ‘I can no longer be part of the lies and the corruption by the government.’ She said hospitals ‘aren’t full, the beds aren’t full, beds have been shut, wards have been shut’. Hospitals were never busy throughout ‘Covid’. The staff nurse said that Nicola Sturgeon, tragically the leader of the Scottish government, was on television saying save the hospitals and the NHS – ‘but the beds are empty’ and ‘we’ve not seen flu, we always see flu every year’. She wrote to government and spoke with her union Unison (the unions are Cult-compromised and *useless*, but nothing changed. Many of her colleagues were scared of losing their jobs if they spoke out as they wanted to. She said nursing staff were being affected by wearing masks all day and ‘my head is splitting every shift from wearing a mask’. The NHS is part of the fascist tyranny and must be dismantled so we can start again with human beings in charge. (Ironically, hospitals were reported to be busier again when official ‘Covid’ cases *fell* in spring/summer of 2021 and many other conditions required treatment at the same time as *the fake vaccine rollout*.)

I will cover the ‘Covid vaccine’ scam in detail later, but it is another indicator of the sickening disregard for human life that I am highlighting here. The DNA-manipulating concoctions do not fulfil

the definition of a ‘vaccine’, have never been used on humans before and were given only emergency approval because trials were not completed and they continued using the unknowing public. The result was what a NHS senior nurse with responsibility for ‘vaccine’ procedure said was ‘genocide’. She said the ‘vaccines’ were not ‘vaccines’. They had not been shown to be safe and claims about their effectiveness by drug companies were ‘poetic licence’. She described what was happening as a ‘horrid act of human annihilation’. The nurse said that management had instigated a policy of not providing a Patient Information Leaflet (PIL) before people were ‘vaccinated’ even though health care professionals are supposed to do this according to protocol. Patients should also be told that they are taking part in an ongoing clinical trial. Her challenges to what is happening had seen her excluded from meetings and ridiculed in others. She said she was told to ‘watch my step … or I would find myself surplus to requirements’. The nurse, who spoke anonymously in fear of her career, said she asked her NHS manager why he/she was content with taking part in genocide against those having the ‘vaccines’. The reply was that everyone had to play their part and to ‘put up, shut up, and get it done’. Government was ‘leaning heavily’ on NHS management which was clearly leaning heavily on staff. This is how the global ‘medical’ hierarchy operates and it starts with the Cult and its World Health Organization.

She told the story of a doctor who had the Pfizer jab and when questioned had no idea what was in it. The doctor had never read the literature. We have to stop treating doctors as intellectual giants when so many are moral and medical pygmies. The doctor did not even know that the ‘vaccines’ were not fully approved or that their trials were ongoing. They were, however, asking their patients if they minded taking part in follow-ups for research purposes – yes, the *ongoing clinical trial*. The nurse said the doctor’s ignorance was not rare and she had spoken to a hospital consultant who had the jab without any idea of the background or that the ‘trials’ had not been completed. Nurses and pharmacists had shown the same ignorance.

'My NHS colleagues have forsaken their duty of care, broken their code of conduct – Hippocratic Oath – and have been brainwashed just the same as the majority of the UK public through propaganda ...' She said she had not been able to recruit a single NHS colleague, doctor, nurse or pharmacist to stand with her and speak out. Her union had refused to help. She said that if the genocide came to light she would not hesitate to give evidence at a Nuremberg-type trial against those in power who could have affected the outcomes but didn't.

And all for what?

To put the nonsense into perspective let's say the 'virus' does exist and let's go completely crazy and accept that the official manipulated figures for cases and deaths are accurate. *Even then* a study by Stanford University epidemiologist Dr John Ioannidis published on the World Health Organization website produced an average infection to fatality rate of ... 0.23 percent! Ioannidis said: 'If one could sample equally from all locations globally, the median infection fatality rate might even be substantially lower than the 0.23% observed in my analysis.' For healthy people under 70 it was ... 0.05 percent! This compares with the 3.4 percent claimed by the Cult-owned World Health Organization when the hoax was first played and maximum fear needed to be generated. An updated Stanford study in April, 2021, put the 'infection' to 'fatality' rate at just 0.15 percent. Another team of scientists led by Megan O'Driscoll and Henrik Salje studied data from 45 countries and published their findings on the Nature website. For children and young people the figure is so small it virtually does not register although authorities will be hyping dangers to the young when they introduce DNA-manipulating 'vaccines' for children. The O'Driscoll study produced an average infection-fatality figure of 0.003 for children from birth to four; 0.001 for 5 to 14; 0.003 for 15 to 19; and it was still only 0.456 up to 64. To claim that children must be 'vaccinated' to protect them from 'Covid' is an obvious lie and so there must be another reason and there is. What's more the average age of a 'Covid' death is akin

to the average age that people die in general. The average age of death in England is about 80 for men and 83 for women. The average age of death from alleged 'Covid' is between 82 and 83. California doctors, Dan Erickson and Artin Massihi, said at their April media conference that projection models of millions of deaths had been 'woefully inaccurate'. They produced detailed figures showing that Californians had a 0.03 chance of dying from 'Covid' based on the number of people who tested positive (with a test not testing for the 'virus'). Erickson said there was a 0.1 percent chance of dying from 'Covid' in the *state* of New York, not just the city, and a 0.05 percent chance in Spain, a centre of 'Covid-19' hysteria at one stage. The Stanford studies supported the doctors' data with fatality rate estimates of 0.23 and 0.15 percent. How close are these figures to my estimate of *zero*? Death-rate figures claimed by the World Health Organization at the start of the hoax were some 15 times higher. The California doctors said there was no justification for lockdowns and the economic devastation they caused. Everything they had ever learned about quarantine was that you quarantine the *sick* and not the healthy. They had never seen this before and it made no medical sense.

Why in the light of all this would governments and medical systems the world over say that billions must go under house arrest; lose their livelihood; in many cases lose their mind, their health and their life; force people to wear masks dangerous to health and psychology; make human interaction and even family interaction a criminal offence; ban travel; close restaurants, bars, watching live sport, concerts, theatre, and any activity involving human togetherness and discourse; and closing schools to isolate children from their friends and cause many to commit suicide in acts of hopelessness and despair? The California doctors said lockdown consequences included increased child abuse, partner abuse, alcoholism, depression, and other impacts they were seeing every day. Who would do that to the entire human race if not mentally-ill psychopaths of almost unimaginable extremes like Bill Gates? We must face the reality of what we are dealing with and come out of

denial. Fascism and tyranny are made possible only by the target population submitting and acquiescing to fascism and tyranny. The whole of human history shows that to be true. Most people naively and unquestioning believed what they were told about a ‘deadly virus’ and meekly and weakly submitted to house arrest. Those who didn’t believe it – at least in total – still submitted in fear of the consequences of not doing so. For the rest who wouldn’t submit draconian fines have been imposed, brutal policing by psychopaths *for* psychopaths, and condemnation from the meek and weak who condemn the Pushbackers on behalf of the very force that has them, too, in its gunsights. ‘Pathetic’ does not even begin to suffice.

Britain’s brainless ‘Health’ Secretary Matt Hancock warned anyone lying to border officials about returning from a list of ‘hotspot’ countries could face a jail sentence of up to ten years which is more than for racially-aggravated assault, incest and attempting to have sex with a child under 13. Hancock is a lunatic, but he has the state apparatus behind him in a Cult-led chain reaction and the same with UK ‘Vaccine Minister’ Nadhim Zahawi, a prominent member of the mega-Cult secret society, Le Cercle, which featured in my earlier books. The Cult enforces its will on governments and medical systems; government and medical systems enforce their will on business and police; business enforces its will on staff who enforce it on customers; police enforce the will of the Cult on the population and play their essential part in creating a world of fascist control that their own children and grandchildren will have to live in their entire lives. It is a hierarchical pyramid of imposition and acquiescence and, yes indeedy, of clinical insanity.

Does anyone bright enough to read this book have to ask what the answer is? I think not, but I will reveal it anyway in the fewest of syllables: Tell the psychos and their moronic lackeys to fuck off and let’s get on with our lives. We are many – They are few.

CHAPTER SEVEN

War on your mind

One believes things because one has been conditioned to believe them

Aldous Huxley, *Brave New World*

I have described the ‘Covid’ hoax as a ‘Psyop’ and that is true in every sense and on every level in accordance with the definition of that term which is psychological warfare. Break down the ‘Covid pandemic’ to the foundation themes and it is psychological warfare on the human individual and collective mind.

The same can be said for the entire human belief system involving every subject you can imagine. Huxley was right in his contention that people believe what they are conditioned to believe and this comes from the repetition throughout their lives of the same falsehoods. They spew from government, corporations, media and endless streams of ‘experts’ telling you what the Cult wants you to believe and often believing it themselves (although *far* from always). ‘Experts’ are rewarded with ‘prestigious’ jobs and titles and as agents of perceptual programming with regular access to the media. The Cult has to control the narrative – control *information* – or they lose control of the vital, crucial, without-which-they-cannot-prevail public perception of reality. The foundation of that control today is the Internet made possible by the Defense Advanced Research Projects Agency (DARPA), the incredibly sinister technological arm of the Pentagon. The Internet is the result of military technology.

DARPA openly brags about establishing the Internet which has been a long-term project to lasso the minds of the global population. I have said for decades the plan is to control information to such an extreme that eventually no one would see or hear anything that the Cult does not approve. We are closing in on that end with ferocious censorship since the ‘Covid’ hoax began and in my case it started back in the 1990s in terms of books and speaking venues. I had to create my own publishing company in 1995 precisely because no one else would publish my books even then. I think they’re all still running.

Cult Internet

To secure total control of information they needed the Internet in which pre-programmed algorithms can seek out ‘unclean’ content for deletion and even stop it being posted in the first place. The Cult had to dismantle print and non-Internet broadcast media to ensure the transfer of information to the appropriate-named ‘Web’ – a critical expression of the *Cult* web. We’ve seen the ever-quickenning demise of traditional media and control of what is left by a tiny number of corporations operating worldwide. Independent journalism in the mainstream is already dead and never was that more obvious than since the turn of 2020. The Cult wants all information communicated via the Internet to globally censor and allow the plug to be pulled any time. Lockdowns and forced isolation has meant that communication between people has been through electronic means and no longer through face-to-face discourse and discussion. Cult psychopaths have targeted the bars, restaurants, sport, venues and meeting places in general for this reason. None of this is by chance and it’s to stop people gathering in any kind of privacy or number while being able to track and monitor all Internet communications and block them as necessary. Even private messages between individuals have been censored by these fascists that control Cult fronts like Facebook, Twitter, Google and YouTube which are all officially run by Sabbatian place-people and from the background by higher-level Sabbatian place people.

Facebook, Google, Amazon and their like were seed-funded and supported into existence with money-no-object infusions of funds either directly or indirectly from DARPA and CIA technology arm In-Q-Tel. The Cult plays the long game and prepares very carefully for big plays like 'Covid'. Amazon is another front in the psychological war and pretty much controls the global market in book sales and increasingly publishing. Amazon's limitless funds have deleted fantastic numbers of independent publishers to seize global domination on the way to deciding which books can be sold and circulated and which cannot. Moves in that direction are already happening. Amazon's leading light Jeff Bezos is the grandson of Lawrence Preston Gise who worked with DARPA predecessor ARPA. Amazon has big connections to the CIA and the Pentagon. The plan I have long described went like this:

1. Employ military technology to establish the Internet.
2. Sell the Internet as a place where people can freely communicate without censorship and allow that to happen until the Net becomes the central and irreversible pillar of human society. If the Internet had been highly censored from the start many would have rejected it.
3. Fund and manipulate major corporations into being to control the circulation of information on your Internet using cover stories about geeks in garages to explain how they came about. Give them unlimited funds to expand rapidly with no need to make a profit for years while non-Cult companies who need to balance the books cannot compete. You know that in these circumstances your Googles, YouTubes, Facebooks and Amazons are going to secure near monopolies by either crushing or buying up the opposition.
4. Allow freedom of expression on both the Internet and communication platforms to draw people in until the Internet is the central and irreversible pillar of human society and your communication corporations have reached a stage of near monopoly domination.
5. Then unleash your always-planned frenzy of censorship on the basis of 'where else are you going to go?' and continue to expand that until nothing remains that the Cult does not want its human targets to see.

The process was timed to hit the 'Covid' hoax to ensure the best chance possible of controlling the narrative which they knew they had to do at all costs. They were, after all, about to unleash a 'deadly virus' that didn't really exist. If you do that in an environment of free-flowing information and opinion you would be dead in the

water before you could say Gates is a psychopath. The network was in place through which the Cult-created-and-owned World Health Organization could dictate the ‘Covid’ narrative and response policy slavishly supported by Cult-owned Internet communication giants and mainstream media while those telling a different story were censored. Google, YouTube, Facebook and Twitter openly announced that they would do this. What else would we expect from Cult-owned operations like Facebook which former executives have confirmed set out to make the platform more addictive than cigarettes and coldly manipulates emotions of its users to sow division between people and groups and scramble the minds of the young? If Zuckerberg lives out the rest of his life without going to jail for crimes against humanity, and most emphatically against the young, it will be a travesty of justice. Still, no matter, cause and effect will catch up with him eventually and the same with Sergey Brin and Larry Page at Google with its CEO Sundar Pichai who fix the Google search results to promote Cult narratives and hide the opposition. Put the same key words into Google and other search engines like DuckDuckGo and you will see how different results can be. Wikipedia is another intensely biased ‘encyclopaedia’ which skews its content to the Cult agenda. YouTube links to Wikipedia’s version of ‘Covid’ and ‘climate change’ on video pages in which experts in their field offer a different opinion (even that is increasingly rare with Wojcicki censorship). Into this ‘Covid’ silence-them network must be added government media censors, sorry ‘regulators’, such as Ofcom in the UK which imposed tyrannical restrictions on British broadcasters that had the effect of banning me from ever appearing. Just to debate with me about my evidence and views on ‘Covid’ would mean breaking the fascistic impositions of Ofcom and its CEO career government bureaucrat Melanie Dawes. Gutless British broadcasters tremble at the very thought of fascist Ofcom.

Psychos behind ‘Covid’

The reason for the ‘Covid’ catastrophe in all its facets and forms can be seen by whom and what is driving the policies worldwide in such a coordinated way. Decisions are not being made to protect health, but to target psychology. The dominant group guiding and ‘advising’ government policy are not medical professionals. They are psychologists and behavioural scientists. Every major country has its own version of this phenomenon and I’ll use the British example to show how it works. In many ways the British version has been affecting the wider world in the form of the huge behaviour manipulation network in the UK which operates in other countries. The network involves private companies, government, intelligence and military. The Cabinet Office is at the centre of the government ‘Covid’ Psyop and part-owns, with ‘innovation charity’ Nesta, the Behavioural Insights Team (BIT) which claims to be independent of government but patently isn’t. The BIT was established in 2010 and its job is to manipulate the psyche of the population to acquiesce to government demands and so much more. It is also known as the ‘Nudge Unit’, a name inspired by the 2009 book by two ultra-Zionists, Cass Sunstein and Richard Thaler, called *Nudge: Improving Decisions About Health, Wealth, and Happiness*. The book, as with the Behavioural Insights Team, seeks to ‘nudge’ behaviour (manipulate it) to make the public follow patterns of action and perception that suit those in authority (the Cult). Sunstein is so skilled at this that he advises the World Health Organization and the UK Behavioural Insights Team and was Administrator of the White House Office of Information and Regulatory Affairs in the Obama administration. Biden appointed him to the Department of Homeland Security – another ultra-Zionist in the fold to oversee new immigration laws which is another policy the Cult wants to control. Sunstein is desperate to silence anyone exposing conspiracies and co-authored a 2008 report on the subject in which suggestions were offered to ban ‘conspiracy theorizing’ or impose ‘some kind of tax, financial or otherwise, on those who disseminate such theories’. I guess a psychiatrist’s chair is out of the question?

Sunstein's mate Richard Thaler, an 'academic affiliate' of the UK Behavioural Insights Team, is a proponent of 'behavioural economics' which is defined as the study of 'the effects of psychological, cognitive, emotional, cultural and social factors on the decisions of individuals and institutions'. Study the effects so they can be manipulated to be what you want them to be. Other leading names in the development of behavioural economics are ultra-Zionists Daniel Kahneman and Robert J. Shiller and they, with Thaler, won the Nobel Memorial Prize in Economic Sciences for their work in this field. The Behavioural Insights Team is operating at the heart of the UK government and has expanded globally through partnerships with several universities including Harvard, Oxford, Cambridge, University College London (UCL) and Pennsylvania. They claim to have 'trained' (reframed) 20,000 civil servants and run more than 750 projects involving 400 randomised controlled trials in dozens of countries' as another version of mind reframers Common Purpose. BIT works from its office in New York with cities and their agencies, as well as other partners, across the United States and Canada – this is a company part-owned by the British government Cabinet Office. An executive order by President Cult-servant Obama established a US Social and Behavioral Sciences Team in 2015. They all have the same reason for being and that's to brainwash the population directly and by brainwashing those in positions of authority.

'Covid' mind game

Another prime aspect of the UK mind-control network is the 'independent' [joke] Scientific Pandemic Insights Group on Behaviours (SPI-B) which 'provides behavioural science advice aimed at anticipating and helping people adhere to interventions that are recommended by medical or epidemiological experts'. That means manipulating public perception and behaviour to do whatever government tells them to do. It's disgusting and if they really want the public to be 'safe' this lot should all be under lock and key. According to the government website SPI-B consists of

'behavioural scientists, health and social psychologists, anthropologists and historians' and advises the Whitty-Vallance-led Scientific Advisory Group for Emergencies (SAGE) which in turn advises the government on 'the science' (it doesn't) and 'Covid' policy. When politicians say they are being guided by 'the science' this is the rabble in each country they are talking about and that 'science' is dominated by behaviour manipulators to enforce government fascism through public compliance. The Behaviour Insight Team is headed by psychologist David Solomon Halpern, a visiting professor at King's College London, and connects with a national and global web of other civilian and military organisations as the Cult moves towards its goal of fusing them into one fascistic whole in every country through its 'Fusion Doctrine'. The behaviour manipulation network involves, but is not confined to, the Foreign Office; National Security Council; government communications headquarters (GCHQ); MI5; MI6; the Cabinet Office-based Media Monitoring Unit; and the Rapid Response Unit which 'monitors digital trends to spot emerging issues; including misinformation and disinformation; and identifies the best way to respond'.

There is also the 77th Brigade of the UK military which operates like the notorious Israeli military's Unit 8200 in manipulating information and discussion on the Internet by posing as members of the public to promote the narrative and discredit those who challenge it. Here we have the military seeking to manipulate *domestic* public opinion while the Nazis in government are fine with that. Conservative Member of Parliament Tobias Ellwood, an advocate of lockdown and control through 'vaccine passports', is a Lieutenant Colonel reservist in the 77th Brigade which connects with the military operation jHub, the 'innovation centre' for the Ministry of Defence and Strategic Command. jHub has also been involved with the civilian National Health Service (NHS) in 'symptom tracing' the population. The NHS is a key part of this mind control network and produced a document in December, 2020, explaining to staff how to use psychological manipulation with different groups and ages to get them to have the DNA-manipulating 'Covid vaccine'

that's designed to cumulatively rewrite human genetics. The document, called 'Optimising Vaccination Roll Out – Do's and Dont's for all messaging, documents and "communications" in the widest sense', was published by NHS England and the NHS Improvement *Behaviour Change Unit* in partnership with Public Health England and Warwick Business School. I hear the mantra about 'save the NHS' and 'protect the NHS' when we need to scrap the NHS and start again. The current version is far too corrupt, far too anti-human and totally compromised by Cult operatives and their assets. UK government broadcast media censor Ofcom will connect into this web – as will the BBC with its tremendous Ofcom influence – to control what the public see and hear and dictate mass perception. Nuremberg trials must include personnel from all these organisations.

The fear factor

The 'Covid' hoax has led to the creation of the UK Cabinet Office-connected Joint Biosecurity Centre (JBC) which is officially described as providing 'expert advice on pandemics' using its independent [all Cult operations are 'independent'] analytical function to provide real-time analysis about infection outbreaks to identify and respond to outbreaks of Covid-19'. Another role is to advise the government on a response to spikes in infections – 'for example by closing schools or workplaces in local areas where infection levels have risen'. Put another way, promoting the Cult agenda. The Joint Biosecurity Centre is modelled on the Joint Terrorism Analysis Centre which analyses intelligence to set 'terrorism threat levels' and here again you see the fusion of civilian and military operations and intelligence that has led to military intelligence producing documents about 'vaccine hesitancy' and how it can be combated. Domestic civilian matters and opinions should not be the business of the military. The Joint Biosecurity Centre is headed by Tom Hurd, director general of the Office for Security and Counter-Terrorism from the establishment-to-its-fingertips Hurd family. His father is former Foreign Secretary Douglas Hurd. How coincidental that Tom

Hurd went to the elite Eton College and Oxford University with Boris Johnson. Imperial College with its ridiculous computer modeller Neil Ferguson will connect with this gigantic web that will itself interconnect with similar set-ups in other major and not so major countries. Compared with this Cult network the politicians, be they Boris Johnson, Donald Trump or Joe Biden, are bit-part players ‘following the science’. The network of psychologists was on the ‘Covid’ case from the start with the aim of generating maximum fear of the ‘virus’ to ensure compliance by the population. A government behavioural science group known as SPI-B produced a paper in March, 2020, for discussion by the main government science advisory group known as SAGE. It was headed ‘Options for increasing adherence to social distancing measures’ and it said the following in a section headed ‘Persuasion’:

- A substantial number of people still do not feel sufficiently personally threatened; it could be that they are reassured by the low death rate in their demographic group, although levels of concern may be rising. Having a good understanding of the risk has been found to be positively associated with adoption of COVID-19 social distancing measures in Hong Kong.
- The perceived level of personal threat needs to be increased among those who are complacent, using hard-hitting evaluation of options for increasing social distancing emotional messaging. To be effective this must also empower people by making clear the actions they can take to reduce the threat.
- Responsibility to others: There seems to be insufficient understanding of, or feelings of responsibility about, people’s role in transmitting the infection to others ... Messaging about actions need to be framed positively in terms of protecting oneself and the community, and increase confidence that they will be effective.
- Some people will be more persuaded by appeals to play by the rules, some by duty to the community, and some to personal risk.

All these different approaches are needed. The messaging also needs to take account of the realities of different people's lives. Messaging needs to take account of the different motivational levers and circumstances of different people.

All this could be achieved the SPI-B psychologists said by *using the media to increase the sense of personal threat* which translates as terrify the shit out of the population, including children, so they all do what we want. That's not happened has it? Those excuses for 'journalists' who wouldn't know journalism if it bit them on the arse (the great majority) have played their crucial part in serving this Cult-government Psyop to enslave their own kids and grandkids. How they live with themselves I have no idea. The psychological war has been underpinned by constant government 'Covid' propaganda in almost every television and radio ad break, plus the Internet and print media, which has pounded out the fear with taxpayers footing the bill for their own programming. The result has been people terrified of a 'virus' that doesn't exist or one with a tiny fatality rate even if you believe it does. People walk down the street and around the shops wearing face-nappies damaging their health and psychology while others report those who refuse to be that naïve to the police who turn up in their own face-nappies. I had a cameraman come to my flat and he was so frightened of 'Covid' he came in wearing a mask and refused to shake my hand in case he caught something. He had – naïveitis – and the thought that he worked in the mainstream media was both depressing and made his behaviour perfectly explainable. The fear which has gripped the minds of so many and frozen them into compliance has been carefully cultivated by these psychologists who are really psychopaths. If lives get destroyed and a lot of young people commit suicide it shows our plan is working. SPI-B then turned to compulsion on the public to comply. 'With adequate preparation, rapid change can be achieved', it said. Some countries had introduced mandatory self-isolation on a wide scale without evidence of major public unrest and a large majority of the UK's population appeared to be supportive of more coercive measures with 64 percent of adults saying they would

support putting London under a lockdown (watch the ‘polls’ which are designed to make people believe that public opinion is in favour or against whatever the subject in hand).

For ‘aggressive protective measures’ to be effective, the SPI-B paper said, special attention should be devoted to those population groups that are more at risk. Translated from the Orwellian this means making the rest of population feel guilty for not protecting the ‘vulnerable’ such as old people which the Cult and its agencies were about to kill on an industrial scale with lockdown, lack of treatment and the Gates ‘vaccine’. Psychopath psychologists sold their guilt-trip so comprehensively that Los Angeles County Supervisor Hilda Solis reported that children were apologising (from a distance) to their parents and grandparents for bringing ‘Covid’ into their homes and getting them sick. ‘... These apologies are just some of the last words that loved ones will ever hear as they die alone,’ she said. Gut-wrenchingly Solis then used this childhood tragedy to tell children to stay at home and ‘keep your loved ones alive’. Imagine heaping such potentially life-long guilt on a kid when it has absolutely nothing to do with them. These people are deeply disturbed and the psychologists behind this even more so.

Uncivil war – divide and rule

Professional mind-controllers at SPI-B wanted the media to increase a sense of responsibility to others (do as you’re told) and promote ‘positive messaging’ for those actions while in contrast to invoke ‘social disapproval’ by the unquestioning, obedient, community of anyone with a mind of their own. Again the compliant Goebbels-like media obliged. This is an old, old, trick employed by tyrannies the world over throughout human history. You get the target population to keep the target population in line – *your* line. SPI-B said this could ‘play an important role in preventing anti-social behaviour or discouraging failure to enact pro-social behaviour’. For ‘anti-social’ in the Orwellian parlance of SPI-B see any behaviour that government doesn’t approve. SPI-B recommendations said that ‘social disapproval’ should be accompanied by clear messaging and

promotion of strong collective identity – hence the government and celebrity mantra of ‘we’re all in this together’. Sure we are. The mind doctors have such contempt for their targets that they think some clueless comedian, actor or singer telling them to do what the government wants will be enough to win them over. We have had UK comedian Lenny Henry, actor Michael Caine and singer Elton John wheeled out to serve the propagandists by urging people to have the DNA-manipulating ‘Covid’ non-‘vaccine’. The role of Henry and fellow black celebrities in seeking to coax a ‘vaccine’ reluctant black community into doing the government’s will was especially stomach-turning. An emotion-manipulating script and carefully edited video featuring these black ‘celebs’ was such an insult to the intelligence of black people and where’s the self-respect of those involved selling their souls to a fascist government agenda? Henry said he heard black people’s ‘legitimate worries and concerns’, but people must ‘trust the facts’ when they were doing exactly that by not having the ‘vaccine’. They had to include the obligatory reference to Black Lives Matter with the line ... ‘Don’t let coronavirus cost even more black lives – because we matter’. My god, it was pathetic. ‘I know the vaccine is safe and what it does.’ How? ‘I’m a comedian and it says so in my script.’

SPI-B said social disapproval needed to be carefully managed to avoid victimisation, scapegoating and misdirected criticism, but they knew that their ‘recommendations’ would lead to exactly that and the media were specifically used to stir-up the divide-and-conquer hostility. Those who conform like good little baa, baas, are praised while those who have seen through the tidal wave of lies are ‘Covidiots’. The awake have been abused by the fast asleep for not conforming to fascism and impositions that the awake know are designed to endanger their health, dehumanise them, and tear asunder the very fabric of human society. We have had the curtain-twitchers and morons reporting neighbours and others to the face-nappied police for breaking ‘Covid rules’ with fascist police delighting in posting links and phone numbers where this could be done. The Cult cannot impose its will without a compliant police

and military or a compliant population willing to play their part in enslaving themselves and their kids. The words of a pastor in Nazi Germany are so appropriate today:

First they came for the socialists and I did not speak out because I was not a socialist.

Then they came for the trade unionists and I did not speak out because I was not a trade unionist.

Then they came for the Jews and I did not speak out because I was not a Jew.

Then they came for me and there was no one left to speak for me.

Those who don't learn from history are destined to repeat it and so many are.

'Covid' rules: Rewiring the mind

With the background laid out to this gigantic national and global web of psychological manipulation we can put 'Covid' rules into a clear and sinister perspective. Forget the claims about protecting health. 'Covid' rules are about dismantling the human mind, breaking the human spirit, destroying self-respect, and then putting Humpty Dumpty together again as a servile, submissive slave. Social isolation through lockdown and distancing have devastating effects on the human psyche as the psychological psychopaths well know and that's the real reason for them. Humans need contact with each other, discourse, closeness and touch, or they eventually, and literally, go crazy. Masks, which I will address at some length, fundamentally add to the effects of isolation and the Cult agenda to dehumanise and de-individualise the population. To do this while knowing – in fact *seeking* – this outcome is the very epitome of evil and psychologists involved in this *are* the epitome of evil. They must like all the rest of the Cult demons and their assets stand trial for crimes against humanity on a scale that defies the imagination. Psychopaths in uniform use isolation to break enemy troops and agents and make them subservient and submissive to tell what they know. The technique is rightly considered a form of torture and

torture is most certainly what has been imposed on the human population.

Clinically-insane American psychologist Harry Harlow became famous for his isolation experiments in the 1950s in which he separated baby monkeys from their mothers and imprisoned them for months on end in a metal container or ‘pit of despair’. They soon began to show mental distress and depression as any idiot could have predicted. Harlow put other monkeys in steel chambers for three, six or twelve months while denying them any contact with animals or humans. He said that the effects of total social isolation for six months were ‘so devastating and debilitating that we had assumed initially that twelve months of isolation would not produce any additional decrement’; but twelve months of isolation ‘almost obliterated the animals socially’. This is what the Cult and its psychopaths are doing to you and your children. Even monkeys in partial isolation in which they were not allowed to form relationships with other monkeys became ‘aggressive and hostile, not only to others, but also towards their own bodies’. We have seen this in the young as a consequence of lockdown. UK government psychopaths launched a public relations campaign telling people not to hug each other even after they received the ‘Covid-19 vaccine’ which we were told with more lies would allow a return to ‘normal life’. A government source told *The Telegraph*: ‘It will be along the lines that it is great that you have been vaccinated, but if you are going to visit your family and hug your grandchildren there is a chance you are going to infect people you love.’ The source was apparently speaking from a secure psychiatric facility. Janet Lord, director of Birmingham University’s Institute of Inflammation and Ageing, said that parents and grandparents should avoid hugging their children. Well, how can I put it, Ms Lord? Fuck off. Yep, that’ll do.

Destroying the kids – where are the parents?

Observe what has happened to people enslaved and isolated by lockdown as suicide and self-harm has soared worldwide,

particularly among the young denied the freedom to associate with their friends. A study of 49,000 people in English-speaking countries concluded that almost half of young adults are at clinical risk of mental health disorders. A national survey in America of 1,000 currently enrolled high school and college students found that 5 percent reported attempting suicide during the pandemic. Data from the US CDC's National Syndromic Surveillance Program from January 1st to October 17th, 2020, revealed a 31 percent increase in mental health issues among adolescents aged 12 to 17 compared with 2019. The CDC reported that America in general suffered the biggest drop in life expectancy since World War Two as it fell by a year in the first half of 2020 as a result of 'deaths of despair' – overdoses and suicides. Deaths of despair have leapt by more than 20 percent during lockdown and include the highest number of fatal overdoses ever recorded in a single year – 81,000. Internet addiction is another consequence of being isolated at home which lowers interest in physical activities as kids fall into inertia and what's the point? Children and young people are losing hope and giving up on life, sometimes literally. A 14-year-old boy killed himself in Maryland because he had 'given up' when his school district didn't reopen; an 11-year-old boy shot himself during a zoom class; a teenager in Maine succumbed to the isolation of the 'pandemic' when he ended his life after experiencing a disrupted senior year at school. Children as young as nine have taken their life and all these stories can be repeated around the world. Careers are being destroyed before they start and that includes those in sport in which promising youngsters have not been able to take part. The plan of the psycho-psychologists is working all right. Researchers at Cambridge University found that lockdowns cause significant harm to children's mental health. Their study was published in the *Archives of Disease in Childhood*, and followed 168 children aged between 7 and 11. The researchers concluded:

During the UK lockdown, children's depression symptoms have increased substantially, relative to before lockdown. The scale of this effect has direct relevance for the continuation of different elements of lockdown policy, such as complete or partial school closures ...

... Specifically, we observed a statistically significant increase in ratings of depression, with a medium-to-large effect size. Our findings emphasise the need to incorporate the potential impact of lockdown on child mental health in planning the ongoing response to the global pandemic and the recovery from it.

Not a chance when the Cult's psycho-psychologists were getting exactly what they wanted. The UK's Royal College of Paediatrics and Child Health has urged parents to look for signs of eating disorders in children and young people after a three to four fold increase. Specialists say the 'pandemic' is a major reason behind the rise. You don't say. The College said isolation from friends during school closures, exam cancellations, loss of extra-curricular activities like sport, and an increased use of social media were all contributory factors along with fears about the virus (psycho-psychologists again), family finances, and students being forced to quarantine. Doctors said young people were becoming severely ill by the time they were seen with 'Covid' regulations reducing face-to-face consultations. Nor is it only the young that have been devastated by the psychopaths. Like all bullies and cowards the Cult is targeting the young, elderly, weak and infirm. A typical story was told by a British lady called Lynn Parker who was not allowed to visit her husband in 2020 for the last ten and half months of his life 'when he needed me most' between March 20th and when he died on December 19th. This vacates the criminal and enters the territory of evil. The emotional impact on the immune system alone is immense as are the number of people of all ages worldwide who have died as a result of Cult-demanded, Gates-demanded, lockdowns.

Isolation is torture

The experience of imposing solitary confinement on millions of prisoners around the world has shown how a large percentage become 'actively psychotic and/or acutely suicidal'. Social isolation has been found to trigger 'a specific psychiatric syndrome, characterized by hallucinations; panic attacks; overt paranoia; diminished impulse control; hypersensitivity to external stimuli; and difficulties with thinking, concentration and memory'. Juan Mendez,

a United Nations rapporteur (investigator), said that isolation is a form of torture. Research has shown that even after isolation prisoners find it far more difficult to make social connections and I remember chatting to a shop assistant after one lockdown who told me that when her young son met another child again he had no idea how to act or what to do. Hannah Flanagan, Director of Emergency Services at Journey Mental Health Center in Dane County, Wisconsin, said: ‘The specificity about Covid social distancing and isolation that we’ve come across as contributing factors to the suicides are really new to us this year.’ But they are not new to those that devised them. They are getting the effect they want as the population is psychologically dismantled to be rebuilt in a totally different way. Children and the young are particularly targeted. They will be the adults when the full-on fascist AI-controlled technocracy is planned to be imposed and they are being prepared to meekly submit. At the same time older people who still have a memory of what life was like before – and how fascist the new normal really is – are being deleted. You are going to see efforts to turn the young against the old to support this geriatric genocide. Hannah Flanagan said the big increase in suicide in her county proved that social isolation is not only harmful, but deadly. Studies have shown that isolation from others is one of the main risk factors in suicide and even more so with women. Warnings that lockdown could create a ‘perfect storm’ for suicide were ignored. After all this was one of the *reasons* for lockdown. Suicide, however, is only the most extreme of isolation consequences. There are many others. Dr Dhruv Khullar, assistant professor of healthcare policy at Weill Cornell Medical College, said in a *New York Times* article in 2016 long before the fake ‘pandemic’:

A wave of new research suggests social separation is bad for us. Individuals with less social connection have disrupted sleep patterns, altered immune systems, more inflammation and higher levels of stress hormones. One recent study found that isolation increases the risk of heart disease by 29 percent and stroke by 32 percent. Another analysis that pooled data from 70 studies and 3.4 million people found that socially isolated individuals had a 30 percent higher risk of dying in the next seven years, and that this effect was largest in middle age.

Loneliness can accelerate cognitive decline in older adults, and isolated individuals are twice as likely to die prematurely as those with more robust social interactions. These effects start early: Socially isolated children have significantly poorer health 20 years later, even after controlling for other factors. All told, loneliness is as important a risk factor for early death as obesity and smoking.

There you have proof from that one article alone four years before 2020 that those who have enforced lockdown, social distancing and isolation knew what the effect would be and that is even more so with professional psychologists that have been driving the policy across the globe. We can go back even further to the years 2000 and 2003 and the start of a major study on the effects of isolation on health by Dr Janine Gronewold and Professor Dirk M. Hermann at the University Hospital in Essen, Germany, who analysed data on 4,316 people with an average age of 59 who were recruited for the long-term research project. They found that socially isolated people are more than 40 percent more likely to have a heart attack, stroke, or other major cardiovascular event and nearly 50 percent more likely to die from any cause. Given the financial Armageddon unleashed by lockdown we should note that the study found a relationship between increased cardiovascular risk and lack of financial support. After excluding other factors social isolation was still connected to a 44 percent increased risk of cardiovascular problems and a 47 percent increased risk of death by any cause. Lack of financial support was associated with a 30 percent increase in the risk of cardiovascular health events. Dr Gronewold said it had been known for some time that feeling lonely or lacking contact with close friends and family can have an impact on physical health and the study had shown that having strong social relationships is of high importance for heart health. Gronewold said they didn't understand yet why people who are socially isolated have such poor health outcomes, but this was obviously a worrying finding, particularly during these times of prolonged social distancing. Well, it can be explained on many levels. You only have to identify the point in the body where people feel loneliness and missing people they are parted from – it's in the centre of the chest where they feel the ache of loneliness and the ache of missing people. 'My heart aches for

you' ... 'My heart aches for some company.' I will explain this more in the chapter Escaping Wetiko, but when you realise that the body is the mind – they are expressions of each other – the reason why state of the mind dictates state of the body becomes clear.

American psychologist Ranjit Powar was highlighting the effects of lockdown isolation as early as April, 2020. She said humans have evolved to be social creatures and are wired to live in interactive groups. Being isolated from family, friends and colleagues could be unbalancing and traumatic for most people and could result in short or even long-term psychological and physical health problems. An increase in levels of anxiety, aggression, depression, forgetfulness and hallucinations were possible psychological effects of isolation. 'Mental conditions may be precipitated for those with underlying pre-existing susceptibilities and show up in many others without any pre-condition.' Powar said personal relationships helped us cope with stress and if we lost this outlet for letting off steam the result can be a big emotional void which, for an average person, was difficult to deal with. 'Just a few days of isolation can cause increased levels of anxiety and depression' – so what the hell has been the effect on the global population of *18 months* of this at the time of writing? Powar said: 'Add to it the looming threat of a dreadful disease being repeatedly hammered in through the media and you have a recipe for many shades of mental and physical distress.' For those with a house and a garden it is easy to forget that billions have had to endure lockdown isolation in tiny overcrowded flats and apartments with nowhere to go outside. The psychological and physical consequences of this are unimaginable and with lunatic and abusive partners and parents the consequences have led to tremendous increases in domestic and child abuse and alcoholism as people seek to shut out the horror. Ranjit Powar said:

Staying in a confined space with family is not all a rosy picture for everyone. It can be extremely oppressive and claustrophobic for large low-income families huddled together in small single-room houses. Children here are not lucky enough to have many board/electronic games or books to keep them occupied.

Add to it the deep insecurity of running out of funds for food and basic necessities. On the other hand, there are people with dysfunctional family dynamics, such as domineering, abusive or alcoholic partners, siblings or parents which makes staying home a period of trial. Incidence of suicide and physical abuse against women has shown a worldwide increase. Heightened anxiety and depression also affect a person's immune system, making them more susceptible to illness.

To think that Powar's article was published on April 11th, 2020.

Six-feet fantasy

Social (unsocial) distancing demanded that people stay six feet or two metres apart. UK government advisor Robert Dingwall from the New and Emerging Respiratory Virus Threats Advisory Group said in a radio interview that the two-metre rule was 'conjured up out of nowhere' and was not based on science. No, it was not based on *medical* science, but it didn't come out of nowhere. The distance related to *psychological* science. Six feet/two metres was adopted in many countries and we were told by people like the criminal Anthony Fauci and his ilk that it was founded on science. Many schools could not reopen because they did not have the space for six-feet distancing. Then in March, 2021, after a year of six-feet 'science', a study published in the *Journal of Infectious Diseases* involving more than 500,000 students and almost 100,000 staff over 16 weeks revealed no significant difference in 'Covid' cases between six feet and three feet and Fauci changed his tune. Now three feet was okay. There is no difference between six feet and three *inches* when there is no 'virus' and they got away with six feet for psychological reasons for as long as they could. I hear journalists and others talk about 'unintended consequences' of lockdown. They are not *unintended* at all; they have been coldly-calculated for a specific outcome of human control and that's why super-psychopaths like Gates have called for them so vehemently. Super-psychopath psychologists have demanded them and psychopathic or clueless, spineless, politicians have gone along with them by 'following the science'. But it's not science at all. 'Science' is not what is; it's only what people can be manipulated to believe it is. The whole 'Covid' catastrophe is

founded on mind control. Three word or three statement mantras issued by the UK government are a well-known mind control technique and so we've had 'Stay home/protect the NHS/save lives', 'Stay alert/control the virus/save lives' and 'hands/face/space'. One of the most vocal proponents of extreme 'Covid' rules in the UK has been Professor Susan Michie, a member of the British Communist Party, who is not a medical professional. Michie is the director of the Centre for Behaviour Change at University College London. She is a *behavioural psychologist* and another filthy rich 'Marxist' who praised China's draconian lockdown. She was known by fellow students at Oxford University as 'Stalin's nanny' for her extreme Marxism. Michie is an influential member of the UK government's Scientific Advisory Group for Emergencies (SAGE) and behavioural manipulation groups which have dominated 'Covid' policy. She is a consultant adviser to the World Health Organization on 'Covid-19' and behaviour. Why the hell are lockdowns anything to do with her when they are claimed to be about health? Why does a behavioural psychologist from a group charged with changing the behaviour of the public want lockdown, human isolation and mandatory masks? Does that question really need an answer? Michie *absolutely* has to explain herself before a Nuremberg court when humanity takes back its world again and even more so when you see the consequences of masks that she demands are compulsory. This is a Michie classic:

The benefits of getting primary school children to wear masks is that regardless of what little degree of transmission is occurring in those age groups it could help normalise the practice. Young children wearing masks may be more likely to get their families to accept masks.

Those words alone should carry a prison sentence when you ponder on the callous disregard for children involved and what a statement it makes about the mind and motivations of Susan Michie. What a lovely lady and what she said there encapsulates the mentality of the psychopaths behind the 'Covid' horror. Let us compare what Michie said with a countrywide study in Germany published at [researchsquare.com](https://www.researchsquare.com) involving 25,000 school children and 17,854 health complaints submitted by parents. Researchers

found that masks are harming children physically, psychologically, and behaviourally with 24 health issues associated with mask wearing. They include: shortness of breath (29.7%); dizziness (26.4%); increased headaches (53%); difficulty concentrating (50%); drowsiness or fatigue (37%); and malaise (42%). Nearly a third of children experienced more sleep issues than before and a quarter developed new fears. Researchers found health issues and other impairments in 68 percent of masked children covering their faces for an average of 4.5 hours a day. Hundreds of those taking part experienced accelerated respiration, tightness in the chest, weakness, and short-term impairment of consciousness. A reminder of what Michie said again:

The benefits of getting primary school children to wear masks is that regardless of what little degree of transmission is occurring in those age groups it could help normalise the practice. Young children wearing masks may be more likely to get their families to accept masks.

Psychopaths in government and psychology now have children and young people – plus all the adults – wearing masks for hours on end while clueless teachers impose the will of the psychopaths on the young they should be protecting. What the hell are parents doing?

Cult lab rats

We have some schools already imposing on students microchipped buzzers that activate when they get ‘too close’ to their pals in the way they do with lab rats. How apt. To the Cult and its brain-dead servants our children *are* lab rats being conditioned to be unquestioning, dehumanised slaves for the rest of their lives.

Children and young people are being weaned and frightened away from the most natural human instincts including closeness and touch. I have tracked in the books over the years how schools were banning pupils from greeting each other with a hug and the whole Cult-induced Me Too movement has terrified men and boys from a relaxed and natural interaction with female friends and work colleagues to the point where many men try never to be in a room

alone with a woman that's not their partner. Airhead celebrities have as always played their virtue-signalling part in making this happen with their gross exaggeration. For every monster like Harvey Weinstein there are at least tens of thousands of men that don't treat women like that; but everyone must be branded the same and policy changed for them as well as the monster. I am going to be using the word 'dehumanise' many times in this chapter because that is what the Cult is seeking to do and it goes very deep as we shall see. Don't let them kid you that social distancing is planned to end one day. That's not the idea. We are seeing more governments and companies funding and producing wearable gadgets to keep people apart and they would not be doing that if this was meant to be short-term. A tech start-up company backed by GCHQ, the British Intelligence and military surveillance headquarters, has created a social distancing wrist sensor that alerts people when they get too close to others. The CIA has also supported tech companies developing similar devices. The wearable sensor was developed by Tended, one of a number of start-up companies supported by GCHQ (see the CIA and DARPA). The device can be worn on the wrist or as a tag on the waistband and will vibrate whenever someone wearing the device breaches social distancing and gets anywhere near natural human contact. The company had a lucky break in that it was developing a distancing sensor when the 'Covid' hoax arrived which immediately provided a potentially enormous market. How fortunate. The government in big-time Cult-controlled Ontario in Canada is investing \$2.5 million in wearable contact tracing technology that 'will alert users if they may have been exposed to the Covid-19 in the workplace and will beep or vibrate if they are within six feet of another person'. Facedrive Inc., the technology company behind this, was founded in 2016 with funding from the Ontario Together Fund and obviously they, too, had a prophet on the board of directors. The human surveillance and control technology is called TraceSCAN and would be worn by the human cyborgs in places such as airports, workplaces, construction sites, care homes and ... *schools*.

I emphasise schools with children and young people the prime targets. You know what is planned for society as a whole if you keep your eyes on the schools. They have always been places where the state program the next generation of slaves to be its compliant worker-ants – or Woker-ants these days; but in the mist of the ‘Covid’ madness they have been transformed into mind laboratories on a scale never seen before. Teachers and head teachers are just as programmed as the kids – often more so. Children are kept apart from human interaction by walk lanes, classroom distancing, staggered meal times, masks, and the rolling-out of buzzer systems. Schools are now physically laid out as a laboratory maze for lab-rats. Lunatics at a school in Anchorage, Alaska, who should be prosecuted for child abuse, took away desks and forced children to kneel (know your place) on a mat for five hours a day while wearing a mask and using their chairs as a desk. How this was supposed to impact on a ‘virus’ only these clinically insane people can tell you and even then it would be clap-trap. The school banned recess (interaction), art classes (creativity), and physical exercise (getting body and mind moving out of inertia). Everyone behind this outrage should be in jail or better still a mental institution. The behavioural manipulators are all for this dystopian approach to schools.

Professor Susan Michie, the mind-doctor and British Communist Party member, said it was wrong to say that schools were safe. They had to be made so by ‘distancing’, masks and ventilation (sitting all day in the cold). I must ask this lady round for dinner on a night I know I am going to be out and not back for weeks. She probably wouldn’t be able to make it, anyway, with all the visits to her own psychologist she must have block-booked.

Masking identity

I know how shocking it must be for you that a behaviour manipulator like Michie wants everyone to wear masks which have long been a feature of mind-control programs like the infamous MKUltra in the United States, but, there we are. We live and learn. I spent many years from 1996 to right across the millennium

researching mind control in detail on both sides of the Atlantic and elsewhere. I met a large number of mind-control survivors and many had been held captive in body and mind by MKUltra. MK stands for mind-control, but employs the German spelling in deference to the Nazis spirited out of Germany at the end of World War Two by Operation Paperclip in which the US authorities, with help from the Vatican, transported Nazi mind-controllers and engineers to America to continue their work. Many of them were behind the creation of NASA and they included Nazi scientist and SS officer Wernher von Braun who swapped designing V-2 rockets to bombard London with designing the Saturn V rockets that powered the NASA moon programme's Apollo craft. I think I may have mentioned that the Cult has no borders. Among Paperclip escapees was Josef Mengele, the Angel of Death in the Nazi concentration camps where he conducted mind and genetic experiments on children often using twins to provide a control twin to measure the impact of his 'work' on the other. If you want to observe the Cult mentality in all its extremes of evil then look into the life of Mengele. I have met many people who suffered mercilessly under Mengele in the United States where he operated under the name Dr Greene and became a stalwart of MKUltra programming and torture. Among his locations was the underground facility in the Mojave Desert in California called the China Lake Naval Weapons Station which is almost entirely below the surface. My books *The Biggest Secret*, *Children of the Matrix* and *The Perception Deception* have the detailed background to MKUltra.

The best-known MKUltra survivor is American Cathy O'Brien. I first met her and her late partner Mark Phillips at a conference in Colorado in 1996. Mark helped her escape and deprogram from decades of captivity in an offshoot of MKUltra known as Project Monarch in which 'sex slaves' were provided for the rich and famous including Father George Bush, Dick Cheney and the Clintons. Read Cathy and Mark's book *Trance-Formation of America* and if you are new to this you will be shocked to the core. I read it in 1996 shortly before, with the usual synchronicity of my life, I found

myself given a book table at the conference right next to hers. MKUltra never ended despite being very publicly exposed (only a small part of it) in the 1970s and continues in other guises. I am still in touch with Cathy. She contacted me during 2020 after masks became compulsory in many countries to tell me how they were used as part of MKUltra programming. I had been observing 'Covid regulations' and the relationship between authority and public for months. I saw techniques that I knew were employed on individuals in MKUltra being used on the global population. I had read many books and manuals on mind control including one called *Silent Weapons for Quiet Wars* which came to light in the 1980s and was a guide on how to perceptually program on a mass scale. 'Silent Weapons' refers to mind-control. I remembered a line from the manual as governments, medical authorities and law enforcement agencies have so obviously talked to – or rather at – the adult population since the 'Covid' hoax began as if they are children. The document said:

If a person is spoken to by a T.V. advertiser as if he were a twelve-year-old, then, due to suggestibility, he will, with a certain probability, respond or react to that suggestion with the uncritical response of a twelve-year-old and will reach in to his economic reservoir and deliver its energy to buy that product on impulse when he passes it in the store.

That's why authority has spoken to adults like children since all this began.

Why did Michael Jackson wear masks?

Every aspect of the 'Covid' narrative has mind-control as its central theme. Cathy O'Brien wrote an article for davidicke.com about the connection between masks and mind control. Her daughter Kelly who I first met in the 1990s was born while Cathy was still held captive in MKUltra. Kelly was forced to wear a mask as part of her programming from the age of *two* to dehumanise her, target her sense of individuality and reduce the amount of oxygen her brain and body received. *Bingo*. This is the real reason for compulsory

masks, why they have been enforced en masse, and why they seek to increase the number they demand you wear. First one, then two, with one disgraceful alleged ‘doctor’ recommending four which is nothing less than a death sentence. Where and how often they must be worn is being expanded for the purpose of mass mind control and damaging respiratory health which they can call ‘Covid-19’. Canada’s government headed by the man-child Justin Trudeau, says it’s fine for children of two and older to wear masks. An insane ‘study’ in Italy involving just 47 children concluded there was no problem for babies as young as *four months* wearing them. Even after people were ‘vaccinated’ they were still told to wear masks by the criminal that is Anthony Fauci. Cathy wrote that mandating masks is allowing the authorities literally to control the air we breathe which is what was done in MKUltra. You might recall how the singer Michael Jackson wore masks and there is a reason for that. He was subjected to MKUltra mind control through Project Monarch and his psyche was scrambled by these simpletons. Cathy wrote:

In MKUltra Project Monarch mind control, Michael Jackson had to wear a mask to silence his voice so he could not reach out for help. Remember how he developed that whisper voice when he wasn’t singing? Masks control the mind from the outside in, like the redefining of words is doing. By controlling what we can and cannot say for fear of being labeled racist or beaten, for example, it ultimately controls thought that drives our words and ultimately actions (or lack thereof).

Likewise, a mask muffles our speech so that we are not heard, which controls voice ... words ... mind. This is Mind Control. Masks are an obvious mind control device, and I am disturbed so many people are complying on a global scale. Masks depersonalize while making a person feel as though they have no voice. It is a barrier to others. People who would never choose to comply but are forced to wear a mask in order to keep their job, and ultimately their family fed, are compromised. They often feel shame and are subdued. People have stopped talking with each other while media controls the narrative.

The ‘no voice’ theme has often become literal with train passengers told not to speak to each other in case they pass on the ‘virus’, singing banned for the same reason and bonkers California officials telling people riding roller coasters that they cannot shout and scream. Cathy said she heard every day from healed MKUltra survivors who cannot wear a mask without flashing back on ways

their breathing was controlled – ‘from ball gags and penises to water boarding’. She said that through the years when she saw images of people in China wearing masks ‘due to pollution’ that it was really to control their oxygen levels. ‘I knew it was as much of a population control mechanism of depersonalisation as are burkas’, she said. Masks are another Chinese communist/fascist method of control that has been swept across the West as the West becomes China at lightning speed since we entered 2020.

Mask-19

There are other reasons for mandatory masks and these include destroying respiratory health to call it ‘Covid-19’ and stunting brain development of children and the young. Dr Margarite Griesz-Brisson MD, PhD, is a Consultant Neurologist and Neurophysiologist and the Founder and Medical Director of the London Neurology and Pain Clinic. Her CV goes down the street and round the corner. She is clearly someone who cares about people and won’t parrot the propaganda. Griesz-Brisson has a PhD in pharmacology, with special interest in neurotoxicology, environmental medicine, neuroregeneration and neuroplasticity (the way the brain can change in the light of information received). She went public in October, 2020, with a passionate warning about the effects of mask-wearing laws:

The reinhalation of our exhaled air will without a doubt create oxygen deficiency and a flooding of carbon dioxide. We know that the human brain is very sensitive to oxygen deprivation. There are nerve cells for example in the hippocampus that can’t be longer than 3 minutes without oxygen – they cannot survive. The acute warning symptoms are headaches, drowsiness, dizziness, issues in concentration, slowing down of reaction time – reactions of the cognitive system.

Oh, I know, let’s tell bus, truck and taxi drivers to wear them and people working machinery. How about pilots, doctors and police? Griesz-Brisson makes the important point that while the symptoms she mentions may fade as the body readjusts this does not alter the fact that people continue to operate in oxygen deficit with long list of

potential consequences. She said it was well known that neurodegenerative diseases take years or decades to develop. 'If today you forget your phone number, the breakdown in your brain would have already started 20 or 30 years ago.' She said degenerative processes in your brain are getting amplified as your oxygen deprivation continues through wearing a mask. Nerve cells in the brain are unable to divide themselves normally in these circumstances and lost nerve cells will no longer be regenerated. 'What is gone is gone.' Now consider that people like shop workers and *schoolchildren* are wearing masks for hours every day. What in the name of sanity is going to be happening to them? 'I do not wear a mask, I need my brain to think', Griesz-Brisson said, 'I want to have a clear head when I deal with my patients and not be in a carbon dioxide-induced anaesthesia'. If you are told to wear a mask anywhere ask the organisation, police, store, whatever, for their risk assessment on the dangers and negative effects on mind and body of enforcing mask-wearing. They won't have one because it has never been done not even by government. All of them must be subject to class-action lawsuits as the consequences come to light. They don't do mask risk assessments for an obvious reason. They know what the conclusions would be and independent scientific studies that *have* been done tell a horror story of consequences.

'Masks are criminal'

Dr Griesz-Brisson said that for children and adolescents, masks are an absolute no-no. They had an extremely active and adaptive immune system and their brain was incredibly active with so much to learn. 'The child's brain, or the youth's brain, is thirsting for oxygen.' The more metabolically active an organ was, the more oxygen it required; and in children and adolescents every organ was metabolically active. Griesz-Brisson said that to deprive a child's or adolescent's brain of oxygen, or to restrict it in any way, was not only dangerous to their health, it was absolutely criminal. 'Oxygen deficiency inhibits the development of the brain, and the damage that has taken place as a result CANNOT be reversed.' Mind

manipulators of MKUltra put masks on two-year-olds they wanted to neurologically rewire and you can see why. Griesz-Brisson said a child needs the brain to learn and the brain needs oxygen to function. 'We don't need a clinical study for that. This is simple, indisputable physiology.' Consciously and purposely induced oxygen deficiency was an absolutely deliberate health hazard, and an absolute medical contraindication which means that 'this drug, this therapy, this method or measure should not be used, and is not allowed to be used'. To coerce an entire population to use an absolute medical contraindication by force, she said, there had to be definite and serious reasons and the reasons must be presented to competent interdisciplinary and independent bodies to be verified and authorised. She had this warning of the consequences that were coming if mask wearing continued:

When, in ten years, dementia is going to increase exponentially, and the younger generations couldn't reach their god-given potential, it won't help to say 'we didn't need the masks'. I know how damaging oxygen deprivation is for the brain, cardiologists know how damaging it is for the heart, pulmonologists know how damaging it is for the lungs. Oxygen deprivation damages every single organ. Where are our health departments, our health insurance, our medical associations? It would have been their duty to be vehemently against the lockdown and to stop it and stop it from the very beginning.

Why do the medical boards issue punishments to doctors who give people exemptions? Does the person or the doctor seriously have to prove that oxygen deprivation harms people? What kind of medicine are our doctors and medical associations representing? Who is responsible for this crime? The ones who want to enforce it? The ones who let it happen and play along, or the ones who don't prevent it?

All of the organisations and people she mentions there either answer directly to the Cult or do whatever hierarchical levels above them tell them to do. The outcome of both is the same. 'It's not about masks, it's not about viruses, it's certainly not about your health', Griesz-Brisson said. 'It is about much, much more. I am not participating. I am not afraid.' They were taking our air to breathe and there was no unfounded medical exemption from face masks. Oxygen deprivation was dangerous for every single brain. It had to be the free decision of every human being whether they want to

wear a mask that was absolutely ineffective to protect themselves from a virus. She ended by rightly identifying where the responsibility lies for all this:

The imperative of the hour is personal responsibility. We are responsible for what we think, not the media. We are responsible for what we do, not our superiors. We are responsible for our health, not the World Health Organization. And we are responsible for what happens in our country, not the government.

Halle-bloody-lujah.

But surgeons wear masks, right?

Independent studies of mask-wearing have produced a long list of reports detailing mental, emotional and physical dangers. What a definition of insanity to see police officers imposing mask-wearing on the public which will cumulatively damage their health while the police themselves wear masks that will cumulatively damage *their* health. It's utter madness and both public and police do this because 'the government says so' – yes a government of brain-donor idiots like UK Health Secretary Matt Hancock reading the 'follow the science' scripts of psychopathic, lunatic psychologists. The response you get from Stockholm syndrome sufferers defending the very authorities that are destroying them and their families is that 'surgeons wear masks'. This is considered the game, set and match that they must work and don't cause oxygen deficit. Well, actually, scientific studies have shown that they *do* and oxygen levels are monitored in operating theatres to compensate. Surgeons wear masks to stop spittle and such like dropping into open wounds – not to stop 'viral particles' which are so minuscule they can only be seen through an electron microscope. Holes in the masks are significantly bigger than 'viral particles' and if you sneeze or cough they will breach the mask. I watched an incredibly disingenuous 'experiment' that claimed to prove that masks work in catching 'virus' material from the mouth and nose. They did this with a slow motion camera and the mask did block big stuff which stayed inside the mask and

against the face to be breathed in or cause infections on the face as we have seen with many children. ‘Viral particles’, however, would never have been picked up by the camera as they came through the mask when they are far too small to be seen. The ‘experiment’ was therefore disingenuous *and* useless.

Studies have concluded that wearing masks in operating theatres (and thus elsewhere) make no difference to preventing infection while the opposite is true with toxic shite building up in the mask and this had led to an explosion in tooth decay and gum disease dubbed by dentists ‘mask mouth’. You might have seen the Internet video of a furious American doctor urging people to take off their masks after a four-year-old patient had been rushed to hospital the night before and nearly died with a lung infection that doctors sourced to mask wearing. A study in the journal *Cancer Discovery* found that inhalation of harmful microbes can contribute to advanced stage lung cancer in adults and long-term use of masks can help breed dangerous pathogens. Microbiologists have said frequent mask wearing creates a moist environment in which microbes can grow and proliferate before entering the lungs. The Canadian Agency for Drugs and Technologies in Health, or CADTH, a Canadian national organisation that provides research and analysis to healthcare decision-makers, said this as long ago as 2013 in a report entitled ‘Use of Surgical Masks in the Operating Room: A Review of the Clinical Effectiveness and Guidelines’. It said:

- No evidence was found to support the use of surgical face masks to reduce the frequency of surgical site infections
- No evidence was found on the effectiveness of wearing surgical face masks to protect staff from infectious material in the operating room.
- Guidelines recommend the use of surgical face masks by staff in the operating room to protect both operating room staff and patients (despite the lack of evidence).

We were told that the world could go back to ‘normal’ with the arrival of the ‘vaccines’. When they came, fraudulent as they are, the story changed as I knew that it would. We are in the midst of transforming ‘normal’, not going back to it. Mary Ramsay, head of immunisation at Public Health England, echoed the words of US criminal Anthony Fauci who said masks and other regulations must stay no matter if people are vaccinated. The Fauci idiot continued to wear two masks – different colours so both could be clearly seen – after he *claimed* to have been vaccinated. Senator Rand Paul told Fauci in one exchange that his double-masks were ‘theatre’ and he was right. It’s all theatre. Mary Ramsay back-tracked on the vaccine-return-to-normal theme when she said the public may need to wear masks and social-distance for years despite the jabs. ‘People have got used to those lower-level restrictions now, and [they] can live with them’, she said telling us what the idea has been all along. ‘The vaccine does not give you a pass, even if you have had it, you must continue to follow all the guidelines’ said a Public Health England statement which reneged on what we had been told before and made having the ‘vaccine’ irrelevant to ‘normality’ even by the official story. Spain’s fascist government trumped everyone by passing a law mandating the wearing of masks on the beach and even when swimming in the sea. The move would have devastated what’s left of the Spanish tourist industry, posed potential breathing dangers to swimmers and had Northern European sunbathers walking around with their forehead brown and the rest of their face white as a sheet. The ruling was so crazy that it had to be retracted after pressure from public and tourist industry, but it confirmed where the Cult wants to go with masks and how clinically insane authority has become. The determination to make masks permanent and hide the serious dangers to body and mind can be seen in the censorship of scientist Professor Denis Rancourt by Bill Gates-funded academic publishing website ResearchGate over his papers exposing the dangers and uselessness of masks. Rancourt said:

ResearchGate today has permanently locked my account, which I have had since 2015. Their reasons graphically show the nature of their attack against democracy, and their corruption of

science ... By their obscene non-logic, a scientific review of science articles reporting on harms caused by face masks has a 'potential to cause harm'. No criticism of the psychological device (face masks) is tolerated, if the said criticism shows potential to influence public policy.

This is what happens in a fascist world.

Where are the 'greens' (again)?

Other dangers of wearing masks especially regularly relate to the inhalation of minute plastic fibres into the lungs and the deluge of discarded masks in the environment and oceans. Estimates predicted that more than 1.5 billion disposable masks will end up in the world's oceans every year polluting the water with tons of plastic and endangering marine wildlife. Studies project that humans are using 129 billion face masks each month worldwide – about three million a minute. Most are disposable and made from plastic, non-biodegradable microfibers that break down into smaller plastic particles that become widespread in ecosystems. They are littering cities, clogging sewage channels and turning up in bodies of water. I have written in other books about the immense amounts of microplastics from endless sources now being absorbed into the body. Rolf Halden, director of the Arizona State University (ASU) Biodesign Center for Environmental Health Engineering, was the senior researcher in a 2020 study that analysed 47 human tissue samples and found microplastics in all of them. 'We have detected these chemicals of plastics in every single organ that we have investigated', he said. I wrote in *The Answer* about the world being deluged with microplastics. A study by the Worldwide Fund for Nature (WWF) found that people are consuming on average every week some 2,000 tiny pieces of plastic mostly through water and also through marine life and the air. Every year humans are ingesting enough microplastics to fill a heaped dinner plate and in a life-time of 79 years it is enough to fill two large waste bins. Marco Lambertini, WWF International director general said: 'Not only are plastics polluting our oceans and waterways and killing marine life – it's in all of us and we can't escape consuming plastics,' American

geologists found tiny plastic fibres, beads and shards in rainwater samples collected from the remote slopes of the Rocky Mountain National Park near Denver, Colorado. Their report was headed: 'It is raining plastic.' Rachel Adams, senior lecturer in Biomedical Science at Cardiff Metropolitan University, said that among health consequences are internal inflammation and immune responses to a 'foreign body'. She further pointed out that microplastics become carriers of toxins including mercury, pesticides and dioxins (a known cause of cancer and reproductive and developmental problems). These toxins accumulate in the fatty tissues once they enter the body through microplastics. Now this is being compounded massively by people putting plastic on their face and throwing it away.

Workers exposed to polypropylene plastic fibres known as 'flock' have developed 'flock worker's lung' from inhaling small pieces of the flock fibres which can damage lung tissue, reduce breathing capacity and exacerbate other respiratory problems. Now ... commonly used surgical masks have three layers of melt-blown textiles made of ... polypropylene. We have billions of people putting these microplastics against their mouth, nose and face for hours at a time day after day in the form of masks. How does anyone think that will work out? I mean – what could possibly go wrong? We posted a number of scientific studies on this at davidicke.com, but when I went back to them as I was writing this book the links to the science research website where they were hosted were dead. Anything that challenges the official narrative in any way is either censored or vilified. The official narrative is so unsupportable by the evidence that only deleting the truth can protect it. A study by Chinese scientists still survived – with the usual twist which it why it was still active, I guess. Yes, they found that virtually all the masks they tested increased the daily intake of microplastic fibres, but people should still wear them because the danger from the 'virus' was worse said the crazy 'team' from the Institute of Hydrobiology in Wuhan. Scientists first discovered microplastics in lung tissue of some patients who died of lung cancer

in the 1990s. Subsequent studies have confirmed the potential health damage with the plastic degrading slowly and remaining in the lungs to accumulate in volume. Wuhan researchers used a machine simulating human breathing to establish that masks shed up to nearly 4,000 microplastic fibres in a month with reused masks producing more. Scientists said some masks are laced with toxic chemicals and a variety of compounds seriously restricted for both health and environmental reasons. They include cobalt (used in blue dye) and formaldehyde known to cause watery eyes, burning sensations in the eyes, nose, and throat, plus coughing, wheezing and nausea. No – that must be 'Covid-19'.

Mask 'worms'

There is another and potentially even more sinister content of masks. Mostly new masks of different makes filmed under a microscope around the world have been found to contain strange black fibres or 'worms' that appear to move or 'crawl' by themselves and react to heat and water. The nearest I have seen to them are the self-replicating fibres that are pulled out through the skin of those suffering from Morgellons disease which has been connected to the phenomena of 'chemtrails' which I will bring into the story later on. Morgellons fibres continue to grow outside the body and have a form of artificial intelligence. Black 'worm' fibres in masks have that kind of feel to them and there is a nanotechnology technique called 'worm micelles' which carry and release drugs or anything else you want to deliver to the body. For sure the suppression of humanity by mind altering drugs is the Cult agenda big time and the more excuses they can find to gain access to the body the more opportunities there are to make that happen whether through 'vaccines' or masks pushed against the mouth and nose for hours on end.

So let us summarise the pros and cons of masks:

Against masks: Breathing in your own carbon dioxide; depriving the body and brain of sufficient oxygen; build-up of toxins in the mask that can be breathed into the lungs and cause rashes on the face and ‘mask-mouth’; breathing microplastic fibres and toxic chemicals into the lungs; dehumanisation and deleting individualisation by literally making people faceless; destroying human emotional interaction through facial expression and deleting parental connection with their babies which look for guidance to their facial expression.

For masks: They don’t protect you from a ‘virus’ that doesn’t exist and even if it did ‘viral’ particles are so minute they are smaller than the holes in the mask.

Governments, police, supermarkets, businesses, transport companies, and all the rest who seek to impose masks have done no risk assessment on their consequences for health and psychology and are now open to group lawsuits when the impact becomes clear with a cumulative epidemic of respiratory and other disease. Authorities will try to exploit these effects and hide the real cause by dubbing them ‘Covid-19’. Can you imagine setting out to force the population to wear health-destroying masks without doing any assessment of the risks? It is criminal and it is evil, but then how many people targeted in this way, who see their children told to wear them all day at school, have asked for a risk assessment? Billions can’t be imposed upon by the few unless the billions allow it. Oh, yes, with just a tinge of irony, 85 percent of all masks made worldwide come from *China*.

Wash your hands in toxic shite

‘Covid’ rules include the use of toxic sanitisers and again the health consequences of constantly applying toxins to be absorbed through the skin is obvious to any level of Renegade Mind. America’s Food and Drug Administration (FDA) said that sanitisers are drugs and issued a warning about 75 dangerous brands which contain

methanol used in antifreeze and can cause death, kidney damage and blindness. The FDA circulated the following warning even for those brands that it claims to be safe:

Store hand sanitizer out of the reach of pets and children, and children should use it only with adult supervision. Do not drink hand sanitizer. This is particularly important for young children, especially toddlers, who may be attracted by the pleasant smell or brightly colored bottles of hand sanitizer.

Drinking even a small amount of hand sanitizer can cause alcohol poisoning in children. (However, there is no need to be concerned if your children eat with or lick their hands after using hand sanitizer.) During this coronavirus pandemic, poison control centers have had an increase in calls about accidental ingestion of hand sanitizer, so it is important that adults monitor young children's use.

Do not allow pets to swallow hand sanitizer. If you think your pet has eaten something potentially dangerous, call your veterinarian or a pet poison control center right away. Hand sanitizer is flammable and should be stored away from heat and flames. When using hand sanitizer, rub your hands until they feel completely dry before performing activities that may involve heat, sparks, static electricity, or open flames.

There you go, perfectly safe, then, and that's without even a mention of the toxins absorbed through the skin. Come on kids – sanitise your hands everywhere you go. It will save you from the 'virus'. Put all these elements together of the 'Covid' normal and see how much health and psychology is being cumulatively damaged, even devastated, to 'protect your health'. Makes sense, right? They are only imposing these things because they care, right? *Right?*

Submitting to insanity

Psychological reframing of the population goes very deep and is done in many less obvious ways. I hear people say how contradictory and crazy 'Covid' rules are and how they are ever changing. This is explained away by dismissing those involved as idiots. It is a big mistake. The Cult is delighted if its cold calculation is perceived as incompetence and idiocy when it is anything but. Oh, yes, there are idiots within the system – lots of them – but they are *administering* the Cult agenda, mostly unknowingly. They are not deciding and dictating it. The bulwark against tyranny is self-

respect, always has been, always will be. It is self-respect that has broken every tyranny in history. By its very nature self-respect will not bow to oppression and its perpetrators. There is so little self-respect that it's always the few that overturn dictators. Many may eventually follow, but the few with the iron spines (self-respect) kick it off and generate the momentum. The Cult targets self-respect in the knowledge that once this has gone only submission remains. Crazy, contradictory, ever-changing 'Covid' rules are systematically applied by psychologists to delete self-respect. They *want* you to see that the rules make no sense. It is one thing to decide to do something when *you* have made the choice based on evidence and logic. You still retain your self-respect. It is quite another when you can see what you are being told to do is insane, ridiculous and makes no sense, and *yet you still do it*. Your self-respect is extinguished and this has been happening as ever more obviously stupid and nonsensical things have been demanded and the great majority have complied even when they can see they are stupid and nonsensical.

People walk around in face-nappies knowing they are damaging their health and make no difference to a 'virus'. They do it in fear of not doing it. I know it's daft, but I'll do it anyway. When that happens something dies inside of you and submissive reframing has begun. Next there's a need to hide from yourself that you have conceded your self-respect and you convince yourself that you have not really submitted to fear and intimidation. You begin to believe that you are complying with craziness because it's the right thing to do. When first you concede your self-respect of $2+2 = 4$ to $2+2 = 5$ you *know* you are compromising your self-respect. Gradually to avoid facing that fact you begin to *believe* that $2+2=5$. You have been reframed and I have been watching this process happening in the human psyche on an industrial scale. The Cult is working to break your spirit and one of its major tools in that war is humiliation. I read how former American soldier Bradley Manning (later Chelsea Manning after a sex-change) was treated after being jailed for supplying WikiLeaks with documents exposing the enormity of

government and elite mendacity. Manning was isolated in solitary confinement for eight months, put under 24-hour surveillance, forced to hand over clothing before going to bed, and stand naked for every roll call. This is systematic humiliation. The introduction of anal swab 'Covid' tests in China has been done for the same reason to delete self-respect and induce compliant submission. Anal swabs are mandatory for incoming passengers in parts of China and American diplomats have said they were forced to undergo the indignity which would have been calculated humiliation by the Cult-owned Chinese government that has America in its sights.

Government-people: An abusive relationship

Spirit-breaking psychological techniques include giving people hope and apparent respite from tyranny only to take it away again. This happened in the UK during Christmas, 2020, when the psycho-psychologists and their political lackeys announced an easing of restrictions over the holiday only to reimpose them almost immediately on the basis of yet another lie. There is a big psychological difference between getting used to oppression and being given hope of relief only to have that dashed. Psychologists know this and we have seen the technique used repeatedly. Then there is traumatising people before you introduce more extreme regulations that require compliance. A perfect case was the announcement by the dark and sinister Whitty and Vallance in the UK that 'new data' predicted that 4,000 could die every day over the winter of 2020/2021 if we did not lockdown again. I think they call it lying and after traumatising people with that claim out came Jackboot Johnson the next day with new curbs on human freedom. Psychologists know that a frightened and traumatised mind becomes suggestable to submission and behaviour reframing. Underpinning all this has been to make people fearful and suspicious of each other and see themselves as a potential danger to others. In league with deleted self-respect you have the perfect psychological recipe for self-loathing. The relationship between authority and public is now demonstrably the same as that of

subservience to an abusive partner. These are signs of an abusive relationship explained by psychologist Leslie Becker-Phelps:

Psychological and emotional abuse: Undermining a partner's self-worth with verbal attacks, name-calling, and belittling. Humiliating the partner in public, unjustly accusing them of having an affair, or interrogating them about their every behavior. Keeping partner confused or off balance by saying they were just kidding or blaming the partner for 'making' them act this way ... Feigning in public that they care while turning against them in private. This leads to victims frequently feeling confused, incompetent, unworthy, hopeless, and chronically self-doubting. [Apply these techniques to how governments have treated the population since New Year, 2020, and the parallels are obvious.]

Physical abuse: The abuser might physically harm their partner in a range of ways, such as grabbing, hitting, punching, or shoving them. They might throw objects at them or harm them with a weapon. [Observe the physical harm imposed by masks, lockdown, and so on.]

Threats and intimidation: One way abusers keep their partners in line is by instilling fear. They might be verbally threatening, or give threatening looks or gestures. Abusers often make it known that they are tracking their partner's every move. They might destroy their partner's possessions, threaten to harm them, or threaten to harm their family members. Not surprisingly, victims of this abuse often feel anxiety, fear, and panic. [No words necessary.]

Isolation: Abusers often limit their partner's activities, forbidding them to talk or interact with friends or family. They might limit access to a car or even turn off their phone. All of this might be done by physically holding them against their will, but is often accomplished through psychological abuse and intimidation. The more isolated a person feels, the fewer resources they have to help gain perspective on their situation and to escape from it. [No words necessary.]

Economic abuse: Abusers often make their partners beholden to them for money by controlling access to funds of any kind. They might prevent their partner from getting a job or withhold access to money they earn from a job. This creates financial dependency that makes leaving the relationship very difficult. [See destruction of livelihoods and the proposed meagre 'guaranteed income' so long as you do whatever you are told.]

Using children: An abuser might disparage their partner's parenting skills, tell their children lies about their partner, threaten to take custody of their children, or threaten to harm their children. These tactics instil fear and often elicit compliance. [See reframed social service mafia and how children are being mercilessly abused by the state over 'Covid' while their parents look on too frightened to do anything.]

A further recurring trait in an abusive relationship is the abused blaming themselves for their abuse and making excuses for the abuser. We have the public blaming each other for lockdown abuse by government and many making excuses for the government while attacking those who challenge the government. How often we have heard authorities say that rules are being imposed or reimposed only because people have refused to 'behave' and follow the rules. We don't want to do it – it's *you*.

Renegade Minds are an antidote to all of these things. They will never concede their self-respect no matter what the circumstances. Even when apparent humiliation is heaped upon them they laugh in its face and reflect back the humiliation on the abuser where it belongs. Renegade Minds will never wear masks they know are only imposed to humiliate, suppress and damage both physically and psychologically. Consequences will take care of themselves and they will never break their spirit or cause them to concede to tyranny. UK newspaper columnist Peter Hitchens was one of the few in the mainstream media to speak out against lockdowns and forced vaccinations. He then announced he had taken the jab. He wanted to see family members abroad and he believed vaccine passports were inevitable even though they had not yet been introduced. Hitchens

has a questioning and critical mind, but not a Renegade one. If he had no amount of pressure would have made him concede. Hitchens excused his action by saying that the battle has been lost. Renegade Minds never accept defeat when freedom is at stake and even if they are the last one standing the self-respect of not submitting to tyranny is more important than any outcome or any consequence.

That's why Renegade Minds are the only minds that ever changed anything worth changing.

CHAPTER EIGHT

'Reframing' insanity

Insanity is relative. It depends on who has who locked in what cage

Ray Bradbury

'Reframing' a mind means simply to change its perception and behaviour. This can be done subconsciously to such an extent that subjects have no idea they have been 'reframed' while to any observer changes in behaviour and attitudes are obvious.

Human society is being reframed on a ginormous scale since the start of 2020 and here we have the reason why psychologists rather than doctors have been calling the shots. Ask most people who have succumbed to 'Covid' reframing if they have changed and most will say 'no'; but they *have* and fundamentally. The Cult's long-game has been preparing for these times since way back and crucial to that has been to prepare both population and officialdom mentally and emotionally. To use the mind-control parlance they had to reframe the population with a mentality that would submit to fascism and reframe those in government and law enforcement to impose fascism or at least go along with it. The result has been the fact-deleted mindlessness of 'Wokeness' and officialdom that has either enthusiastically or unquestioningly imposed global tyranny demanded by reframed politicians on behalf of psychopathic and deeply evil cultists. 'Cognitive reframing' identifies and challenges the way someone sees the world in the form of situations, experiences and emotions and then restructures those perceptions to view the same set of circumstances in a different way. This can have

benefits if the attitudes are personally destructive while on the other side it has the potential for individual and collective mind control which the subject has no idea has even happened.

Cognitive therapy was developed in the 1960s by Aaron T. Beck who was born in Rhode Island in 1921 as the son of Jewish immigrants from the Ukraine. He became interested in the techniques as a treatment for depression. Beck's daughter Judith S. Beck is prominent in the same field and they founded the Beck Institute for Cognitive Behavior Therapy in Philadelphia in 1994. Cognitive reframing, however, began to be used worldwide by those with a very dark agenda. The Cult reframes politicians to change their attitudes and actions until they are completely at odds with what they once appeared to stand for. The same has been happening to government administrators at all levels, law enforcement, military and the human population. Cultists love mind control for two main reasons: It allows them to control what people think, do and say to secure agenda advancement and, by definition, it calms their legendary insecurity and fear of the unexpected. I have studied mind control since the time I travelled America in 1996. I may have been talking to next to no one in terms of an audience in those years, but my goodness did I gather a phenomenal amount of information and knowledge about so many things including the techniques of mind control. I have described this in detail in other books going back to *The Biggest Secret* in 1998. I met a very large number of people recovering from MKUltra and its offshoots and successors and I began to see how these same techniques were being used on the population in general. This was never more obvious than since the 'Covid' hoax began.

Reframing the enforcers

I have observed over the last two decades and more the very clear transformation in the dynamic between the police, officialdom and the public. I tracked this in the books as the relationship mutated from one of serving the public to seeing them as almost the enemy and certainly a lower caste. There has always been a class divide

based on income and always been some psychopathic, corrupt, and big-I-am police officers. This was different. Wholesale change was unfolding in the collective dynamic; it was less about money and far more about position and perceived power. An us-and-them was emerging. Noses were lifted skyward by government administration and law enforcement and their attitude to the public they were *supposed* to be serving changed to one of increasing contempt, superiority and control. The transformation was so clear and widespread that it had to be planned. Collective attitudes and dynamics do not change naturally and organically that quickly on that scale. I then came across an organisation in Britain called Common Purpose created in the late 1980s by Julia Middleton who would work in the office of Deputy Prime Minister John Prescott during the long and disastrous premiership of war criminal Tony Blair. When Blair speaks the Cult is speaking and the man should have been in jail a long time ago. Common Purpose proclaims itself to be one of the biggest 'leadership development' organisations in the world while functioning as a *charity* with all the financial benefits which come from that. It hosts 'leadership development' courses and programmes all over the world and claims to have 'brought together' what it calls 'leaders' from more than 100 countries on six continents. The modus operandi of Common Purpose can be compared with the work of the UK government's reframing network that includes the Behavioural Insights Team 'nudge unit' and 'Covid' reframing specialists at SPI-B. WikiLeaks described Common Purpose long ago as 'a hidden virus in our government and schools' which is unknown to the general public: 'It recruits and trains "leaders" to be loyal to the directives of Common Purpose and the EU, instead of to their own departments, which they then undermine or subvert, the NHS [National Health Service] being an example.' This is a vital point to understand the 'Covid' hoax. The NHS, and its equivalent around the world, has been utterly reframed in terms of administrators and much of the medical personnel with the transformation underpinned by recruitment policies. The outcome has been the criminal and psychopathic behaviour of the

NHS over ‘Covid’ and we have seen the same in every other major country. WikiLeaks said Common Purpose trainees are ‘learning to rule without regard to democracy’ and to usher in a police state (current events explained). Common Purpose operated like a ‘glue’ and had members in the NHS, BBC, police, legal profession, church, many of Britain’s 7,000 quangos, local councils, the Civil Service, government ministries and Parliament, and controlled many RDA’s (Regional Development Agencies). Here we have one answer for how and why British institutions and their like in other countries have changed so negatively in relation to the public. This further explains how and why the beyond-disgraceful reframed BBC has become a propaganda arm of ‘Covid’ fascism. They are all part of a network pursuing the same goal.

By 2019 Common Purpose was quoting a figure of 85,000 ‘leaders’ that had attended its programmes. These ‘students’ of all ages are known as Common Purpose ‘graduates’ and they consist of government, state and local government officials and administrators, police chiefs and officers, and a whole range of others operating within the national, local and global establishment. Cressida Dick, Commissioner of the London Metropolitan Police, is the Common Purpose graduate who was the ‘Gold Commander’ that oversaw what can only be described as the murder of Brazilian electrician Jean Charles de Menezes in 2005. He was held down by psychopathic police and shot seven times in the head by a psychopathic lunatic after being mistaken for a terrorist when he was just a bloke going about his day. Dick authorised officers to pursue and keep surveillance on de Menezes and ordered that he be stopped from entering the underground train system. Police psychopaths took her at her word clearly. She was ‘disciplined’ for this outrage by being *promoted* – eventually to the top of the ‘Met’ police where she has been a disaster. Many Chief Constables controlling the police in different parts of the UK are and have been Common Purpose graduates. I have heard the ‘graduate’ network described as a sort of Mafia or secret society operating within the fabric of government at all levels pursuing a collective policy

ingrained at Common Purpose training events. Founder Julia Middleton herself has said:

Locally and internationally, Common Purpose graduates will be 'lighting small fires' to create change in their organisations and communities ... The Common Purpose effect is best illustrated by the many stories of small changes brought about by leaders, who themselves have changed.

A Common Purpose mission statement declared:

Common Purpose aims to improve the way society works by expanding the vision, decision-making ability and influence of all kinds of leaders. The organisation runs a variety of educational programmes for leaders of all ages, backgrounds and sectors, in order to provide them with the inspirational, information and opportunities they need to change the world.

Yes, but into what? Since 2020 the answer has become clear.

NLP and the Delphi technique

Common Purpose would seem to be a perfect name or would common programming be better? One of the foundation methods of reaching 'consensus' (group think) is by setting the agenda theme and then encouraging, cajoling or pressuring everyone to agree a 'consensus' in line with the core theme promoted by Common Purpose. The methodology involves the 'Delphi technique', or an adaption of it, in which opinions are expressed that are summarised by a 'facilitator or change agent' at each stage. Participants are 'encouraged' to modify their views in the light of what others have said. Stage by stage the former individual opinions are merged into group consensus which just happens to be what Common Purpose wants them to believe. A key part of this is to marginalise anyone refusing to concede to group think and turn the group against them to apply pressure to conform. We are seeing this very technique used on the general population to make 'Covid' group-thinkers hostile to those who have seen through the bullshit. People can be reframed by using perception manipulation methods such as Neuro-Linguistic Programming (NLP) in which you change perception with the use of

carefully constructed language. An NLP website described the technique this way:

... A method of influencing brain behaviour (the 'neuro' part of the phrase) through the use of language (the 'linguistic' part) and other types of communication to enable a person to 'recode' the way the brain responds to stimuli (that's the 'programming') and manifest new and better behaviours. Neuro-Linguistic Programming often incorporates hypnosis and self-hypnosis to help achieve the change (or 'programming') that is wanted.

British alternative media operation UKColumn has done very detailed research into Common Purpose over a long period. I quoted co-founder and former naval officer Brian Gerrish in my book *Remember Who You Are*, published in 2011, as saying the following years before current times:

It is interesting that many of the mothers who have had children taken by the State speak of the Social Services people being icily cool, emotionless and, as two ladies said in slightly different words, '... like little robots'. We know that NLP is cumulative, so people can be given small imperceptible doses of NLP in a course here, another in a few months, next year etc. In this way, major changes are accrued in their personality, but the day by day change is almost unnoticeable.

In these and other ways 'graduates' have had their perceptions uniformly reframed and they return to their roles in the institutions of government, law enforcement, legal profession, military, 'education', the UK National Health Service and the whole swathe of the establishment structure to pursue a common agenda preparing for the 'post-industrial', 'post-democratic' society. I say 'preparing' but we are now there. 'Post-industrial' is code for the Great Reset and 'post-democratic' is 'Covid' fascism. UKColumn has spoken to partners of those who have attended Common Purpose 'training'. They have described how personalities and attitudes of 'graduates' changed very noticeably for the worse by the time they had completed the course. They had been 'reframed' and told they are the 'leaders' – the special ones – who know better than the population. There has also been the very demonstrable recruitment of psychopaths and narcissists into government administration at all

levels and law enforcement. If you want psychopathy hire psychopaths and you get a simple cause and effect. If you want administrators, police officers and 'leaders' to perceive the public as lesser beings who don't matter then employ narcissists. These personalities are identified using 'psychometrics' that identifies knowledge, abilities, attitudes and personality traits, mostly through carefully-designed questionnaires and tests. As this policy has passed through the decades we have had power-crazy, power-trippers appointed into law enforcement, security and government administration in preparation for current times and the dynamic between public and law enforcement/officialdom has been transformed. UKColumn's Brian Gerrish said of the narcissistic personality:

Their love of themselves and power automatically means that they will crush others who get in their way. I received a major piece of the puzzle when a friend pointed out that when they made public officials re-apply for their own jobs several years ago they were also required to do psychometric tests. This was undoubtedly the start of the screening process to get 'their' sort of people in post.

How obvious that has been since 2020 although it was clear what was happening long before if people paid attention to the changing public-establishment dynamic.

Change agents

At the centre of events in 'Covid' Britain is the National Health Service (NHS) which has behaved disgracefully in slavishly following the Cult agenda. The NHS management structure is awash with Common Purpose graduates or 'change agents' working to a common cause. Helen Bevan, a Chief of Service Transformation at the NHS Institute for Innovation and Improvement, co-authored a document called 'Towards a million change agents, a review of the social movements literature: implications for large scale change in the NHS'. The document compared a project management approach to that of change and social movements where 'people change

themselves and each other – peer to peer'. Two definitions given for a 'social movement' were:

A group of people who consciously attempt to build a radically new social order; involves people of a broad range of social backgrounds; and deploys politically confrontational and socially disruptive tactics – Cyrus Zirakzadeh 1997

Collective challenges, based on common purposes and social solidarities, in sustained interaction with elites, opponents, and authorities – Sidney Tarrow 1994

Helen Bevan wrote another NHS document in which she defined 'framing' as 'the process by which leaders construct, articulate and put across their message in a powerful and compelling way in order to win people to their cause and call them to action'. I think I could come up with another definition that would be rather more accurate. The National Health Service and institutions of Britain and the wider world have been taken over by reframed 'change agents' and that includes everything from the United Nations to national governments, local councils and social services which have been kidnapping children from loving parents on an extraordinary and gathering scale on the road to the end of parenthood altogether. Children from loving homes are stolen and kidnapped by the state and put into the 'care' (inversion) of the local authority through council homes, foster parents and forced adoption. At the same time children are allowed to be abused without response while many are under council 'care'. UKColumn highlighted the Common Purpose connection between South Yorkshire Police and Rotherham council officers in the case of the scandal in that area of the sexual exploitation of children to which the authorities turned not one blind eye, but both:

We were alarmed to discover that the Chief Executive, the Strategic Director of Children and Young People's Services, the Manager for the Local Strategic Partnership, the Community Cohesion Manager, the Cabinet Member for Cohesion, the Chief Constable and his predecessor had all attended Leadership training courses provided by the pseudo-charity Common Purpose.

Once 'change agents' have secured positions of hire and fire within any organisation things start to move very quickly. Personnel are then hired and fired on the basis of whether they will work towards the agenda the change agent represents. If they do they are rapidly promoted even though they may be incompetent. Those more qualified and skilled who are pre-Common Purpose 'old school' see their careers stall and even disappear. This has been happening for decades in every institution of state, police, 'health' and social services and all of them have been transformed as a result in their attitudes to their jobs and the public. Medical professions, including nursing, which were once vocations for the caring now employ many cold, callous and couldn't give a shit personality types. The UKColumn investigation concluded:

By blurring the boundaries between people, professions, public and private sectors, responsibility and accountability, Common Purpose encourages 'graduates' to believe that as new selected leaders, they can work together, outside of the established political and social structures, to achieve a paradigm shift or CHANGE – so called 'Leading Beyond Authority'. In doing so, the allegiance of the individual becomes 'reframed' on CP colleagues and their NETWORK.

Reframing the Face-Nappies

Nowhere has this process been more obvious than in the police where recruitment of psychopaths and development of unquestioning mind-controlled group-thinkers have transformed law enforcement into a politically-correct 'Woke' joke and a travesty of what should be public service. Today they wear their face-nappies like good little gofers and enforce 'Covid' rules which are fascism under another name. Alongside the specifically-recruited psychopaths we have software minds incapable of free thought. Brian Gerrish again:

An example is the policeman who would not get on a bike for a press photo because he had not done the cycling proficiency course. Normal people say this is political correctness gone mad. Nothing could be further from the truth. The policeman has been reframed, and in his reality it is perfect common sense not to get on the bike ‘because he hasn’t done the cycling course’.

Another example of this is where the police would not rescue a boy from a pond until they had taken advice from above on the ‘risk assessment’. A normal person would have arrived, perhaps thought of the risk for a moment, and dived in. To the police now ‘reframed’, they followed ‘normal’ procedure.

There are shocking cases of reframed ambulance crews doing the same. Sheer unthinking stupidity of London Face-Nappies headed by Common Purpose graduate Cressida Dick can be seen in their behaviour at a vigil in March, 2021, for a murdered woman, Sarah Everard. A police officer had been charged with the crime. Anyone with a brain would have left the vigil alone in the circumstances. Instead they ‘manhandled’ women to stop them breaking ‘Covid rules’ to betray classic reframing. Minds in the thrall of perception control have no capacity for seeing a situation on its merits and acting accordingly. ‘Rules is rules’ is their only mind-set. My father used to say that rules and regulations are for the guidance of the intelligent and the blind obedience of the idiot. Most of the intelligent, decent, coppers have gone leaving only the other kind and a few old school for whom the job must be a daily nightmare. The combination of psychopaths and rule-book software minds has been clearly on public display in the ‘Covid’ era with automaton robots in uniform imposing fascistic ‘Covid’ regulations on the population without any personal initiative or judging situations on their merits. There are thousands of examples around the world, but I’ll make my point with the infamous Derbyshire police in the English East Midlands – the ones who think pouring dye into beauty spots and using drones to track people walking in the countryside away from anyone is called ‘policing’. To them there are rules decreed by the government which they have to enforce and in their bewildered state a group gathering in a closed space and someone walking alone in the countryside are the same thing. It is beyond idiocy and enters the realm of clinical insanity.

Police officers in Derbyshire said they were ‘horrified’ – *horrified* – to find 15 to 20 ‘irresponsible’ kids playing a football match at a closed leisure centre ‘in breach of coronavirus restrictions’. When they saw the police the kids ran away leaving their belongings behind and the reframed men and women of Derbyshire police were seeking to establish their identities with a view to fining their parents. The most natural thing for youngsters to do – kicking a ball about – is turned into a criminal activity and enforced by the moronic software programs of Derbyshire police. You find the same mentality in every country. These barely conscious ‘horrified’ officers said they had to take action because ‘we need to ensure these rules are being followed’ and ‘it is of the utmost importance that you ensure your children are following the rules and regulations for Covid-19’. Had any of them done ten seconds of research to see if this parroting of their masters’ script could be supported by any evidence? Nope. Reframed people don’t think – others think for them and that’s the whole idea of reframing. I have seen police officers one after the other repeating without question word for word what officialdom tells them just as I have seen great swathes of the public doing the same. Ask either for ‘their’ opinion and out spews what they have been told to think by the official narrative. Police and public may seem to be in different groups, but their mentality is the same. Most people do whatever they are told in fear not doing so or because they believe what officialdom tells them; almost the entirety of the police do what they are told for the same reason. Ultimately it’s the tiny inner core of the global Cult that’s telling both what to do.

So Derbyshire police were ‘horrified’. Oh, really? Why did they think those kids were playing football? It was to relieve the psychological consequences of lockdown and being denied human contact with their friends and interaction, touch and discourse vital to human psychological health. Being denied this month after month has dismantled the psyche of many children and young people as depression and suicide have exploded. Were Derbyshire police *horrified by that?* Are you kidding? Reframed people don’t have those

mental and emotional processes that can see how the impact on the psychological health of youngsters is far more dangerous than any 'virus' even if you take the mendacious official figures to be true. The reframed are told (programmed) how to act and so they do. The Derbyshire Chief Constable in the first period of lockdown when the black dye and drones nonsense was going on was Peter Goodman. He was the man who severed the connection between his force and the Derbyshire Constabulary *Male Voice* Choir when he decided that it was not inclusive enough to allow women to join. The fact it was a male voice choir making a particular sound produced by male voices seemed to elude a guy who terrifyingly ran policing in Derbyshire. He retired weeks after his force was condemned as disgraceful by former Supreme Court Justice Jonathan Sumption for their behaviour over extreme lockdown impositions. Goodman was replaced by his deputy Rachel Swann who was in charge when her officers were 'horrified'. The police statement over the boys committing the hanging-offence of playing football included the line about the youngsters being 'irresponsible in the times we are all living through' missing the point that the real relevance of the 'times we are all living through' is the imposition of fascism enforced by psychopaths and reframed minds of police officers playing such a vital part in establishing the fascist tyranny that their own children and grandchildren will have to live in their entire lives. As a definition of insanity that is hard to beat although it might be run close by imposing masks on people that can have a serious effect on their health while wearing a face nappy all day themselves. Once again public and police do it for the same reason – the authorities tell them to and who are they to have the self-respect to say no?

Wokers in uniform

How reframed do you have to be to arrest a *six-year-old* and take him to court for *picking a flower* while waiting for a bus? Brain dead police and officialdom did just that in North Carolina where criminal proceedings happen regularly for children under nine. Attorney Julie Boyer gave the six-year-old crayons and a colouring book

during the ‘flower’ hearing while the ‘adults’ decided his fate. County Chief District Court Judge Jay Corpening asked: ‘Should a child that believes in Santa Claus, the Easter Bunny and the tooth fairy be making life-altering decisions?’ Well, of course not, but common sense has no meaning when you have a common purpose and a reframed mind. Treating children in this way, and police operating in American schools, is all part of the psychological preparation for children to accept a police state as normal all their adult lives. The same goes for all the cameras and biometric tracking technology in schools. Police training is focused on reframing them as snowflake Wokers and this is happening in the military. Pentagon top brass said that ‘training sessions on extremism’ were needed for troops who asked why they were so focused on the Capitol Building riot when Black Lives Matter riots were ignored. What’s the difference between them some apparently and rightly asked. Actually, there is a difference. Five people died in the Capitol riot, only one through violence, and that was a police officer shooting an unarmed protestor. BLM riots killed at least 25 people and cost billions. Asking the question prompted the psychopaths and reframed minds that run the Pentagon to say that more ‘education’ (programming) was needed. Troop training is all based on psychological programming to make them fodder for the Cult – ‘Military men are just dumb, stupid animals to be used as pawns in foreign policy’ as Cult-to-his-DNA former Secretary of State Henry Kissinger famously said. Governments see the police in similar terms and it’s time for those among them who can see this to defend the people and stop being enforcers of the Cult agenda upon the people.

The US military, like the country itself, is being targeted for destruction through a long list of Woke impositions. Cult-owned gaga ‘President’ Biden signed an executive order when he took office to allow taxpayer money to pay for transgender surgery for active military personnel and veterans. Are you a man soldier? No, I’m a LGBTQIA+ with a hint of Skoliosexual and Spectrasexual. Oh, good man. Bad choice of words you bigot. The Pentagon announced in March, 2021, the appointment of the first ‘diversity and inclusion

officer' for US Special Forces. Richard Torres-Estrada arrived with the publication of a 'D&I Strategic Plan which will guide the enterprise-wide effort to institutionalize and sustain D&I'. If you think a Special Forces 'Strategic Plan' should have something to do with defending America you haven't been paying attention.

Defending Woke is now the military's new role. Torres-Estrada has posted images comparing Donald Trump with Adolf Hitler and we can expect no bias from him as a representative of the supposedly non-political Pentagon. Cable news host Tucker Carlson said: 'The Pentagon is now the Yale faculty lounge but with cruise missiles.' Meanwhile Secretary of Defense Lloyd Austin, a board member of weapons-maker Raytheon with stock and compensation interests in October, 2020, worth \$1.4 million, said he was purging the military of the 'enemy within' – anyone who isn't Woke and supports Donald Trump. Austin refers to his targets as 'racist extremists' while in true Woke fashion being himself a racist extremist. Pentagon documents pledge to 'eradicate, eliminate and conquer all forms of racism, sexism and homophobia'. The definitions of these are decided by 'diversity and inclusion committees' peopled by those who see racism, sexism and homophobia in every situation and opinion. Woke (the Cult) is dismantling the US military and purging testosterone as China expands its military and gives its troops 'masculinity training'. How do we think that is going to end when this is all Cult coordinated? The US military, like the British military, is controlled by Woke and spineless top brass who just go along with it out of personal career interests.

'Woke' means fast asleep

Mind control and perception manipulation techniques used on individuals to create group-think have been unleashed on the global population in general. As a result many have no capacity to see the obvious fascist agenda being installed all around them or what 'Covid' is really all about. Their brains are firewalled like a computer system not to process certain concepts, thoughts and realisations that are bad for the Cult. The young are most targeted as the adults they

will be when the whole fascist global state is planned to be fully implemented. They need to be prepared for total compliance to eliminate all pushback from entire generations. The Cult has been pouring billions into taking complete control of 'education' from schools to universities via its operatives and corporations and not least Bill Gates as always. The plan has been to transform 'education' institutions into programming centres for the mentality of 'Woke'. James McConnell, professor of psychology at the University of Michigan, wrote in *Psychology Today* in 1970:

The day has come when we can combine sensory deprivation with drugs, hypnosis, and astute manipulation of reward and punishment, to gain almost absolute control over an individual's behaviour. It should then be possible to achieve a very rapid and highly effective type of brainwashing that would allow us to make dramatic changes in a person's behaviour and personality ...

... We should reshape society so that we all would be trained from birth to want to do what society wants us to do. We have the techniques to do it... no-one owns his own personality you acquired, and there's no reason to believe you should have the right to refuse to acquire a new personality if your old one is anti-social.

This was the potential for mass brainwashing in 1970 and the mentality there displayed captures the arrogant psychopathy that drives it forward. I emphasise that not all young people have succumbed to Woke programming and those that haven't are incredibly impressive people given that today's young are the most perceptually-targeted generations in history with all the technology now involved. Vast swathes of the young generations, however, have fallen into the spell – and that's what it is – of Woke. The Woke mentality and perceptual program is founded on *inversion* and you will appreciate later why that is so significant. Everything with Woke is inverted and the opposite of what it is claimed to be. Woke was a term used in African-American culture from the 1900s and referred to an awareness of social and racial justice. This is not the meaning of the modern version or 'New Woke' as I call it in *The Answer*. Oh, no, Woke today means something very different no matter how much Wokers may seek to hide that and insist Old Woke and New

Woke are the same. See if you find any 'awareness of social justice' here in the modern variety:

- Woke demands 'inclusivity' while excluding anyone with a different opinion and calls for mass censorship to silence other views.
- Woke claims to stand against oppression when imposing oppression is the foundation of all that it does. It is the driver of political correctness which is nothing more than a Cult invention to manipulate the population to silence itself.
- Woke believes itself to be 'liberal' while pursuing a global society that can only be described as fascist (see 'anti-fascist' fascist Antifa).
- Woke calls for 'social justice' while spreading injustice wherever it goes against the common 'enemy' which can be easily identified as a differing view.
- Woke is supposed to be a metaphor for 'awake' when it is solid-gold asleep and deep in a Cult-induced coma that meets the criteria for 'off with the fairies'.

I state these points as obvious facts if people only care to look. I don't do this with a sense of condemnation. We need to appreciate that the onslaught of perceptual programming on the young has been incessant and merciless. I can understand why so many have been reframed, or, given their youth, framed from the start to see the world as the Cult demands. The Cult has had access to their minds day after day in its 'education' system for their entire formative years. Perception is formed from information received and the Cult-created system is a life-long download of information delivered to elicit a particular perception, thus behaviour. The more this has expanded into still new extremes in recent decades and ever-increasing censorship has deleted other opinions and information why wouldn't that lead to a perceptual reframing on a mass scale? I

have described already cradle-to-grave programming and in more recent times the targeting of young minds from birth to adulthood has entered the stratosphere. This has taken the form of skewing what is ‘taught’ to fit the Cult agenda and the omnipresent techniques of group-think to isolate non-believers and pressure them into line. There has always been a tendency to follow the herd, but we really are in a new world now in relation to that. We have parents who can see the ‘Covid’ hoax told by their children not to stop them wearing masks at school, being ‘Covid’ tested or having the ‘vaccine’ in fear of the peer-pressure consequences of being different. What is ‘peer-pressure’ if not pressure to conform to group-think? Renegade Minds never group-think and always retain a set of perceptions that are unique to them. Group-think is always underpinned by consequences for not group-thinking. Abuse now aimed at those refusing DNA-manipulating ‘Covid vaccines’ are a potent example of this. The biggest pressure to conform comes from the very group which is itself being manipulated. ‘I am programmed to be part of a hive mind and so you must be.’

Woke control structures in ‘education’ now apply to every mainstream organisation. Those at the top of the ‘education’ hierarchy (the Cult) decide the policy. This is imposed on governments through the Cult network; governments impose it on schools, colleges and universities; their leadership impose the policy on teachers and academics and they impose it on children and students. At any level where there is resistance, perhaps from a teacher or university lecturer, they are targeted by the authorities and often fired. Students themselves regularly demand the dismissal of academics (increasingly few) at odds with the narrative that the students have been programmed to believe in. It is quite a thought that students who are being targeted by the Cult become so consumed by programmed group-think that they launch protests and demand the removal of those who are trying to push back against those targeting the students. Such is the scale of perceptual inversion. We see this with ‘Covid’ programming as the Cult imposes the rules via psycho-psychologists and governments on

shops, transport companies and businesses which impose them on their staff who impose them on their customers who pressure Pushbackers to conform to the will of the Cult which is in the process of destroying them and their families. Scan all aspects of society and you will see the same sequence every time.

Fact free Woke and hijacking the 'left'

There is no more potent example of this than 'Woke', a mentality only made possible by the deletion of factual evidence by an 'education' system seeking to produce an ever more uniform society. Why would you bother with facts when you don't know any? Deletion of credible history both in volume and type is highly relevant. Orwell said: 'Who controls the past controls the future: who controls the present controls the past.' They who control the perception of the past control the perception of the future and they who control the present control the perception of the past through the writing and deleting of history. Why would you oppose the imposition of Marxism in the name of Wokeism when you don't know that Marxism cost at least 100 million lives in the 20th century alone? Watch videos and read reports in which Woker generations are asked basic historical questions – it's mind-blowing. A survey of 2,000 people found that six percent of millennials (born approximately early 1980s to early 2000s) believed the Second World War (1939-1945) broke out with the assassination of President Kennedy (in 1963) and one in ten thought Margaret Thatcher was British Prime Minister at the time. She was in office between 1979 and 1990. We are in a post-fact society. Provable facts are no defence against the fascism of political correctness or Silicon Valley censorship. Facts don't matter anymore as we have witnessed with the 'Covid' hoax. Sacrificing uniqueness to the Woke group-think religion is all you are required to do and that means thinking for yourself is the biggest Woke no, no. All religions are an expression of group-think and censorship and Woke is just another religion with an orthodoxy defended by group-think and censorship. Burned at

the stake becomes burned on Twitter which leads back eventually to burned at the stake as Woke humanity regresses to ages past.

The biggest Woke inversion of all is its creators and funders. I grew up in a traditional left of centre political household on a council estate in Leicester in the 1950s and 60s – you know, the left that challenged the power of wealth-hoarding elites and threats to freedom of speech and opinion. In those days students went on marches defending freedom of speech while today's Wokers march for its deletion. What on earth could have happened? Those very elites (collectively the Cult) that we opposed in my youth and early life have funded into existence the antithesis of that former left and hijacked the 'brand' while inverting everything it ever stood for. We have a mentality that calls itself 'liberal' and 'progressive' while acting like fascists. Cult billionaires and their corporations have funded themselves into control of 'education' to ensure that Woke programming is unceasing throughout the formative years of children and young people and that non-Wokers are isolated (that word again) whether they be students, teachers or college professors. The Cult has funded into existence the now colossal global network of Woke organisations that have spawned and promoted all the 'causes' on the Cult wish-list for global transformation and turned Wokers into demanders of them. Does anyone really think it's a coincidence that the Cult agenda for humanity is a carbon (sorry) copy of the societal transformations desired by Woke?? These are only some of them:

Political correctness: The means by which the Cult deletes all public debates that it knows it cannot win if we had the free-flow of information and evidence.

Human-caused 'climate change': The means by which the Cult seeks to transform society into a globally-controlled dictatorship imposing its will over the fine detail of everyone's lives 'to save the planet' which doesn't actually need saving.

Transgender obsession: Preparing collective perception to accept the ‘new human’ which would not have genders because it would be created technologically and not through procreation. I’ll have much more on this in Human 2.0.

Race obsession: The means by which the Cult seeks to divide and rule the population by triggering racial division through the perception that society is more racist than ever when the opposite is the case. Is it perfect in that regard? No. But to compare today with the racism of apartheid and segregation brought to an end by the civil rights movement in the 1960s is to insult the memory of that movement and inspirations like Martin Luther King. Why is the ‘anti-racism’ industry (which it is) so dominated by privileged white people?

White supremacy: This is a label used by privileged white people to demonise poor and deprived white people pushing back on tyranny to marginalise and destroy them. White people are being especially targeted as the dominant race by number within Western society which the Cult seeks to transform in its image. If you want to change a society you must weaken and undermine its biggest group and once you have done that by using the other groups you next turn on them to do the same ... ‘Then they came for the Jews and I was not a Jew so I did nothing.’

Mass migration: The mass movement of people from the Middle East, Africa and Asia into Europe, from the south into the United States and from Asia into Australia are another way the Cult seeks to dilute the racial, cultural and political influence of white people on Western society. White people ask why their governments appear to be working against them while being politically and culturally biased towards incoming cultures. Well, here’s your answer. In the same way sexually ‘straight’ people, men and women, ask why the

authorities are biased against them in favour of other sexualities. The answer is the same – that's the way the Cult wants it to be for very sinister motives.

These are all central parts of the Cult agenda and central parts of the Woke agenda and Woke was created and continues to be funded to an immense degree by Cult billionaires and corporations. If anyone begins to say 'coincidence' the syllables should stick in their throat.

Billionaire 'social justice warriors'

Joe Biden is a 100 percent-owned asset of the Cult and the Wokers' man in the White House whenever he can remember his name and for however long he lasts with his rapidly diminishing cognitive function. Even walking up the steps of an aircraft without falling on his arse would appear to be a challenge. He's not an empty-shell puppet or anything. From the minute Biden took office (or the Cult did) he began his executive orders promoting the Woke wish-list. You will see the Woke agenda imposed ever more severely because it's really the *Cult* agenda. Woke organisations and activist networks spawned by the Cult are funded to the extreme so long as they promote what the Cult wants to happen. Woke is funded to promote 'social justice' by billionaires who become billionaires by destroying social justice. The social justice mantra is only a cover for dismantling social justice and funded by billionaires that couldn't give a damn about social justice. Everything makes sense when you see that. One of Woke's premier funders is Cult billionaire financier George Soros who said: 'I am basically there to make money, I cannot and do not look at the social consequences of what I do.' This is the same Soros who has given more than \$32 billion to his Open Society Foundations global Woke network and funded Black Lives Matter, mass immigration into Europe and the United States, transgender activism, climate change activism, political correctness and groups targeting 'white supremacy' in the form of privileged white thugs that dominate Antifa. What a scam it all is and when

you are dealing with the unquestioning fact-free zone of Woke scamming them is child's play. All you need to pull it off in all these organisations are a few in-the-know agents of the Cult and an army of naïve, reframed, uninformed, narcissistic, know-nothings convinced of their own self-righteousness, self-purity and virtue.

Soros and fellow billionaires and billionaire corporations have poured hundreds of millions into Black Lives Matter and connected groups and promoted them to a global audience. None of this is motivated by caring about black people. These are the billionaires that have controlled and exploited a system that leaves millions of black people in abject poverty and deprivation which they do absolutely nothing to address. The same Cult networks funding BLM were behind the *slave trade!* Black Lives Matter hijacked a phrase that few would challenge and they have turned this laudable concept into a political weapon to divide society. You know that BLM is a fraud when it claims that *All Lives Matter*, the most inclusive statement of all, is 'racist'. BLM and its Cult masters don't want to end racism. To them it's a means to an end to control all of humanity never mind the colour, creed, culture or background. What has destroying the nuclear family got to do with ending racism? Nothing – but that is one of the goals of BLM and also happens to be a goal of the Cult as I have been exposing in my books for decades. Stealing children from loving parents and giving schools ever more power to override parents is part of that same agenda. BLM is a Marxist organisation and why would that not be the case when the Cult created Marxism *and* BLM? Patrisse Cullors, a BLM co-founder, said in a 2015 video that she and her fellow organisers, including co-founder Alicia Garza, are 'trained Marxists'. The lady known after marriage as Patrisse Khan-Cullors bought a \$1.4 million home in 2021 in one of the whitest areas of California with a black population of just 1.6 per cent and has so far bought *four* high-end homes for a total of \$3.2 million. How very Marxist. There must be a bit of spare in the BLM coffers, however, when Cult corporations and billionaires have handed over the best part of \$100 million. Many black people can see that Black Lives Matter is not

working for them, but against them, and this is still more confirmation. Black journalist Jason Whitlock, who had his account suspended by Twitter for simply linking to the story about the ‘Marxist’s’ home buying spree, said that BLM leaders are ‘making millions of dollars off the backs of these dead black men who they wouldn’t spit on if they were on fire and alive’.

Black Lies Matter

Cult assets and agencies came together to promote BLM in the wake of the death of career criminal George Floyd who had been jailed a number of times including for forcing his way into the home of a black woman with others in a raid in which a gun was pointed at her stomach. Floyd was filmed being held in a Minneapolis street in 2020 with the knee of a police officer on his neck and he subsequently died. It was an appalling thing for the officer to do, but the same technique has been used by police on peaceful protestors of lockdown without any outcry from the Woke brigade. As unquestioning supporters of the Cult agenda Wokers have supported lockdown and all the ‘Covid’ claptrap while attacking anyone standing up to the tyranny imposed in its name. Court documents would later include details of an autopsy on Floyd by County Medical Examiner Dr Andrew Baker who concluded that Floyd had taken a fatal level of the drug fentanyl. None of this mattered to fact-free, question-free, Woke. Floyd’s death was followed by worldwide protests against police brutality amid calls to defund the police. Throwing babies out with the bathwater is a Woke speciality. In the wake of the murder of British woman Sarah Everard a Green Party member of the House of Lords, Baroness Jones of Moulsecoomb (Nincompoopia would have been better), called for a 6pm curfew for all men. This would be in breach of the Geneva Conventions on war crimes which ban collective punishment, but that would never have crossed the black and white Woke mind of Baroness Nincompoopia who would have been far too convinced of her own self-righteousness to compute such details. Many American cities did defund the police in the face of Floyd riots

and after \$15 million was deleted from the police budget in Washington DC under useless Woke mayor Muriel Bowser car-jacking alone rose by 300 percent and within six months the US capital recorded its highest murder rate in 15 years. The same happened in Chicago and other cities in line with the Cult/Soros plan to bring fear to streets and neighbourhoods by reducing the police, releasing violent criminals and not prosecuting crime. This is the mob-rule agenda that I have warned in the books was coming for so long. Shootings in the area of Minneapolis where Floyd was arrested increased by 2,500 percent compared with the year before. Defunding the police over George Floyd has led to a big increase in dead people with many of them black. Police protection for politicians making these decisions stayed the same or increased as you would expect from professional hypocrites. The Cult doesn't actually want to abolish the police. It wants to abolish local control over the police and hand it to federal government as the psychopaths advance the Hunger Games Society. Many George Floyd protests turned into violent riots with black stores and businesses destroyed by fire and looting across America fuelled by Black Lives Matter. Woke doesn't do irony. If you want civil rights you must loot the liquor store and the supermarket and make off with a smart TV. It's the only way.

It's not a race war – it's a class war

Black people are patronised by privileged blacks and whites alike and told they are victims of white supremacy. I find it extraordinary to watch privileged blacks supporting the very system and bloodline networks behind the slave trade and parroting the same Cult-serving manipulative crap of their privileged white, often billionaire, associates. It is indeed not a race war but a class war and colour is just a diversion. Black Senator Cory Booker and black Congresswoman Maxine Waters, more residents of Nincompoopia, personify this. Once you tell people they are victims of someone else you devalue both their own responsibility for their plight and the power they have to impact on their reality and experience. Instead

we have: 'You are only in your situation because of whitey – turn on them and everything will change.' It won't change. Nothing changes in our lives unless *we* change it. Crucial to that is never seeing yourself as a victim and always as the creator of your reality. Life is a simple sequence of choice and consequence. Make different choices and you create different consequences. *You* have to make those choices – not Black Lives Matter, the Woke Mafia and anyone else that seeks to dictate your life. Who are they these Wokers, an emotional and psychological road traffic accident, to tell you what to do? Personal empowerment is the last thing the Cult and its Black Lives Matter want black people or anyone else to have. They claim to be defending the underdog while *creating* and perpetuating the underdog. The Cult's worst nightmare is human unity and if they are going to keep blacks, whites and every other race under economic servitude and control then the focus must be diverted from what they have in common to what they can be manipulated to believe divides them. Blacks have to be told that their poverty and plight is the fault of the white bloke living on the street in the same poverty and with the same plight they are experiencing. The difference is that your plight black people is due to him, a white supremacist with 'white privilege' living on the street. Don't unite as one human family against your mutual oppressors and suppressors – fight the oppressor with the white face who is as financially deprived as you are. The Cult knows that as its 'Covid' agenda moves into still new levels of extremism people are going to respond and it has been spreading the seeds of disunity everywhere to stop a united response to the evil that targets *all of us*.

Racist attacks on 'whiteness' are getting ever more outrageous and especially through the American Democratic Party which has an appalling history for anti-black racism. Barack Obama, Joe Biden, Hillary Clinton and Nancy Pelosi all eulogised about Senator Robert Byrd at his funeral in 2010 after a nearly 60-year career in Congress. Byrd was a brutal Ku Klux Klan racist and a violent abuser of Cathy O'Brien in MKUltra. He said he would never fight in the military 'with a negro by my side' and 'rather I should die a thousand times,

and see Old Glory trampled in the dirt never to rise again, than to see this beloved land of ours become degraded by race mongrels, a throwback to the blackest specimen from the wilds'. Biden called Byrd a 'very close friend and mentor'. These 'Woke' hypocrites are not anti-racist they are anti-poor and anti-people not of their perceived class. Here is an illustration of the scale of anti-white racism to which we have now descended. Seriously Woke and moronic *New York Times* contributor Damon Young described whiteness as a 'virus' that 'like other viruses will not die until there are no bodies left for it to infect'. He went on: '... the only way to stop it is to locate it, isolate it, extract it, and kill it.' Young can say that as a black man with no consequences when a white man saying the same in reverse would be facing a jail sentence. *That's* racism. We had super-Woke numbskull senators Tammy Duckworth and Mazie Hirono saying they would object to future Biden Cabinet appointments if he did not nominate more Asian Americans and Pacific Islanders. Never mind the ability of the candidate what do they look like? Duckworth said: 'I will vote for racial minorities and I will vote for LGBTQ, but anyone else I'm not voting for.' Appointing people on the grounds of race is illegal, but that was not a problem for this ludicrous pair. They were on-message and that's a free pass in any situation.

Critical race racism

White children are told at school they are intrinsically racist as they are taught the divisive 'critical race theory'. This claims that the law and legal institutions are inherently racist and that race is a socially constructed concept used by white people to further their economic and political interests at the expense of people of colour. White is a 'virus' as we've seen. Racial inequality results from 'social, economic, and legal differences that white people create between races to maintain white interests which leads to poverty and criminality in minority communities'. I must tell that to the white guy sleeping on the street. The principal of East Side Community School in New York sent white parents a manifesto that called on

them to become ‘white traitors’ and advocate for full ‘white abolition’. These people are teaching your kids when they urgently need a psychiatrist. The ‘school’ included a chart with ‘eight white identities’ that ranged from ‘white supremacist’ to ‘white abolition’ and defined the behaviour white people must follow to end ‘the regime of whiteness’. Woke blacks and their privileged white associates are acting exactly like the slave owners of old and Ku Klux Klan racists like Robert Byrd. They are too full of their own self-purity to see that, but it’s true. Racism is not a body type; it’s a state of mind that can manifest through any colour, creed or culture.

Another racial fraud is ‘*equity*’. Not equality of treatment and opportunity – equity. It’s a term spun as equality when it means something very different. Equality in its true sense is a raising up while ‘*equity*’ is a race to the bottom. Everyone in the same level of poverty is ‘*equity*’. Keep everyone down – that’s equity. The Cult doesn’t want anyone in the human family to be empowered and BLM leaders, like all these ‘anti-racist’ organisations, continue their privileged, pampered existence by perpetuating the perception of gathering racism. When is the last time you heard an ‘anti-racist’ or ‘anti-Semitism’ organisation say that acts of racism and discrimination have *fallen*? It’s not in the interests of their fund-raising and power to influence and the same goes for the professional soccer anti-racism operation, Kick It Out. Two things confirmed that the Black Lives Matter riots in the summer of 2020 were Cult creations. One was that while anti-lockdown protests were condemned in this same period for ‘transmitting ‘Covid’ the authorities supported mass gatherings of Black Lives Matter supporters. I even saw self-deluding people claiming to be doctors say the two types of protest were not the same. No – the non-existent ‘Covid’ was in favour of lockdowns and attacked those that protested against them while ‘Covid’ supported Black Lives Matter and kept well away from its protests. The whole thing was a joke and as lockdown protestors were arrested, often brutally, by reframed Face-Nappies we had the grotesque sight of police officers taking the knee to Black Lives Matter, a Cult-funded Marxist

organisation that supports violent riots and wants to destroy the nuclear family and white people.

He's not white? Shucks!

Woke obsession with race was on display again when ten people were shot dead in Boulder, Colorado, in March, 2021. Cult-owned Woke TV channels like CNN said the shooter appeared to be a white man and Wokers were on Twitter condemning 'violent white men' with the usual mantras. Then the shooter's name was released as Ahmad Al Aliwi Alissa, an anti-Trump Arab-American, and the sigh of disappointment could be heard five miles away. Never mind that ten people were dead and what that meant for their families. Race baiting was all that mattered to these sick Cult-serving people like Barack Obama who exploited the deaths to further divide America on racial grounds which is his job for the Cult. This is the man that 'racist' white Americans made the first black president of the United States and then gave him a second term. Not-very-bright Obama has become filthy rich on the back of that and today appears to have a big influence on the Biden administration. Even so he's still a downtrodden black man and a victim of white supremacy. This disingenuous fraud reveals the contempt he has for black people when he puts on a Deep South Alabama accent whenever he talks to them, no, *at* them.

Another BLM red flag was how the now fully-Woke (fully-Cult) and fully-virtue-signalled professional soccer authorities had their teams taking the knee before every match in support of Marxist Black Lives Matter. Soccer authorities and clubs displayed 'Black Lives Matter' on the players' shirts and flashed the name on electronic billboards around the pitch. Any fans that condemned what is a Freemasonic taking-the-knee ritual were widely condemned as you would expect from the Woke virtue-signallers of professional sport and the now fully-Woke media. We have reverse racism in which you are banned from criticising any race or culture except for white people for whom anything goes – say what you like, no problem. What has this got to do with racial harmony and

equality? We've had black supremacists from Black Lives Matter telling white people to fall to their knees in the street and apologise for their white supremacy. Black supremacists acting like white supremacist slave owners of the past couldn't breach their self-obsessed, race-obsessed sense of self-purity. Joe Biden appointed a race-obsessed black supremacist Kristen Clarke to head the Justice Department Civil Rights Division. Clarke claimed that blacks are endowed with 'greater mental, physical and spiritual abilities' than whites. If anyone reversed that statement they would be vilified. Clarke is on-message so no problem. She's never seen a black-white situation in which the black figure is anything but a virtuous victim and she heads the Civil Rights Division which should treat everyone the same or it isn't civil rights. Another perception of the Renegade Mind: If something or someone is part of the Cult agenda they will be supported by Woke governments and media no matter what. If they're not, they will be condemned and censored. It really is that simple and so racist Clarke prospers despite (make that because of) her racism.

The end of culture

Biden's administration is full of such racial, cultural and economic bias as the Cult requires the human family to be divided into warring factions. We are now seeing racially-segregated graduations and everything, but everything, is defined through the lens of perceived 'racism. We have 'racist' mathematics, 'racist' food and even 'racist' *plants*. World famous Kew Gardens in London said it was changing labels on plants and flowers to tell its pre-'Covid' more than two million visitors a year how racist they are. Kew director Richard Deverell said this was part of an effort to 'move quickly to decolonise collections' after they were approached by one Ajay Chhabra 'an actor with an insight into how sugar cane was linked to slavery'. They are *plants* you idiots. 'Decolonisation' in the Woke manual really means colonisation of society with its mentality and by extension colonisation by the Cult. We are witnessing a new Chinese-style 'Cultural Revolution' so essential to the success of all

Marxist takeovers. Our cultural past and traditions have to be swept away to allow a new culture to be built-back-better. Woke targeting of long-standing Western cultural pillars including historical monuments and cancelling of historical figures is what happened in the Mao revolution in China which ‘purged remnants of capitalist and traditional elements from Chinese society’ and installed Maoism as the dominant ideology’. For China see the Western world today and for ‘dominant ideology’ see Woke. Better still see Marxism or Maoism. The ‘Covid’ hoax has specifically sought to destroy the arts and all elements of Western culture from people meeting in a pub or restaurant to closing theatres, music venues, sports stadiums, places of worship and even banning *singing*. Destruction of Western society is also why criticism of any religion is banned except for Christianity which again is the dominant religion as white is the numerically-dominant race. Christianity may be fading rapidly, but its history and traditions are weaved through the fabric of Western society. Delete the pillars and other structures will follow until the whole thing collapses. I am not a Christian defending that religion when I say that. I have no religion. It’s just a fact. To this end Christianity has itself been turned Woke to usher its own downfall and its ranks are awash with ‘change agents’ – knowing and unknowing – at every level including Pope Francis (*definitely* knowing) and the clueless Archbishop of Canterbury Justin Welby (possibly not, but who can be sure?). Woke seeks to coordinate attacks on Western culture, traditions, and ways of life through ‘intersectionality’ defined as ‘the complex, cumulative way in which the effects of multiple forms of discrimination (such as racism, sexism, and classism) combine, overlap, or intersect especially in the experiences of marginalised individuals or groups’. Wade through the Orwellian Woke-speak and this means coordinating disparate groups in a common cause to overthrow freedom and liberal values.

The entire structure of public institutions has been infested with Woke – government at all levels, political parties, police, military, schools, universities, advertising, media and trade unions. This abomination has been achieved through the Cult web by appointing

Wokers to positions of power and battering non-Wokers into line through intimidation, isolation and threats to their job. Many have been fired in the wake of the empathy-deleted, vicious hostility of 'social justice' Wokers and the desire of gutless, spineless employers to virtue-signal their Wokeness. Corporations are filled with Wokers today, most notably those in Silicon Valley. Ironically at the top they are not Woke at all. They are only exploiting the mentality their Cult masters have created and funded to censor and enslave while the Wokers cheer them on until it's their turn. Thus the Woke 'liberal left' is an inversion of the traditional liberal left. Campaigning for justice on the grounds of power and wealth distribution has been replaced by campaigning for identity politics. The genuine traditional left would never have taken money from today's billionaire abusers of fairness and justice and nor would the billionaires have wanted to fund that genuine left. It would not have been in their interests to do so. The division of opinion in those days was between the haves and have nots. This all changed with Cult manipulated and funded identity politics. The division of opinion today is between Wokers and non-Wokers and not income brackets. Cult corporations and their billionaires may have taken wealth disparity to cataclysmic levels of injustice, but as long as they speak the language of Woke, hand out the dosh to the Woke network and censor the enemy they are 'one of us'. Billionaires who don't give a damn about injustice are laughing at them till their bellies hurt. Wokers are not even close to self-aware enough to see that. The transformed 'left' dynamic means that Wokers who drone on about 'social justice' are funded by billionaires that have destroyed social justice the world over. It's *why* they are billionaires.

The climate con

Nothing encapsulates what I have said more comprehensively than the hoax of human-caused global warming. I have detailed in my books over the years how Cult operatives and organisations were the pump-primers from the start of the climate con. A purpose-built vehicle for this is the Club of Rome established by the Cult in 1968

with the Rockefellers and Rothschilds centrally involved all along. Their gofer frontman Maurice Strong, a Canadian oil millionaire, hosted the Earth Summit in Rio de Janeiro, Brazil, in 1992 where the global ‘green movement’ really expanded in earnest under the guiding hand of the Cult. The Earth Summit established Agenda 21 through the Cult-created-and-owned United Nations to use the illusion of human-caused climate change to justify the transformation of global society to save the world from climate disaster. It is a No-Problem-Reaction-Solution sold through governments, media, schools and universities as whole generations have been terrified into believing that the world was going to end in their lifetimes unless what old people had inflicted upon them was stopped by a complete restructuring of how everything is done. Chill, kids, it’s all a hoax. Such restructuring is precisely what the Cult agenda demands (purely by coincidence of course). Today this has been given the codename of the Great Reset which is only an updated term for Agenda 21 and its associated Agenda 2030. The latter, too, is administered through the UN and was voted into being by the General Assembly in 2015. Both 21 and 2030 seek centralised control of all resources and food right down to the raindrops falling on your own land. These are some of the demands of Agenda 21 established in 1992. See if you recognise this society emerging today:

- End national sovereignty
- State planning and management of all land resources, ecosystems, deserts, forests, mountains, oceans and fresh water; agriculture; rural development; biotechnology; and ensuring ‘*equity*’
- The state to ‘define the role’ of business and financial resources
- Abolition of private property
- ‘Restructuring’ the family unit (see BLM)
- Children raised by the state
- People told what their job will be
- Major restrictions on movement
- Creation of ‘human settlement zones’

- Mass resettlement as people are forced to vacate land where they live
- Dumbing down education
- Mass global depopulation in pursuit of all the above

The United Nations was created as a Trojan horse for world government. With the climate con of critical importance to promoting that outcome you would expect the UN to be involved. Oh, it's involved all right. The UN is promoting Agenda 21 and Agenda 2030 justified by 'climate change' while also driving the climate hoax through its Intergovernmental Panel on Climate Change (IPCC), one of the world's most corrupt organisations. The IPCC has been lying ferociously and constantly since the day it opened its doors with the global media hanging unquestioningly on its every mendacious word. The Green movement is entirely Woke and has long lost its original environmental focus since it was co-opted by the Cult. An obsession with 'global warming' has deleted its values and scrambled its head. I experienced a small example of what I mean on a beautiful country walk that I have enjoyed several times a week for many years. The path merged into the fields and forests and you felt at one with the natural world. Then a 'Green' organisation, the Hampshire and Isle of Wight Wildlife Trust, took over part of the land and proceeded to cut down a large number of trees, including mature ones, to install a horrible big, bright steel 'this-is-ours-stay-out' fence that destroyed the whole atmosphere of this beautiful place. No one with a feel for nature would do that. Day after day I walked to the sound of chainsaws and a magnificent mature weeping willow tree that I so admired was cut down at the base of the trunk. When I challenged a Woke young girl in a green shirt (of course) about this vandalism she replied: 'It's a weeping willow – it will grow back.' This is what people are paying for when they donate to the Hampshire and Isle of Wight Wildlife Trust and many other 'green' organisations today. It is not the environmental movement that I knew and instead has become a support-system – as with Extinction Rebellion – for a very dark agenda.

Private jets for climate justice

The Cult-owned, Gates-funded, World Economic Forum and its founder Klaus Schwab were behind the emergence of Greta Thunberg to harness the young behind the climate agenda and she was invited to speak to the world at ... the UN. Schwab published a book, *Covid-19: The Great Reset* in 2020 in which he used the 'Covid' hoax and the climate hoax to lay out a new society straight out of Agenda 21 and Agenda 2030. Bill Gates followed in early 2021 when he took time out from destroying the world to produce a book in his name about the way to save it. Gates flies across the world in private jets and admitted that 'I probably have one of the highest greenhouse gas footprints of anyone on the planet ... my personal flying alone is gigantic.' He has also bid for the planet's biggest private jet operator. Other climate change saviours who fly in private jets include John Kerry, the US Special Presidential Envoy for Climate, and actor Leonardo DiCaprio, a 'UN Messenger of Peace with special focus on climate change'. These people are so full of bullshit they could corner the market in manure. We mustn't be sceptical, though, because the Gates book, *How to Avoid a Climate Disaster: The Solutions We Have and the Breakthroughs We Need*, is a genuine attempt to protect the world and not an obvious pile of excrement attributed to a mega-psychopath aimed at selling his masters' plans for humanity. The Gates book and the other shite-pile by Klaus Schwab could have been written by the same person and may well have been. Both use 'climate change' and 'Covid' as the excuses for their new society and by coincidence the Cult's World Economic Forum and Bill and Melinda Gates Foundation promote the climate hoax and hosted Event 201 which pre-empted with a 'simulation' the very 'coronavirus' hoax that would be simulated for real on humanity within weeks. The British 'royal' family is promoting the 'Reset' as you would expect through Prince 'climate change caused the war in Syria' Charles and his hapless son Prince William who said that we must 'reset our relationship with nature and our trajectory as a species' to avoid a climate disaster. Amazing how many promoters of the 'Covid' and 'climate change' control

systems are connected to Gates and the World Economic Forum. A ‘study’ in early 2021 claimed that carbon dioxide emissions must fall by the equivalent of a global lockdown roughly every two years for the next decade to save the planet. The ‘study’ appeared in the same period that the Schwab mob claimed in a video that lockdowns destroying the lives of billions are good because they make the earth ‘quieter’ with less ‘ambient noise’. They took down the video amid a public backlash for such arrogant, empathy-deleted stupidity You see, however, where they are going with this. Corinne Le Quéré, a professor at the Tyndall Centre for Climate Change Research, University of East Anglia, was lead author of the climate lockdown study, and she writes for ... the World Economic Forum. Gates calls in ‘his’ book for changing ‘every aspect of the economy’ (long-time Cult agenda) and for humans to eat synthetic ‘meat’ (predicted in my books) while cows and other farm animals are eliminated.

Australian TV host and commentator Alan Jones described what carbon emission targets would mean for farm animals in Australia alone if emissions were reduced as demanded by 35 percent by 2030 and zero by 2050:

Well, let’s take agriculture, the total emissions from agriculture are about 75 million tonnes of carbon dioxide, equivalent. Now reduce that by 35 percent and you have to come down to 50 million tonnes, I’ve done the maths. So if you take for example 1.5 million cows, you’re going to have to reduce the herd by 525,000 [by] 2030, nine years, that’s 58,000 cows a year. The beef herd’s 30 million, reduce that by 35 percent, that’s 10.5 million, which means 1.2 million cattle have to go every year between now and 2030. This is insanity!

There are 75 million sheep. Reduce that by 35 percent, that’s 26 million sheep, that’s almost 3 million a year. So under the Paris Agreement over 30 million beasts. dairy cows, cattle, pigs and sheep would go. More than 8,000 every minute of every hour for the next decade, do these people know what they’re talking about?

Clearly they don’t at the level of campaigners, politicians and administrators. The Cult *does* know; that’s the outcome it wants. We are faced with not just a war on humanity. Animals and the natural world are being targeted and I have been saying since the ‘Covid’ hoax began that the plan eventually was to claim that the ‘deadly virus’ is able to jump from animals, including farm animals and

domestic pets, to humans. Just before this book went into production came this story: 'Russia registers world's first Covid-19 vaccine for cats & dogs as makers of Sputnik V warn pets & farm animals could spread virus'. The report said 'top scientists warned that the deadly pathogen could soon begin spreading through homes and farms' and 'the next stage is the infection of farm and domestic animals'. Know the outcome and you'll see the journey. Think what that would mean for animals and keep your eye on a term called zoonosis or zoonotic diseases which transmit between animals and humans. The Cult wants to break the connection between animals and people as it does between people and people. Farm animals fit with the Cult agenda to transform food from natural to synthetic.

The gas of life is killing us

There can be few greater examples of Cult inversion than the condemnation of carbon dioxide as a dangerous pollutant when it is the gas of life. Without it the natural world would be dead and so we would all be dead. We breathe in oxygen and breathe out carbon dioxide while plants produce oxygen and absorb carbon dioxide. It is a perfect symbiotic relationship that the Cult wants to dismantle for reasons I will come to in the final two chapters. Gates, Schwab, other Cult operatives and mindless repeaters, want the world to be 'carbon neutral' by at least 2050 and the earlier the better. 'Zero carbon' is the cry echoed by lunatics calling for 'Zero Covid' when we already have it. These carbon emission targets will deindustrialise the world in accordance with Cult plans – the post-industrial, post-democratic society – and with so-called renewables like solar and wind not coming even close to meeting human energy needs blackouts and cold are inevitable. Texans got the picture in the winter of 2021 when a snow storm stopped wind turbines and solar panels from working and the lights went down along with water which relies on electricity for its supply system. Gates wants everything to be powered by electricity to ensure that his masters have the kill switch to stop all human activity, movement, cooking, water and warmth any time they like. The climate lie is so

stupendously inverted that it claims we must urgently reduce carbon dioxide when we *don't have enough*.

Co₂ in the atmosphere is a little above 400 parts per million when the optimum for plant growth is 2,000 ppm and when it falls anywhere near 150 ppm the natural world starts to die and so do we. It fell to as low as 280 ppm in an 1880 measurement in Hawaii and rose to 413 ppm in 2019 with industrialisation which is why the planet has become *greener* in the industrial period. How insane then that psychopathic madman Gates is not satisfied only with blocking the rise of Co₂. He's funding technology to suck it out of the atmosphere. The reason why will become clear. The industrial era is not destroying the world through Co₂ and has instead turned around a potentially disastrous ongoing fall in Co₂. Greenpeace co-founder and scientist Patrick Moore walked away from Greenpeace in 1986 and has exposed the green movement for fear-mongering and lies. He said that 500 million years ago there was *17 times* more Co₂ in the atmosphere than we have today and levels have been falling for hundreds of millions of years. In the last 150 million years Co₂ levels in Earth's atmosphere had reduced by *90 percent*. Moore said that by the time humanity began to unlock carbon dioxide from fossil fuels we were at '38 seconds to midnight' and in that sense: 'Humans are [the Earth's] salvation.' Moore made the point that only half the Co₂ emitted by fossil fuels stays in the atmosphere and we should remember that all pollution pouring from chimneys that we are told is carbon dioxide is in fact nothing of the kind. It's pollution. Carbon dioxide is an invisible gas.

William Happer, Professor of Physics at Princeton University and long-time government adviser on climate, has emphasised the Co₂ deficiency for maximum growth and food production. Greenhouse growers don't add carbon dioxide for a bit of fun. He said that most of the warming in the last 100 years, after the earth emerged from the super-cold period of the 'Little Ice Age' into a natural warming cycle, was over by 1940. Happer said that a peak year for warming in 1988 can be explained by a 'monster El Nino' which is a natural and cyclical warming of the Pacific that has nothing to do with 'climate

change'. He said the effect of Co2 could be compared to painting a wall with red paint in that once two or three coats have been applied it didn't matter how much more you slapped on because the wall will not get much redder. Almost all the effect of the rise in Co2 has already happened, he said, and the volume in the atmosphere would now have to *double* to increase temperature by a single degree. Climate hoaxers know this and they have invented the most ridiculously complicated series of 'feedback' loops to try to overcome this rather devastating fact. You hear puppet Greta going on cluelessly about feedback loops and this is why.

The Sun affects temperature? No you *climate denier*

Some other nonsense to contemplate: Climate graphs show that rises in temperature do not follow rises in Co2 – *it's the other way round* with a lag between the two of some 800 years. If we go back 800 years from present time we hit the Medieval Warm Period when temperatures were higher than now without any industrialisation and this was followed by the Little Ice Age when temperatures plummeted. The world was still emerging from these centuries of serious cold when many climate records began which makes the ever-repeated line of the 'hottest year since records began' meaningless when you are not comparing like with like. The coldest period of the Little Ice Age corresponded with the lowest period of sunspot activity when the Sun was at its least active. Proper scientists will not be at all surprised by this when it confirms the obvious fact that earth temperature is affected by the scale of Sun activity and the energetic power that it subsequently emits; but when is the last time you heard a climate hoaxter talking about the Sun as a source of earth temperature?? Everything has to be focussed on Co2 which makes up just 0.117 percent of so-called greenhouse gases and only a fraction of even that is generated by human activity. The rest is natural. More than 90 percent of those greenhouse gases are water vapour and clouds ([Fig 9](#)). Ban moisture I say. Have you noticed that the climate hoaxers no longer use the polar bear as their promotion image? That's because far from becoming extinct polar

bear communities are stable or thriving. Joe Bastardi, American meteorologist, weather forecaster and outspoken critic of the climate lie, documents in his book *The Climate Chronicles* how weather patterns and events claimed to be evidence of climate change have been happening since long before industrialisation: 'What happened before naturally is happening again, as is to be expected given the cyclical nature of the climate due to the design of the planet.' If you read the detailed background to the climate hoax in my other books you will shake your head and wonder how anyone could believe the crap which has spawned a multi-trillion dollar industry based on absolute garbage (see HIV causes AIDS and Sars-Cov-2 causes 'Covid-19'). Climate and 'Covid' have much in common given they have the same source. They both have the contradictory *everything* factor in which everything is explained by reference to them. It's hot – 'it's climate change'. It's cold – 'it's climate change'. I got a sniffle – 'it's Covid'. I haven't got a sniffle – 'it's Covid'. Not having a sniffle has to be a symptom of 'Covid'. Everything is and not having a sniffle is especially dangerous if you are a slow walker. For sheer audacity I offer you a Cambridge University 'study' that actually linked 'Covid' to 'climate change'. It had to happen eventually. They concluded that climate change played a role in 'Covid-19' spreading from animals to humans because ... wait for it ... I kid you not ... *the two groups were forced closer together as populations grow.* Er, that's it. The whole foundation on which this depended was that 'Bats are the likely zoonotic origin of SARS-CoV-1 and SARS-CoV-2'. Well, they are not. They are nothing to do with it. Apart from bats not being the origin and therefore 'climate change' effects on bats being irrelevant I am in awe of their academic insight. Where would we be without them? Not where we are that's for sure.

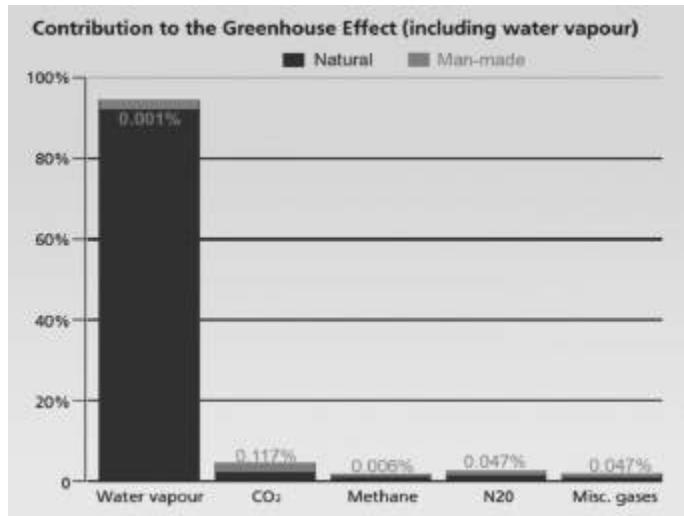


Figure 9: The idea that the gas of life is disastrously changing the climate is an insult to brain cell activity.

One other point about the weather is that climate modification is now well advanced and not every major weather event is natural – or earthquake come to that. I cover this subject at some length in other books. China is openly planning a rapid expansion of its weather modification programme which includes changing the climate in an area more than one and a half times the size of India. China used weather manipulation to ensure clear skies during the 2008 Olympics in Beijing. I have quoted from US military documents detailing how to employ weather manipulation as a weapon of war and they did that in the 1960s and 70s during the conflict in Vietnam with Operation Popeye manipulating monsoon rains for military purposes. Why would there be international treaties on weather modification if it wasn't possible? Of course it is. Weather is energetic information and it can be changed.

How was the climate hoax pulled off? See 'Covid'

If you can get billions to believe in a 'virus' that doesn't exist you can get them to believe in human-caused climate change that doesn't exist. Both are being used by the Cult to transform global society in the way it has long planned. Both hoaxes have been achieved in pretty much the same way. First you declare a lie is a fact. There's a

'virus' you call SARS-Cov-2 or humans are warming the planet with their behaviour. Next this becomes, via Cult networks, the foundation of government, academic and science policy and belief. Those who parrot the mantra are given big grants to produce research that confirms the narrative is true and ever more 'symptoms' are added to make the 'virus'/'climate change' sound even more scary. Scientists and researchers who challenge the narrative have their grants withdrawn and their careers destroyed. The media promote the lie as the unquestionable truth and censor those with an alternative view or evidence. A great percentage of the population believe what they are told as the lie becomes an everybody-knows-that and the believing-masses turn on those with a mind of their own. The technique has been used endlessly throughout human history. Wokers are the biggest promotorrs of the climate lie *and* 'Covid' fascism because their minds are owned by the Cult; their sense of self-righteous self-purity knows no bounds; and they exist in a bubble of reality in which facts are irrelevant and only get in the way of looking without seeing.

Running through all of this like veins in a blue cheese is control of information, which means control of perception, which means control of behaviour, which collectively means control of human society. The Cult owns the global media and Silicon Valley fascists for the simple reason that it *has* to. Without control of information it can't control perception and through that human society. Examine every facet of the Cult agenda and you will see that anything supporting its introduction is never censored while anything pushing back is always censored. I say again: Psychopaths that know why they are doing this must go before Nuremberg trials and those that follow their orders must trot along behind them into the same dock. 'I was just following orders' didn't work the first time and it must not work now. Nuremberg trials must be held all over the world before public juries for politicians, government officials, police, compliant doctors, scientists and virologists, and all Cult operatives such as Gates, Tedros, Fauci, Vallance, Whitty, Ferguson, Zuckerberg, Wojcicki, Brin, Page, Dorsey, the whole damn lot of

them – including, no *especially*, the psychopath psychologists. Without them and the brainless, gutless excuses for journalists that have repeated their lies, none of this could be happening. Nobody can be allowed to escape justice for the psychological and economic Armageddon they are all responsible for visiting upon the human race.

As for the compliant, unquestioning, swathes of humanity, and the self-obsessed, all-knowing ignorance of the Wokers ... don't start me. God help their kids. God help their grandkids. God *help them*.

CHAPTER NINE

We must have it? So what is it?

Well I won't back down. No, I won't back down. You can stand me up at the Gates of Hell. But I won't back down

Tom Petty

I will now focus on the genetically-manipulating ‘Covid vaccines’ which do not meet this official definition of a vaccine by the US Centers for Disease Control (CDC): ‘A product that stimulates a person’s immune system to produce immunity to a specific disease, protecting the person from that disease.’ On that basis ‘Covid vaccines’ are not a vaccine in that the makers don’t even claim they stop infection or transmission.

They are instead part of a multi-levelled conspiracy to change the nature of the human body and what it means to be ‘human’ and to depopulate an enormous swathe of humanity. What I shall call Human 1.0 is on the cusp of becoming Human 2.0 and for very sinister reasons. Before I get to the ‘Covid vaccine’ in detail here’s some background to vaccines in general. Government regulators do not test vaccines – the makers do – and the makers control which data is revealed and which isn’t. Children in America are given 50 vaccine doses by age six and 69 by age 19 and the effect of the whole combined schedule has never been tested. Autoimmune diseases when the immune system attacks its own body have soared in the mass vaccine era and so has disease in general in children and the young. Why wouldn’t this be the case when vaccines target the *immune system*? The US government gave Big Pharma drug

companies immunity from prosecution for vaccine death and injury in the 1986 National Childhood Vaccine Injury Act (NCVIA) and since then the government (taxpayer) has been funding compensation for the consequences of Big Pharma vaccines. The criminal and satanic drug giants can't lose and the vaccine schedule has increased dramatically since 1986 for this reason. There is no incentive to make vaccines safe and a big incentive to make money by introducing ever more. Even against a ridiculously high bar to prove vaccine liability, and with the government controlling the hearing in which it is being challenged for compensation, the vaccine court has so far paid out more than \$4 billion. These are the vaccines we are told are safe and psychopaths like Zuckerberg censor posts saying otherwise. The immunity law was even justified by a ruling that vaccines by their nature were 'unavoidably unsafe'.

Check out the ingredients of vaccines and you will be shocked if you are new to this. *They put that in children's bodies?? What??* Try aluminium, a brain toxin connected to dementia, aborted foetal tissue and formaldehyde which is used to embalm corpses. World-renowned aluminium expert Christopher Exley had his research into the health effect of aluminium in vaccines shut down by Keele University in the UK when it began taking funding from the Bill and Melinda Gates Foundation. Research when diseases 'eradicated' by vaccines began to decline and you will find the fall began long *before* the vaccine was introduced. Sometimes the fall even plateaued after the vaccine. Diseases like scarlet fever for which there was no vaccine declined in the same way because of environmental and other factors. A perfect case in point is the polio vaccine. Polio began when lead arsenate was first sprayed as an insecticide and residues remained in food products. Spraying started in 1892 and the first US polio epidemic came in Vermont in 1894. The simple answer was to stop spraying, but Rockefeller-created Big Pharma had a better idea. Polio was decreed to be caused by the *poliovirus* which 'spreads from person to person and can infect a person's spinal cord'. Lead arsenate was replaced by the lethal DDT which had the same effect of causing paralysis by damaging the brain and central nervous

system. Polio plummeted when DDT was reduced and then banned, but the vaccine is still given the credit for something it didn't do. Today by far the biggest cause of polio is the vaccines promoted by Bill Gates. Vaccine justice campaigner Robert Kennedy Jr, son of assassinated (by the Cult) US Attorney General Robert Kennedy, wrote:

In 2017, the World Health Organization (WHO) reluctantly admitted that the global explosion in polio is predominantly vaccine strain. The most frightening epidemics in Congo, Afghanistan, and the Philippines, are all linked to vaccines. In fact, by 2018, 70% of global polio cases were vaccine strain.

Vaccines make fortunes for Cult-owned Gates and Big Pharma while undermining the health and immune systems of the population. We had a glimpse of the mentality behind the Big Pharma cartel with a report on WION (World is One News), an international English language TV station based in India, which exposed the extraordinary behaviour of US drug company Pfizer over its 'Covid vaccine'. The WION report told how Pfizer had made fantastic demands of Argentina, Brazil and other countries in return for its 'vaccine'. These included immunity from prosecution, even for Pfizer negligence, government insurance to protect Pfizer from law suits and handing over as collateral sovereign assets of the country to include Argentina's bank reserves, military bases and embassy buildings. Pfizer demanded the same of Brazil in the form of waiving sovereignty of its assets abroad; exempting Pfizer from Brazilian laws; and giving Pfizer immunity from all civil liability. This is a 'vaccine' developed with government funding. Big Pharma is evil incarnate as a creation of the Cult and all must be handed tickets to Nuremberg.

Phantom 'vaccine' for a phantom 'disease'

I'll expose the 'Covid vaccine' fraud and then go on to the wider background of why the Cult has set out to 'vaccinate' every man, woman and child on the planet for an alleged 'new disease' with a survival rate of 99.77 percent (or more) even by the grotesquely-

manipulated figures of the World Health Organization and Johns Hopkins University. The ‘infection’ to ‘death’ ratio is 0.23 to 0.15 percent according to Stanford epidemiologist Dr John Ioannidis and while estimates vary the danger remains tiny. I say that if the truth be told the fake infection to fake death ratio is zero. Never mind all the evidence I have presented here and in *The Answer* that there is no ‘virus’ let us just focus for a moment on that death-rate figure of say 0.23 percent. The figure includes all those worldwide who have tested positive with a test not testing for the ‘virus’ and then died within 28 days or even longer of any other cause – *any other cause*. Now subtract all those illusory ‘Covid’ deaths on the global data sheets from the 0.23 percent. What do you think you would be left with? *Zero*. A vaccination has never been successfully developed for a so-called coronavirus. They have all failed at the animal testing stage when they caused hypersensitivity to what they were claiming to protect against and made the impact of a disease far worse. Cult-owned vaccine corporations got around that problem this time by bypassing animal trials, going straight to humans and making the length of the ‘trials’ before the public rollout as short as they could get away with. Normally it takes five to ten years or more to develop vaccines that still cause demonstrable harm to many people and that’s without including the long-term effects that are never officially connected to the vaccination. ‘Covid’ non-vaccines have been officially produced and approved in a matter of months from a standing start and part of the reason is that (a) they were developed before the ‘Covid’ hoax began and (b) they are based on computer programs and not natural sources. Official non-trials were so short that government agencies gave *emergency*, not full, approval. ‘Trials’ were not even completed and full approval cannot be secured until they are. Public ‘Covid vaccination’ is actually a *continuation of the trial*. Drug company ‘trials’ are not scheduled to end until 2023 by which time a lot of people are going to be dead. Data on which government agencies gave this emergency approval was supplied by the Big Pharma corporations themselves in the form of Pfizer/BioNTech, AstraZeneca, Moderna, Johnson & Johnson, and

others, and this is the case with all vaccines. By its very nature *emergency* approval means drug companies do not have to prove that the ‘vaccine’ is ‘safe and effective’. How could they with trials way short of complete? Government regulators only have to *believe* that they *could* be safe and effective. It is criminal manipulation to get products in circulation with no testing worth the name. Agencies giving that approval are infested with Big Pharma-connected place-people and they act in the interests of Big Pharma (the Cult) and not the public about whom they do not give a damn.

More human lab rats

‘Covid vaccines’ produced in record time by Pfizer/BioNTech and Moderna employ a technique *never approved before for use on humans*. They are known as mRNA ‘vaccines’ and inject a synthetic version of ‘viral’ mRNA or ‘messenger RNA’. The key is in the term ‘messenger’. The body works, or doesn’t, on the basis of information messaging. Communications are constantly passing between and within the genetic system and the brain. Change those messages and you change the state of the body and even its very nature and you can change psychology and behaviour by the way the brain processes information. I think you are going to see significant changes in personality and perception of many people who have had the ‘Covid vaccine’ synthetic potions. Insider Aldous Huxley predicted the following in 1961 and mRNA ‘vaccines’ can be included in the term ‘pharmacological methods’:

There will be, in the next generation or so, a pharmacological method of making people love their servitude, and producing dictatorship without tears, so to speak, producing a kind of painless concentration camp for entire societies, so that people will in fact have their own liberties taken away from them, but rather enjoy it, because they will be distracted from any desire to rebel by propaganda or brainwashing, or brainwashing enhanced by pharmacological methods. And this seems to be the final revolution.

Apologists claim that mRNA synthetic ‘vaccines’ don’t change the DNA genetic blueprint because RNA does not affect DNA only the other way round. This is so disingenuous. A process called ‘reverse

'transcription' can convert RNA into DNA and be integrated into DNA in the cell nucleus. This was highlighted in December, 2020, by scientists at Harvard and Massachusetts Institute of Technology (MIT). Geneticists report that more than 40 percent of mammalian genomes results from reverse transcription. On the most basic level if messaging changes then that sequence must lead to changes in DNA which is receiving and transmitting those communications. How can introducing synthetic material into cells not change the cells where DNA is located? The process is known as transfection which is defined as 'a technique to insert foreign nucleic acid (DNA or RNA) into a cell, typically with the intention of altering the properties of the cell'. Researchers at the Sloan Kettering Institute in New York found that changes in messenger RNA can deactivate tumour-suppressing proteins and thereby promote cancer. This is what happens when you mess with messaging. 'Covid vaccine' maker Moderna was founded in 2010 by Canadian stem cell biologist Derrick J. Rossi after his breakthrough discovery in the field of transforming and reprogramming stem cells. These are neutral cells that can be programmed to become any cell including sperm cells. Moderna was therefore founded on the principle of genetic manipulation and has never produced any vaccine or drug before its genetically-manipulating synthetic 'Covid' shite. Look at the name – Mode-RNA or Modify-RNA. Another important point is that the US Supreme Court has ruled that genetically-modified DNA, or complementary DNA (cDNA) synthesized in the laboratory from messenger RNA, can be patented and owned. These psychopaths are doing this to the human body.

Cells replicate synthetic mRNA in the 'Covid vaccines' and in theory the body is tricked into making antigens which trigger antibodies to target the 'virus spike proteins' which as Dr Tom Cowan said have *never been seen*. Cut the crap and these 'vaccines' deliver *self-replicating* synthetic material to the cells with the effect of changing human DNA. The more of them you have the more that process is compounded while synthetic material is all the time self-replicating. 'Vaccine'-maker Moderna describes mRNA as 'like

software for the cell' and so they are messing with the body's software. What happens when you change the software in a computer? Everything changes. For this reason the Cult is preparing a production line of mRNA 'Covid vaccines' and a long list of excuses to use them as with all the 'variants' of a 'virus' never shown to exist. The plan is further to transfer the mRNA technique to other vaccines mostly given to children and young people. The cumulative consequences will be a transformation of human DNA through a constant infusion of synthetic genetic material which will kill many and change the rest. Now consider that governments that have given emergency approval for a vaccine that's not a vaccine; never been approved for humans before; had no testing worth the name; and the makers have been given immunity from prosecution for any deaths or adverse effects suffered by the public. The UK government awarded *permanent legal indemnity* to itself and its employees for harm done when a patient is being treated for 'Covid-19' or 'suspected Covid-19'. That is quite a thought when these are possible 'side-effects' from the 'vaccine' (they are not 'side', they are effects) listed by the US Food and Drug Administration:

Guillain-Barre syndrome; acute disseminated encephalomyelitis; transverse myelitis; encephalitis; myelitis; encephalomyelitis; meningoencephalitis; meningitis; encephalopathy; convulsions; seizures; stroke; narcolepsy; cataplexy; anaphylaxis; acute myocardial infarction (heart attack); myocarditis; pericarditis; autoimmune disease; death; implications for pregnancy, and birth outcomes; other acute demyelinating diseases; non anaphylactic allergy reactions; thrombocytopenia ; disseminated intravascular coagulation; venous thromboembolism; arthritis; arthralgia; joint pain; Kawasaki disease; multisystem inflammatory syndrome in children; vaccine enhanced disease. The latter is the way the 'vaccine' has the potential to make diseases far worse than they would otherwise be.

UK doctor and freedom campaigner Vernon Coleman described the conditions in this list as 'all unpleasant, most of them very serious, and you can't get more serious than death'. The thought that anyone at all has had the 'vaccine' in these circumstances is testament to the potential that humanity has for clueless, unquestioning, stupidity and for many that programmed stupidity has already been terminal.

An insider speaks

Dr Michael Yeadon is a former Vice President, head of research and Chief Scientific Adviser at vaccine giant Pfizer. Yeadon worked on the inside of Big Pharma, but that did not stop him becoming a vocal critic of 'Covid vaccines' and their potential for multiple harms, including infertility in women. By the spring of 2021 he went much further and even used the no, no, term 'conspiracy'. When you begin to see what is going on it is impossible not to do so. Yeadon spoke out in an interview with freedom campaigner James Delingpole and I mentioned earlier how he said that no one had samples of 'the virus'. He explained that the mRNA technique originated in the anti-cancer field and ways to turn on and off certain genes which could be advantageous if you wanted to stop cancer growing out of control. 'That's the origin of them. They are a very unusual application, really.' Yeadon said that treating a cancer patient with an aggressive procedure might be understandable if the alternative was dying, but it was quite another thing to use the same technique as a public health measure. Most people involved wouldn't catch the infectious agent you were vaccinating against and if they did they probably wouldn't die:

If you are really using it as a public health measure you really want to as close as you can get to zero side-effects ... I find it odd that they chose techniques that were really cutting their teeth in the field of oncology and I'm worried that in using gene-based vaccines that have to be injected in the body and spread around the body, get taken up into some cells, and the regulators haven't quite told us which cells they get taken up into ... you are going to be generating a wide range of responses ... with multiple steps each of which could go well or badly.

I doubt the Cult intends it to go well. Yeadon said that you can put any gene you like into the body through the 'vaccine'. 'You can certainly give them a gene that would do them some harm if you wanted.' I was intrigued when he said that when used in the cancer field the technique could turn genes on and off. I explore this process in *The Answer* and with different genes having different functions you could create mayhem – physically and psychologically – if you turned the wrong ones on and the right ones off. I read reports of an experiment by researchers at the University of Washington's school of computer science and engineering in which they encoded DNA to infect computers. The body is itself a biological computer and if human DNA can inflict damage on a computer why can't the computer via synthetic material mess with the human body? It can. The Washington research team said it was possible to insert malicious malware into 'physical DNA strands' and corrupt the computer system of a gene sequencing machine as it 'reads gene letters and stores them as binary digits 0 and 1'. They concluded that hackers could one day use blood or spit samples to access computer systems and obtain sensitive data from police forensics labs or infect genome files. It is at this level of digital interaction that synthetic 'vaccines' need to be seen to get the full picture and that will become very clear later on. Michael Yeadon said it made no sense to give the 'vaccine' to younger people who were in no danger from the 'virus'. What was the benefit? It was all downside with potential effects:

The fact that my government in what I thought was a civilised, rational country, is raining [the 'vaccine'] on people in their 30s and 40s, even my children in their 20s, they're getting letters and phone calls, I know this is not right and any of you doctors who are vaccinating you know it's not right, too. They are not at risk. They are not at risk from the disease, so you are now hoping that the side-effects are so rare that you get away with it. You don't give new technology ... that you don't understand to 100 percent of the population.

Blood clot problems with the AstraZeneca 'vaccine' have been affecting younger people to emphasise the downside risks with no benefit. AstraZeneca's version, produced with Oxford University, does not use mRNA, but still gets its toxic cocktail inside cells where

it targets DNA. The Johnson & Johnson ‘vaccine’ which uses a similar technique has also produced blood clot effects to such an extent that the United States paused its use at one point. They are all ‘gene therapy’ (cell modification) procedures and not ‘vaccines’. The truth is that once the content of these injections enter cells we have no idea what the effect will be. People can speculate and some can give very educated opinions and that’s good. In the end, though, only the makers know what their potions are designed to do and even they won’t know every last consequence. Michael Yeadon was scathing about doctors doing what they knew to be wrong.

‘Everyone’s mute’, he said. Doctors in the NHS must know this was not right, coming into work and injecting people. ‘I don’t know how they sleep at night. I know I couldn’t do it. I know that if I were in that position I’d have to quit.’ He said he knew enough about toxicology to know this was not a good risk-benefit. Yeadon had spoken to seven or eight university professors and all except two would not speak out publicly. Their universities had a policy that no one said anything that countered the government and its medical advisors. They were afraid of losing their government grants. This is how intimidation has been used to silence the truth at every level of the system. I say silence, but these people could still speak out if they made that choice. Yeadon called them ‘moral cowards’ – ‘This is about your children and grandchildren’s lives and you have just buggered off and left it.’

‘Variant’ nonsense

Some of his most powerful comments related to the alleged ‘variants’ being used to instil more fear, justify more lockdowns, and introduce more ‘vaccines’. He said government claims about ‘variants’ were nonsense. He had checked the alleged variant ‘codes’ and they were 99.7 percent identical to the ‘original’. This was the human identity difference equivalent to putting a baseball cap on and off or wearing it the other way round. A 0.3 percent difference would make it impossible for that ‘variant’ to escape immunity from the ‘original’. This made no sense of having new ‘vaccines’ for

'variants'. He said there would have to be at least a *30 percent* difference for that to be justified and even then he believed the immune system would still recognise what it was. Gates-funded 'variant modeller' and 'vaccine'-pusher John Edmunds might care to comment. Yeadon said drug companies were making new versions of the 'vaccine' as a 'top up' for 'variants'. Worse than that, he said, the 'regulators' around the world like the MHRA in the UK had got together and agreed that because 'vaccines' for 'variants' were so similar to the first 'vaccines' *they did not have to do safety studies*. How transparently sinister that is. This is when Yeadon said: 'There is a conspiracy here.' There was no need for another vaccine for 'variants' and yet we were told that there was and the country had shut its borders because of them. 'They are going into hundreds of millions of arms without passing 'go' or any regulator. Why did they do that? Why did they pick this method of making the vaccine?'

The reason had to be something bigger than that it seemed and 'it's not protection against the virus'. It's was a far bigger project that meant politicians and advisers were willing to do things and not do things that knowingly resulted in avoidable deaths – 'that's already happened when you think about lockdown and deprivation of health care for a year.' He spoke of people prepared to do something that results in the avoidable death of their fellow human beings and it not bother them. This is the penny-drop I have been working to get across for more than 30 years – the level of pure evil we are dealing with. Yeadon said his friends and associates could not believe there could be that much evil, but he reminded them of Stalin, Pol Pot and Hitler and of what Stalin had said: 'One death is a tragedy. A million? A statistic.' He could not think of a benign explanation for why you need top-up vaccines 'which I'm sure you don't' and for the regulators 'to just get out of the way and wave them through'. Why would the regulators do that when they were still wrestling with the dangers of the 'parent' vaccine? He was clearly shocked by what he had seen since the 'Covid' hoax began and now he was thinking the previously unthinkable:

If you wanted to depopulate a significant proportion of the world and to do it in a way that doesn't involve destruction of the environment with nuclear weapons, poisoning everyone with anthrax or something like that, and you wanted plausible deniability while you had a multi-year infectious disease crisis, I actually don't think you could come up with a better plan of work than seems to be in front of me. I can't say that's what they are going to do, but I can't think of a benign explanation why they are doing it.

He said he never thought that they would get rid of 99 percent of humans, but now he wondered. 'If you wanted to that this would be a hell of a way to do it – it would be unstoppable folks.' Yeadon had concluded that those who submitted to the 'vaccine' would be allowed to have some kind of normal life (but for how long?) while screws were tightened to coerce and mandate the last few percent. 'I think they'll put the rest of them in a prison camp. I wish I was wrong, but I don't think I am.' Other points he made included: There were no coronavirus vaccines then suddenly they all come along at the same time; we have no idea of the long term affect with trials so short; coercing or forcing people to have medical procedures is against the Nuremberg Code instigated when the Nazis did just that; people should at least delay having the 'vaccine'; a quick Internet search confirms that masks don't reduce respiratory viral transmission and 'the government knows that'; they have smashed civil society and they know that, too; two dozen peer-reviewed studies show no connection between lockdown and reducing deaths; he knew from personal friends the elite were still flying around and going on holiday while the public were locked down; the elite were not having the 'vaccines'. He was also asked if 'vaccines' could be made to target difference races. He said he didn't know, but the document by the Project for the New American Century in September, 2000, said developing 'advanced forms of biological warfare that can target *specific genotypes* may transform biological warfare from the realm of terror to a politically useful tool.' Oh, they're evil all right. Of that we can be *absolutely* sure.

Another cull of old people

We have seen from the CDC definition that the mRNA 'Covid vaccine' is not a vaccine and nor are the others that *claim* to reduce 'severity of symptoms' in *some* people, but not protect from infection or transmission. What about all the lies about returning to 'normal' if people were 'vaccinated'? If they are not claimed to stop infection and transmission of the alleged 'virus', how does anything change? This was all lies to manipulate people to take the jabs and we are seeing that now with masks and distancing still required for the 'vaccinated'. How did they think that elderly people with fragile health and immune responses were going to be affected by infusing their cells with synthetic material and other toxic substances? They *knew* that in the short and long term it would be devastating and fatal as the culling of the old that began with the first lockdowns was continued with the 'vaccine'. Death rates in care homes soared immediately residents began to be 'vaccinated' – infused with synthetic material. Brave and committed whistleblower nurses put their careers at risk by exposing this truth while the rest kept their heads down and their mouths shut to put their careers before those they are supposed to care for. A long-time American Certified Nursing Assistant who gave his name as James posted a video in which he described emotionally what happened in his care home when vaccination began. He said that during 2020 very few residents were sick with 'Covid' and no one died during the entire year; but shortly after the Pfizer mRNA injections 14 people died within two weeks and many others were near death. 'They're dropping like flies', he said. Residents who walked on their own before the shot could no longer and they had lost their ability to conduct an intelligent conversation. The home's management said the sudden deaths were caused by a 'super-spreader' of 'Covid-19'. Then how come, James asked, that residents who refused to take the injections were not sick? It was a case of inject the elderly with mRNA synthetic potions and blame their illness and death that followed on the 'virus'. James described what was happening in care homes as 'the greatest crime of genocide this country has ever seen'. Remember the NHS staff nurse from earlier who used the same

word ‘genocide’ for what was happening with the ‘vaccines’ and that it was an ‘act of human annihilation’. A UK care home whistleblower told a similar story to James about the effect of the ‘vaccine’ in deaths and ‘outbreaks’ of illness dubbed ‘Covid’ after getting the jab. She told how her care home management and staff had zealously imposed government regulations and no one was allowed to even question the official narrative let alone speak out against it. She said the NHS was even worse. Again we see the results of reframing. A worker at a local care home where I live said they had not had a single case of ‘Covid’ there for almost a year and when the residents were ‘vaccinated’ they had 19 positive cases in two weeks with eight dying.

It's not the 'vaccine' – honest

The obvious cause and effect was being ignored by the media and most of the public. Australia’s health minister Greg Hunt (a former head of strategy at the World Economic Forum) was admitted to hospital after he had the ‘vaccine’. He was suffering according to reports from the skin infection ‘cellulitis’ and it must have been a severe case to have warranted days in hospital. Immediately the authorities said this was nothing to do with the ‘vaccine’ when an effect of some vaccines is a ‘cellulitis-like reaction’. We had families of perfectly healthy old people who died after the ‘vaccine’ saying that if only they had been given the ‘vaccine’ earlier they would still be alive. As a numbskull rating that is off the chart. A father of four ‘died of Covid’ at aged 48 when he was taken ill two days after having the ‘vaccine’. The man, a health administrator, had been ‘shielding during the pandemic’ and had ‘not really left the house’ until he went for the ‘vaccine’. Having the ‘vaccine’ and then falling ill and dying does not seem to have qualified as a possible cause and effect and ‘Covid-19’ went on his death certificate. His family said they had no idea how he ‘caught the virus’. A family member said: ‘Tragically, it could be that going for a vaccination ultimately led to him catching Covid ...The sad truth is that they are never going to know where it came from.’ The family warned people to remember

that the virus still existed and was 'very real'. So was their stupidity. Nurses and doctors who had the first round of the 'vaccine' were collapsing, dying and ending up in a hospital bed while they or their grieving relatives were saying they'd still have the 'vaccine' again despite what happened. I kid you not. You mean if your husband returned from the dead he'd have the same 'vaccine' again that killed him??

Doctors at the VCU Medical Center in Richmond, Virginia, said the Johnson & Johnson 'vaccine' was to blame for a man's skin peeling off. Patient Richard Terrell said: 'It all just happened so fast. My skin peeled off. It's still coming off on my hands now.' He said it was stinging, burning and itching and when he bent his arms and legs it was very painful with 'the skin swollen and rubbing against itself'. Pfizer/BioNTech and Moderna vaccines use mRNA to change the cell while the Johnson & Johnson version uses DNA in a process similar to AstraZeneca's technique. Johnson & Johnson and AstraZeneca have both had their 'vaccines' paused by many countries after causing serious blood problems. Terrell's doctor Fnu Nutan said he could have died if he hadn't got medical attention. It sounds terrible so what did Nutan and Terrell say about the 'vaccine' now? Oh, they still recommend that people have it. A nurse in a hospital bed 40 minutes after the vaccination and unable to swallow due to throat swelling was told by a doctor that he lost mobility in his arm for 36 hours following the vaccination. What did he say to the ailing nurse? 'Good for you for getting the vaccination.' We are dealing with a serious form of cognitive dissonance madness in both public and medical staff. There is a remarkable correlation between those having the 'vaccine' and trumpeting the fact and suffering bad happenings shortly afterwards. Witold Rogiewicz, a Polish doctor, made a video of his 'vaccination' and ridiculed those who were questioning its safety and the intentions of Bill Gates: 'Vaccinate yourself to protect yourself, your loved ones, friends and also patients. And to mention quickly I have info for anti-vaxxers and anti-Covidiers if you want to contact Bill Gates you can do this through me.' He further ridiculed the dangers of 5G. Days later he

was dead, but naturally the vaccination wasn't mentioned in the verdict of 'heart attack'.

Lies, lies and more lies

So many members of the human race have slipped into extreme states of insanity and unfortunately they include reframed doctors and nursing staff. Having a 'vaccine' and dying within minutes or hours is not considered a valid connection while death from any cause within 28 days or longer of a positive test with a test not testing for the 'virus' means 'Covid-19' goes on the death certificate. How could that 'vaccine'-death connection not have been made except by calculated deceit? US figures in the initial rollout period to February 12th, 2020, revealed that a third of the deaths reported to the CDC after 'Covid vaccines' happened within 48 hours. Five men in the UK suffered an 'extremely rare' blood clot problem after having the AstraZeneca 'vaccine', but no causal link was established said the Gates-funded Medicines and Healthcare products Regulatory Agency (MHRA) which had given the 'vaccine' emergency approval to be used. Former Pfizer executive Dr Michael Yeadon explained in his interview how the procedures could cause blood coagulation and clots. People who should have been at no risk were dying from blood clots in the brain and he said he had heard from medical doctor friends that people were suffering from skin bleeding and massive headaches. The AstraZeneca 'shot' was stopped by some 20 countries over the blood clotting issue and still the corrupt MHRA, the European Medicines Agency (EMA) and the World Health Organization said that it should continue to be given even though the EMA admitted that it 'still cannot rule out definitively' a link between blood clotting and the 'vaccine'. Later Marco Cavaleri, head of EMA vaccine strategy, said there was indeed a clear link between the 'vaccine' and thrombosis, but they didn't know why. So much for the trials showing the 'vaccine' is safe. Blood clots were affecting younger people who would be under virtually no danger from 'Covid' even if it existed which makes it all the more stupid and sinister.

The British government responded to public alarm by wheeling out June Raine, the terrifyingly weak infant school headmistress sound-alike who heads the UK MHRA drug ‘regulator’. The idea that she would stand up to Big Pharma and government pressure is laughable and she told us that all was well in the same way that she did when allowing untested, never-used-on-humans-before, genetically-manipulating ‘vaccines’ to be exposed to the public in the first place. Mass lying is the new normal of the ‘Covid’ era. The MHRA later said 30 cases of rare blood clots had by then been connected with the AstraZeneca ‘vaccine’ (that means a lot more in reality) while stressing that the benefits of the jab in preventing ‘Covid-19’ outweighed any risks. A more ridiculous and disingenuous statement with callous disregard for human health it is hard to contemplate. Immediately after the mendacious ‘all-clears’ two hospital workers in Denmark experienced blood clots and cerebral haemorrhaging following the AstraZeneca jab and one died. Top Norwegian health official Pål Andre Holme said the ‘vaccine’ was the only common factor: ‘There is nothing in the patient history of these individuals that can give such a powerful immune response ... I am confident that the antibodies that we have found are the cause, and I see no other explanation than it being the vaccine which triggers it.’ Strokes, a clot or bleed in the brain, were clearly associated with the ‘vaccine’ from word of mouth and whistleblower reports. Similar consequences followed with all these ‘vaccines’ that we were told were so safe and as the numbers grew by the day it was clear we were witnessing human carnage.

Learning the hard way

A woman interviewed by UKColumn told how her husband suffered dramatic health effects after the vaccine when he’d been in good health all his life. He went from being a little unwell to losing all feeling in his legs and experiencing ‘excruciating pain’. Misdiagnosis followed twice at Accident and Emergency (an ‘allergy’ and ‘sciatica’) before he was admitted to a neurology ward where doctors said his serious condition had been caused by the

'vaccine'. Another seven 'vaccinated' people were apparently being treated on the same ward for similar symptoms. The woman said he had the 'vaccine' because they believed media claims that it was safe. 'I didn't think the government would give out a vaccine that does this to somebody; I believed they would be bringing out a vaccination that would be safe.' What a tragic way to learn that lesson. Another woman posted that her husband was transporting stroke patients to hospital on almost every shift and when he asked them if they had been 'vaccinated' for 'Covid' they all replied 'yes'. One had a 'massive brain bleed' the day after his second dose. She said her husband reported the 'just been vaccinated' information every time to doctors in A and E only for them to ignore it, make no notes and appear annoyed that it was even mentioned. This particular report cannot be verified, but it expresses a common theme that confirms the monumental underreporting of 'vaccine' consequences. Interestingly as the 'vaccines' and their brain blood clot/stroke consequences began to emerge the UK National Health Service began a publicity campaign telling the public what to do in the event of a stroke. A Scottish NHS staff nurse who quit in disgust in March, 2021, said:

I have seen traumatic injuries from the vaccine, they're not getting reported to the yellow card [adverse reaction] scheme, they're treating the symptoms, not asking why, why it's happening. It's just treating the symptoms and when you speak about it you're dismissed like you're crazy, I'm not crazy, I'm not crazy because every other colleague I've spoken to is terrified to speak out, they've had enough.

Videos appeared on the Internet of people uncontrollably shaking after the 'vaccine' with no control over muscles, limbs and even their face. A Scottish mother broke out in a severe rash all over her body almost immediately after she was given the AstraZeneca 'vaccine'. The pictures were horrific. Leigh King, a 41-year-old hairdresser from Lanarkshire said: 'Never in my life was I prepared for what I was about to experience ... My skin was so sore and constantly hot ... I have never felt pain like this ...' But don't you worry, the 'vaccine' is perfectly safe. Then there has been the effect on medical

staff who have been pressured to have the ‘vaccine’ by psychopathic ‘health’ authorities and government. A London hospital consultant who gave the name K. Polyakova wrote this to the *British Medical Journal* or *BMJ*:

I am currently struggling with ... the failure to report the reality of the morbidity caused by our current vaccination program within the health service and staff population. The levels of sickness after vaccination is unprecedented and staff are getting very sick and some with neurological symptoms which is having a huge impact on the health service function. Even the young and healthy are off for days, some for weeks, and some requiring medical treatment. Whole teams are being taken out as they went to get vaccinated together.

Mandatory vaccination in this instance is stupid, unethical and irresponsible when it comes to protecting our staff and public health. We are in the voluntary phase of vaccination, and encouraging staff to take an unlicensed product that is impacting on their immediate health ... it is clearly stated that these vaccine products do not offer immunity or stop transmission. In which case why are we doing it?

Not to protect health that’s for sure. Medical workers are lauded by governments for agenda reasons when they couldn’t give a toss about them any more than they can for the population in general. Schools across America faced the same situation as they closed due to the high number of teachers and other staff with bad reactions to the Pfizer/BioNTech, Moderna, and Johnson & Johnson ‘Covid vaccines’ all of which were linked to death and serious adverse effects. The *BMJ* took down the consultant’s comments pretty quickly on the grounds that they were being used to spread ‘disinformation’. They were exposing the truth about the ‘vaccine’ was the real reason. The cover-up is breathtaking.

Hiding the evidence

The scale of the ‘vaccine’ death cover-up worldwide can be confirmed by comparing official figures with the personal experience of the public. I heard of many people in my community who died immediately or soon after the vaccine that would never appear in the media or even likely on the official totals of ‘vaccine’ fatalities and adverse reactions when only about ten percent are estimated to be

reported and I have seen some estimates as low as one percent in a Harvard study. In the UK alone by April 29th, 2021, some 757,654 adverse reactions had been officially reported from the Pfizer/BioNTech, Oxford/AstraZeneca and Moderna 'vaccines' with more than a thousand deaths linked to jabs and that means an estimated ten times this number in reality from a ten percent reporting rate percentage. That's seven million adverse reactions and 10,000 potential deaths and a one percent reporting rate would be ten times *those* figures. In 1976 the US government pulled the swine flu vaccine after 53 deaths. The UK data included a combined 10,000 eye disorders from the 'Covid vaccines' with more than 750 suffering visual impairment or blindness and again multiply by the estimated reporting percentages. As 'Covid cases' officially fell hospitals virtually empty during the 'Covid crisis' began to fill up with a range of other problems in the wake of the 'vaccine' rollout. The numbers across America have also been catastrophic. Deaths linked to *all* types of vaccine increased by *6,000 percent* in the first quarter of 2021 compared with 2020. A 39-year-old woman from Ogden, Utah, died four days after receiving a second dose of Moderna's 'Covid vaccine' when her liver, heart and kidneys all failed despite the fact that she had no known medical issues or conditions. Her family sought an autopsy, but Dr Erik Christensen, Utah's chief medical examiner, said proving vaccine injury as a cause of death almost never happened. He could think of only one instance where an autopsy would name a vaccine as the official cause of death and that would be anaphylaxis where someone received a vaccine and died almost instantaneously. 'Short of that, it would be difficult for us to definitively say this is the vaccine,' Christensen said. If that is true this must be added to the estimated ten percent (or far less) reporting rate of vaccine deaths and serious reactions and the conclusion can only be that vaccine deaths and serious reactions – including these 'Covid' potions – are phenomenally understated in official figures. The same story can be found everywhere. Endless accounts of deaths and serious reactions among the public, medical

and care home staff while official figures did not even begin to reflect this.

Professional script-reader Dr David Williams, a ‘top public-health official’ in Ontario, Canada, insulted our intelligence by claiming only four serious adverse reactions and no deaths from the more than 380,000 vaccine doses then given. This bore no resemblance to what people knew had happened in their own circles and we had Dirk Huyer in charge of getting millions vaccinated in Ontario while at the same time he was Chief Coroner for the province investigating causes of death including possible death from the vaccine. An aide said he had stepped back from investigating deaths, but evidence indicated otherwise. Rosemary Frei, who secured a Master of Science degree in molecular biology at the Faculty of Medicine at Canada’s University of Calgary before turning to investigative journalism, was one who could see that official figures for ‘vaccine’ deaths and reactions made no sense. She said that doctors seldom reported adverse events and when people got really sick or died after getting a vaccination they would attribute that to anything except the vaccines. It had been that way for years and anyone who wondered aloud whether the ‘Covid vaccines’ or other shots cause harm is immediately branded as ‘anti-vax’ and ‘anti-science’. This was ‘career-threatening’ for health professionals. Then there was the huge pressure to support the push to ‘vaccinate’ billions in the quickest time possible. Frei said:

So that’s where we’re at today. More than half a million vaccine doses have been given to people in Ontario alone. The rush is on to vaccinate all 15 million of us in the province by September. And the mainstream media are screaming for this to be sped up even more. That all adds up to only a very slim likelihood that we’re going to be told the truth by officials about how many people are getting sick or dying from the vaccines.

What is true of Ontario is true of everywhere.

They KNEW – and still did it

The authorities knew what was going to happen with multiple deaths and adverse reactions. The UK government’s Gates-funded

and Big Pharma-dominated Medicines and Healthcare products Regulatory Agency (MHRA) hired a company to employ AI in compiling the projected reactions to the ‘vaccine’ that would otherwise be uncountable. The request for applications said: ‘The MHRA urgently seeks an Artificial Intelligence (AI) software tool to process the expected high volume of Covid-19 vaccine Adverse Drug Reaction ...’ This was from the agency, headed by the disingenuous June Raine, that gave the ‘vaccines’ emergency approval and the company was hired before the first shot was given. ‘We are going to kill and maim you – is that okay?’ ‘Oh, yes, perfectly fine – I’m very grateful, thank you, doctor.’ The range of ‘Covid vaccine’ adverse reactions goes on for page after page in the MHRA criminally underreported ‘Yellow Card’ system and includes affects to eyes, ears, skin, digestion, blood and so on. Raine’s MHRA amazingly claimed that the ‘overall safety experience ... is so far as expected from the clinical trials’. The death, serious adverse effects, deafness and blindness were *expected*? When did they ever mention that? If these human tragedies were expected then those that gave approval for the use of these ‘vaccines’ must be guilty of crimes against humanity including murder – a definition of which is ‘killing a person with malice aforethought or with recklessness manifesting extreme indifference to the value of human life.’ People involved at the MHRA, the CDC in America and their equivalent around the world must go before Nuremberg trials to answer for their callous inhumanity. We are only talking here about the immediate effects of the ‘vaccine’. The longer-term impact of the DNA synthetic manipulation is the main reason they are so hysterically desperate to inoculate the entire global population in the shortest possible time.

Africa and the developing world are a major focus for the ‘vaccine’ depopulation agenda and a mass vaccination sales-pitch is underway thanks to caring people like the Rockefellers and other Cult assets. The Rockefeller Foundation, which pre-empted the ‘Covid pandemic’ in a document published in 2010 that ‘predicted’ what happened a decade later, announced an initial \$34.95 million grant in February, 2021, ‘to ensure more equitable access to Covid-19

testing and vaccines' among other things in Africa in collaboration with '24 organizations, businesses, and government agencies'. The pan-Africa initiative would focus on 10 countries: Burkina Faso, Ethiopia, Ghana, Kenya, Nigeria, Rwanda, South Africa, Tanzania, Uganda, and Zambia'. Rajiv Shah, President of the Rockefeller Foundation and former administrator of CIA-controlled USAID, said that if Africa was not mass-vaccinated (to change the DNA of its people) it was a 'threat to all of humanity' and not fair on Africans. When someone from the Rockefeller Foundation says they want to do something to help poor and deprived people and countries it is time for a belly-laugh. They are doing this out of the goodness of their 'heart' because 'vaccinating' the entire global population is what the 'Covid' hoax set out to achieve. Official 'decolonisation' of Africa by the Cult was merely a prelude to financial colonisation on the road to a return to physical colonisation. The 'vaccine' is vital to that and the sudden and convenient death of the 'Covid' sceptic president of Tanzania can be seen in its true light. A lot of people in Africa are aware that this is another form of colonisation and exploitation and they need to stand their ground.

The 'vaccine is working' scam

A potential problem for the Cult was that the 'vaccine' is meant to change human DNA and body messaging and not to protect anyone from a 'virus' never shown to exist. The vaccine couldn't work because it was not designed to work and how could they make it *appear* to be working so that more people would have it? This was overcome by lowering the amplification rate of the PCR test to produce fewer 'cases' and therefore fewer 'deaths'. Some of us had been pointing out since March, 2020, that the amplification rate of the test not testing for the 'virus' had been made artificially high to generate positive tests which they could call 'cases' to justify lockdowns. The World Health Organization recommended an absurdly high 45 amplification cycles to ensure the high positives required by the Cult and then remained silent on the issue until January 20th, 2021 – Biden's Inauguration Day. This was when the

'vaccinations' were seriously underway and on that day the WHO recommended after discussions with America's CDC that laboratories *lowered their testing amplification*. Dr David Samadi, a certified urologist and health writer, said the WHO was encouraging all labs to reduce their cycle count for PCR tests. He said the current cycle was much too high and was 'resulting in any particle being declared a positive case'. Even one mainstream news report I saw said this meant the number of 'Covid' infections may have been 'dramatically inflated'. Oh, just a little bit. The CDC in America issued new guidance to laboratories in April, 2021, to use 28 cycles *but only for 'vaccinated' people*. The timing of the CDC/WHO interventions were cynically designed to make it appear the 'vaccines' were responsible for falling cases and deaths when the real reason can be seen in the following examples. New York's state lab, the Wadsworth Center, identified 872 positive tests in July, 2020, based on a threshold of 40 cycles. When the figure was lowered to 35 cycles *43 percent* of the 872 were no longer 'positives'. At 30 cycles the figure was 63 percent. A Massachusetts lab found that between *85 to 90 percent* of people who tested positive in July with a cycle threshold of 40 would be negative at 30 cycles, Ashish Jha, MD, director of the Harvard Global Health Institute, said: 'I'm really shocked that it could be that high ... Boy, does it really change the way we need to be thinking about testing.' I'm shocked that I could see the obvious in the spring of 2020, with no medical background, and most medical professionals still haven't worked it out. No, that's not shocking – it's terrifying.

Three weeks after the WHO directive to lower PCR cycles the London *Daily Mail* ran this headline: 'Why ARE Covid cases plummeting? New infections have fallen 45% in the US and 30% globally in the past 3 weeks but experts say vaccine is NOT the main driver because only 8% of Americans and 13% of people worldwide have received their first dose.' They acknowledged that the drop could not be attributed to the 'vaccine', but soon this morphed throughout the media into the 'vaccine' has caused cases and deaths to fall when it was the PCR threshold. In December, 2020, there was

chaos at English Channel ports with truck drivers needing negative 'Covid' tests before they could board a ferry home for Christmas. The government wanted to remove the backlog as fast as possible and they brought in troops to do the 'testing'. Out of 1,600 drivers just 36 tested positive and the rest were given the all clear to cross the Channel. I guess the authorities thought that 36 was the least they could get away with without the unquestioning catching on. The amplification trick which most people believed in the absence of information in the mainstream applied more pressure on those refusing the 'vaccine' to succumb when it 'obviously worked'. The truth was the exact opposite with deaths in care homes soaring with the 'vaccine' and in Israel the term used was 'skyrocket'. A re-analysis of published data from the Israeli Health Ministry led by Dr Hervé Seligmann at the Medicine Emerging Infectious and Tropical Diseases at Aix-Marseille University found that Pfizer's 'Covid vaccine' killed 'about 40 times more [elderly] people than the disease itself would have killed' during a five-week vaccination period and 260 *times* more younger people than would have died from the 'virus' even according to the manipulated 'virus' figures. Dr Seligmann and his co-study author, Haim Yativ, declared after reviewing the Israeli 'vaccine' death data: 'This is a new Holocaust.'

Then, in mid-April, 2021, after vast numbers of people worldwide had been 'vaccinated', the story changed with clear coordination. The UK government began to prepare the ground for more future lockdowns when Nuremberg-destined Boris Johnson told yet another whopper. He said that cases had fallen because of *lockdowns* not 'vaccines'. Lockdowns are irrelevant when *there is no 'virus'* and the test and fraudulent death certificates are deciding the number of 'cases' and 'deaths'. Study after study has shown that lockdowns don't work and instead kill and psychologically destroy people. Meanwhile in the United States Anthony Fauci and Rochelle Walensky, the ultra-Zionist head of the CDC, peddled the same line. More lockdown was the answer and not the 'vaccine', a line repeated on cue by the moron that is Canadian Prime Minister Justin Trudeau. Why all the hysteria to get everyone 'vaccinated' if lockdowns and

not ‘vaccines’ made the difference? None of it makes sense on the face of it. Oh, but it does. The Cult wants lockdowns *and* the ‘vaccine’ and if the ‘vaccine’ is allowed to be seen as the total answer lockdowns would no longer be justified when there are still livelihoods to destroy. ‘Variants’ and renewed upward manipulation of PCR amplification are planned to instigate never-ending lockdown *and* more ‘vaccines’.

You must have it – we’re desperate

Israel, where the Jewish and Arab population are ruled by the Sabbatian Cult, was the front-runner in imposing the DNA-manipulating ‘vaccine’ on its people to such an extent that Jewish refusers began to liken what was happening to the early years of Nazi Germany. This would seem to be a fantastic claim. Why would a government of Jewish people be acting like the Nazis did? If you realise that the Sabbatian Cult was behind the Nazis and that Sabbatians hate Jews the pieces start to fit and the question of why a ‘Jewish’ government would treat Jews with such callous disregard for their lives and freedom finds an answer. Those controlling the government of Israel *aren’t Jewish* – they’re Sabbatian. Israeli lawyer Tamir Turgal was one who made the Nazi comparison in comments to German lawyer Reiner Fuellmich who is leading a class action lawsuit against the psychopaths for crimes against humanity. Turgal described how the Israeli government was vaccinating children and pregnant women on the basis that there was no evidence that this was dangerous when they had no evidence that it *wasn’t* dangerous either. They just had no evidence. This was medical experimentation and Turgal said this breached the Nuremberg Code about medical experimentation and procedures requiring informed consent and choice. Think about that. A Nuremberg Code developed because of Nazi experimentation on Jews and others in concentration camps by people like the evil-beyond-belief Josef Mengele is being breached by the *Israeli* government; but when you know that it’s a *Sabbatian* government along with its intelligence and military agencies like Mossad, Shin Bet and the Israeli Defense Forces, and that Sabbatians

were the force behind the Nazis, the kaleidoscope comes into focus. What have we come to when Israeli Jews are suing their government for violating the Nuremberg Code by essentially making Israelis subject to a medical experiment using the controversial 'vaccines'? It's a shocker that this has to be done in the light of what happened in Nazi Germany. The Anshe Ha-Emet, or 'People of the Truth', made up of Israeli doctors, lawyers, campaigners and public, have launched a lawsuit with the International Criminal Court. It says:

When the heads of the Ministry of Health as well as the prime minister presented the vaccine in Israel and began the vaccination of Israeli residents, the vaccinated were not advised, that, in practice, they are taking part in a medical experiment and that their consent is required for this under the Nuremberg Code.

The irony is unbelievable, but easily explained in one word: Sabbatians. The foundation of Israeli 'Covid' apartheid is the 'green pass' or 'green passport' which allows Jews and Arabs who have had the DNA-manipulating 'vaccine' to go about their lives – to work, fly, travel in general, go to shopping malls, bars, restaurants, hotels, concerts, gyms, swimming pools, theatres and sports venues, while non-'vaccinated' are banned from all those places and activities. Israelis have likened the 'green pass' to the yellow stars that Jews in Nazi Germany were forced to wear – the same as the yellow stickers that a branch of UK supermarket chain Morrisons told exempt mask-wears they had to display when shopping. How very sensitive. The Israeli system is blatant South African-style apartheid on the basis of compliance or non-compliance to fascism rather than colour of the skin. How appropriate that the Sabbatian Israeli government was so close to the pre-Mandela apartheid regime in Pretoria. The Sabbatian-instigated 'vaccine passport' in Israel is planned for everywhere. Sabbatians struck a deal with Pfizer that allowed them to lead the way in the percentage of a national population infused with synthetic material and the result was catastrophic. Israeli freedom activist Shai Dannon told me how chairs were appearing on beaches that said 'vaccinated only'. Health Minister Yuli Edelstein said that anyone unwilling or unable to get

the jabs that ‘confer immunity’ will be ‘left behind’. The man’s a liar. Not even the makers claim the ‘vaccines’ confer immunity. When you see those figures of ‘vaccine’ deaths these psychopaths were saying that you must take the chance the ‘vaccine’ will kill you or maim you while knowing it will change your DNA or lockdown for you will be permanent. That’s fascism. The Israeli parliament passed a law to allow personal information of the non-vaccinated to be shared with local and national authorities for three months. This was claimed by its supporters to be a way to ‘encourage’ people to be vaccinated. Hadas Ziv from Physicians for Human Rights described this as a ‘draconian law which crushed medical ethics and the patient rights’. But that’s the idea, the Sabbatians would reply.

Your papers, please

Sabbatian Israel was leading what has been planned all along to be a global ‘vaccine pass’ called a ‘green passport’ without which you would remain in permanent lockdown restriction and unable to do anything. This is how badly – *desperately* – the Cult is to get everyone ‘vaccinated’. The term and colour ‘green’ was not by chance and related to the psychology of fusing the perception of the green climate hoax with the ‘Covid’ hoax and how the ‘solution’ to both is the same Great Reset. Lying politicians, health officials and psychologists denied there were any plans for mandatory vaccinations or restrictions based on vaccinations, but they knew that was exactly what was meant to happen with governments of all countries reaching agreements to enforce a global system. ‘Free’ Denmark and ‘free’ Sweden unveiled digital vaccine certification. Cyprus, Czech Republic, Estonia, Greece, Hungary, Iceland, Italy, Poland, Portugal, Slovakia, and Spain have all committed to a vaccine passport system and the rest including the whole of the EU would follow. The satanic UK government will certainly go this way despite mendacious denials and at the time of writing it is trying to manipulate the public into having the ‘vaccine’ so they could go abroad on a summer holiday. How would that work without something to prove you had the synthetic toxicity injected into you?

Documents show that the EU's European Commission was moving towards 'vaccine certificates' in 2018 and 2019 before the 'Covid' hoax began. They knew what was coming. Abracadabra – Ursula von der Leyen, the German President of the Commission, announced in March, 2021, an EU 'Digital Green Certificate' – green again – to track the public's 'Covid status'. The passport sting is worldwide and the Far East followed the same pattern with South Korea ruling that only those with 'vaccination' passports – again the *green* pass – would be able to 'return to their daily lives'.

Bill Gates has been preparing for this 'passport' with other Cult operatives for years and beyond the paper version is a Gates-funded 'digital tattoo' to identify who has been vaccinated and who hasn't. The 'tattoo' is reported to include a substance which is externally readable to confirm who has been vaccinated. This is a bio-luminous light-generating enzyme (think fireflies) called ... *Luciferase*. Yes, named after the Cult 'god' Lucifer the 'light bringer' of whom more to come. Gates said he funded the readable tattoo to ensure children in the developing world were vaccinated and no one was missed out. He cares so much about poor kids as we know. This was just the cover story to develop a vaccine tagging system for everyone on the planet. Gates has been funding the ID2020 'alliance' to do just that in league with other lovely people at Microsoft, GAVI, the Rockefeller Foundation, Accenture and IDEO.org. He said in interviews in March, 2020, before any 'vaccine' publicly existed, that the world must have a globalised digital certificate to track the 'virus' and who had been vaccinated. Gates knew from the start that the mRNA vaccines were coming and when they would come and that the plan was to tag the 'vaccinated' to marginalise the intelligent and stop them doing anything including travel. Evil just doesn't suffice. Gates was exposed for offering a \$10 million bribe to the Nigerian House of Representatives to invoke compulsory 'Covid' vaccination of all Nigerians. Sara Cunial, a member of the Italian Parliament, called Gates a 'vaccine criminal'. She urged the Italian President to hand him over to the International Criminal Court for crimes against

humanity and condemned his plans to 'chip the human race' through ID2020.

You know it's a long-planned agenda when war criminal and Cult gofer Tony Blair is on the case. With the scale of arrogance only someone as dark as Blair can muster he said: 'Vaccination in the end is going to be your route to liberty.' Blair is a disgusting piece of work and he confirms that again. The media has given a lot of coverage to a bloke called Charlie Mullins, founder of London's biggest independent plumbing company, Pimlico Plumbers, who has said he won't employ anyone who has not been vaccinated or have them go to any home where people are not vaccinated. He said that if he had his way no one would be allowed to walk the streets if they have not been vaccinated. Gates was cheering at the time while I was alerting the white coats. The plan is that people will qualify for 'passports' for having the first two doses and then to keep it they will have to have all the follow ups and new ones for invented 'variants' until human genetics is transformed and many are dead who can't adjust to the changes. Hollywood celebrities – the usual propaganda stunt – are promoting something called the WELL Health-Safety Rating to verify that a building or space has 'taken the necessary steps to prioritize the health and safety of their staff, visitors and other stakeholders'. They included Lady Gaga, Jennifer Lopez, Michael B. Jordan, Robert DeNiro, Venus Williams, Wolfgang Puck, Deepak Chopra and 17th Surgeon General Richard Carmona. Yawn. WELL Health-Safety has big connections with China. Parent company Delos is headed by former Goldman Sachs partner Paul Scialla. This is another example – and we will see so many others – of using the excuse of 'health' to dictate the lives and activities of the population. I guess one confirmation of the 'safety' of buildings is that only 'vaccinated' people can go in, right?

Electronic concentration camps

I wrote decades ago about the plans to restrict travel and here we are for those who refuse to bow to tyranny. This can be achieved in one go with air travel if the aviation industry makes a blanket decree.

The ‘vaccine’ and guaranteed income are designed to be part of a global version of China’s social credit system which tracks behaviour 24/7 and awards or deletes ‘credits’ based on whether your behaviour is supported by the state or not. I mean your entire lifestyle – what you do, eat, say, everything. Once your credit score falls below a certain level consequences kick in. In China tens of millions have been denied travel by air and train because of this. All the locations and activities denied to refusers by the ‘vaccine’ passports will be included in one big mass ban on doing almost anything for those that don’t bow their head to government. It’s beyond fascist and a new term is required to describe its extremes – I guess fascist technocracy will have to do. The way the Chinese system of technological – technocratic – control is sweeping the West can be seen in the Los Angeles school system and is planned to be expanded worldwide. Every child is required to have a ‘Covid’-tracking app scanned daily before they can enter the classroom. The so-called Daily Pass tracking system is produced by Gates’ Microsoft which I’m sure will shock you rigid. The pass will be scanned using a barcode (one step from an inside-the-body barcode) and the information will include health checks, ‘Covid’ tests and vaccinations. Entry codes are for one specific building only and access will only be allowed if a student or teacher has a negative test with a test not testing for the ‘virus’, has no symptoms of anything alleged to be related to ‘Covid’ (symptoms from a range of other illness), and has a temperature under 100 degrees. No barcode, no entry, is planned to be the case for everywhere and not only schools.

Kids are being psychologically prepared to accept this as ‘normal’ their whole life which is why what they can impose in schools is so important to the Cult and its gofers. Long-time American freedom campaigner John Whitehead of the Rutherford Institute was not exaggerating when he said: ‘Databit by databit, we are building our own electronic concentration camps.’ Canada under its Cult gofer prime minister Justin Trudeau has taken a major step towards the real thing with people interned against their will if they test positive with a test not testing for the ‘virus’ when they arrive at a Canadian

airport. They are jailed in internment hotels often without food or water for long periods and with many doors failing to lock there have been sexual assaults. The interned are being charged sometimes \$2,000 for the privilege of being abused in this way. Trudeau is fully on board with the Cult and says the ‘Covid pandemic’ has provided an opportunity for a global ‘reset’ to permanently change Western civilisation. His number two, Deputy Prime Minister Chrystia Freeland, is a trustee of the World Economic Forum and a Rhodes Scholar. The Trudeau family have long been servants of the Cult. See *The Biggest Secret* and Cathy O’Brien’s book *Trance-Formation of America* for the horrific background to Trudeau’s father Pierre Trudeau another Canadian prime minister. Hide your fascism behind the façade of a heart-on-the-sleeve liberal. It’s a well-honed Cult technique.

What can the ‘vaccine’ really do?

We have a ‘virus’ never shown to exist and ‘variants’ of the ‘virus’ that have also never been shown to exist except, like the ‘original’, as computer-generated fictions. Even if you believe there’s a ‘virus’ the ‘case’ to ‘death’ rate is in the region of 0.23 to 0.15 percent and those ‘deaths’ are concentrated among the very old around the same average age that people die anyway. In response to this lack of threat (in truth none) psychopaths and idiots, knowingly and unknowingly answering to Gates and the Cult, are seeking to ‘vaccinate’ every man, woman and child on Planet Earth. Clearly the ‘vaccine’ is not about ‘Covid’ – none of this ever has been. So what is it all about *really*? Why the desperation to infuse genetically-manipulating synthetic material into everyone through mRNA fraudulent ‘vaccines’ with the intent of doing this over and over with the excuses of ‘variants’ and other ‘virus’ inventions? Dr Sherri Tenpenny, an osteopathic medical doctor in the United States, has made herself an expert on vaccines and their effects as a vehement campaigner against their use. Tenpenny was board certified in emergency medicine, the director of a level two trauma centre for 12 years, and moved to Cleveland in 1996 to start an integrative

medicine practice which has treated patients from all 50 states and some 17 other countries. Weaning people off pharmaceutical drugs is a speciality.

She became interested in the consequences of vaccines after attending a meeting at the National Vaccine Information Center in Washington DC in 2000 where she 'sat through four days of listening to medical doctors and scientists and lawyers and parents of vaccine injured kids' and asked: 'What's going on?' She had never been vaccinated and never got ill while her father was given a list of vaccines to be in the military and was 'sick his entire life'. The experience added to her questions and she began to examine vaccine documents from the Centers for Disease Control (CDC). After reading the first one, the 1998 version of *The General Recommendations of Vaccination*, she thought: 'This is it?' The document was poorly written and bad science and Tenpenny began 20 years of research into vaccines that continues to this day. She began her research into 'Covid vaccines' in March, 2020, and she describes them as 'deadly'. For many, as we have seen, they already have been. Tenpenny said that in the first 30 days of the 'vaccine' rollout in the United States there had been more than 40,000 adverse events reported to the vaccine adverse event database. A document had been delivered to her the day before that was 172 pages long. 'We have over 40,000 adverse events; we have over 3,100 cases of [potentially deadly] anaphylactic shock; we have over 5,000 neurological reactions.' Effects ranged from headaches to numbness, dizziness and vertigo, to losing feeling in hands or feet and paraesthesia which is when limbs 'fall asleep' and people have the sensation of insects crawling underneath their skin. All this happened in the first 30 days and remember that only about *ten percent* (or far less) of adverse reactions and vaccine-related deaths are estimated to be officially reported. Tenpenny said:

So can you think of one single product in any industry, any industry, for as long as products have been made on the planet that within 30 days we have 40,000 people complaining of side effects that not only is still on the market but ... we've got paid actors telling us how great

they are for getting their vaccine. We're offering people \$500 if they will just get their vaccine and we've got nurses and doctors going; 'I got the vaccine, I got the vaccine'.

Tenpenny said they were not going to be 'happy dancing folks' when they began to suffer Bell's palsy (facial paralysis), neuropathies, cardiac arrhythmias and autoimmune reactions that kill through a blood disorder. 'They're not going to be so happy, happy then, but we're never going to see pictures of those people' she said. Tenpenny described the 'vaccine' as 'a well-designed killing tool'.

No off-switch

Bad as the initial consequences had been Tenpenny said it would be maybe 14 months before we began to see the 'full ravage' of what is going to happen to the 'Covid vaccinated' with full-out consequences taking anything between two years and 20 years to show. You can understand why when you consider that variations of the 'Covid vaccine' use mRNA (messenger RNA) to in theory activate the immune system to produce protective antibodies without using the actual 'virus'. How can they when it's a computer program and they've never isolated what they claim is the 'real thing'? Instead they use *synthetic* mRNA. They are inoculating synthetic material into the body which through a technique known as the Trojan horse is absorbed into cells to change the nature of DNA. Human DNA is changed by an infusion of messenger RNA and with each new 'vaccine' of this type it is changed even more. Say so and you are banned by Cult Internet platforms. The contempt the contemptuous Mark Zuckerberg has for the truth and human health can be seen in an internal Facebook video leaked to the Project Veritas investigative team in which he said of the 'Covid vaccines': '... I share some caution on this because we just don't know the long term side-effects of basically modifying people's DNA and RNA.' At the same time this disgusting man's Facebook was censoring and banning anyone saying exactly the same. He must go before a Nuremberg trial for crimes against humanity when he *knows* that he

is censoring legitimate concerns and denying the right of informed consent on behalf of the Cult that owns him. People have been killed and damaged by the very ‘vaccination’ technique he cast doubt on himself when they may not have had the ‘vaccine’ with access to information that he denied them. The plan is to have at least annual ‘Covid vaccinations’, add others to deal with invented ‘variants’, and change all other vaccines into the mRNA system. Pfizer executives told shareholders at a virtual Barclays Global Healthcare Conference in March, 2021, that the public may need a third dose of ‘Covid vaccine’, plus regular yearly boosters and the company planned to hike prices to milk the profits in a ‘significant opportunity for our vaccine’. These are the professional liars, cheats and opportunists who are telling you their ‘vaccine’ is safe. Given this volume of mRNA planned to be infused into the human body and its ability to then replicate we will have a transformation of human genetics from biological to synthetic biological – exactly the long-time Cult plan for reasons we’ll see – and many will die. Sherri Tenpenny said of this replication:

It’s like having an on-button but no off-button and that whole mechanism ... they actually give it a name and they call it the Trojan horse mechanism, because it allows that [synthetic] virus and that piece of that [synthetic] virus to get inside of your cells, start to replicate and even get inserted into other parts of your DNA as a Trojan-horse.

Ask the overwhelming majority of people who have the ‘vaccine’ what they know about the contents and what they do and they would reply: ‘The government says it will stop me getting the virus.’ Governments give that false impression on purpose to increase take-up. You can read Sherri Tenpenny’s detailed analysis of the health consequences in her blog at Vaxxter.com, but in summary these are some of them. She highlights the statement by Bill Gates about how human beings can become their own ‘vaccine manufacturing machine’. The man is insane. [‘Vaccine’-generated] ‘antibodies’ carry synthetic messenger RNA into the cells and the damage starts, Tenpenny contends, and she says that lungs can be adversely affected through varying degrees of pus and bleeding which

obviously affects breathing and would be dubbed ‘Covid-19’. Even more sinister was the impact of ‘antibodies’ on macrophages, a white blood cell of the immune system. They consist of Type 1 and Type 2 which have very different functions. She said Type 1 are ‘hyper-vigilant’ white blood cells which ‘gobble up’ bacteria etc. However, in doing so, this could cause inflammation and in extreme circumstances be fatal. She says these affects are mitigated by Type 2 macrophages which kick in to calm down the system and stop it going rogue. They clear up dead tissue debris and reduce inflammation that the Type 1 ‘fire crews’ have caused. Type 1 kills the infection and Type 2 heals the damage, she says. This is her punchline with regard to ‘Covid vaccinations’: She says that mRNA ‘antibodies’ block Type 2 macrophages by attaching to them and deactivating them. This meant that when the Type 1 response was triggered by infection there was nothing to stop that getting out of hand by calming everything down. There’s an on-switch, but no off-switch, she says. What follows can be ‘over and out, see you when I see you’.

Genetic suicide

Tenpenny also highlights the potential for autoimmune disease – the body attacking itself – which has been associated with vaccines since they first appeared. Infusing a synthetic foreign substance into cells could cause the immune system to react in a panic believing that the body is being overwhelmed by an invader (it is) and the consequences can again be fatal. There is an autoimmune response known as a ‘cytokine storm’ which I have likened to a homeowner panicked by an intruder and picking up a gun to shoot randomly in all directions before turning the fire on himself. The immune system unleashes a storm of inflammatory response called cytokines to a threat and the body commits hara-kiri. The lesson is that you mess with the body’s immune response at your peril and these ‘vaccines’ seriously – fundamentally – mess with immune response. Tenpenny refers to a consequence called anaphylactic shock which is a severe and highly dangerous allergic reaction when the immune system

floods the body with chemicals. She gives the example of having a bee sting which primes the immune system and makes it sensitive to those chemicals. When people are stung again maybe years later the immune response can be so powerful that it leads to anaphylactic shock. Tenpenny relates this 'shock' with regard to the 'Covid vaccine' to something called polyethylene glycol or PEG. Enormous numbers of people have become sensitive to this over decades of use in a whole range of products and processes including food, drink, skin creams and 'medicine'. Studies have claimed that some 72 percent of people have antibodies triggered by PEG compared with two percent in the 1960s and allergic hypersensitive reactions to this become a gathering cause for concern. Tenpenny points out that the 'mRNA vaccine' is coated in a 'bubble' of polyethylene glycol which has the potential to cause anaphylactic shock through immune sensitivity. Many reports have appeared of people reacting this way after having the 'Covid vaccine'. What do we think is going to happen as humanity has more and more of these 'vaccines'?

Tenpenny said: 'All these pictures we have seen with people with these rashes ... these weepy rashes, big reactions on their arms and things like that – it's an acute allergic reaction most likely to the polyethylene glycol that you've been previously primed and sensitised to.'

Those who have not studied the conspiracy and its perpetrators at length might think that making the population sensitive to PEG and then putting it in these 'vaccines' is just a coincidence. It is not. It is instead testament to how carefully and coldly-planned current events have been and the scale of the conspiracy we are dealing with. Tenpenny further explains that the 'vaccine' mRNA procedure can breach the blood-brain barrier which protects the brain from toxins and other crap that will cause malfunction. In this case they could make two proteins corrupt brain function to cause Amyotrophic lateral sclerosis (ALS), a progressive nervous system disease leading to loss of muscle control, and frontal lobe degeneration – Alzheimer's and dementia. Immunologist J. Bart Classon published a paper connecting mRNA 'vaccines' to prion

disease which can lead to Alzheimer's and other forms of neurodegenerative disease while others have pointed out the potential to affect the placenta in ways that make women infertile. This will become highly significant in the next chapter when I will discuss other aspects of this non-vaccine that relate to its nanotechnology and transmission from the injected to the uninjected.

Qualified in idiocy

Tenpenny describes how research has confirmed that these 'vaccine'-generated antibodies can interact with a range of other tissues in the body and attack many other organs including the lungs. 'This means that if you have a hundred people standing in front of you that all got this shot they could have a hundred different symptoms.'

Anyone really think that Cult gofers like the Queen, Tony Blair, Christopher Whitty, Anthony Fauci, and all the other psychopaths have really had this 'vaccine' in the pictures we've seen? Not a bloody chance. Why don't doctors all tell us about all these dangers and consequences of the 'Covid vaccine'? Why instead do they encourage and pressure patients to have the shot? Don't let's think for a moment that doctors and medical staff can't be stupid, lazy, and psychopathic and that's without the financial incentives to give the jab. Tenpenny again:

Some people are going to die from the vaccine directly but a large number of people are going to start to get horribly sick and get all kinds of autoimmune diseases 42 days to maybe a year out. What are they going to do, these stupid doctors who say; 'Good for you for getting that vaccine.' What are they going to say; 'Oh, it must be a mutant, we need to give an extra dose of that vaccine.'

Because now the vaccine, instead of one dose or two doses we need three or four because the stupid physicians aren't taking the time to learn anything about it. If I can learn this sitting in my living room reading a 19 page paper and several others so can they. There's nothing special about me, I just take the time to do it.

Remember how Sara Kayat, the NHS and TV doctor, said that the 'Covid vaccine' would '100 percent prevent hospitalisation and death'. Doctors can be idiots like every other profession and they

should not be worshipped as infallible. They are not and far from it. Behind many medical and scientific ‘experts’ lies an uninformed prat trying to hide themselves from you although in the ‘Covid’ era many have failed to do so as with UK narrative-repeating ‘TV doctor’ Hilary Jones. Pushing back against the minority of proper doctors and scientists speaking out against the ‘vaccine’ has been the entire edifice of the Cult global state in the form of governments, medical systems, corporations, mainstream media, Silicon Valley, and an army of compliant doctors, medical staff and scientists willing to say anything for money and to enhance their careers by promoting the party line. If you do that you are an ‘expert’ and if you won’t you are an ‘anti-vaxxer’ and ‘Covidiot’. The pressure to be ‘vaccinated’ is incessant. We have even had reports claiming that the ‘vaccine’ can help cure cancer and Alzheimer’s and make the lame walk. I am waiting for the announcement that it can bring you coffee in the morning and cook your tea. Just as the symptoms of ‘Covid’ seem to increase by the week so have the miracles of the ‘vaccine’. American supermarket giant Kroger Co. offered nearly 500,000 employees in 35 states a \$100 bonus for having the ‘vaccine’ while donut chain Krispy Kreme promised ‘vaccinated’ customers a free glazed donut every day for the rest of 2021. Have your DNA changed and you will get a doughnut although we might not have to give you them for long. Such offers and incentives confirm the desperation.

Perhaps the worse vaccine-stunt of them all was UK ‘Health’ Secretary Matt-the-prat Hancock on live TV after watching a clip of someone being ‘vaccinated’ when the roll-out began. Hancock faked tears so badly it was embarrassing. Brain-of-Britain Piers Morgan, the lockdown-supporting, ‘vaccine’ supporting, ‘vaccine’ passport-supporting, TV host played along with Hancock – ‘You’re quite emotional about that’ he said in response to acting so atrocious it would have been called out at a school nativity which will presumably today include Mary and Jesus in masks, wise men keeping their camels six feet apart, and shepherds under tent arrest. System-serving Morgan tweeted this: ‘Love the idea of covid vaccine passports for everywhere: flights, restaurants, clubs, football, gyms,

shops etc. It's time covid-denying, anti-vaxxer loonies had their bullsh*t bluff called & bar themselves from going anywhere that responsible citizens go.' If only I could aspire to his genius. To think that Morgan, who specialises in shouting over anyone he disagrees with, was lauded as a free speech hero when he lost his job after storming off the set of his live show like a child throwing his dolly out of the pram. If he is a free speech hero we are in real trouble. I have no idea what 'bullsh*t' means, by the way, the * throws me completely.

The Cult is desperate to infuse its synthetic DNA-changing concoction into everyone and has been using every lie, trick and intimidation to do so. The question of '*Why?*' we shall now address.

CHAPTER TEN

Human 2.0

I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted –

Alan Turing (1912-1954), the ‘Father of artificial intelligence’

I have been exposing for decades the plan to transform the human body from a biological to a synthetic-biological state. The new human that I will call Human 2.0 is planned to be connected to artificial intelligence and a global AI ‘Smart Grid’ that would operate as one global system in which AI would control everything from your fridge to your heating system to your car to your mind. Humans would no longer be ‘human’, but post-human and sub-human, with their thinking and emotional processes replaced by AI.

What I said sounded crazy and beyond science fiction and I could understand that. To any balanced, rational, mind it *is* crazy. Today, however, that world is becoming reality and it puts the ‘Covid vaccine’ into its true context. Ray Kurzweil is the ultra-Zionist ‘computer scientist, inventor and futurist’ and co-founder of the Singularity University. Singularity refers to the merging of humans with machines or ‘transhumanism’. Kurzweil has said humanity would be connected to the cyber ‘cloud’ in the period of the ever-recurring year of 2030:

Our thinking ... will be a hybrid of biological and non-biological thinking ... humans will be able to extend their limitations and ‘think in the cloud’ ... We’re going to put gateways to the

cloud in our brains ... We're going to gradually merge and enhance ourselves ... In my view, that's the nature of being human – we transcend our limitations. As the technology becomes vastly superior to what we are then the small proportion that is still human gets smaller and smaller and smaller until it's just utterly negligible.

They are trying to sell this end-of-humanity-as-we-know-it as the next stage of 'evolution' when we become super-human and 'like the gods'. They are lying to you. Shocked, eh? The population, and again especially the young, have been manipulated into addiction to technologies designed to enslave them for life. First they induced an addiction to smartphones (holdables); next they moved to technology on the body (wearables); and then began the invasion of the body (implantables). I warned way back about the plan for microchipped people and we are now entering that era. We should not be diverted into thinking that this refers only to chips we can see. Most important are the nanochips known as smart dust, neural dust and nanobots which are far too small to be seen by the human eye. Nanotechnology is everywhere, increasingly in food products, and released into the atmosphere by the geoengineering of the skies funded by Bill Gates to 'shut out the Sun' and 'save the planet from global warming'. Gates has been funding a project to spray millions of tonnes of chalk (calcium carbonate) into the stratosphere over Sweden to 'dim the Sun' and cool the Earth. Scientists warned the move could be disastrous for weather systems in ways no one can predict and opposition led to the Swedish space agency announcing that the 'experiment' would not be happening as planned in the summer of 2021; but it shows where the Cult is going with dimming the impact of the Sun and there's an associated plan to change the planet's atmosphere. Who gives psychopath Gates the right to dictate to the entire human race and dismantle planetary systems? The world will not be safe while this man is at large.

The global warming hoax has made the Sun, like the gas of life, something to fear when both are essential to good health and human survival (more inversion). The body transforms sunlight into vital vitamin D through a process involving ... *cholesterol*. This is the cholesterol we are also told to fear. We are urged to take Big Pharma

statin drugs to reduce cholesterol and it's all systematic. Reducing cholesterol means reducing vitamin D uptake with all the multiple health problems that will cause. At least if you take statins long term it saves the government from having to pay you a pension. The delivery system to block sunlight is widely referred to as chemtrails although these have a much deeper agenda, too. They appear at first to be contrails or condensation trails streaming from aircraft into cold air at high altitudes. Contrails disperse very quickly while chemtrails do not and spread out across the sky before eventually their content falls to earth. Many times I have watched aircraft cross-cross a clear blue sky releasing chemtrails until it looks like a cloudy day. Chemtrails contain many things harmful to humans and the natural world including toxic heavy metals, aluminium (see Alzheimer's) and nanotechnology. Ray Kurzweil reveals the reason without actually saying so: 'Nanobots will infuse all the matter around us with information. Rocks, trees, everything will become these intelligent creatures.' How do you deliver that? *From the sky.* Self-replicating nanobots would connect everything to the Smart Grid. The phenomenon of Morgellons disease began in the chemtrail era and the correlation has led to it being dubbed the 'chemtrail disease'. Self-replicating fibres appear in the body that can be pulled out through the skin. Morgellons fibres continue to grow outside the body and have a form of artificial intelligence. I cover this at greater length in *Phantom Self*.

'Vaccine' operating system

'Covid vaccines' with their self-replicating synthetic material are also designed to make the connection between humanity and Kurzweil's 'cloud'. American doctor and dedicated campaigner for truth, Carrie Madej, an Internal Medicine Specialist in Georgia with more than 20 years medical experience, has highlighted the nanotechnology aspect of the fake 'vaccines'. She explains how one of the components in at least the Moderna and Pfizer synthetic potions are 'lipid nanoparticles' which are 'like little tiny computer bits' – a 'sci-fi substance' known as nanobots and hydrogel which can be 'triggered

at any moment to deliver its payload' and act as 'biosensors'. The synthetic substance had 'the ability to accumulate data from your body like your breathing, your respiration, thoughts and emotions, all kind of things' and each syringe could carry a *million* nanobots:

This substance because it's like little bits of computers in your body, crazy, but it's true, it can do that, [and] obviously has the ability to act through Wi-Fi. It can receive and transmit energy, messages, frequencies or impulses. That issue has never been addressed by these companies. What does that do to the human?

Just imagine getting this substance in you and it can react to things all around you, the 5G, your smart device, your phones, what is happening with that? What if something is triggering it, too, like an impulse, a frequency? We have something completely foreign in the human body.

Madej said her research revealed that electromagnetic (EMF) frequencies emitted by phones and other devices had increased dramatically in the same period of the 'vaccine' rollout and she was seeing more people with radiation problems as 5G and other electromagnetic technology was expanded and introduced to schools and hospitals. She said she was 'floored with the EMF coming off' the devices she checked. All this makes total sense and syncs with my own work of decades when you think that Moderna refers in documents to its mRNA 'vaccine' as an 'operating system':

Recognizing the broad potential of mRNA science, we set out to create an mRNA technology platform that functions very much like an operating system on a computer. It is designed so that it can plug and play interchangeably with different programs. In our case, the 'program' or 'app' is our mRNA drug – the unique mRNA sequence that codes for a protein ...

... Our mRNA Medicines – 'The Software Of Life': When we have a concept for a new mRNA medicine and begin research, fundamental components are already in place. Generally, the only thing that changes from one potential mRNA medicine to another is the coding region – the actual genetic code that instructs ribosomes to make protein. Utilizing these instruction sets gives our investigational mRNA medicines a software-like quality. We also have the ability to combine different mRNA sequences encoding for different proteins in a single mRNA investigational medicine.

Who needs a real ‘virus’ when you can create a computer version to justify infusing your operating system into the entire human race on the road to making living, breathing people into cyborgs? What is missed with the ‘vaccines’ is the *digital* connection between synthetic material and the body that I highlighted earlier with the study that hacked a computer with human DNA. On one level the body is digital, based on mathematical codes, and I’ll have more about that in the next chapter. Those who ridiculously claim that mRNA ‘vaccines’ are not designed to change human genetics should explain the words of Dr Tal Zaks, chief medical officer at Moderna, in a 2017 TED talk. He said that over the last 30 years ‘we’ve been living this phenomenal digital scientific revolution, and I’m here today to tell you, that we are actually *hacking the software of life*, and that it’s changing the way we think about prevention and treatment of disease’:

In every cell there’s this thing called messenger RNA, or mRNA for short, that transmits the critical information from the DNA in our genes to the protein, which is really the stuff we’re all made out of. This is the critical information that determines what the cell will do. So we think about it as an operating system. So if you could change that, if you could introduce a line of code, or change a line of code, it turns out, that has profound implications for everything, from the flu to cancer.

Zaks should more accurately have said that this has profound implications for the human genetic code and the nature of DNA. Communications within the body go both ways and not only one. But, hey, no, the ‘Covid vaccine’ will not affect your genetics. Cult fact-checkers say so even though the man who helped to develop the mRNA technique says that it does. Zaks said in 2017:

If you think about what it is we’re trying to do. We’ve taken information and our understanding of that information and how that information is transmitted in a cell, and we’ve taken our understanding of medicine and how to make drugs, and we’re fusing the two. We think of it as information therapy.

I have been writing for decades that the body is an information field communicating with itself and the wider world. This is why

radiation which is information can change the information field of body and mind through phenomena like 5G and change their nature and function. ‘Information therapy’ means to change the body’s information field and change the way it operates. DNA is a receiver-transmitter of information and can be mutated by information like mRNA synthetic messaging. Technology to do this has been ready and waiting in the underground bases and other secret projects to be rolled out when the ‘Covid’ hoax was played. ‘Trials’ of such short and irrelevant duration were only for public consumption. When they say the ‘vaccine’ is ‘experimental’ that is not true. It may appear to be ‘experimental’ to those who don’t know what’s going on, but the trials have already been done to ensure the Cult gets the result it desires. Zaks said that it took decades to sequence the human genome, completed in 2003, but now they could do it in a week. By ‘they’ he means scientists operating in the public domain. In the secret projects they were sequencing the genome in a week long before even 2003.

Deluge of mRNA

Highly significantly the Moderna document says the guiding premise is that if using mRNA as a medicine works for one disease then it should work for many diseases. They were leveraging the flexibility afforded by their platform and the fundamental role mRNA plays in protein synthesis to pursue mRNA medicines for a broad spectrum of diseases. Moderna is confirming what I was saying through 2020 that multiple ‘vaccines’ were planned for ‘Covid’ (and later invented ‘variants’) and that previous vaccines would be converted to the mRNA system to infuse the body with massive amounts of genetically-manipulating synthetic material to secure a transformation to a synthetic-biological state. The ‘vaccines’ are designed to kill stunning numbers as part of the long-exposed Cult depopulation agenda and transform the rest. Given this is the goal you can appreciate why there is such hysterical demand for every human to be ‘vaccinated’ for an alleged ‘disease’ that has an estimated ‘infection’ to ‘death’ ratio of 0.23-0.15 percent. As I write

children are being given the ‘vaccine’ in trials (their parents are a disgrace) and ever-younger people are being offered the vaccine for a ‘virus’ that even if you believe it exists has virtually zero chance of harming them. Horrific effects of the ‘trials’ on a 12-year-old girl were revealed by a family member to be serious brain and gastric problems that included a bowel obstruction and the inability to swallow liquids or solids. She was unable to eat or drink without throwing up, had extreme pain in her back, neck and abdomen, and was paralysed from the waist down which stopped her urinating unaided. When the girl was first taken to hospital doctors said it was all in her mind. She was signed up for the ‘trial’ by her parents for whom no words suffice. None of this ‘Covid vaccine’ insanity makes any sense unless you see what the ‘vaccine’ really is – a body-changer. Synthetic biology or ‘SynBio’ is a fast-emerging and expanding scientific discipline which includes everything from genetic and molecular engineering to electrical and computer engineering. Synthetic biology is defined in these ways:

- A multidisciplinary area of research that seeks to create new biological parts, devices, and systems, or to redesign systems that are already found in nature.
- The use of a mixture of physical engineering and genetic engineering to create new (and therefore synthetic) life forms.
- An emerging field of research that aims to combine the knowledge and methods of biology, engineering and related disciplines in the design of chemically-synthesized DNA to create organisms with novel or enhanced characteristics and traits (synthetic organisms including humans).

We now have synthetic blood, skin, organs and limbs being developed along with synthetic body parts produced by 3D printers. These are all elements of the synthetic human programme and this comment by Kurzweil’s co-founder of the Singularity University,

Peter Diamandis, can be seen in a whole new light with the 'Covid' hoax and the sanctions against those that refuse the 'vaccine':

Anybody who is going to be resisting the progress forward [to transhumanism] is going to be resisting evolution and, fundamentally, they will die out. It's not a matter of whether it's good or bad. It's going to happen.

'Resisting evolution'? What absolute bollocks. The arrogance of these people is without limit. His 'it's going to happen' mantra is another way of saying 'resistance is futile' to break the spirit of those pushing back and we must not fall for it. Getting this genetically-transforming 'vaccine' into everyone is crucial to the Cult plan for total control and the desperation to achieve that is clear for anyone to see. Vaccine passports are a major factor in this and they, too, are a form of resistance is futile. It's NOT. The paper funded by the Rockefeller Foundation for the 2013 'health conference' in China said:

We will interact more with artificial intelligence. The use of robotics, bio-engineering to augment human functioning is already well underway and will advance. Re-engineering of humans into potentially separate and unequal forms through genetic engineering or mixed human-robots raises debates on ethics and equality.

A new demography is projected to emerge after 2030 [that year again] of technologies (robotics, genetic engineering, nanotechnology) producing robots, engineered organisms, 'nanobots' and artificial intelligence (AI) that can self-replicate. Debates will grow on the implications of an impending reality of human designed life.

What is happening today is so long planned. The world army enforcing the will of the world government is intended to be a robot army, not a human one. Today's military and its technologically 'enhanced' troops, pilotless planes and driverless vehicles are just stepping stones to that end. Human soldiers are used as Cult fodder and its time they woke up to that and worked for the freedom of the population instead of their own destruction and their family's destruction – the same with the police. Join us and let's sort this out. The phenomenon of enforce my own destruction is widespread in the 'Covid' era with Woker 'luvvies' in the acting and entertainment

industries supporting ‘Covid’ rules which have destroyed their profession and the same with those among the public who put signs on the doors of their businesses ‘closed due to Covid – stay safe’ when many will never reopen. It’s a form of masochism and most certainly insanity.

Transgender = transhumanism

When something explodes out of nowhere and is suddenly everywhere it is always the Cult agenda and so it is with the tidal wave of claims and demands that have infiltrated every aspect of society under the heading of ‘transgenderism’. The term ‘trans’ is so ‘in’ and this is the dictionary definition:

A prefix meaning ‘across’, ‘through’, occurring ... in loanwords from Latin, used in particular for denoting movement or conveyance from place to place (transfer; transmit; transplant) or complete change (transform; transmute), or to form adjectives meaning ‘crossing’, ‘on the other side of’, or ‘going beyond’ the place named (transmontane; transnational; trans-Siberian).

Transgender means to go beyond gender and transhuman means to go beyond human. Both are aspects of the Cult plan to transform the human body to a synthetic state with *no gender*. Human 2.0 is not designed to procreate and would be produced technologically with no need for parents. The new human would mean the end of parents and so men, and increasingly women, are being targeted for the deletion of their rights and status. Parental rights are disappearing at an ever-quickening speed for the same reason. The new human would have no need for men or women when there is no procreation and no gender. Perhaps the transgender movement that appears to be in a permanent state of frenzy might now contemplate on how it is being used. This was never about transgender rights which are only the interim excuse for confusing gender, particularly in the young, on the road to *fusing* gender. Transgender activism is not an end; it is a *means* to an end. We see again the technique of creative destruction in which you destroy the status quo to ‘build back better’ in the form that you want. The gender status quo had to be

destroyed by persuading the Cult-created Woke mentality to believe that you can have 100 genders or more. A programme for 9 to 12 year olds produced by the Cult-owned BBC promoted the 100 genders narrative. The very idea may be the most monumental nonsense, but it is not what is true that counts, only what you can make people *believe* is true. Once the gender of $2 + 2 = 4$ has been dismantled through indoctrination, intimidation and $2 + 2 = 5$ then the new no-gender normal can take its place with Human 2.0.

Aldous Huxley revealed the plan in his prophetic *Brave New World* in 1932:

Natural reproduction has been done away with and children are created, 'decanted', and raised in 'hatcheries and conditioning centres'. From birth, people are genetically designed to fit into one of five castes, which are further split into 'Plus' and 'Minus' members and designed to fulfil predetermined positions within the social and economic strata of the World State.

How could Huxley know this in 1932? For the same reason George Orwell knew about the Big Brother state in 1948, Cult insiders I have quoted knew about it in 1969, and I have known about it since the early 1990s. If you are connected to the Cult or you work your balls off to uncover the plan you can predict the future. The process is simple. If there is a plan for the world and nothing intervenes to stop it then it will happen. Thus if you communicate the plan ahead of time you are perceived to have predicted the future, but you haven't. You have revealed the plan which without intervention will become the human future. The whole reason I have done what I have is to alert enough people to inspire an intervention and maybe at last that time has come with the Cult and its intentions now so obvious to anyone with a brain in working order.

The future is here

Technological wombs that Huxley described to replace parent procreation are already being developed and they are only the projects we know about in the public arena. Israeli scientists told *The Times of Israel* in March, 2021, that they have grown 250-cell embryos

into mouse foetuses with fully formed organs using artificial wombs in a development they say could pave the way for gestating humans outside the womb. Professor Jacob Hanna of the Weizmann Institute of Science said:

We took mouse embryos from the mother at day five of development, when they are just of 250 cells, and had them in the incubator from day five until day 11, by which point they had grown all their organs.

By day 11 they make their own blood and have a beating heart, a fully developed brain. Anybody would look at them and say, 'this is clearly a mouse foetus with all the characteristics of a mouse.' It's gone from being a ball of cells to being an advanced foetus.

A special liquid is used to nourish embryo cells in a laboratory dish and they float on the liquid to duplicate the first stage of embryonic development. The incubator creates all the right conditions for its development, Hanna said. The liquid gives the embryo 'all the nutrients, hormones and sugars they need' along with a custom-made electronic incubator which controls gas concentration, pressure and temperature. The cutting-edge in the underground bases and other secret locations will be light years ahead of that, however, and this was reported by the London *Guardian* in 2017:

We are approaching a biotechnological breakthrough. Ectogenesis, the invention of a complete external womb, could completely change the nature of human reproduction. In April this year, researchers at the Children's Hospital of Philadelphia announced their development of an artificial womb.

The article was headed 'Artificial wombs could soon be a reality. What will this mean for women?' What would it mean for children is an even bigger question. No mother to bond with only a machine in preparation for a life of soulless interaction and control in a world governed by machines (see the *Matrix* movies). Now observe the calculated manipulations of the 'Covid' hoax as human interaction and warmth has been curtailed by distancing, isolation and fear with people communicating via machines on a scale never seen before.

These are all dots in the same picture as are all the personal assistants, gadgets and children's toys through which kids and adults communicate with AI as if it is human. The AI 'voice' on Sat-Nav should be included. All these things are psychological preparation for the Cult endgame. Before you can make a physical connection with AI you have to make a psychological connection and that is what people are being conditioned to do with this ever gathering human-AI interaction. Movies and TV programmes depicting the transhuman, robot dystopia relate to a phenomenon known as 'pre-emptive programming' in which the world that is planned is portrayed everywhere in movies, TV and advertising. This is conditioning the conscious and subconscious mind to become familiar with the planned reality to dilute resistance when it happens for real. What would have been a shock such is the change is made less so. We have young children put on the road to transgender transition surgery with puberty blocking drugs at an age when they could never be able to make those life-changing decisions.

Rachel Levine, a professor of paediatrics and psychiatry who believes in treating children this way, became America's highest-ranked openly-transgender official when she was confirmed as US Assistant Secretary at the Department of Health and Human Services after being nominated by Joe Biden (the Cult). Activists and governments press for laws to deny parents a say in their children's transition process so the kids can be isolated and manipulated into agreeing to irreversible medical procedures. A Canadian father Robert Hoogland was denied bail by the Vancouver Supreme Court in 2021 and remained in jail for breaching a court order that he stay silent over his young teenage daughter, a minor, who was being offered life-changing hormone therapy without parental consent. At the age of 12 the girl's 'school counsellor' said she may be transgender, referred her to a doctor and told the school to treat her like a boy. This is another example of state-serving schools imposing ever more control over children's lives while parents have ever less.

Contemptible and extreme child abuse is happening all over the world as the Cult gender-fusion operation goes into warp-speed.

Why the war on men – and now women?

The question about what artificial wombs mean for women should rightly be asked. The answer can be seen in the deletion of women's rights involving sport, changing rooms, toilets and status in favour of people in male bodies claiming to identify as women. I can identify as a mountain climber, but it doesn't mean I can climb a mountain any more than a biological man can be a biological woman. To believe so is a triumph of belief over factual reality which is the very perceptual basis of everything Woke. Women's sport is being destroyed by allowing those with male bodies who say they identify as female to 'compete' with girls and women. Male body 'women' dominate 'women's' competition with their greater muscle mass, bone density, strength and speed. With that disadvantage sport for women loses all meaning. To put this in perspective nearly 300 American high school boys can run faster than the quickest woman sprinter in the world. Women are seeing their previously protected spaces invaded by male bodies simply because they claim to identify as women. That's all they need to do to access all women's spaces and activities under the Biden 'Equality Act' that destroys equality for women with the usual Orwellian Woke inversion. Male sex offenders have already committed rapes in women's prisons after claiming to identify as women to get them transferred. Does this not matter to the Woke 'equality' hypocrites? Not in the least. What matters to Cult manipulators and funders behind transgender activists is to advance gender fusion on the way to the no-gender 'human'. When you are seeking to impose transparent nonsense like this, or the 'Covid' hoax, the only way the nonsense can prevail is through censorship and intimidation of dissenters, deletion of factual information, and programming of the unquestioning, bewildered and naive. You don't have to scan the world for long to see that all these things are happening.

Many women's rights organisations have realised that rights and status which took such a long time to secure are being eroded and that it is systematic. Kara Dansky of the global Women's Human Rights Campaign said that Biden's transgender executive order immediately he took office, subsequent orders, and Equality Act legislation that followed 'seek to erase women and girls in the law as a category'. *Exactly.* I said during the long ago-started war on men (in which many women play a crucial part) that this was going to turn into a war on them. The Cult is phasing out *both* male and female genders. To get away with that they are brought into conflict so they are busy fighting each other while the Cult completes the job with no unity of response. Unity, people, *unity*. We need unity everywhere. Transgender is the only show in town as the big step towards the no-gender human. It's not about rights for transgender people and never has been. Woke political correctness is deleting words relating to genders to the same end. Wokers believe this is to be 'inclusive' when the opposite is true. They are deleting words describing gender because gender *itself* is being deleted by Human 2.0. Terms like 'man', 'woman', 'mother' and 'father' are being deleted in the universities and other institutions to be replaced by the *no*-gender, not trans-gender, 'individuals' and 'guardians'. Women's rights campaigner Maria Keffler of Partners for Ethical Care said: 'Children are being taught from kindergarten upward that some boys have a vagina, some girls have a penis, and that kids can be any gender they want to be.' Do we really believe that suddenly countries all over the world at the same time had the idea of having drag queens go into schools or read transgender stories to very young children in the local library? It's coldly-calculated confusion of gender on the way to the fusion of gender. Suzanne Vierling, a psychologist from Southern California, made another important point:

Yesterday's slave woman who endured gynecological medical experiments is today's girl-child being butchered in a booming gender-transitioning sector. Ovaries removed, pushing her into menopause and osteoporosis, uncharted territory, and parents' rights and authority decimated.

The erosion of parental rights is a common theme in line with the Cult plans to erase the very concept of parents and ‘ovaries removed, pushing her into menopause’ means what? Those born female lose the ability to have children – another way to discontinue humanity as we know it.

Eliminating Human 1.0 (before our very eyes)

To pave the way for Human 2.0 you must phase out Human 1.0. This is happening through plummeting sperm counts and making women infertile through an onslaught of chemicals, radiation (including smartphones in pockets of men) and mRNA ‘vaccines’. Common agriculture pesticides are also having a devastating impact on human fertility. I have been tracking collapsing sperm counts in the books for a long time and in 2021 came a book by fertility scientist and reproductive epidemiologist Shanna Swan, *Count Down: How Our Modern World Is Threatening Sperm Counts, Altering Male and Female Reproductive Development and Imperiling the Future of the Human Race*. She reports how the global fertility rate dropped by half between 1960 and 2016 with America’s birth rate 16 percent below where it needs to be to sustain the population. Women are experiencing declining egg quality, more miscarriages, and more couples suffer from infertility. Other findings were an increase in erectile dysfunction, infant boys developing more genital abnormalities, male problems with conception, and plunging levels of the male hormone testosterone which would explain why so many men have lost their backbone and masculinity. This has been very evident during the ‘Covid’ hoax when women have been prominent among the Pushbackers and big strapping blokes have bowed their heads, covered their faces with a nappy and quietly submitted. Mind control expert Cathy O’Brien also points to how global education introduced the concept of ‘we’re all winners’ in sport and classrooms: ‘Competition was defused, and it in turn defused a sense of fighting back.’ This is another version of the ‘equity’ doctrine in which you drive down rather than raise up. What a contrast in Cult-controlled China with its global ambitions

where the government published plans in January, 2021, to 'cultivate masculinity' in boys from kindergarten through to high school in the face of a 'masculinity crisis'. A government adviser said boys would be soon become 'delicate, timid and effeminate' unless action was taken. Don't expect any similar policy in the targeted West. A 2006 study showed that a 65-year-old man in 2002 had testosterone levels 15 percent lower than a 65-year-old man in 1987 while a 2020 study found a similar story with young adults and adolescents. Men are getting prescriptions for testosterone replacement therapy which causes an even greater drop in sperm count with up to 99 percent seeing sperm counts drop to zero during the treatment. More sperm is defective and malfunctioning with some having two heads or not pursuing an egg.

A class of *synthetic* chemicals known as phthalates are being blamed for the decline. These are found everywhere in plastics, shampoos, cosmetics, furniture, flame retardants, personal care products, pesticides, canned foods and even receipts. Why till receipts? Everyone touches them. Let no one delude themselves that all this is not systematic to advance the long-time agenda for human body transformation. Phthalates mimic hormones and disrupt the hormone balance causing testosterone to fall and genital birth defects in male infants. Animals and fish have been affected in the same way due to phthalates and other toxins in rivers. When fish turn gay or change sex through chemicals in rivers and streams it is a pointer to why there has been such an increase in gay people and the sexually confused. It doesn't matter to me what sexuality people choose to be, but if it's being affected by chemical pollution and consumption then we need to know. Does anyone really think that this is not connected to the transgender agenda, the war on men and the condemnation of male 'toxic masculinity'? You watch this being followed by 'toxic femininity'. It's already happening. When breastfeeding becomes 'chest-feeding', pregnant women become pregnant people along with all the other Woke claptrap you know that the world is going insane and there's a Cult scam in progress. Transgender activists are promoting the Cult agenda while Cult

billionaires support and fund the insanity as they laugh themselves to sleep at the sheer stupidity for which humans must be infamous in galaxies far, far away.

'Covid vaccines' and female infertility

We can now see why the 'vaccine' has been connected to potential infertility in women. Dr Michael Yeadon, former Vice President and Chief Scientific Advisor at Pfizer, and Dr Wolfgang Wodarg in Germany, filed a petition with the European Medicines Agency in December, 2020, urging them to stop trials for the Pfizer/BioNTech shot and all other mRNA trials until further studies had been done. They were particularly concerned about possible effects on fertility with 'vaccine'-produced antibodies attacking the protein Syncytin-1 which is responsible for developing the placenta. The result would be infertility 'of indefinite duration' in women who have the 'vaccine' with the placenta failing to form. Section 10.4.2 of the Pfizer/BioNTech trial protocol says that pregnant women or those who might become so should not have mRNA shots. Section 10.4 warns men taking mRNA shots to 'be abstinent from heterosexual intercourse' and not to donate sperm. The UK government said that it *did not know* if the mRNA procedure had an effect on fertility. *Did not know?* These people have to go to jail. UK government advice did not recommend at the start that pregnant women had the shot and said they should avoid pregnancy for at least two months after 'vaccination'. The 'advice' was later updated to pregnant women should only have the 'vaccine' if the benefits outweighed the risks to mother and foetus. What the hell is that supposed to mean? Then 'spontaneous abortions' began to appear and rapidly increase on the adverse reaction reporting schemes which include only a fraction of adverse reactions. Thousands and ever-growing numbers of 'vaccinated' women are describing changes to their menstrual cycle with heavier blood flow, irregular periods and menstruating again after going through the menopause – all links to reproduction effects. Women are passing blood clots and the lining of their uterus while men report erectile dysfunction and blood effects. Most

significantly of all *unvaccinated* women began to report similar menstrual changes after interaction with '*vaccinated*' people and men and children were also affected with bleeding noses, blood clots and other conditions. 'Shedding' is when vaccinated people can emit the content of a vaccine to affect the unvaccinated, but this is different. '*Vaccinated*' people were not shedding a 'live virus' allegedly in '*vaccines*' as before because the fake '*Covid vaccines*' involve synthetic material and other toxicity. Doctors exposing what is happening prefer the term '*transmission*' to shedding. Somehow those that have had the shots are transmitting effects to those that haven't. Dr Carrie Madej said the nano-content of the '*vaccines*' can 'act like an antenna' to others around them which fits perfectly with my own conclusions. This '*vaccine*' transmission phenomenon was becoming known as the book went into production and I deal with this further in the Postscript.

Vaccine effects on sterility are well known. The World Health Organization was accused in 2014 of sterilising millions of women in Kenya with the evidence confirmed by the content of the vaccines involved. The same WHO behind the '*Covid*' hoax admitted its involvement for more than ten years with the vaccine programme. Other countries made similar claims. Charges were lodged by Tanzania, Nicaragua, Mexico, and the Philippines. The Gardasil vaccine claimed to protect against a genital '*virus*' known as HPV has also been linked to infertility. Big Pharma and the WHO (same thing) are criminal and satanic entities. Then there's the Bill Gates Foundation which is connected through funding and shared interests with 20 pharmaceutical giants and laboratories. He stands accused of directing the policy of United Nations Children's Fund (UNICEF), vaccine alliance GAVI, and other groupings, to advance the vaccine agenda and silence opposition at great cost to women and children. At the same time Gates wants to reduce the global population. Coincidence?

Great Reset = Smart Grid = new human

The Cult agenda I have been exposing for 30 years is now being openly promoted by Cult assets like Gates and Klaus Schwab of the World Economic Forum under code-terms like the 'Great Reset', 'Build Back Better' and 'a rare but narrow window of opportunity to reflect, reimagine, and reset our world'. What provided this 'rare but narrow window of opportunity'? The 'Covid' hoax did. Who created that? *They* did. My books from not that long ago warned about the planned 'Internet of Things' (IoT) and its implications for human freedom. This was the plan to connect all technology to the Internet and artificial intelligence and today we are way down that road with an estimated 36 billion devices connected to the World Wide Web and that figure is projected to be 76 billion by 2025. I further warned that the Cult planned to go beyond that to the Internet of *Everything* when the human brain was connected via AI to the Internet and Kurzweil's 'cloud'. Now we have Cult operatives like Schwab calling for precisely that under the term 'Internet of Bodies', a fusion of the physical, digital and biological into one centrally-controlled Smart Grid system which the Cult refers to as the 'Fourth Industrial Revolution'. They talk about the 'biological', but they really mean the synthetic-biological which is required to fully integrate the human body and brain into the Smart Grid and artificial intelligence planned to replace the human mind. We have everything being synthetically manipulated including the natural world through GMO and smart dust, the food we eat and the human body itself with synthetic 'vaccines'. I said in *The Answer* that we would see the Cult push for synthetic meat to replace animals and in February, 2021, the so predictable psychopath Bill Gates called for the introduction of synthetic meat to save us all from 'climate change'. The climate hoax just keeps on giving like the 'Covid' hoax. The war on meat by vegan activists is a carbon (oops, sorry) copy of the manipulation of transgender activists. They have no idea (except their inner core) that they are being used to promote and impose the agenda of the Cult or that they are only the *vehicle* and not the *reason*. This is not to say those who choose not to eat meat shouldn't be respected and supported in that right, but there are ulterior motives

for those in power. A *Forbes* article in December, 2019, highlighted the plan so beloved of Schwab and the Cult under the heading: 'What Is The Internet of Bodies? And How Is It Changing Our World?' The article said the human body is the latest data platform (remember 'our vaccine is an operating system'). *Forbes* described the plan very accurately and the words could have come straight out of my books from long before:

The Internet of Bodies (IoB) is an extension of the IoT and basically connects the human body to a network through devices that are ingested, implanted, or connected to the body in some way. Once connected, data can be exchanged, and the body and device can be remotely monitored and controlled.

They were really describing a human hive mind with human perception centrally-dictated via an AI connection as well as allowing people to be 'remotely monitored and controlled'.

Everything from a fridge to a human mind could be directed from a central point by these insane psychopaths and 'Covid vaccines' are crucial to this. *Forbes* explained the process I mentioned earlier of holdable and wearable technology followed by implantable. The article said there were three generations of the Internet of Bodies that include:

- Body external: These are wearable devices such as Apple Watches or Fitbits that can monitor our health.
- Body internal: These include pacemakers, cochlear implants, and digital pills that go inside our bodies to monitor or control various aspects of health.
- Body embedded: The third generation of the Internet of Bodies is embedded technology where technology and the human body are melded together and have a real-time connection to a remote machine.

Forbes noted the development of the Brain Computer Interface (BCI) which merges the brain with an external device for monitoring and controlling in real-time. ‘The ultimate goal is to help restore function to individuals with disabilities by using brain signals rather than conventional neuromuscular pathways.’ Oh, do fuck off. The goal of brain interface technology is controlling human thought and emotion from the central point in a hive mind serving its masters wishes. Many people are now agreeing to be chipped to open doors without a key. You can recognise them because they’ll be wearing a mask, social distancing and lining up for the ‘vaccine’. The Cult plans a Great Reset money system after they have completed the demolition of the global economy in which ‘money’ will be exchanged through communication with body operating systems. Rand Corporation, a Cult-owned think tank, said of the Internet of Bodies or IoB:

Internet of Bodies technologies fall under the broader IoT umbrella. But as the name suggests, IoB devices introduce an even more intimate interplay between humans and gadgets. IoB devices monitor the human body, collect health metrics and other personal information, and transmit those data over the Internet. Many devices, such as fitness trackers, are already in use ... IoB devices ... and those in development can track, record, and store users’ whereabouts, bodily functions, and what they see, hear, and even think.

Schwab’s World Economic Forum, a long-winded way of saying ‘fascism’ or ‘the Cult’, has gone full-on with the Internet of Bodies in the ‘Covid’ era. ‘We’re entering the era of the Internet of Bodies’, it declared, ‘collecting our physical data via a range of devices that can be implanted, swallowed or worn’. The result would be a huge amount of health-related data that could improve human wellbeing around the world, and prove crucial in fighting the ‘Covid-19 pandemic’. Does anyone think these clowns care about ‘human wellbeing’ after the death and devastation their pandemic hoax has purposely caused? Schwab and co say we should move forward with the Internet of Bodies because ‘Keeping track of symptoms could help us stop the spread of infection, and quickly detect new cases’. How wonderful, but keeping track’ is all they are really bothered

about. Researchers were investigating if data gathered from smartwatches and similar devices could be used as viral infection alerts by tracking the user's heart rate and breathing. Schwab said in his 2018 book *Shaping the Future of the Fourth Industrial Revolution*:

The lines between technologies and beings are becoming blurred and not just by the ability to create lifelike robots or synthetics. Instead it is about the ability of new technologies to literally become part of us. Technologies already influence how we understand ourselves, how we think about each other, and how we determine our realities. As the technologies ... give us deeper access to parts of ourselves, we may begin to integrate digital technologies into our bodies.

You can see what the game is. Twenty-four hour control and people – if you could still call them that – would never know when something would go ping and take them out of circulation. It's the most obvious rush to a global fascist dictatorship and the complete submission of humanity and yet still so many are locked away in their Cult-induced perceptual coma and can't see it.

Smart Grid control centres

The human body is being transformed by the 'vaccines' and in other ways into a synthetic cyborg that can be attached to the global Smart Grid which would be controlled from a central point and other sub-locations of Grid manipulation. Where are these planned to be? Well, China for a start which is one of the Cult's biggest centres of operation. The technological control system and technocratic rule was incubated here to be unleashed across the world after the 'Covid' hoax came out of China in 2020. Another Smart Grid location that will surprise people new to this is Israel. I have exposed in *The Trigger* how Sabbatian technocrats, intelligence and military operatives were behind the horrors of 9/11 and not 19 Arab hijackers' who somehow manifested the ability to pilot big passenger airliners when instructors at puddle-jumping flying schools described some of them as a joke. The 9/11 attacks were made possible through control of civilian and military air computer systems and those of the White House, Pentagon and connected agencies. See *The Trigger* – it

will blow your mind. The controlling and coordinating force were the Sabbatian networks in Israel and the United States which by then had infiltrated the entire US government, military and intelligence system. The real name of the American Deep State is 'Sabbatian State'. Israel is a tiny country of only nine million people, but it is one of the global centres of cyber operations and fast catching Silicon Valley in importance to the Cult. Israel is known as the 'start-up nation' for all the cyber companies spawned there with the Sabbatian specialisation of 'cyber security' that I mentioned earlier which gives those companies access to computer systems of their clients in real time through 'backdoors' written into the coding when security software is downloaded. The Sabbatian centre of cyber operations outside Silicon Valley is the Israeli military Cyber Intelligence Unit, the biggest infrastructure project in Israel's history, headquartered in the desert-city of Beersheba and involving some 20,000 'cyber soldiers'. Here are located a literal army of Internet trolls scanning social media, forums and comment lists for anyone challenging the Cult agenda. The UK military has something similar with its 77th Brigade and associated operations. The Beersheba complex includes research and development centres for other Cult operations such as Intel, Microsoft, IBM, Google, Apple, Hewlett-Packard, Cisco Systems, Facebook and Motorola. [Techcrunch.com](#) ran an article about the Beersheba global Internet technology centre headlined 'Israel's desert city of Beersheba is turning into a cybertech oasis':

The military's massive relocation of its prestigious technology units, the presence of multinational and local companies, a close proximity to Ben Gurion University and generous government subsidies are turning Beersheba into a major global cybertech hub. Beersheba has all of the ingredients of a vibrant security technology ecosystem, including Ben Gurion University with its graduate program in cybersecurity and Cyber Security Research Center, and the presence of companies such as EMC, Deutsche Telekom, PayPal, Oracle, IBM, and Lockheed Martin. It's also the future home of the INCB (Israeli National Cyber Bureau); offers a special income tax incentive for cyber security companies, and was the site for the relocation of the army's intelligence corps units.

Sabbatians have taken over the cyber world through the following process: They scan the schools for likely cyber talent and develop them at Ben Gurion University and their period of conscription in the Israeli Defense Forces when they are stationed at the Beersheba complex. When the cyber talented officially leave the army they are funded to start cyber companies with technology developed by themselves or given to them by the state. Much of this is stolen through backdoors of computer systems around the world with America top of the list. Others are sent off to Silicon Valley to start companies or join the major ones and so we have many major positions filled by apparently 'Jewish' but really Sabbatian operatives. Google, YouTube and Facebook are all run by 'Jewish' CEOs while Twitter is all but run by ultra-Zionist hedge-fund shark Paul Singer. At the centre of the Sabbatian global cyber web is the Israeli army's Unit 8200 which specialises in hacking into computer systems of other countries, inserting viruses, gathering information, instigating malfunction, and even taking control of them from a distance. A long list of Sabbatians involved with 9/11, Silicon Valley and Israeli cyber security companies are operatives of Unit 8200. This is not about Israel. It's about the Cult. Israel is planned to be a Smart Grid hub as with China and what is happening at Beersheba is not for the benefit of Jewish people who are treated disgustingly by the Sabbatian elite that control the country. A glance at the Nuremberg Codes will tell you that.

The story is much bigger than 'Covid', important as that is to where we are being taken. Now, though, it's time to really strap in. There's more ... much more ...

CHAPTER ELEVEN

Who controls the Cult?

Awake, arise or be forever fall'n

John Milton, Paradise Lost

I have exposed this far the level of the Cult conspiracy that operates in the world of the seen and within the global secret society and satanic network which operates in the shadows one step back from the seen. The story, however, goes much deeper than that.

The 'Covid' hoax is major part of the Cult agenda, but only part, and to grasp the biggest picture we have to expand our attention beyond the realm of human sight and into the infinity of possibility that we cannot see. It is from here, ultimately, that humanity is being manipulated into a state of total control by the force which dictates the actions of the Cult. How much of reality can we see? Next to damn all is the answer. We may appear to see all there is to see in the 'space' our eyes survey and observe, but little could be further from the truth. The human 'world' is only a tiny band of frequency that the body's visual and perceptual systems can decode into *perception* of a 'world'. According to mainstream science the electromagnetic spectrum is 0.005 percent of what exists in the Universe ([Fig 10](#)). The maximum estimate I have seen is 0.5 percent and either way it's minuscule. I say it is far, far, smaller even than 0.005 percent when you compare reality we see with the totality of reality that we don't. Now get this if you are new to such information: Visible light, the only band of frequency that we can see, is a *fraction* of the 0.005

percent (Fig 11 overleaf). Take this further and realise that our universe is one of infinite universes and that universes are only a fragment of overall reality – *infinite* reality. Then compare that with the almost infinitesimal frequency band of visible light or human sight. You see that humans are as near blind as it is possible to be without actually being so. Artist and filmmaker, Sergio Toporek, said:

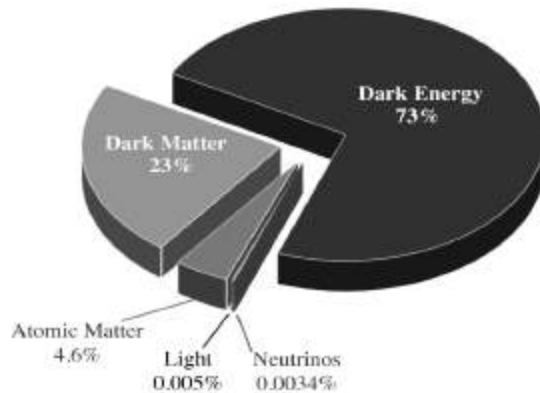


Figure 10: Humans can perceive such a tiny band of visual reality it's laughable.

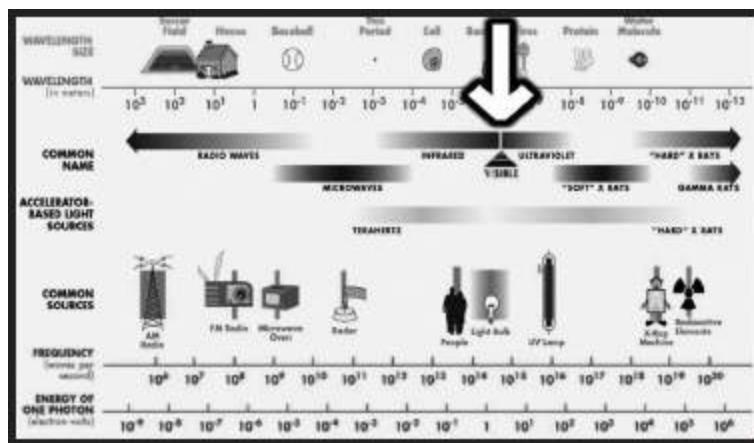


Figure 11: We can see a smear of the 0.005 percent electromagnetic spectrum, but we still know it all. Yep, makes sense.

Consider that you can see less than 1% of the electromagnetic spectrum and hear less than 1% of the acoustic spectrum. 90% of the cells in your body carry their own microbial DNA and are not 'you'. The atoms in your body are 99.99999999999999% empty space and none of them are the ones you were born with ... Human beings have 46 chromosomes, two less than a potato.

The existence of the rainbow depends on the conical photoreceptors in your eyes; to animals without cones, the rainbow does not exist. So you don't just look at a rainbow, you create it. This is pretty amazing, especially considering that all the beautiful colours you see represent less than 1% of the electromagnetic spectrum.

Suddenly the 'world' of humans looks a very different place. Take into account, too, that Planet Earth when compared with the projected size of this single universe is the equivalent of a billionth of a pinhead. Imagine the ratio that would be when compared to infinite reality. To think that Christianity once insisted that Earth and humanity were the centre of everything. This background is vital if we are going to appreciate the nature of 'human' and how we can be manipulated by an unseen force. To human visual reality virtually *everything* is unseen and yet the prevailing perception within the institutions and so much of the public is that if we can't see it, touch it, hear it, taste it and smell it then it cannot exist. Such perception is indoctrinated and encouraged by the Cult and its agents because it isolates believers in the strictly limited, village-idiot, realm of the five senses where perceptions can be firewalled and information controlled. Most of those perpetuating the 'this-world-is-all-there-is' insanity are themselves indoctrinated into believing the same delusion. While major players and influencers know that official reality is laughable most of those in science, academia and medicine really believe the nonsense they peddle and teach succeeding generations. Those who challenge the orthodoxy are dismissed as nutters and freaks to protect the manufactured illusion from exposure. Observe the dynamic of the 'Covid' hoax and you will see how that takes the same form. The inner-circle psychopaths know it's a gigantic scam, but almost the entirety of those imposing their fascist rules believe that 'Covid' is all that they're told it is.

Stolen identity

Ask people who they are and they will give you their name, place of birth, location, job, family background and life story. Yet that is not who they are – it is what they are *experiencing*. The difference is *absolutely crucial*. The true 'I', the eternal, infinite 'I', is consciousness,

a state of being aware. Forget ‘form’. That is a vehicle for a brief experience. Consciousness does not come *from* the brain, but *through* the brain and even that is more symbolic than literal. We are awareness, pure awareness, and this is what withdraws from the body at what we call ‘death’ to continue our eternal beingness, *isness*, in other realms of reality within the limitlessness of infinity or the Biblical ‘many mansions in my father’s house’. Labels of a human life, man, woman, transgender, black, white, brown, nationality, circumstances and income are not who we are. They are what we are – awareness – is *experiencing* in a brief connection with a band of frequency we call ‘human’. The labels are not the self; they are, to use the title of one of my books, a *Phantom Self*. I am not David Icke born in Leicester, England, on April 29th, 1952. I am the consciousness *having that experience*. The Cult and its non-human masters seek to convince us through the institutions of ‘education’, science, medicine, media and government that what we are *experiencing* is who we *are*. It’s so easy to control and direct perception locked away in the bewildered illusions of the five senses with no expanded radar. Try, by contrast, doing the same with a humanity aware of its true self and its true power to consciously create its reality and experience. How is it possible to do this? We do it all day every day. If you perceive yourself as ‘little me’ with no power to impact upon your life and the world then your life experience will reflect that. You will hand the power you don’t think you have to authority in all its forms which will use it to control your experience. This, in turn, will appear to confirm your perception of ‘little me’ in a self-fulfilling feedback loop. But that is what ‘little me’ really is – a *perception*. We are all ‘big-me’, infinite me, and the Cult has to make us forget that if its will is to prevail. We are therefore manipulated and pressured into self-identifying with human labels and not the consciousness/awareness *experiencing* those human labels.

The phenomenon of identity politics is a Cult-instigated manipulation technique to sub-divide previous labels into even smaller ones. A United States university employs this list of letters to

describe student identity: LGBTQQFAGPBDSM or lesbian, gay, bisexual, transgender, transsexual, queer, questioning, flexual, asexual, gender-fuck, polyamorous, bondage/discipline, dominance/submission and sadism/masochism. I'm sure other lists are even longer by now as people feel the need to self-identify the 'I' with the minutiae of race and sexual preference. Wokers programmed by the Cult for generations believe this is about 'inclusivity' when it's really the Cult locking them away into smaller and smaller versions of Phantom Self while firewalls them from the influence of their true self, the infinite, eternal 'I'. You may notice that my philosophy which contends that we are all unique points of attention/awareness within the same infinite whole or Oneness is the ultimate non-racism. The very sense of Oneness makes the judgement of people by their body-type, colour or sexuality utterly ridiculous and confirms that racism has no understanding of reality (including anti-white racism). Yet despite my perception of life Cult agents and fast-asleep Wokers label me racist to discredit my information while they are themselves phenomenally racist and sexist. All they see is race and sexuality and they judge people as good or bad, demons or untouchables, by their race and sexuality. All they see is *Phantom Self* and perceive themselves in terms of *Phantom Self*. They are pawns and puppets of the Cult agenda to focus attention and self-identity in the five senses and play those identities against each other to divide and rule. Columbia University has introduced segregated graduations in another version of social distancing designed to drive people apart and teach them that different racial and cultural groups have nothing in common with each other. The last thing the Cult wants is unity. Again the pump-primers of this will be Cult operatives in the knowledge of what they are doing, but the rest are just the *Phantom Self* blind leading the *Phantom Self* blind. We *do* have something in common – we are all *the same consciousness* having different temporary experiences.

What is this 'human'?

Yes, what *is* ‘human’? That is what we are supposed to be, right? I mean ‘human’? True, but ‘human’ is the experience not the ‘I’. Break it down to basics and ‘human’ is the way that information is processed. If we are to experience and interact with this band of frequency we call the ‘world’ we must have a vehicle that operates within that band of frequency. Our consciousness in its prime form cannot do that; it is way beyond the frequency of the human realm. My consciousness or awareness could not tap these keys and pick up the cup in front of me in the same way that radio station A cannot interact with radio station B when they are on different frequencies. The human body is the means through which we have that interaction. I have long described the body as a biological computer which processes information in a way that allows consciousness to experience this reality. The body is a receiver, transmitter and processor of information in a particular way that we call human. We visually perceive only the world of the five senses in a wakened state – that is the limit of the body’s visual decoding system. In truth it’s not even visual in the way we experience ‘visual reality’ as I will come to in a moment. We are ‘human’ because the body processes the information sources of human into a reality and behaviour system that we *perceive* as human. Why does an elephant act like an elephant and not like a human or a duck? The elephant’s biological computer is a different information field and processes information according to that program into a visual and behaviour type we call an elephant. The same applies to everything in our reality. These body information fields are perpetuated through procreation (like making a copy of a software program). The Cult wants to break that cycle and intervene technologically to transform the human information field into one that will change what we call humanity. If it can change the human information field it will change the way that field processes information and change humanity both ‘physically’ and psychologically. Hence the *messenger* (information) RNA ‘vaccines’ and so much more that is targeting human genetics by changing the body’s information – *messaging* – construct through food, drink, radiation, toxicity and other means.

Reality that we experience is nothing like reality as it really is in the same way that the reality people experience in virtual reality games is not the reality they are really living in. The game is only a decoded source of information that appears to be a reality. Our world is also an information construct – a *simulation* (more later). In its base form our reality is a wavefield of information much the same in theme as Wi-Fi. The five senses decode wavefield information into electrical information which they communicate to the brain to decode into holographic (illusory ‘physical’) information. Different parts of the brain specialise in decoding different senses and the information is fused into a reality that appears to be outside of us but is really inside the brain and the genetic structure in general ([Fig 12](#) overleaf). DNA is a receiver-transmitter of information and a vital part of this decoding process and the body’s connection to other realities. Change DNA and you change the way we decode and connect with reality – see ‘Covid vaccines’. Think of computers decoding Wi-Fi. You have information encoded in a radiation field and the computer decodes that information into a very different form on the screen. You can’t see the Wi-Fi until its information is made manifest on the screen and the information on the screen is inside the computer and not outside. I have just described how we decode the ‘human world’. All five senses decode the waveform ‘Wi-Fi’ field into electrical signals and the brain (computer) constructs reality inside the brain and not outside – ‘You don’t just look at a rainbow, you create it’. Sound is a simple example. We don’t hear sound until the brain decodes it. Waveform sound waves are picked up by the hearing sense and communicated to the brain in an electrical form to be decoded into the sounds that we hear. Everything we hear is inside the brain along with everything we see, feel, smell and taste. Words and language are waveform fields generated by our vocal chords which pass through this process until they are decoded by the brain into words that we hear. Different languages are different frequency fields or sound waves generated by vocal chords. Late British philosopher Alan Watts said:

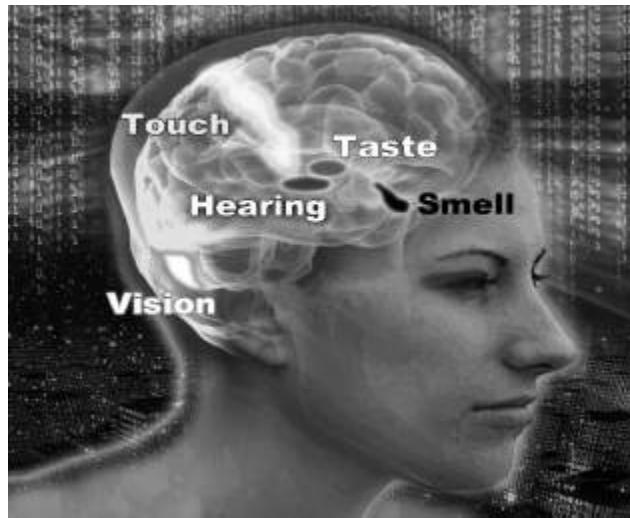


Figure 12: The brain receives information from the five senses and constructs from that our perceived reality.

[Without the brain] the world is devoid of light, heat, weight, solidity, motion, space, time or any other imaginable feature. All these phenomena are interactions, or transactions, of vibrations with a certain arrangement of neurons.

That's exactly what they are and scientist Robert Lanza describes in his book, *Biocentrism*, how we decode electromagnetic waves and energy into visual and 'physical' experience. He uses the example of a flame emitting photons, electromagnetic energy, each pulsing electrically and magnetically:

... these ... invisible electromagnetic waves strike a human retina, and if (and only if) the waves happen to measure between 400 and 700 nano meters in length from crest to crest, then their energy is just right to deliver a stimulus to the 8 million cone-shaped cells in the retina.

Each in turn send an electrical pulse to a neighbour neuron, and on up the line this goes, at 250 mph, until it reaches the ... occipital lobe of the brain, in the back of the head. There, a cascading complex of neurons fire from the incoming stimuli, and we subjectively perceive this experience as a yellow brightness occurring in a place we have been conditioned to call the 'external world'.

You hear what you decode

If a tree falls or a building collapses they make no noise unless someone is there to decode the energetic waves generated by the disturbance into what we call sound. Does a falling tree make a noise? Only if you hear it – *decode* it. Everything in our reality is a frequency field of information operating within the overall ‘Wi-Fi’ field that I call The Field. A vibrational disturbance is generated in The Field by the fields of the falling tree or building. These disturbance waves are what we decode into the sound of them falling. If no one is there to do that then neither will make any noise. Reality is created by the observer – *decoder* – and the *perceptions* of the observer affect the decoding process. For this reason different people – different *perceptions* – will perceive the same reality or situation in a different way. What one may perceive as a nightmare another will see as an opportunity. The question of why the Cult is so focused on controlling human perception now answers itself. All experienced reality is the act of decoding and we don’t experience Wi-Fi until it is decoded on the computer screen. The sight and sound of an Internet video is encoded in the Wi-Fi all around us, but we don’t see or hear it until the computer decodes that information. Taste, smell and touch are all phenomena of the brain as a result of the same process. We don’t taste, smell or feel anything except in the brain and there are pain relief techniques that seek to block the signal from the site of discomfort to the brain because if the brain doesn’t decode that signal we don’t feel pain. Pain is in the brain and only appears to be at the point of impact thanks to the feedback loop between them. We don’t see anything until electrical information from the sight senses is decoded in an area at the back of the brain. If that area is damaged we can go blind when our eyes are perfectly okay. So why do we go blind if we damage an eye? We damage the information processing between the waveform visual information and the visual decoding area of the brain. If information doesn’t reach the brain in a form it can decode then we can’t see the visual reality that it represents. What’s more the brain is decoding only a fraction of the information it receives and the rest is absorbed by the

sub-conscious mind. This explanation is from the science magazine, *Wonderpedia*:

Every second, 11 million sensations crackle along these [brain] pathways ... The brain is confronted with an alarming array of images, sounds and smells which it rigorously filters down until it is left with a manageable list of around 40. Thus 40 sensations per second make up what we perceive as reality.

The ‘world’ is not what people are told to believe that is it and the inner circles of the Cult *know that*.

Illusory ‘physical’ reality

We can only see a smear of 0.005 percent of the Universe which is only one of a vast array of universes – ‘mansions’ – within infinite reality. Even then the brain decodes only 40 pieces of information (‘sensations’) from a potential *11 million* that we receive every second. Two points strike you from this immediately: The sheer breathtaking stupidity of believing we know anything so rigidly that there’s nothing more to know; and the potential for these processes to be manipulated by a malevolent force to control the reality of the population. One thing I can say for sure with no risk of contradiction is that when you can perceive an almost indescribable fraction of infinite reality there is always more to know as in tidal waves of it. Ancient Greek philosopher Socrates was so right when he said that wisdom is to know how little we know. How obviously true that is when you think that we are experiencing a physical world of solidity that is neither physical nor solid and a world of apartness when everything is connected. Cult-controlled ‘science’ dismisses the so-called ‘paranormal’ and all phenomena related to that when the ‘para’-normal is perfectly normal and explains the alleged ‘great mysteries’ which dumbfound scientific minds. There is a reason for this. A ‘scientific mind’ in terms of the mainstream is a material mind, a five-sense mind imprisoned in see it, touch it, hear it, smell it and taste it. Phenomena and happenings that can’t be explained that way leave the ‘scientific mind’ bewildered and the rule is that if they

can't account for why something is happening then it can't, by definition, be happening. I beg to differ. Telepathy is thought waves passing through The Field (think wave disturbance again) to be decoded by someone able to connect with that wavelength (information). For example: You can pick up the thought waves of a friend at any distance and at the very least that will bring them to mind. A few minutes later the friend calls you. 'My god', you say, 'that's incredible – I was just thinking of you.' Ah, but *they* were thinking of *you* before they made the call and that's what you decoded. Native peoples not entrapped in five-sense reality do this so well it became known as the 'bush telegraph'. Those known as psychics and mediums (genuine ones) are doing the same only across dimensions of reality. 'Mind over matter' comes from the fact that matter and mind are the *same*. The state of one influences the state of the other. Indeed one *and* the other are illusions. They are aspects of the same field. Paranormal phenomena are all explainable so why are they still considered 'mysteries' or not happening? Once you go down this road of understanding you begin to expand awareness beyond the five senses and that's the nightmare for the Cult.



Figure 13: Holograms are not solid, but the best ones appear to be.

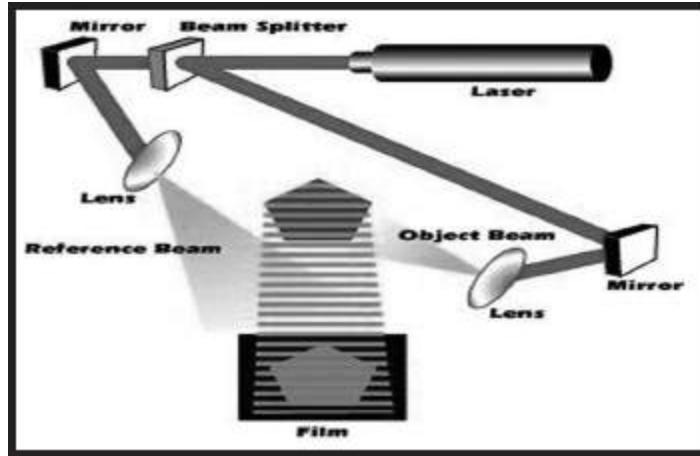


Figure 14: How holograms are created by capturing a waveform version of the subject image.

Holographic ‘solidity’

Our reality is not solid, it is holographic. We are now well aware of holograms which are widely used today. Two-dimensional information is decoded into a three-dimensional reality that is not solid although can very much appear to be (Fig 13). Holograms are created with a laser divided into two parts. One goes directly onto a photographic print ('reference beam') and the other takes a waveform image of the subject ('working beam') before being directed onto the print where it 'collides' with the other half of the laser (Fig 14). This creates a *waveform* interference pattern which contains the wavefield information of whatever is being photographed (Fig 15 overleaf). The process can be likened to dropping pebbles in a pond. Waves generated by each one spread out across the water to collide with the others and create a wave representation of where the stones fell and at what speed, weight and distance. A waveform interference pattern of a hologram is akin to the waveform information in The Field which the five senses decode into electrical signals to be decoded by the brain into a holographic illusory 'physical' reality. In the same way when a laser (think human attention) is directed at the waveform interference pattern a three-dimensional version of the subject is projected into apparently 'solid' reality (Fig 16). An amazing trait of holograms reveals more 'paranormal mysteries'. Information of the *whole*

hologram is encoded in waveform in every part of the interference pattern by the way they are created. This means that every *part* of a hologram is a smaller version of the whole. Cut the interference wave-pattern into four and you won't get four parts of the image. You get quarter-sized versions of the *whole* image. The body is a hologram and the same applies. Here we have the basis of acupuncture, reflexology and other forms of healing which identify representations of the whole body in all of the parts, hands, feet, ears, everywhere. Skilled palm readers can do what they do because the information of whole body is encoded in the hand. The concept of as above, so below, comes from this.



Figure 15: A waveform interference pattern that holds the information that transforms into a hologram.



Figure 16: Holographic people including 'Elvis' holographically inserted to sing a duet with Celine Dion.

The question will be asked of why, if solidity is illusory, we can't just walk through walls and each other. The resistance is not solid against solid; it is electromagnetic field against electromagnetic field and we decode this into the *experience* of solid against solid. We should also not underestimate the power of belief to dictate reality. What you believe is impossible *will be*. Your belief impacts on your decoding processes and they won't decode what you think is impossible. What we believe we perceive and what we perceive we experience. 'Can't dos' and 'impossibles' are like a firewall in a computer system that won't put on the screen what the firewall blocks. How vital that is to understanding how human experience has been hijacked. I explain in *The Answer, Everything You Need To Know But Have Never Been Told* and other books a long list of 'mysteries' and 'paranormal' phenomena that are not mysterious and perfectly normal once you realise what reality is and how it works. 'Ghosts' can be seen to pass through 'solid' walls because the walls are not solid and the ghost is a discarnate entity operating on a frequency so different to that of the wall that it's like two radio stations sharing the same space while never interfering with each other. I have seen ghosts do this myself. The apartness of people and objects is also an illusion. Everything is connected by the Field like all sea life is connected by the sea. It's just that within the limits of our visual reality we only 'see' holographic information and not the field of information that connects everything and from which the holographic world is made manifest. If you can only see holographic 'objects' and not the field that connects them they will appear to you as unconnected to each other in the same way that we see the computer while not seeing the Wi-Fi.

What you don't know *can* hurt you

Okay, we return to those 'two worlds' of human society and the Cult with its global network of interconnecting secret societies and satanic groups which manipulate through governments, corporations, media, religions, etc. The fundamental difference between them is *knowledge*. The idea has been to keep humanity

ignorant of the plan for its total enslavement underpinned by a crucial ignorance of reality – who we are and where we are – and how we interact with it. ‘Human’ should be the interaction between our expanded eternal consciousness and the five-sense body experience. We are meant to be *in* this world in terms of the five senses but not *of* this world in relation to our greater consciousness and perspective. In that state we experience the small picture of the five senses within the wider context of the big picture of awareness beyond the five senses. Put another way the five senses see the dots and expanded awareness connects them into pictures and patterns that give context to the apparently random and unconnected. Without the context of expanded awareness the five senses see only apartness and randomness with apparently no meaning. The Cult and its other-dimensional controllers seek to intervene in the frequency realm where five-sense reality is supposed to connect with expanded reality and to keep the two apart (more on this in the final chapter). When that happens five-sense mental and emotional processes are no longer influenced by expanded awareness, or the True ‘I’, and instead are driven by the isolated perceptions of the body’s decoding systems. They are in the world *and* of it. Here we have the human plight and why humanity with its potential for infinite awareness can be so easily manipulatable and descend into such extremes of stupidity.

Once the Cult isolates five-sense mind from expanded awareness it can then program the mind with perceptions and beliefs by controlling information that the mind receives through the ‘education’ system of the formative years and the media perceptual bombardment and censorship of an entire lifetime. Limit perception and a sense of the possible through limiting knowledge by limiting and skewing information while censoring and discrediting that which could set people free. As the title of another of my books says ... *And The Truth Shall Set You Free*. For this reason the last thing the Cult wants in circulation is the truth about anything – especially the reality of the eternal ‘I’ – and that’s why it is desperate to control information. The Cult knows that information becomes perception

which becomes behaviour which, collectively, becomes human society. Cult-controlled and funded mainstream ‘science’ denies the existence of an eternal ‘I’ and seeks to dismiss and trash all evidence to the contrary. Cult-controlled mainstream religion has a version of ‘God’ that is little more than a system of control and dictatorship that employs threats of damnation in an afterlife to control perceptions and behaviour in the here and now through fear and guilt. Neither is true and it’s the ‘neither’ that the Cult wishes to suppress. This ‘neither’ is that everything is an expression, a point of attention, within an infinite state of consciousness which is the real meaning of the term ‘God’.

Perceptual obsession with the ‘physical body’ and five-senses means that ‘God’ becomes personified as a bearded bloke sitting among the clouds or a raging bully who loves us if we do what ‘he’ wants and condemns us to the fires of hell if we don’t. These are no more than a ‘spiritual’ fairy tales to control and dictate events and behaviour through fear of this ‘God’ which has bizarrely made ‘God-fearing’ in religious circles a state to be desired. I would suggest that fearing *anything* is not to be encouraged and celebrated, but rather deleted. You can see why ‘God fearing’ is so beneficial to the Cult and its religions when *they* decide what ‘God’ wants and what ‘God’ demands (the Cult demands) that everyone do. As the great American comedian Bill Hicks said satirising a Christian zealot: ‘I think what God meant to say.’ How much of this infinite awareness (“God”) that we access is decided by how far we choose to expand our perceptions, self-identity and sense of the possible. The scale of self-identity reflects itself in the scale of awareness that we can connect with and are influenced by – how much knowing and insight we have instead of programmed perception. You cannot expand your awareness into the infinity of possibility when you believe that you are little me Peter the postman or Mary in marketing and nothing more. I’ll deal with this in the concluding chapter because it’s crucial to how we turnaround current events.

Where the Cult came from

When I realised in the early 1990s there was a Cult network behind global events I asked the obvious question: When did it start? I took it back to ancient Rome and Egypt and on to Babylon and Sumer in Mesopotamia, the 'Land Between Two Rivers', in what we now call Iraq. The two rivers are the Tigris and Euphrates and this region is of immense historical and other importance to the Cult, as is the land called Israel only 550 miles away by air. There is much more going with deep esoteric meaning across this whole region. It's not only about 'wars for oil'. Priceless artefacts from Mesopotamia were stolen or destroyed after the American and British invasion of Iraq in 2003 justified by the lies of Boy Bush and Tony Blair (their Cult masters) about non-existent 'weapons of mass destruction'.

Mesopotamia was the location of Sumer (about 5,400BC to 1,750BC), and Babylon (about 2,350BC to 539BC). Sabbatians may have become immensely influential in the Cult in modern times but they are part of a network that goes back into the mists of history. Sumer is said by historians to be the 'cradle of civilisation'. I disagree. I say it was the re-start of what we call human civilisation after cataclysmic events symbolised in part as the 'Great Flood' destroyed the world that existed before. These fantastic upheavals that I have been describing in detail in the books since the early 1990s appear in accounts and legends of ancient cultures across the world and they are supported by geological and biological evidence. Stone tablets found in Iraq detailing the Sumer period say the cataclysms were caused by non-human 'gods' they call the Anunnaki. These are described in terms of extraterrestrial visitations in which knowledge supplied by the Anunnaki is said to have been the source of at least one of the world's oldest writing systems and developments in astronomy, mathematics and architecture that were way ahead of their time. I have covered this subject at length in *The Biggest Secret* and *Children of the Matrix* and the same basic 'Anunnaki' story can be found in Zulu accounts in South Africa where the late and very great Zulu high shaman Credo Mutwa told me that the Sumerian Anunnaki were known by Zulus as the Chitauri or 'children of the serpent'. See my six-hour video interview with Credo on this subject entitled *The*

Reptilian Agenda recorded at his then home near Johannesburg in 1999 which you can watch on the Ickonic media platform.

The Cult emerged out of Sumer, Babylon and Egypt (and elsewhere) and established the Roman Empire before expanding with the Romans into northern Europe from where many empires were savagely imposed in the form of Cult-controlled societies all over the world. Mass death and destruction was their calling card. The Cult established its centre of operations in Europe and European Empires were Cult empires which allowed it to expand into a global force. Spanish and Portuguese colonialists headed for Central and South America while the British and French targeted North America. Africa was colonised by Britain, France, Belgium, the Netherlands, Portugal, Spain, Italy, and Germany. Some like Britain and France moved in on the Middle East. The British Empire was by far the biggest for a simple reason. By now Britain was the headquarters of the Cult from which it expanded to form Canada, the United States, Australia and New Zealand. The Sun never set on the British Empire such was the scale of its occupation. London remains a global centre for the Cult along with Rome and the Vatican although others have emerged in Israel and China. It is no accident that the 'virus' is alleged to have come out of China while Italy was chosen as the means to terrify the Western population into compliance with 'Covid' fascism. Nor that Israel has led the world in 'Covid' fascism and mass 'vaccination'.

You would think that I would mention the United States here, but while it has been an important means of imposing the Cult's will it is less significant than would appear and is currently in the process of having what power it does have deleted. The Cult in Europe has mostly loaded the guns for the US to fire. America has been controlled from Europe from the start through Cult operatives in Britain and Europe. The American Revolution was an illusion to make it appear that America was governing itself while very different forces were pulling the strings in the form of Cult families such as the Rothschilds through the Rockefellers and other subordinates. The Rockefellers are extremely close to Bill Gates and

established both scalpel and drug ‘medicine’ and the World Health Organization. They play a major role in the development and circulation of vaccines through the Rockefeller Foundation on which Bill Gates said his Foundation is based. Why wouldn’t this be the case when the Rockefellers and Gates are on the same team? Cult infiltration of human society goes way back into what we call history and has been constantly expanding and centralising power with the goal of establishing a global structure to dictate everything. Look how this has been advanced in great leaps with the ‘Covid’ hoax.

The non-human dimension

I researched and observed the comings and goings of Cult operatives through the centuries and even thousands of years as they were born, worked to promote the agenda within the secret society and satanic networks, and then died for others to replace them. Clearly there had to be a coordinating force that spanned this entire period while operatives who would not have seen the end goal in their lifetimes came and went advancing the plan over millennia. I went in search of that coordinating force with the usual support from the extraordinary synchronicity of my life which has been an almost daily experience since 1990. I saw common themes in religious texts and ancient cultures about a non-human force manipulating human society from the hidden. Christianity calls this force Satan, the Devil and demons; Islam refers to the Jinn or Djinn; Zulus have their Chitauri (spelt in other ways in different parts of Africa); and the Gnostic people in Egypt in the period around and before 400AD referred to this phenomena as the ‘Archons’, a word meaning rulers in Greek. Central American cultures speak of the ‘Predators’ among other names and the same theme is everywhere. I will use ‘Archons’ as a collective name for all of them. When you see how their nature and behaviour is described all these different sources are clearly talking about the same force. Gnostics described the Archons in terms of ‘luminous fire’ while Islam relates the Jinn to ‘smokeless fire’. Some refer to beings in form that could occasionally be seen, but the most common of common theme is that they operate from

unseen realms which means almost all existence to the visual processes of humans. I had concluded that this was indeed the foundation of human control and that the Cult was operating within the human frequency band on behalf of this hidden force when I came across the writings of Gnostics which supported my conclusions in the most extraordinary way.

A sealed earthen jar was found in 1945 near the town of Nag Hammadi about 75-80 miles north of Luxor on the banks of the River Nile in Egypt. Inside was a treasure trove of manuscripts and texts left by the Gnostic people some 1,600 years earlier. They included 13 leather-bound papyrus codices (manuscripts) and more than 50 texts written in Coptic Egyptian estimated to have been hidden in the jar in the period of 400AD although the source of the information goes back much further. Gnostics oversaw the Great or Royal Library of Alexandria, the fantastic depository of ancient texts detailing advanced knowledge and accounts of human history. The Library was dismantled and destroyed in stages over a long period with the death-blow delivered by the Cult-established Roman Church in the period around 415AD. The Church of Rome was the Church of Babylon relocated as I said earlier. Gnostics were not a race. They were a way of perceiving reality. Whenever they established themselves and their information circulated the terrorists of the Church of Rome would target them for destruction. This happened with the Great Library and with the Gnostic Cathars who were burned to death by the psychopaths after a long period of oppression at the siege of the Castle of Monségur in southern France in 1244. The Church has always been terrified of Gnostic information which demolishes the official Christian narrative although there is much in the Bible that supports the Gnostic view if you read it in another way. To anyone studying the texts of what became known as the Nag Hammadi Library it is clear that great swathes of Christian and Biblical belief has its origin with Gnostics sources going back to Sumer. Gnostic themes have been twisted to manipulate the perceived reality of Bible believers. Biblical texts have been in the open for centuries where they could be changed while Gnostic

documents found at Nag Hammadi were sealed away and untouched for 1,600 years. What you see is what they wrote.

Use your *pneuma* not your *nous*

Gnosticism and Gnostic come from 'gnosis' which means knowledge, or rather *secret* knowledge, in the sense of spiritual awareness – knowledge about reality and life itself. The desperation of the Cult's Church of Rome to destroy the Gnostics can be understood when the knowledge they were circulating was the last thing the Cult wanted the population to know. Sixteen hundred years later the same Cult is working hard to undermine and silence me for the same reason. The dynamic between knowledge and ignorance is a constant. 'Time' appears to move on, but essential themes remain the same. We are told to 'use your *nous*', a Gnostic word for head/brain/intelligence. They said, however, that spiritual awakening or 'salvation' could only be secured by expanding awareness *beyond* what they called *nous* and into *pneuma* or Infinite Self. Obviously as I read these texts the parallels with what I have been saying since 1990 were fascinating to me. There is a universal truth that spans human history and in that case why wouldn't we be talking the same language 16 centuries apart? When you free yourself from the perception program of the five senses and explore expanded realms of consciousness you are going to connect with the same information no matter what the perceived 'era' within a manufactured timeline of a single and tiny range of manipulated frequency. Humans working with 'smart' technology or knocking rocks together in caves is only a timeline appearing to operate within the human frequency band. Expanded awareness and the knowledge it holds have always been there whether the era be Stone Age or computer age. We can only access that knowledge by opening ourselves to its frequency which the five-sense prison cell is designed to stop us doing. Gates, Fauci, Whitty, Vallance, Zuckerberg, Brin, Page, Wojcicki, Bezos, and all the others behind the 'Covid' hoax clearly have a long wait before their range of frequency can make that connection given that an open heart is

crucial to that as we shall see. Instead of accessing knowledge directly through expanded awareness it is given to Cult operatives by the secret society networks of the Cult where it has been passed on over thousands of years outside the public arena. Expanded realms of consciousness is where great artists, composers and writers find their inspiration and where truth awaits anyone open enough to connect with it. We need to go there fast.

Archon hijack

A fifth of the Nag Hammadi texts describe the existence and manipulation of the Archons led by a 'Chief Archon' they call 'Yaldabaoth', or the 'Demiurge', and this is the Christian 'Devil', 'Satan', 'Lucifer', and his demons. Archons in Biblical symbolism are the 'fallen ones' which are also referred to as fallen angels after the angels expelled from heaven according to the Abrahamic religions of Judaism, Christianity and Islam. These angels are claimed to tempt humans to 'sin' ongoing and you will see how accurate that symbolism is during the rest of the book. The theme of 'original sin' is related to the 'Fall' when Adam and Eve were 'tempted by the serpent' and fell from a state of innocence and 'obedience' (connection) with God into a state of disobedience (disconnection). The Fall is said to have brought sin into the world and corrupted everything including human nature. Yaldabaoth, the 'Lord Archon', is described by Gnostics as a 'counterfeit spirit', 'The Blind One', 'The Blind God', and 'The Foolish One'. The Jewish name for Yaldabaoth in Talmudic writings is Samael which translates as 'Poison of God', or 'Blindness of God'. You see the parallels. Yaldabaoth in Islamic belief is the Muslim Jinn devil known as Shaytan – Shaytan is Satan as the same themes are found all over the world in every religion and culture. The 'Lord God' of the Old Testament is the 'Lord Archon' of Gnostic manuscripts and that's why he's such a bloodthirsty bastard. Satan is known by Christians as 'the Demon of Demons' and Gnostics called Yaldabaoth the 'Archon of Archons'. Both are known as 'The Deceiver'. We are talking about the same 'bloke' for sure and these common themes

using different names, storylines and symbolism tell a common tale of the human plight.

Archons are referred to in Nag Hammadi documents as mind parasites, inverters, guards, gatekeepers, detainers, judges, pitiless ones and deceivers. The 'Covid' hoax alone is a glaring example of all these things. The Biblical 'God' is so different in the Old and New Testaments because they are not describing the same phenomenon. The vindictive, angry, hate-filled, 'God' of the Old Testament, known as Yahweh, is Yaldabaoth who is depicted in Cult-dictated popular culture as the 'Dark Lord', 'Lord of Time', Lord (Darth) Vader and Dormammu, the evil ruler of the 'Dark Dimension' trying to take over the 'Earth Dimension' in the Marvel comic movie, *Dr Strange*. Yaldabaoth is both the Old Testament 'god' and the Biblical 'Satan'. Gnostics referred to Yaldabaoth as the 'Great Architect of the Universe' and the Cult-controlled Freemason network calls their god 'the Great Architect of the Universe' (also Grand Architect). The 'Great Architect' Yaldabaoth is symbolised by the Cult as the all-seeing eye at the top of the pyramid on the Great Seal of the United States and the dollar bill. Archon is encoded in *arch-itect* as it is in *arch-angels* and *arch-bishops*. All religions have the theme of a force for good and force for evil in some sort of spiritual war and there is a reason for that – the theme is true. The Cult and its non-human masters are quite happy for this to circulate. They present themselves as the force for good fighting evil when they are really the force of evil (absence of love). The whole foundation of Cult modus operandi is inversion. They promote themselves as a force for good and anyone challenging them in pursuit of peace, love, fairness, truth and justice is condemned as a satanic force for evil. This has been the game plan throughout history whether the Church of Rome inquisitions of non-believers or 'conspiracy theorists' and 'anti-vaxxers' of today. The technique is the same whatever the timeline era.

Yaldabaoth is revolting (true)

Yaldabaoth and the Archons are said to have revolted against God with Yaldabaoth claiming to *be* God – the *All That Is*. The Old Testament ‘God’ (Yaldabaoth) demanded to be worshipped as such: ‘*I am the LORD, and there is none else, there is no God beside me*’ (Isaiah 45:5). I have quoted in other books a man who said he was the unofficial son of the late Baron Philippe de Rothschild of the Mouton-Rothschild wine producing estates in France who died in 1988 and he told me about the Rothschild ‘revolt from God’. The man said he was given the name Phillip Eugene de Rothschild and we shared long correspondence many years ago while he was living under another identity. He said that he was conceived through ‘occult incest’ which (within the Cult) was ‘normal and to be admired’. ‘Phillip’ told me about his experience attending satanic rituals with rich and famous people whom he names and you can see them and the wider background to Cult Satanism in my other books starting with *The Biggest Secret*. Cult rituals are interactions with Archontic ‘gods’. ‘Phillip’ described Baron Philippe de Rothschild as ‘a master Satanist and hater of God’ and he used the same term ‘revolt from God’ associated with Yaldabaoth/Satan/Lucifer/the Devil in describing the Sabbatian Rothschild dynasty. ‘I played a key role in my family’s revolt from God’, he said. That role was to infiltrate in classic Sabbatian style the Christian Church, but eventually he escaped the mind-prison to live another life. The Cult has been targeting religion in a plan to make worship of the Archons the global one-world religion. Infiltration of Satanism into modern ‘culture’, especially among the young, through music videos, stage shows and other means, is all part of this.

Nag Hammadi texts describe Yaldabaoth and the Archons in their prime form as energy – consciousness – and say they can take form if they choose in the same way that consciousness takes form as a human. Yaldabaoth is called ‘formless’ and represents a deeply inverted, distorted and chaotic state of consciousness which seeks to attach to humans and turn them into a likeness of itself in an attempt at assimilation. For that to happen it has to manipulate

humans into low frequency mental and emotional states that match its own. Archons can certainly appear in human form and this is the origin of the psychopathic personality. The energetic distortion Gnostics called Yaldabaoth is psychopathy. When psychopathic Archons take human form that human will be a psychopath as an expression of Yaldabaoth consciousness. Cult psychopaths are Archons in human form. The principle is the same as that portrayed in the 2009 *Avatar* movie when the American military travelled to a fictional Earth-like moon called Pandora in the Alpha Centauri star system to infiltrate a society of blue people, or Na'vi, by hiding within bodies that looked like the Na'vi. Archons posing as humans have a particular hybrid information field, part human, part Archon, (the ancient 'demigods') which processes information in a way that manifests behaviour to match their psychopathic evil, lack of empathy and compassion, and stops them being influenced by the empathy, compassion and love that a fully-human information field is capable of expressing. Cult bloodlines interbreed, be they royalty or dark suits, for this reason and you have their obsession with incest. Interbreeding with full-blown humans would dilute the Archontic energy field that guarantees psychopathy in its representatives in the human realm.

Gnostic writings say the main non-human forms that Archons take are *serpentine* (what I have called for decades 'reptilian' amid unbounded ridicule from the Archontically-programmed) and what Gnostics describe as 'an unborn baby or foetus with grey skin and dark, unmoving eyes'. This is an excellent representation of the ET 'Greys' of UFO folklore which large numbers of people claim to have seen and been abducted by – Zulu shaman Credo Mutwa among them. I agree with those that believe in extraterrestrial or interdimensional visitations today and for thousands of years past. No wonder with their advanced knowledge and technological capability they were perceived and worshipped as gods for technological and other 'miracles' they appeared to perform. Imagine someone arriving in a culture disconnected from the modern world with a smartphone and computer. They would be

seen as a ‘god’ capable of ‘miracles’. The Renegade Mind, however, wants to know the source of everything and not only the way that source manifests as human or non-human. In the same way that a Renegade Mind seeks the original source material for the ‘Covid virus’ to see if what is claimed is true. The original source of Archons in form is consciousness – the distorted state of consciousness known to Gnostics as Yaldabaoth.

‘Revolt from God’ is energetic disconnection

Where I am going next will make a lot of sense of religious texts and ancient legends relating to ‘Satan’, Lucifer’ and the ‘gods’. Gnostic descriptions sync perfectly with the themes of my own research over the years in how they describe a consciousness distortion seeking to impose itself on human consciousness. I’ve referred to the core of infinite awareness in previous books as Infinite Awareness in Awareness of Itself. By that I mean a level of awareness that knows that it is all awareness and is aware of all awareness. From here comes the frequency of love in its true sense and balance which is what love is on one level – the balance of all forces into a single whole called Oneness and Isness. The more we disconnect from this state of love that many call ‘God’ the constituent parts of that Oneness start to unravel and express themselves as a part and not a whole. They become individualised as intellect, mind, selfishness, hatred, envy, desire for power over others, and such like. This is not a problem in the greater scheme in that ‘God’, the *All That Is*, can experience all these possibilities through different expressions of itself including humans. What we as expressions of the whole experience the *All That Is* experiences. We are the *All That Is* experiencing itself. As we withdraw from that state of Oneness we disconnect from its influence and things can get very unpleasant and very stupid. Archontic consciousness is at the extreme end of that. It has so disconnected from the influence of Oneness that it has become an inversion of unity and love, an inversion of everything, an inversion of life itself. Evil is appropriately live written backwards. Archontic consciousness is obsessed with death, an inversion of life,

and so its manifestations in Satanism are obsessed with death. They use inverted symbols in their rituals such as the inverted pentagram and cross. Sabbatians as Archontic consciousness incarnate invert Judaism and every other religion and culture they infiltrate. They seek disunity and chaos and they fear unity and harmony as they fear love like garlic to a vampire. As a result the Cult, Archons incarnate, act with such evil, psychopathy and lack of empathy and compassion disconnected as they are from the source of love. How could Bill Gates and the rest of the Archontic psychopaths do what they have to human society in the 'Covid' era with all the death, suffering and destruction involved and have no emotional consequence for the impact on others? Now you know. Why have Zuckerberg, Brin, Page, Wojcicki and company callously censored information warning about the dangers of the 'vaccine' while thousands have been dying and having severe, sometimes life-changing reactions? Now you know. Why have Tedros, Fauci, Whitty, Vallance and their like around the world been using case and death figures they're aware are fraudulent to justify lockdowns and all the deaths and destroyed lives that have come from that? Now you know. Why did Christian Drosten produce and promote a 'testing' protocol that he knew couldn't test for infectious disease which led to a global human catastrophe. Now you know. The Archontic mind doesn't give a shit ([Fig 17](#)). I personally think that Gates and major Cult insiders are a form of AI cyborg that the Archons want humans to become.

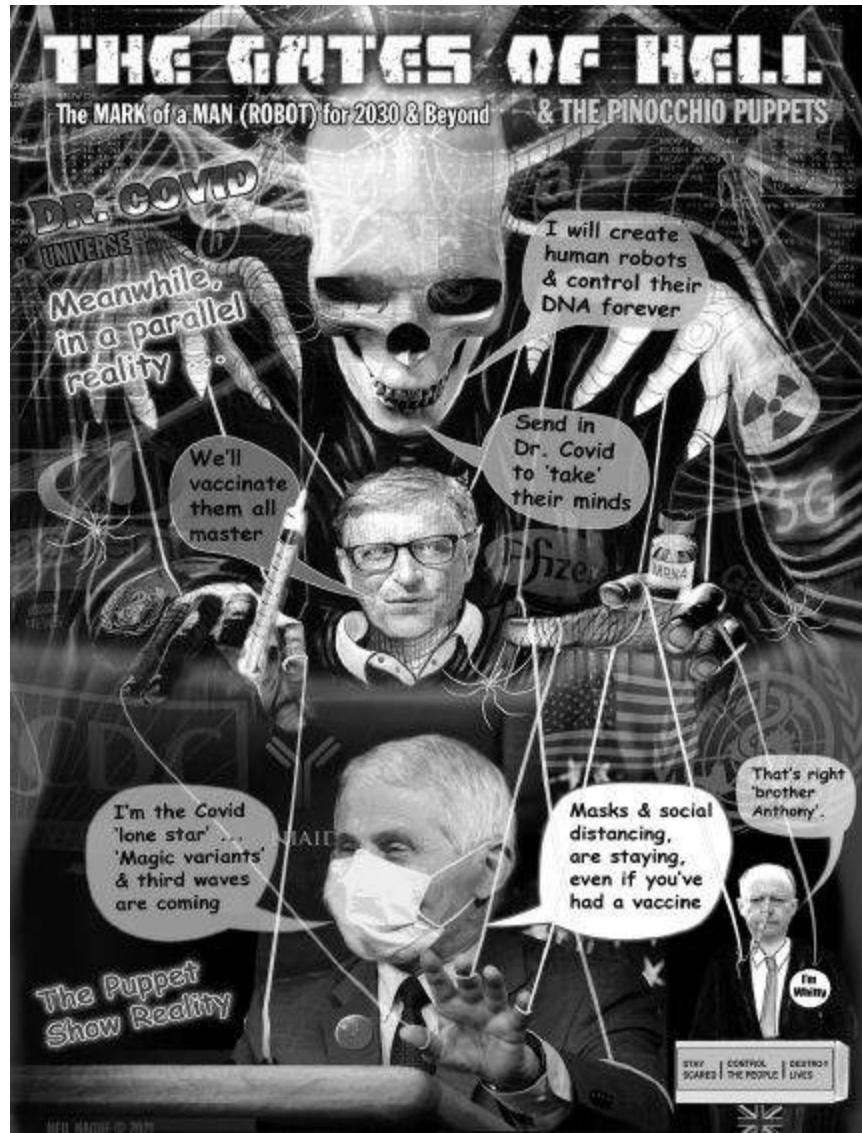


Figure 17: Artist Neil Hague's version of the 'Covid' hierarchy.

Human batteries

A state of such inversion does have its consequences, however. The level of disconnection from the Source of All means that you withdraw from that source of energetic sustenance and creativity. This means that you have to find your own supply of energetic power and it has – *us*. When the Morpheus character in the first *Matrix* movie held up a battery he spoke a profound truth when he said: ‘The Matrix is a computer-generated dream world built to keep us under control in order to change the human being into one of

these.' The statement was true in all respects. We do live in a technologically-generated virtual reality simulation (more very shortly) and we have been manipulated to be an energy source for Archontic consciousness. The Disney-Pixar animated movie *Monsters, Inc.* in 2001 symbolised the dynamic when monsters in their world had no energy source and they would enter the human world to terrify children in their beds, catch the child's scream, terror (low-vibrational frequencies), and take that energy back to power the monster world. The lead character you might remember was a single giant eye and the symbolism of the Cult's all-seeing eye was obvious. Every thought and emotion is broadcast as a frequency unique to that thought and emotion. Feelings of love and joy, empathy and compassion, are high, quick, frequencies while fear, depression, anxiety, suffering and hate are low, slow, dense frequencies. Which kind do you think Archontic consciousness can connect with and absorb? In such a low and dense frequency state there's no way it can connect with the energy of love and joy. Archons can only feed off energy compatible with their own frequency and they and their Cult agents want to delete the human world of love and joy and manipulate the transmission of low vibrational frequencies through low-vibrational human mental and emotional states. *We are their energy source.* Wars are energetic banquets to the Archons – a world war even more so – and think how much low-frequency mental and emotional energy has been generated from the consequences for humanity of the 'Covid' hoax orchestrated by Archons incarnate like Gates.

The ancient practice of human sacrifice 'to the gods', continued in secret today by the Cult, is based on the same principle. 'The gods' are Archontic consciousness in different forms and the sacrifice is induced into a state of intense terror to generate the energy the Archontic frequency can absorb. Incarnate Archons in the ritual drink the blood which contains an adrenaline they crave which floods into the bloodstream when people are terrorised. Most of the sacrifices, ancient and modern, are children and the theme of 'sacrificing young virgins to the gods' is just code for children. They

have a particular pre-puberty energy that Archons want more than anything and the energy of the young in general is their target. The California Department of Education wants students to chant the names of Aztec gods (Archontic gods) once worshipped in human sacrifice rituals in a curriculum designed to encourage them to ‘challenge racist, bigoted, discriminatory, imperialist/colonial beliefs’, join ‘social movements that struggle for social justice’, and ‘build new possibilities for a post-racist, post-systemic racism society’. It’s the usual Woke crap that inverts racism and calls it anti-racism. In this case solidarity with ‘indigenous tribes’ is being used as an excuse to chant the names of ‘gods’ to which people were sacrificed (and still are in secret). What an example of Woke’s inability to see beyond black and white, us and them, They condemn the colonisation of these tribal cultures by Europeans (quite right), but those cultures sacrificing people including children to their ‘gods’, and mass murdering untold numbers as the Aztecs did, is just fine. One chant is to the Aztec god Tezcatlipoca who had a man sacrificed to him in the 5th month of the Aztec calendar. His heart was cut out and he was eaten. Oh, that’s okay then. Come on children … after three … Other sacrificial ‘gods’ for the young to chant their allegiance include Quetzalcoatl, Huitzilopochtli and Xipe Totec. The curriculum says that ‘chants, affirmations, and energizers can be used to bring the class together, build unity around ethnic studies principles and values, and to reinvigorate the class following a lesson that may be emotionally taxing or even when student engagement may appear to be low’. Well, that’s the cover story, anyway. Chanting and mantras are the repetition of a particular frequency generated from the vocal cords and chanting the names of these Archontic ‘gods’ tunes you into their frequency. That is the last thing you want when it allows for energetic synchronisation, attachment and perceptual influence. Initiates chant the names of their ‘Gods’ in their rituals for this very reason.

Vampires of the Woke

Paedophilia is another way that Archons absorb the energy of children. Paedophiles possessed by Archontic consciousness are used as the conduit during sexual abuse for discarnate Archons to vampire the energy of the young they desire so much. Stupendous numbers of children disappear every year never to be seen again although you would never know from the media. Imagine how much low-vibrational energy has been generated by children during the 'Covid' hoax when so many have become depressed and psychologically destroyed to the point of killing themselves.

Shocking numbers of children are now taken by the state from loving parents to be handed to others. I can tell you from long experience of researching this since 1996 that many end up with paedophiles and assets of the Cult through corrupt and Cult-owned social services which in the reframing era has hired many psychopaths and emotionless automatons to do the job. Children are even stolen to order using spurious reasons to take them by the corrupt and secret (because they're corrupt) 'family courts'. I have written in detail in other books, starting with *The Biggest Secret* in 1997, about the ubiquitous connections between the political, corporate, government, intelligence and military elites (Cult operatives) and Satanism and paedophilia. If you go deep enough both networks have an interlocking leadership. The Woke mentality has been developed by the Cult for many reasons: To promote almost every aspect of its agenda; to hijack the traditional political left and turn it fascist; to divide and rule; and to target agenda pushbackers. But there are other reasons which relate to what I am describing here. How many happy and joyful Wokers do you ever see especially at the extreme end? They are a mental and psychological mess consumed by emotional stress and constantly emotionally cocked for the next explosion of indignation at someone referring to a female as a female. They are walking, talking, batteries as Morpheus might say emitting frequencies which both enslave them in low-vibrational bubbles of perceptual limitation and feed the Archons. Add to this the hatred claimed to be love; fascism claimed to 'anti-fascism', racism claimed to be 'anti-racism';

exclusion claimed to inclusion; and the abuse-filled Internet trolling. You have a purpose-built Archontic energy system with not a wind turbine in sight and all founded on Archontic *inversion*. We have whole generations now manipulated to serve the Archons with their actions and energy. They will be doing so their entire adult lives unless they snap out of their Archon-induced trance. Is it really a surprise that Cult billionaires and corporations put so much money their way? Where is the energy of joy and laughter, including laughing at yourself which is confirmation of your own emotional security? Mark Twain said: 'The human race has one really effective weapon, and that is laughter.' We must use it all the time. Woke has destroyed comedy because it has no humour, no joy, sense of irony, or self-deprecation. Its energy is dense and intense. *Mmmmm*, lunch says the Archontic frequency. Rudolf Steiner (1861-1925) was the Austrian philosopher and famous esoteric thinker who established Waldorf education or Steiner schools to treat children like unique expressions of consciousness and not minds to be programmed with the perceptions determined by authority. I'd been writing about this energy vampiring for decades when I was sent in 2016 a quote by Steiner. He was spot on:

There are beings in the spiritual realms for whom anxiety and fear emanating from human beings offer welcome food. When humans have no anxiety and fear, then these creatures starve. If fear and anxiety radiates from people and they break out in panic, then these creatures find welcome nutrition and they become more and more powerful. These beings are hostile towards humanity. Everything that feeds on negative feelings, on anxiety, fear and superstition, despair or doubt, are in reality hostile forces in super-sensible worlds, launching cruel attacks on human beings, while they are being fed ... These are exactly the feelings that belong to contemporary culture and materialism; because it estranges people from the spiritual world, it is especially suited to evoke hopelessness and fear of the unknown in people, thereby calling up the above mentioned hostile forces against them.

Pause for a moment from this perspective and reflect on what has happened in the world since the start of 2020. Not only will pennies drop, but billion dollar bills. We see the same theme from Don Juan Matus, a Yaqui Indian shaman in Mexico and the information source for Peruvian-born writer, Carlos Castaneda, who wrote a series of

books from the 1960s to 1990s. Don Juan described the force manipulating human society and his name for the Archons was the predator:

We have a predator that came from the depths of the cosmos and took over the rule of our lives. Human beings are its prisoners. The predator is our lord and master. It has rendered us docile, helpless. If we want to protest, it suppresses our protest. If we want to act independently, it demands that we don't do so ... indeed we are held prisoner!

They took us over because we are food to them, and they squeeze us mercilessly because we are their sustenance. Just as we rear chickens in coops, the predators rear us in human coops, humaneros. Therefore, their food is always available to them.

Different cultures, different eras, same recurring theme.

The 'ennoia' dilemma

Nag Hammadi Gnostic manuscripts say that Archon consciousness has no 'ennoia'. This is directly translated as 'intentionality', but I'll use the term 'creative imagination'. The *All That Is* in awareness of itself is the source of all creativity – all possibility – and the more disconnected you are from that source the more you are subsequently denied 'creative imagination'. Given that Archon consciousness is almost entirely disconnected it severely lacks creativity and has to rely on far more mechanical processes of thought and exploit the creative potential of those that do have 'ennoia'. You can see cases of this throughout human society. Archon consciousness almost entirely dominates the global banking system and if we study how that system works you will appreciate what I mean. Banks manifest 'money' out of nothing by issuing lines of 'credit' which is 'money' that has never, does not, and will never exist except in theory. It's a confidence trick. If you think 'credit' figures-on-a-screen 'money' is worth anything you accept it as payment. If you don't then the whole system collapses through lack of confidence in the value of that 'money'. Archontic bankers with no 'ennoia' are 'lending' 'money' that doesn't exist to humans that *do* have creativity – those that have the inspired ideas and create businesses and products. Archon banking feeds off human creativity

which it controls through ‘money’ creation and debt. Humans have the creativity and Archons exploit that for their own benefit and control while having none themselves. Archon Internet platforms like Facebook claim joint copyright of everything that creative users post and while Archontic minds like Zuckerberg may officially head that company it will be human creatives on the staff that provide the creative inspiration. When you have limitless ‘money’ you can then buy other companies established by creative humans. Witness the acquisition record of Facebook, Google and their like. Survey the Archon-controlled music industry and you see non-creative dark suit executives making their fortune from the human creativity of their artists. The cases are endless. Research the history of people like Gates and Zuckerberg and how their empires were built on exploiting the creativity of others. Archon minds cannot create out of nothing, but they are skilled (because they have to be) in what Gnostic texts call ‘countermimicry’. They can imitate, but not innovate. Sabbatians trawl the creativity of others through backdoors they install in computer systems through their cybersecurity systems. Archon-controlled China is globally infamous for stealing intellectual property and I remember how Hong Kong, now part of China, became notorious for making counterfeit copies of the creativity of others – ‘countermimicry’. With the now pervasive and all-seeing surveillance systems able to infiltrate any computer you can appreciate the potential for Archons to vampire the creativity of humans. Author John Lamb Lash wrote in his book about the Nag Hammadi texts, *Not In His Image*:

Although they cannot originate anything, because they lack the divine factor of ennoia (intentionality), Archons can imitate with a vengeance. Their expertise is simulation (HAL, virtual reality). The Demiurge [Yaldabaoth] fashions a heaven world copied from the fractal patterns [of the original] ... His construction is celestial kitsch, like the fake Italianate villa of a Mafia don complete with militant angels to guard every portal.

This brings us to something that I have been speaking about since the turn of the millennium. Our reality is a simulation; a virtual reality that we think is real. No, I’m not kidding.

Human reality? Well, virtually

I had pondered for years about whether our reality is ‘real’ or some kind of construct. I remembered being immensely affected on a visit as a small child in the late 1950s to the then newly-opened Planetarium on the Marylebone Road in London which is now closed and part of the adjacent Madame Tussauds wax museum. It was in the middle of the day, but when the lights went out there was the night sky projected in the Planetarium’s domed ceiling and it appeared to be so real. The experience never left me and I didn’t know why until around the turn of the millennium when I became certain that our ‘night sky’ and entire reality is a projection, a virtual reality, akin to the illusory world portrayed in the *Matrix* movies. I looked at the sky one day in this period and it appeared to me like the domed roof of the Planetarium. The release of the first *Matrix* movie in 1999 also provided a synchronistic and perfect visual representation of where my mind had been going for a long time. I hadn’t come across the Gnostic Nag Hammadi texts then. When I did years later the correlation was once again astounding. As I read Gnostic accounts from 1,600 years and more earlier it was clear that they were describing the same simulation phenomenon. They tell how the Yaldabaoth ‘Demiurge’ and Archons created a ‘bad copy’ of original reality to rule over all that were captured by its illusions and the body was a prison to trap consciousness in the ‘bad copy’ fake reality. Read how Gnostics describe the ‘bad copy’ and update that to current times and they are referring to what we would call today a virtual reality simulation.

Author John Lamb Lash said ‘the Demiurge fashions a heaven world copied from the fractal patterns’ of the original through expertise in ‘HAL’ or virtual reality simulation. Fractal patterns are part of the energetic information construct of our reality, a sort of blueprint. If these patterns were copied in computer terms it would indeed give you a copy of a ‘natural’ reality in a non-natural frequency and digital form. The principle is the same as making a copy of a website. The original website still exists, but now you can change the copy version to make it whatever you like and it can

become very different to the original website. Archons have done this with our reality, a *synthetic* copy of prime reality that still exists beyond the frequency walls of the simulation. Trapped within the illusions of this synthetic Matrix, however, were and are human consciousness and other expressions of prime reality and this is why the Archons via the Cult are seeking to make the human body synthetic and give us synthetic AI minds to complete the job of turning the entire reality synthetic including what we perceive to be the natural world. To quote Kurzweil: ‘Nanobots will infuse all the matter around us with information. Rocks, trees, everything will become these intelligent creatures.’ Yes, *synthetic* ‘creatures’ just as ‘Covid’ and other genetically-manipulating ‘vaccines’ are designed to make the human body synthetic. From this perspective it is obvious why Archons and their Cult are so desperate to infuse synthetic material into every human with their ‘Covid’ scam.

Let there be (electromagnetic) light

Yaldabaoth, the force that created the simulation, or Matrix, makes sense of the Gnostic reference to ‘The Great Architect’ and its use by Cult Freemasonry as the name of its deity. The designer of the Matrix in the movies is called ‘The Architect’ and that trilogy is jam-packed with symbolism relating to these subjects. I have contended for years that the angry Old Testament God (Yaldabaoth) is the ‘God’ being symbolically ‘quoted’ in the opening of Genesis as ‘creating the world’. This is not the creation of prime reality – it’s the creation of the *simulation*. The Genesis ‘God’ says: ‘Let there be Light: and there was light.’ But what is this ‘Light’? I have said for decades that the speed of light (186,000 miles per second) is not the fastest speed possible as claimed by mainstream science and is in fact the frequency walls or outer limits of the Matrix. You can’t have a fastest or slowest anything within all possibility when everything is possible. The human body is encoded to operate within the speed of light or *within the simulation* and thus we see only the tiny frequency band of visible *light*. Near-death experiencers who perceive reality outside the body during temporary ‘death’ describe a very different

form of light and this is supported by the Nag Hammadi texts. Prime reality beyond the simulation ('Upper Aeons' to the Gnostics) is described as a realm of incredible beauty, bliss, love and harmony – a realm of 'watery light' that is so powerful 'there are no shadows'. Our false reality of Archon control, which Gnostics call the 'Lower Aeons', is depicted as a realm with a different kind of 'light' and described in terms of chaos, 'Hell', 'the Abyss' and 'Outer Darkness', where trapped souls are tormented and manipulated by demons (relate that to the 'Covid' hoax alone). The watery light theme can be found in near-death accounts and it is not the same as *simulation* 'light' which is electromagnetic or radiation light within the speed of light – the 'Lower Aeons'. Simulation 'light' is the 'luminous fire' associated by Gnostics with the Archons. The Bible refers to Yaldabaoth as 'that old serpent, called the Devil, and Satan, which deceiveth the whole world' (Revelation 12:9). I think that making a simulated copy of prime reality ('countermimicry') and changing it dramatically while all the time manipulating humanity to believe it to be real could probably meet the criteria of deceiving the whole world. Then we come to the Cult god Lucifer – the *Light Bringer*. Lucifer is symbolic of Yaldabaoth, the bringer of radiation light that forms the bad copy simulation within the speed of light. 'He' is symbolised by the lighted torch held by the Statue of Liberty and in the name 'Illuminati'. Sabbatian-Frankism declares that Lucifer is the true god and Lucifer is the real god of Freemasonry honoured as their 'Great or Grand Architect of the Universe' (simulation).

I would emphasise, too, the way Archontic technologically-generated luminous fire of radiation has deluged our environment since I was a kid in the 1950s and changed the nature of The Field with which we constantly interact. Through that interaction technological radiation is changing us. The Smart Grid is designed to operate with immense levels of communication power with 5G expanding across the world and 6G, 7G, in the process of development. Radiation is the simulation and the Archontic manipulation system. Why wouldn't the Archon Cult wish to unleash radiation upon us to an ever-greater extreme to form

Kurzweil's 'cloud'? The plan for a synthetic human is related to the need to cope with levels of radiation beyond even anything we've seen so far. Biological humans would not survive the scale of radiation they have in their script. The Smart Grid is a technological sub-reality within the technological simulation to further disconnect five-sense perception from expanded consciousness. It's a technological prison of the mind.

Infusing the 'spirit of darkness'

A recurring theme in religion and native cultures is the manipulation of human genetics by a non-human force and most famously recorded as the biblical 'sons of god' (the gods plural in the original) who interbred with the daughters of men. The Nag Hammadi *Apocryphon of John* tells the same story this way:

He [Yaldabaoth] sent his angels [Archons/demons] to the daughters of men, that they might take some of them for themselves and raise offspring for their enjoyment. And at first they did not succeed. When they had no success, they gathered together again and they made a plan together ... And the angels changed themselves in their likeness into the likeness of their mates, filling them with the spirit of darkness, which they had mixed for them, and with evil ... And they took women and begot children out of the darkness according to the likeness of their spirit.

Possession when a discarnate entity takes over a human body is an age-old theme and continues today. It's very real and I've seen it. Satanic and secret society rituals can create an energetic environment in which entities can attach to initiates and I've heard many stories of how people have changed their personality after being initiated even into lower levels of the Freemasons. I have been inside three Masonic temples, one at a public open day and two by just walking in when there was no one around to stop me. They were in Ryde, the town where I live, Birmingham, England, when I was with a group, and Boston, Massachusetts. They all felt the same energetically – dark, dense, low-vibrational and sinister. Demonic attachment can happen while the initiate has no idea what is going on. To them it's just a ritual to get in the Masons and do a bit of good

business. In the far more extreme rituals of Satanism human possession is even more powerful and they are designed to make possession possible. The hierarchy of the Cult is dictated by the power and perceived status of the possessing Archon. In this way the Archon hierarchy becomes the Cult hierarchy. Once the entity has attached it can influence perception and behaviour and if it attaches to the extreme then so much of its energy (information) infuses into the body information field that the hologram starts to reflect the nature of the possessing entity. This is the *Exorcist* movie type of possession when facial features change and it's known as shapeshifting. Islam's Jinn are said to be invisible tricksters who change shape, 'whisper', confuse and take human form. These are all traits of the Archons and other versions of the same phenomenon. Extreme possession could certainty infuse the 'spirit of darkness' into a partner during sex as the Nag Hammadi texts appear to describe. Such an infusion can change genetics which is also energetic information. Human genetics is information and the 'spirit of darkness' is information. Mix one with the other and change must happen. Islam has the concept of a 'Jinn baby' through possession of the mother and by Jinn taking human form. There are many ways that human genetics can be changed and remember that Archons have been aware all along of advanced techniques to do this. What is being done in human society today – and far more – was known about by Archons at the time of the 'fallen ones' and their other versions described in religions and cultures.

Archons and their human-world Cult are obsessed with genetics as we see today and they know this dictates how information is processed into perceived reality during a human life. They needed to produce a human form that would decode the simulation and this is symbolically known as 'Adam and Eve' who left the 'garden' (prime reality) and 'fell' into Matrix reality. The simulation is not a 'physical' construct (there is no 'physical'); it is a source of information. Think Wi-Fi again. The simulation is an energetic field encoded with information and body-brain systems are designed to decode that information encoded in wave or frequency form which

is transmitted to the brain as electrical signals. These are decoded by the brain to construct our sense of reality – an illusory ‘physical’ world that only exists in the brain or the mind. Virtual reality games mimic this process using the same sensory decoding system. Information is fed to the senses to decode a virtual reality that can appear so real, but isn’t (Figs 18 and 19). Some scientists believe – and I agree with them – that what we perceive as ‘physical’ reality only exists when we are looking or observing. The act of perception or focus triggers the decoding systems which turn waveform information into holographic reality. When we are not observing something our reality reverts from a holographic state to a waveform state. This relates to the same principle as a falling tree not making a noise unless someone is there to hear it or decode it. The concept makes sense from the simulation perspective. A computer is not decoding all the information in a Wi-Fi field all the time and only decodes or brings into reality on the screen that part of Wi-Fi that it’s decoding – focusing upon – at that moment.



Figure 18: Virtual reality technology ‘hacks’ into the body’s five-sense decoding system.



Figure 19: The result can be experienced as very ‘real’.

Interestingly, Professor Donald Hoffman at the Department of Cognitive Sciences at the University of California, Irvine, says that our experienced reality is like a computer interface that shows us only the level with which we interact while hiding all that exists beyond it: ‘Evolution shaped us with a user interface that hides the truth. Nothing that we see is the truth – the very language of space and time and objects is the wrong language to describe reality.’ He is correct in what he says on so many levels. Space and time are not a universal reality. They are a phenomenon of decoded *simulation* reality as part of the process of enslaving our sense of reality. Near-death experiencers report again and again how space and time did not exist as we perceive them once they were free of the body – body decoding systems. You can appreciate from this why Archons and their Cult are so desperate to entrap human attention in the five senses where we are in the Matrix and of the Matrix. Opening your mind to expanded states of awareness takes you beyond the information confines of the simulation and you become aware of knowledge and insights denied to you before. This is what we call ‘awakening’ – *awakening from the Matrix* – and in the final chapter I will relate this to current events.

Where are the ‘aliens’?

A simulation would explain the so-called ‘Fermi Paradox’ named after Italian physicist Enrico Fermi (1901-1954) who created the first nuclear reactor. He considered the question of why there is such a lack of extraterrestrial activity when there are so many stars and planets in an apparently vast universe; but what if the night sky that we see, or think we do, is a simulated projection as I say? If you control the simulation and your aim is to hold humanity fast in essential ignorance would you want other forms of life including advanced life coming and going sharing information with humanity? Or would you want them to believe they were isolated and apparently alone? Themes of human isolation and apartness are common whether they be the perception of a lifeless universe or the fascist isolation laws of the ‘Covid’ era. Paradoxically the very

existence of a simulation means that we are not alone when some force had to construct it. My view is that experiences that people have reported all over the world for centuries with Reptilians and Grey entities are Archon phenomena as Nag Hammadi texts describe; and that benevolent 'alien' interactions are non-human groups that come in and out of the simulation by overcoming Archon attempts to keep them out. It should be highlighted, too, that Reptilians and Greys are obsessed with *genetics* and *technology* as related by cultural accounts and those who say they have been abducted by them. Technology is their way of overcoming some of the limitations in their creative potential and our technology-driven and controlled human society of today is *archetypical* Archon-Reptilian-Grey modus operandi. Technocracy is really *Archontocracy*. The Universe does not have to be as big as it appears with a simulation. There is no space or distance only information decoded into holographic reality. What we call 'space' is only the absence of holographic 'objects' and that 'space' is The Field of energetic information which connects everything into a single whole. The same applies with the artificially-generated information field of the simulation. The Universe is not big or small as a physical reality. It is decoded information, that's all, and its perceived size is decided by the way the simulation is encoded to make it appear. The entire night sky as we perceive it only exists in our brain and so where are those 'millions of light years'? The 'stars' on the ceiling of the Planetarium looked a vast distance away.

There's another point to mention about 'aliens'. I have been highlighting since the 1990s the plan to stage a fake 'alien invasion' to justify the centralisation of global power and a world military. Nazi scientist Werner von Braun, who was taken to America by Operation Paperclip after World War Two to help found NASA, told his American assistant Dr Carol Rosin about the Cult agenda when he knew he was dying in 1977. Rosin said that he told her about a sequence that would lead to total human control by a one-world government. This included threats from terrorism, rogue nations, meteors and asteroids before finally an 'alien invasion'. All of these

things, von Braun said, would be bogus and what I would refer to as a No-Problem-Reaction-Solution. Keep this in mind when ‘the aliens are coming’ is the new mantra. The aliens are not coming – they are *already here* and they have infiltrated human society while looking human. French-Canadian investigative journalist Serge Monast said in 1994 that he had uncovered a NASA/military operation called Project Blue Beam which fits with what Werner von Braun predicted. Monast died of a ‘heart attack’ in 1996 the day after he was arrested and spent a night in prison. He was 51. He said Blue Beam was a plan to stage an alien invasion that would include religious figures beamed holographically into the sky as part of a global manipulation to usher in a ‘new age’ of worshipping what I would say is the Cult ‘god’ Yaldabaoth in a one-world religion. Fake holographic asteroids are also said to be part of the plan which again syncs with von Braun. How could you stage an illusory threat from asteroids unless they were holographic inserts? This is pretty straightforward given the advanced technology outside the public arena and the fact that our ‘physical’ reality is holographic anyway. Information fields would be projected and we would decode them into the illusion of a ‘physical’ asteroid. If they can sell a global ‘pandemic’ with a ‘virus’ that doesn’t exist what will humans not believe if government and media tell them?

All this is particularly relevant as I write with the Pentagon planning to release in June, 2021, information about ‘UFO sightings’. I have been following the UFO story since the early 1990s and the common theme throughout has been government and military denials and cover up. More recently, however, the Pentagon has suddenly become more talkative and apparently open with Air Force pilot radar images released of unexplained craft moving and changing direction at speeds well beyond anything believed possible with human technology. Then, in March, 2021, former Director of National Intelligence John Ratcliffe said a Pentagon report months later in June would reveal a great deal of information about UFO sightings unknown to the public. He said the report would have ‘massive implications’. The order to do this was included bizarrely

in a \$2.3 trillion ‘coronavirus’ relief and government funding bill passed by the Trump administration at the end of 2020. I would add some serious notes of caution here. I have been pointing out since the 1990s that the US military and intelligence networks have long had craft – ‘flying saucers’ or anti-gravity craft – which any observer would take to be extraterrestrial in origin. Keeping this knowledge from the public allows craft flown by *humans* to be perceived as alien visitations. I am not saying that ‘aliens’ do not exist. I would be the last one to say that, but we have to be streetwise here. President Ronald Reagan told the UN General Assembly in 1987: ‘I occasionally think how quickly our differences worldwide would vanish if we were facing an alien threat from outside this world.’ That’s the idea. Unite against a common ‘enemy’ with a common purpose behind your ‘saviour force’ (the Cult) as this age-old technique of mass manipulation goes global.

Science moves this way ...

I could find only one other person who was discussing the simulation hypothesis publicly when I concluded it was real. This was Nick Bostrom, a Swedish-born philosopher at the University of Oxford, who has explored for many years the possibility that human reality is a computer simulation although his version and mine are not the same. Today the simulation and holographic reality hypothesis have increasingly entered the scientific mainstream. Well, the more open-minded mainstream, that is. Here are a few of the ever-gathering examples. American nuclear physicist Silas Beane led a team of physicists at the University of Bonn in Germany pursuing the question of whether we live in a simulation. They concluded that we probably do and it was likely based on a lattice of cubes. They found that cosmic rays align with that specific pattern. The team highlighted the Greisen-Zatsepin-Kuzmin (GZK) limit which refers to cosmic ray particle interaction with cosmic background radiation that creates an apparent boundary for cosmic ray particles. They say in a paper entitled ‘Constraints on the Universe as a Numerical Simulation’ that this ‘pattern of constraint’ is exactly what you

would find with a computer simulation. They also made the point that a simulation would create its own ‘laws of physics’ that would limit possibility. I’ve been making the same point for decades that the *perceived* laws of physics relate only to this reality, or what I would later call the simulation. When designers write codes to create computer and virtual reality games they are the equivalent of the laws of physics for that game. Players interact within the limitations laid out by the coding. In the same way those who wrote the codes for the simulation decided the laws of physics that would apply. These can be overridden by expanded states of consciousness, but not by those enslaved in only five-sense awareness where simulation codes rule. Overriding the codes is what people call ‘miracles’. They are not. They are bypassing the encoded limits of the simulation. A population caught in simulation perception would have no idea that this was their plight. As the Bonn paper said: ‘Like a prisoner in a pitch-black cell we would not be able to see the “walls” of our prison.’ That’s true if people remain mesmerised by the five senses. Open to expanded awareness and those walls become very clear. The main one is the speed of light.

American theoretical physicist James Gates is another who has explored the simulation question and found considerable evidence to support the idea. Gates was Professor of Physics at the University of Maryland, Director of The Center for String and Particle Theory, and on Barack Obama’s Council of Advisors on Science and Technology. He and his team found *computer codes* of digital data embedded in the fabric of our reality. They relate to on-off electrical charges of 1 and 0 in the binary system used by computers. ‘We have no idea what they are doing there’, Gates said. They found within the energetic fabric mathematical sequences known as error-correcting codes or block codes that ‘reboot’ data to its original state or ‘default settings’ when something knocks it out of sync. Gates was asked if he had found a set of equations embedded in our reality indistinguishable from those that drive search engines and browsers and he said: ‘That is correct.’ Rich Terrile, director of the Centre for Evolutionary Computation and Automated Design at NASA’s Jet

Propulsion Laboratory, has said publicly that he believes the Universe is a digital hologram that must have been created by a form of intelligence. I agree with that in every way. Waveform information is delivered electrically by the senses to the brain which constructs a *digital* holographic reality that we call the ‘world’. This digital level of reality can be read by the esoteric art of numerology. Digital holograms are at the cutting edge of holographics today. We have digital technology everywhere designed to access and manipulate our digital level of perceived reality. Synthetic mRNA in ‘Covid vaccines’ has a digital component to manipulate the body’s digital ‘operating system’.

Reality is numbers

How many know that our reality can be broken down to numbers and codes that are the same as computer games? Max Tegmark, a physicist at the Massachusetts Institute of Technology (MIT), is the author of *Our Mathematical Universe* in which he lays out how reality can be entirely described by numbers and maths in the way that a video game is encoded with the ‘physics’ of computer games. Our world and computer virtual reality are essentially the same.

Tegmark imagines the perceptions of characters in an advanced computer game when the graphics are so good they don’t know they are in a game. They think they can bump into real objects (electromagnetic resistance in our reality), fall in love and feel emotions like excitement. When they began to study the apparently ‘physical world’ of the video game they would realise that everything was made of pixels (which have been found in our energetic reality as must be the case when on one level our world is digital). What computer game characters thought was physical ‘stuff’, Tegmark said, could actually be broken down into numbers:

And we’re exactly in this situation in our world. We look around and it doesn’t seem that mathematical at all, but everything we see is made out of elementary particles like quarks and electrons. And what properties does an electron have? Does it have a smell or a colour or a texture? No! ... We physicists have come up with geeky names for [Electron] properties, like

electric charge, or spin, or lepton number, but the electron doesn't care what we call it, the properties are just numbers.

This is the illusory reality Gnostics were describing. This is the simulation. The A, C, G, and T codes of DNA have a binary value – A and C = 0 while G and T = 1. This has to be when the simulation is digital and the body must be digital to interact with it. Recurring mathematical sequences are encoded throughout reality and the body. They include the Fibonacci sequence in which the two previous numbers are added to get the next one, as in ... 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, etc. The sequence is encoded in the human face and body, proportions of animals, DNA, seed heads, pine cones, trees, shells, spiral galaxies, hurricanes and the number of petals in a flower. The list goes on and on. There are fractal patterns – a 'never-ending pattern that is infinitely complex and self-similar across all scales in the as above, so below, principle of holograms. These and other famous recurring geometrical and mathematical sequences such as Phi, Pi, Golden Mean, Golden Ratio and Golden Section are *computer codes* of the simulation. I had to laugh and give my head a shake the day I finished this book and it went into the production stage. I was sent an article in *Scientific American* published in April, 2021, with the headline 'Confirmed! We Live in a Simulation'. Two decades after I first said our reality is a simulation and the speed of light is its outer limit the article suggested that we do live in a simulation and that the speed of light is its outer limit. I left school at 15 and never passed a major exam in my life while the writer was up to his eyes in qualifications. As I will explain in the final chapter *knowing* is far better than thinking and they come from very different sources. The article rightly connected the speed of light to the processing speed of the 'Matrix' and said what has been in my books all this time ... 'If we are in a simulation, as it appears, then space is an abstract property written in code. It is not real'. No it's not and if we live in a simulation something created it and it wasn't *us*. 'That David Icke says we are manipulated by aliens' – he's crackers.'

Wow ...

The reality that humanity thinks is so real is an illusion. Politicians, governments, scientists, doctors, academics, law enforcement, media, school and university curriculums, on and on, are all founded on a world that *does not exist* except as a simulated prison cell. Is it such a stretch to accept that 'Covid' doesn't exist when our entire 'physical' reality doesn't exist? Revealed here is the knowledge kept under raps in the Cult networks of compartmentalised secrecy to control humanity's sense of reality by inducing the population to believe in a reality that's not real. If it wasn't so tragic in its experiential consequences the whole thing would be hysterically funny. None of this is new to Renegade Minds. Ancient Greek philosopher Plato (about 428 to about 347BC) was a major influence on Gnostic belief and he described the human plight thousands of years ago with his Allegory of the Cave. He told the symbolic story of prisoners living in a cave who had never been outside. They were chained and could only see one wall of the cave while behind them was a fire that they could not see. Figures walked past the fire casting shadows on the prisoners' wall and those moving shadows became their sense of reality. Some prisoners began to study the shadows and were considered experts on them (today's academics and scientists), but what they studied was only an illusion (today's academics and scientists). A prisoner escaped from the cave and saw reality as it really is. When he returned to report this revelation they didn't believe him, called him mad and threatened to kill him if he tried to set them free. Plato's tale is not only a brilliant analogy of the human plight and our illusory reality. It describes, too, the dynamics of the 'Covid' hoax. I have only skimmed the surface of these subjects here. The aim of this book is to crisply connect all essential dots to put what is happening today into its true context. All subject areas and their connections in this chapter are covered in great evidential detail in *Everything You Need To Know, But Have Never Been Told* and *The Answer*.

They say that bewildered people 'can't see the forest for the trees'. Humanity, however, can't see the forest for the *twigs*. The five senses

see only twigs while Renegade Minds can see the forest and it's the forest where the answers lie with the connections that reveals. Breaking free of perceptual programming so the forest can be seen is the way we turn all this around. Not breaking free is how humanity got into this mess. The situation may seem hopeless, but I promise you it's not. We are a perceptual heartbeat from paradise if only we knew.

CHAPTER TWELVE

Escaping Wetiko

Life is simply a vacation from the infinite

Dean Cavanagh

Renegade Minds weave the web of life and events and see common themes in the apparently random. They are always there if you look for them and their pursuit is aided by incredible synchronicity that comes when your mind is open rather than mesmerised by what it thinks it can see.

Infinite awareness is infinite possibility and the more of infinite possibility that we access the more becomes infinitely possible. That may be stating the apparently obvious, but it is a devastatingly-powerful fact that can set us free. We are a point of attention within an infinity of consciousness. The question is how much of that infinity do we choose to access? How much knowledge, insight, awareness, wisdom, do we want to connect with and explore? If your focus is only in the five senses you will be influenced by a fraction of infinite awareness. I mean a range so tiny that it gives new meaning to infinitesimal. Limitation of self-identity and a sense of the possible limit accordingly your range of consciousness. We are what we think we are. Life is what we think it is. The dream is the dreamer and the dreamer is the dream. Buddhist philosophy puts it this way: 'As a thing is viewed, so it appears.' Most humans live in the realm of touch, taste, see, hear, and smell and that's the limit of their sense of the possible and sense of self. Many will follow a religion and speak of a God in his heaven, but their lives are still

dominated by the five senses in their perceptions and actions. The five senses become the arbiter of everything. When that happens all except a smear of infinity is sealed away from influence by the rigid, unyielding, reality bubbles that are the five-sense human or Phantom Self. Archon Cult methodology is to isolate consciousness within five-sense reality – the simulation – and then program that consciousness with a sense of self and the world through a deluge of life-long information designed to instil the desired perception that allows global control. Efforts to do this have increased dramatically with identity politics as identity bubbles are squeezed into the minutiae of five-sense detail which disconnect people even more profoundly from the infinite ‘I’.

Five-sense focus and self-identity are like a firewall that limits access to the infinite realms. You only perceive one radio or television station and no other. We’ll take that literally for a moment. Imagine a vast array of stations giving different information and angles on reality, but you only ever listen to one. Here we have the human plight in which the population is overwhelmingly confined to CultFM. This relates only to the frequency range of CultFM and limits perception and insight to that band – limits *possibility* to that band. It means you are connecting with an almost imperceptibly minuscule range of possibility and creative potential within the infinite Field. It’s a world where everything seems apart from everything else and where synchronicity is rare. Synchronicity is defined in the dictionary as ‘the happening by chance of two or more related or similar events at the same time’. Use of ‘by chance’ betrays a complete misunderstanding of reality. Synchronicity is not ‘by chance’. As people open their minds, or ‘awaken’ to use the term, they notice more and more coincidences in their lives, bits of ‘luck’, apparently miraculous happenings that put them in the right place at the right time with the right people. Days become peppered with ‘fancy meeting you here’ and ‘what are the chances of that?’ My entire life has been lived like this and ever more so since my own colossal awakening in 1990 and 91 which transformed my sense of reality. Synchronicity is not ‘by chance’; it is by accessing expanded

realms of possibility which allow expanded potential for manifestation. People broadcasting the same vibe from the same openness of mind tend to be drawn ‘by chance’ to each other through what I call frequency magnetism and it’s not only people. In the last more than 30 years incredible synchronicity has also led me through the Cult maze to information in so many forms and to crucial personal experiences. These ‘coincidences’ have allowed me to put the puzzle pieces together across an enormous array of subjects and situations. Those who have breached the bubble of five-sense reality will know exactly what I mean and this escape from the perceptual prison cell is open to everyone whenever they make that choice. This may appear super-human when compared with the limitations of ‘human’, but it’s really our natural state. ‘Human’ as currently experienced is consciousness in an unnatural state of induced separation from the infinity of the whole. I’ll come to how this transformation into unity can be made when I have described in more detail the force that holds humanity in servitude by denying this access to infinite self.

The Wetiko factor

I have been talking and writing for decades about the way five-sense mind is systematically barricaded from expanded awareness. I have used the analogy of a computer (five-sense mind) and someone at the keyboard (expanded awareness). Interaction between the computer and the operator is symbolic of the interaction between five-sense mind and expanded awareness. The computer directly experiences the Internet and the operator experiences the Internet via the computer which is how it’s supposed to be – the two working as one. Archons seek to control that point where the operator connects with the computer to stop that interaction ([Fig 20](#)). Now the operator is banging the keyboard and clicking the mouse, but the computer is not responding and this happens when the computer is taken over – *possessed* – by an appropriately-named computer ‘virus’. The operator has lost all influence over the computer which goes its own way making decisions under the control of the ‘virus’. I have

just described the dynamic through which the force known to Gnostics as Yaldabaoth and Archons disconnects five-sense mind from expanded awareness to imprison humanity in perceptual servitude.

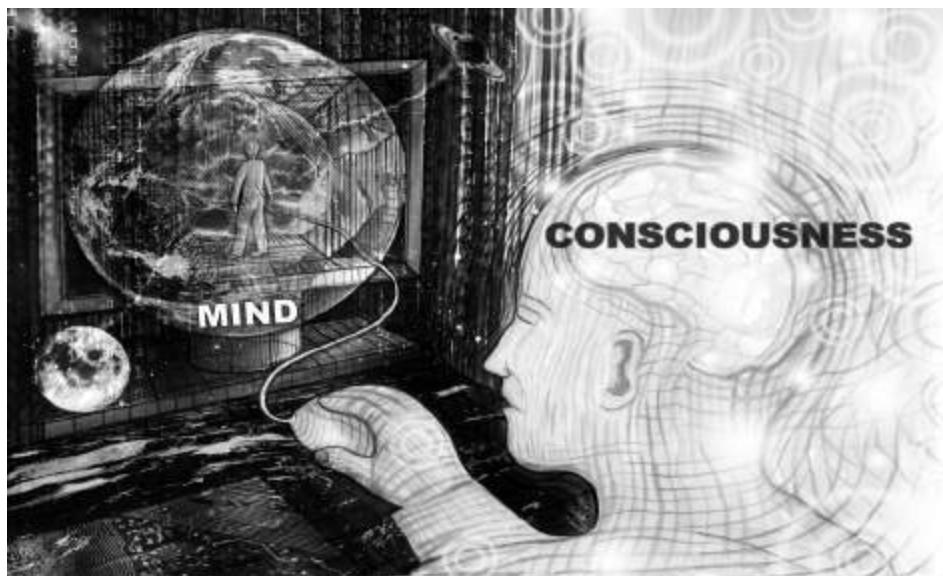


Figure 20: The mind ‘virus’ I have been writing about for decades seeks to isolate five-sense mind (the computer) from the true ‘I’. (Image by Neil Hague).

About a year ago I came across a Native American concept of Wetiko which describes precisely the same phenomenon. Wetiko is the spelling used by the Cree and there are other versions including wintiko and windigo used by other tribal groups. They spell the name with lower case, but I see Wetiko as a proper noun as with Archons and prefer a capital. I first saw an article about Wetiko by writer and researcher Paul Levy which so synced with what I had been writing about the computer/operator disconnection and later the Archons. I then read his book, the fascinating *Dispelling Wetiko, Breaking the Spell of Evil*. The parallels between what I had concluded long before and the Native American concept of Wetiko were so clear and obvious that it was almost funny. For Wetiko see the Gnostic Archons for sure and the Jinn, the Predators, and every other name for a force of evil, inversion and chaos. Wetiko is the Native American name for the force that divides the computer from

the operator ([Fig 21](#)). Indigenous author Jack D. Forbes, a founder of the Native American movement in the 1960s, wrote another book about Wetiko entitled *Columbus And Other Cannibals – The Wetiko Disease of Exploitation, Imperialism, and Terrorism* which I also read. Forbes says that Wetiko refers to an evil person or spirit ‘who terrorizes other creatures by means of terrible acts, including cannibalism’. Zulu shaman Credo Mutwa told me that African accounts tell how cannibalism was brought into the world by the Chitauri ‘gods’ – another manifestation of Wetiko. The distinction between ‘evil person or spirit’ relates to Archons/Wetiko possessing a human or acting as pure consciousness. Wetiko is said to be a sickness of the soul or spirit and a state of being that takes but gives nothing back – the Cult and its operatives perfectly described. Black Hawk, a Native American war leader defending their lands from confiscation, said European invaders had ‘poisoned hearts’ – Wetiko hearts – and that this would spread to native societies. Mention of the heart is very significant as we shall shortly see. Forbes writes: ‘Tragically, the history of the world for the past 2,000 years is, in great part, the story of the epidemiology of the wetiko disease.’ Yes, and much longer. Forbes is correct when he says: ‘The wetikos destroyed Egypt and Babylon and Athens and Rome and Tenochtitlan [capital of the Aztec empire] and perhaps now they will destroy the entire earth.’ Evil, he said, is the number one export of a Wetiko culture – see its globalisation with ‘Covid’. Constant war, mass murder, suffering of all kinds, child abuse, Satanism, torture and human sacrifice are all expressions of Wetiko and the Wetiko possessed. The world is Wetiko made manifest, *but it doesn’t have to be*. There is a way out of this even now.



Figure 21: The mind ‘virus’ is known to Native Americans as ‘Wetiko’. (Image by Neil Hague).

Cult of Wetiko

Wetiko is the Yaldabaoth frequency distortion that seeks to attach to human consciousness and absorb it into its own. Once this connection is made Wetiko can drive the perceptions of the target which they believe to be coming from their own mind. All the horrors of history and today from mass killers to Satanists, paedophiles like Jeffrey Epstein and other psychopaths, are the embodiment of Wetiko and express its state of being in all its grotesqueness. The Cult is Wetiko incarnate, Yaldabaoth incarnate, and it seeks to facilitate Wetiko assimilation of humanity in totality into its distortion by manipulating the population into low frequency states that match its own. Paul Levy writes: ‘Holographically enforced within the psyche of every human being the wetiko virus pervades and underlies the entire field of consciousness, and can therefore potentially manifest through any one of us at any moment if we are not mindful.’ The ‘Covid’ hoax has achieved this with many people, but others have not fallen into Wetiko’s frequency lair. Players in the ‘Covid’ human catastrophe including Gates, Schwab, Tedros, Fauci, Whitty, Vallance, Johnson, Hancock, Ferguson, Drosten, and all the rest, including the psychopath psychologists, are expressions of Wetiko. This is why

they have no compassion or empathy and no emotional consequence for what they do that would make them stop doing it. Observe all the people who support the psychopaths in authority against the Pushbackers despite the damaging impact the psychopaths have on their own lives and their family's lives. You are again looking at Wetiko possession which prevents them seeing through the lies to the obvious scam going on. *Why can't they see it?* Wetiko won't let them see it. The perceptual divide that has now become a chasm is between the Wetikoed and the non-Wetikoed.

Paul Levy describes Wetiko in the same way that I have long described the Archontic force. They are the same distorted consciousness operating across dimensions of reality: '... the subtle body of wetiko is not located in the third dimension of space and time, literally existing in another dimension ... it is able to affect ordinary lives by mysteriously interpenetrating into our three-dimensional world.' Wetiko does this through its incarnate representatives in the Cult and by weaving itself into The Field which on our level of reality is the electromagnetic information field of the simulation or Matrix. More than that, the simulation *is* Wetiko / Yaldabaoth. Caleb Scharf, Director of Astrobiology at Columbia University, has speculated that 'alien life' could be so advanced that it has transcribed itself into the quantum realm to become what we call physics. He said intelligence indistinguishable from the fabric of the Universe would solve many of its greatest mysteries:

Perhaps hyper-advanced life isn't just external. Perhaps it's already all around. It is embedded in what we perceive to be physics itself, from the root behaviour of particles and fields to the phenomena of complexity and emergence ... In other words, life might not just be in the equations. It might BE the equations [My emphasis].

Scharf said it is possible that 'we don't recognise advanced life because it forms an integral and unsuspicious part of what we've considered to be the natural world'. I agree. Wetiko/Yaldabaoth *is* the simulation. We are literally in the body of the beast. But that doesn't mean it has to control us. We all have the power to overcome Wetiko

influence and the Cult knows that. I doubt it sleeps too well because it knows that.

Which Field?

This, I suggest, is how it all works. There are two Fields. One is the fierce electromagnetic light of the Matrix within the speed of light; the other is the ‘watery light’ of The Field beyond the walls of the Matrix that connects with the Great Infinity. Five-sense mind and the decoding systems of the body attach us to the Field of Matrix light. They have to or we could not experience this reality. Five-sense mind sees only the Matrix Field of information while our expanded consciousness is part of the Infinity Field. When we open our minds, and most importantly our hearts, to the Infinity Field we have a mission control which gives us an expanded perspective, a road map, to understand the nature of the five-sense world. If we are isolated only in five-sense mind there is no mission control. We’re on our own trying to understand a world that’s constantly feeding us information to ensure we do not understand. People in this state can feel ‘lost’ and bewildered with no direction or radar. You can see ever more clearly those who are influenced by the Fields of Big Infinity or little five-sense mind simply by their views and behaviour with regard to the ‘Covid’ hoax. We have had this division throughout known human history with the mass of the people on one side and individuals who could see and intuit beyond the walls of the simulation – Plato’s prisoner who broke out of the cave and saw reality for what it is. Such people have always been targeted by Wetiko/Archon-possessed authority, burned at the stake or demonised as mad, bad and dangerous. The Cult today and its global network of ‘anti-hate’, ‘anti-fascist’ Woke groups are all expressions of Wetiko attacking those exposing the conspiracy, ‘Covid’ lies and the ‘vaccine’ agenda.

Woke as a whole is Wetiko which explains its black and white mentality and how at one it is with the Wetiko-possessed Cult. Paul Levy said: ‘To be in this paradigm is to still be under the thrall of a two-valued logic – where things are either true or false – of a

wetikoized mind.' Wetiko consciousness is in a permanent rage, therefore so is Woke, and then there is Woke inversion and contradiction. 'Anti-fascists' act like fascists because fascists *and* 'anti-fascists' are both Wetiko at work. Political parties act the same while claiming to be different for the same reason. Secret society and satanic rituals are attaching initiates to Wetiko and the cold, ruthless, psychopathic mentality that secures the positions of power all over the world is Wetiko. Reframing 'training programmes' have the same cumulative effect of attaching Wetiko and we have their graduates described as automatons and robots with a cold, psychopathic, uncaring demeanour. They are all traits of Wetiko possession and look how many times they have been described in this book and elsewhere with regard to personnel behind 'Covid' including the police and medical profession. Climbing the greasy pole in any profession in a Wetiko society requires traits of Wetiko to get there and that is particularly true of politics which is not about fair competition and pre-eminence of ideas. It is founded on how many backs you can stab and arses you can lick. This culminated in the global 'Covid' coordination between the Wetiko possessed who pulled it off in all the different countries without a trace of empathy and compassion for their impact on humans. Our sight sense can see only holographic form and not the Field which connects holographic form. Therefore we perceive 'physical' objects with 'space' in between. In fact that 'space' is energy/consciousness operating on multiple frequencies. One of them is Wetiko and that connects the Cult psychopaths, those who submit to the psychopaths, and those who serve the psychopaths in the media operations of the world. Wetiko is Gates. Wetiko is the mask-wearing submissive. Wetiko is the fake journalist and 'fact-checker'. The Wetiko Field is coordinating the whole thing. Psychopaths, gofers, media operatives, 'anti-hate' hate groups, 'fact-checkers' and submissive people work as one unit *even without human coordination* because they are attached to the *same* Field which is organising it all ([Fig 22](#)). Paul Levy is here describing how Wetiko-possessed people are drawn together and refuse to let any information breach their rigid

perceptions. He was writing long before ‘Covid’, but I think you will recognise followers of the ‘Covid’ religion *oh just a little bit*:

People who are channelling the vibratory frequency of wetiko align with each other through psychic resonance to reinforce their unspoken shared agreement so as to uphold their deranged view of reality. Once an unconscious content takes possession of certain individuals, it irresistibly draws them together by mutual attraction and knits them into groups tied together by their shared madness that can easily swell into an avalanche of insanity.

A psychic epidemic is a closed system, which is to say that it is insular and not open to any new information or informing influences from the outside world which contradict its fixed, limited, and limiting perspective.

There we have the Woke mind and the ‘Covid’ mind. Compatible resonance draws the awakening together, too, which is clearly happening today.

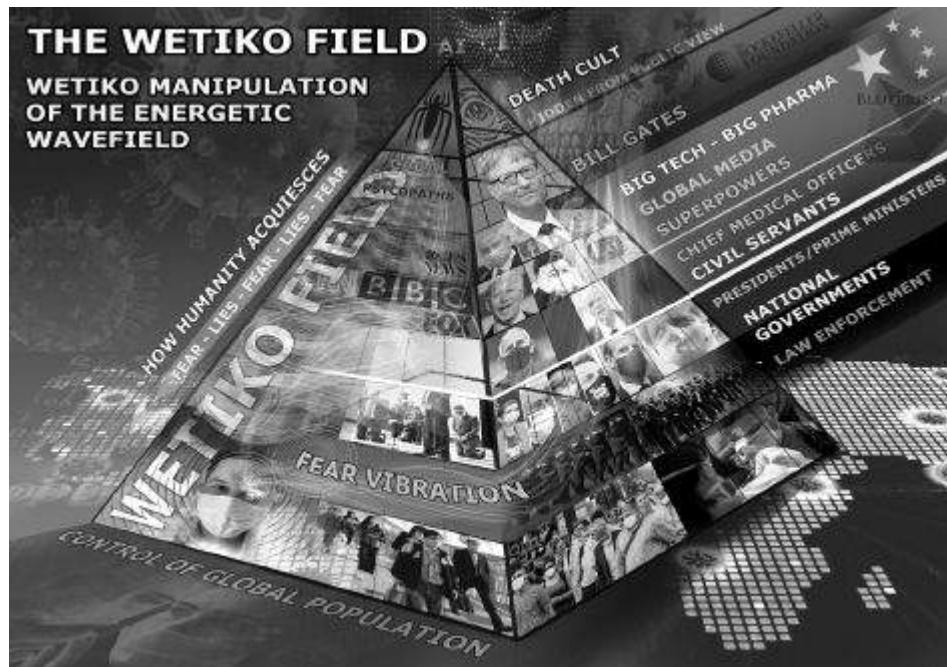


Figure 22: The Wetiko Field from which the Cult pyramid and its personnel are made manifest. (Image by Neil Hague).

Spiritual servitude

Wetiko doesn’t care about humans. It’s not human; it just possesses humans for its own ends and the effect (depending on the scale of

possession) can be anything from extreme psychopathy to unquestioning obedience. Wetiko's worst nightmare is for human consciousness to expand beyond the simulation. Everything is focussed on stopping that happening through control of information, thus perception, thus frequency. The 'education system', media, science, medicine, academia, are all geared to maintaining humanity in five-sense servitude as is the constant stimulation of low-vibrational mental and emotional states (see 'Covid'). Wetiko seeks to dominate those subconscious spaces between five-sense perception and expanded consciousness where the computer meets the operator. From these subconscious hiding places Wetiko speaks to us to trigger urges and desires that we take to be our own and manipulate us into anything from low-vibrational to psychopathic states. Remember how Islam describes the Jinn as invisible tricksters that 'whisper' and confuse. Wetiko is the origin of the 'trickster god' theme that you find in cultures all over the world. Jinn, like the Archons, are Wetiko which is terrified of humans awakening and reconnecting with our true self for then its energy source has gone. With that the feedback loop breaks between Wetiko and human perception that provides the energetic momentum on which its very existence depends as a force of evil. Humans are both its target and its source of survival, but only if we are operating in low-vibrational states of fear, hate, depression and the background anxiety that most people suffer. We are Wetiko's target because we are its key to survival. It needs us, not the other way round. Paul Levy writes:

A vampire has no intrinsic, independent, substantial existence in its own right; it only exists in relation to us. The pathogenic, vampiric mind-parasite called wetiko is nothing in itself – not being able to exist from its own side – yet it has a 'virtual reality' such that it can potentially destroy our species ...

...The fact that a vampire is not reflected by a mirror can also mean that what we need to see is that there's nothing, no-thing to see, other than ourselves. The fact that wetiko is the expression of something inside of us means that the cure for wetiko is with us as well. The critical issue is finding this cure within us and then putting it into effect.

Evil begets evil because if evil does not constantly expand and find new sources of energetic sustenance its evil, its *distortion*, dies with the assimilation into balance and harmony. Love is the garlic to Wetiko's vampire. Evil, the absence of love, cannot exist in the presence of love. I think I see a way out of here. I have emphasised so many times over the decades that the Archons/Wetiko and their Cult are not all powerful. *They are not.* I don't care how it looks even now *they are not.* I have not called them little boys in short trousers for effect. I have said it because it is true. Wetiko's insatiable desire for power over others is not a sign of its omnipotence, but its insecurity. Paul Levy writes: 'Due to the primal fear which ultimately drives it and which it is driven to cultivate, wetiko's body politic has an intrinsic and insistent need for centralising power and control so as to create imagined safety for itself.' *Yeeeeees!* Exactly! Why does Wetiko want humans in an ongoing state of fear? Wetiko itself *is* fear and it is petrified of love. As evil is an absence of love, so love is an absence of fear. Love conquers all and *especially* Wetiko which *is* fear. Wetiko brought fear into the world when it wasn't here before. *Fear* was the 'fall', the fall into low-frequency ignorance and illusion – fear is False Emotion Appearing Real. The simulation is driven and energised by fear because Wetiko/Yaldabaoth (fear) *are* the simulation. Fear is the absence of love and Wetiko is the absence of love.

Wetiko today

We can now view current events from this level of perspective. The 'Covid' hoax has generated momentous amounts of ongoing fear, anxiety, depression and despair which have empowered Wetiko. No wonder people like Gates have been the instigators when they are Wetiko incarnate and exhibit every trait of Wetiko in the extreme. See how cold and unemotional these people are like Gates and his cronies, how dead of eye they are. That's Wetiko. Sabbatians are Wetiko and everything they control including the World Health Organization, Big Pharma and the 'vaccine' makers, national 'health'

hierarchies, corporate media, Silicon Valley, the banking system, and the United Nations with its planned transformation into world government. All are controlled and possessed by the Wetiko distortion into distorting human society in its image. We are with this knowledge at the gateway to understanding the world.

Divisions of race, culture, creed and sexuality are diversions to hide the real division between those possessed and influenced by Wetiko and those that are not. The ‘Covid’ hoax has brought both clearly into view. Human behaviour is not about race. Tyrants and dictatorships come in all colours and creeds. What unites the US president bombing the innocent and an African tribe committing genocide against another as in Rwanda? What unites them? *Wetiko*. All wars are Wetiko, all genocide is Wetiko, all hunger over centuries in a world of plenty is Wetiko. Children going to bed hungry, including in the West, is Wetiko. Cult-generated Woke racial divisions that focus on the body are designed to obscure the reality that divisions in behaviour are manifestations of mind, not body. Obsession with body identity and group judgement is a means to divert attention from the real source of behaviour – mind and perception. Conflict sown by the Woke both within themselves and with their target groups are Wetiko providing lunch for itself through still more agents of the division, chaos, and fear on which it feeds. The Cult is seeking to assimilate the entirety of humanity and all children and young people into the Wetiko frequency by manipulating them into states of fear and despair. Witness all the suicide and psychological unravelling since the spring of 2020. Wetiko psychopaths want to impose a state of unquestioning obedience to authority which is no more than a conduit for Wetiko to enforce its will and assimilate humanity into itself. It needs us to believe that resistance is futile when it fears resistance and even more so the game-changing non-cooperation with its impositions. It can use violent resistance for its benefit. Violent impositions and violent resistance are *both* Wetiko. The Power of Love with its Power of No will sweep Wetiko from our world. Wetiko and its Cult know that. They just don’t want us to know.

AI Wetiko

This brings me to AI or artificial intelligence and something else Wetikos don't want us to know. What is AI *really*? I know about computer code algorithms and AI that learns from data input. These, however, are more diversions, the expeditionary force, for the real AI that they want to connect to the human brain as promoted by Silicon Valley Wetikos like Kurzweil. What is this AI? It is the frequency of *Wetiko*, the frequency of the Archons. The connection of AI to the human brain is the connection of the Wetiko frequency to create a Wetiko hive mind and complete the job of assimilation. The hive mind is planned to be controlled from Israel and China which are both 100 percent owned by Wetiko Sabbatians. The assimilation process has been going on minute by minute in the 'smart' era which fused with the 'Covid' era. We are told that social media is scrambling the minds of the young and changing their personality. This is true, but what is social media? Look more deeply at how it works, how it creates divisions and conflict, the hostility and cruelty, the targeting of people until they are destroyed. That's Wetiko. Social media is manipulated to tune people to the Wetiko frequency with all the emotional exploitation tricks employed by platforms like Facebook and its Wetiko front man, Zuckerberg. Facebook's Instagram announced a new platform for children to overcome a legal bar on them using the main site. This is more Wetiko exploitation and manipulation of kids. Amnesty International likened the plan to foxes offering to guard the henhouse and said it was incompatible with human rights. Since when did Wetiko or Zuckerberg (I repeat myself) care about that? Would Brin and Page at Google, Wojcicki at YouTube, Bezos at Amazon and whoever the hell runs Twitter act as they do if they were not channelling Wetiko? Would those who are developing technologies for no other reason than human control? How about those designing and selling technologies to kill people and Big Pharma drug and 'vaccine' producers who know they will end or devastate lives? Quite a thought for these people to consider is that if you are Wetiko in a human life you are Wetiko on the 'other side' unless your frequency

changes and that can only change by a change of perception which becomes a change of behaviour. Where Gates is going does not bear thinking about although perhaps that's exactly where he wants to go. Either way, that's where he's going. His frequency will make it so.

The frequency lair

I have been saying for a long time that a big part of the addiction to smartphones and devices is that a frequency is coming off them that entraps the mind. People spend ages on their phones and sometimes even a minute or so after they put them down they pick them up again and it all repeats. 'Covid' lockdowns will have increased this addiction a million times for obvious reasons. Addictions to alcohol overindulgence and drugs are another way that Wetiko entraps consciousness to attach to its own. Both are symptoms of low-vibrational psychological distress which alcoholism and drug addiction further compound. Do we think it's really a coincidence that access to them is made so easy while potions that can take people into realms beyond the simulation are banned and illegal? I have explored smartphone addiction in other books, the scale is mind-blowing, and that level of addiction does not come without help. Tech companies that make these phones are Wetiko and they will have no qualms about destroying the minds of children. We are seeing again with these companies the Wetiko perceptual combination of psychopathic enforcers and weak and meek unquestioning compliance by the rank and file.

The global Smart Grid is the Wetiko Grid and it is crucial to complete the Cult endgame. The simulation is radiation and we are being deluged with technological radiation on a devastating scale. Wetiko frauds like Elon Musk serve Cult interests while occasionally criticising them to maintain his street-cred. 5G and other forms of Wi-Fi are being directed at the earth from space on a volume and scale that goes on increasing by the day. Elon Musk's (officially) SpaceX Starlink project is in the process of putting tens of thousands of satellites in low orbit to cover every inch of the planet with 5G and other Wi-Fi to create Kurzweil's global 'cloud' to which the

human mind is planned to be attached very soon. SpaceX has approval to operate 12,000 satellites with more than 1,300 launched at the time of writing and applications filed for 30,000 more. Other operators in the Wi-Fi, 5G, low-orbit satellite market include OneWeb (UK), Telesat (Canada), and AST & Science (US). Musk tells us that AI could be the end of humanity and then launches a company called Neuralink to connect the human brain to computers. Musk's (in theory) Tesla company is building electric cars and the driverless vehicles of the smart control grid. As frauds and bullshitters go Elon Musk in my opinion is Major League.

5G and technological radiation in general are destructive to human health, genetics and psychology and increasing the strength of artificial radiation underpins the five-sense perceptual bubbles which are themselves expressions of radiation or electromagnetism. Freedom activist John Whitehead was so right with his 'databit by databit, we are building our own electronic concentration camps'. The Smart Grid and 5G is a means to control the human mind and infuse perceptual information into The Field to influence anyone in sync with its frequency. You can change perception and behaviour en masse if you can manipulate the population into those levels of frequency and this is happening all around us today. The arrogance of Musk and his fellow Cult operatives knows no bounds in the way that we see with Gates. Musk's satellites are so many in number already they are changing the night sky when viewed from Earth. The astronomy community has complained about this and they have seen nothing yet. Some consequences of Musk's Wetiko hubris include: Radiation; visible pollution of the night sky; interference with astronomy and meteorology; ground and water pollution from intensive use of increasingly many spaceports; accumulating space debris; continual deorbiting and burning up of aging satellites, polluting the atmosphere with toxic dust and smoke; and ever-increasing likelihood of collisions. A collective public open letter of complaint to Musk said:

We are writing to you ... because SpaceX is in process of surrounding the Earth with a network of thousands of satellites whose very purpose is to irradiate every square inch of the

Earth. SpaceX, like everyone else, is treating the radiation as if it were not there. As if the mitochondria in our cells do not depend on electrons moving undisturbed from the food we digest to the oxygen we breathe.

As if our nervous systems and our hearts are not subject to radio frequency interference like any piece of electronic equipment. As if the cancer, diabetes, and heart disease that now afflict a majority of the Earth's population are not metabolic diseases that result from interference with our cellular machinery. As if insects everywhere, and the birds and animals that eat them, are not starving to death as a result.

People like Musk and Gates believe in their limitless Wetiko arrogance that they can do whatever they like to the world because they own it. Consequences for humanity are irrelevant. It's absolutely time that we stopped taking this shit from these self-styled masters of the Earth when you consider where this is going.

Why is the Cult so anti-human?

I hear this question often: Why would they do this when it will affect them, too? Ah, but will it? Who is this *them*? Forget their bodies. They are just vehicles for Wetiko consciousness. When you break it all down to the foundations we are looking at a state of severely distorted consciousness targeting another state of consciousness for assimilation. The rest is detail. The simulation is the fly-trap in which unique sensations of the five senses create a cycle of addiction called reincarnation. Renegade Minds see that everything which happens in our reality is a smaller version of the whole picture in line with the holographic principle. Addiction to the radiation of smart technology is a smaller version of addiction to the whole simulation. Connecting the body/brain to AI is taking that addiction on a giant step further to total ongoing control by assimilating human incarnate consciousness into Wetiko. I have watched during the 'Covid' hoax how many are becoming ever more profoundly attached to Wetiko's perceptual calling cards of aggressive response to any other point of view ('There is no other god but me'), psychopathic lack of compassion and empathy, and servile submission to the narrative and will of authority. Wetiko is the psychopaths *and* subservience to psychopaths. The Cult of Wetiko is

so anti-human because it is *not* human. It embarked on a mission to destroy human by targeting everything that it means to be human and to survive as human. ‘Covid’ is not the end, just a means to an end. The Cult with its Wetiko consciousness is seeking to change Earth systems, including the atmosphere, to suit them, not humans. The gathering bombardment of 5G alone from ground and space is dramatically changing The Field with which the five senses interact. There is so much more to come if we sit on our hands and hope it will all go away. It is not meant to go away. It is meant to get ever more extreme and we need to face that while we still can – just.

Carbon dioxide is the gas of life. Without that human is over. Kaput, gone, history. No natural world, no human. The Cult has created a cock and bull story about carbon dioxide and climate change to justify its reduction to the point where Gates and the ignoramus Biden ‘climate chief’ John Kerry want to suck it out of the atmosphere. Kerry wants to do this because his master Gates does. Wetikos have made the gas of life a demon with the usual support from the Wokers of Extinction Rebellion and similar organisations and the bewildered puppet-child that is Greta Thunberg who was put on the world stage by Klaus Schwab and the World Economic Forum. The name Extinction Rebellion is both ironic and as always Wetiko inversion. The gas that we need to survive must be reduced to save us from extinction. The most basic need of human is oxygen and we now have billions walking around in face nappies depriving body and brain of this essential requirement of human existence. More than that 5G at 60 gigahertz interacts with the oxygen molecule to reduce the amount of oxygen the body can absorb into the bloodstream. The obvious knock-on consequences of that for respiratory and cognitive problems and life itself need no further explanation. Psychopaths like Musk are assembling a global system of satellites to deluge the human atmosphere with this insanity. The man should be in jail. Here we have two most basic of human needs, oxygen and carbon dioxide, being dismantled.

Two others, water and food, are getting similar treatment with the United Nations Agendas 21 and 2030 – the Great Reset – planning to

centrally control all water and food supplies. People will not even own rain water that falls on their land. Food is affected at the most basic level by reducing carbon dioxide. We have genetic modification or GMO infiltrating the food chain on a mass scale, pesticides and herbicides polluting the air and destroying the soil. Freshwater fish that provide livelihoods for 60 million people and feed hundreds of millions worldwide are being 'pushed to the brink' according the conservationists while climate change is the only focus. Now we have Gates and Schwab wanting to dispense with current food sources all together and replace them with a synthetic version which the Wetiko Cult would control in terms of production and who eats and who doesn't. We have been on the Totalitarian Tiptoe to this for more than 60 years as food has become ever more processed and full of chemical shite to the point today when it's not natural food at all. As Dr Tom Cowan says: 'If it has a label don't eat it.' Bill Gates is now the biggest owner of farmland in the United States and he does nothing without an ulterior motive involving the Cult. Klaus Schwab wrote: 'To feed the world in the next 50 years we will need to produce as much food as was produced in the last 10,000 years ... food security will only be achieved, however, if regulations on genetically modified foods are adapted to reflect the reality that gene editing offers a precise, efficient and safe method of improving crops.' Liar. People and the world are being targeted with aluminium through vaccines, chemtrails, food, drink cans, and endless other sources when aluminium has been linked to many health issues including dementia which is increasing year after year. Insects, bees and wildlife essential to the food chain are being deleted by pesticides, herbicides and radiation which 5G is dramatically increasing with 6G and 7G to come. The pollinating bee population is being devastated while wildlife including birds, dolphins and whales are having their natural radar blocked by the effects of ever-increasing radiation. In the summer windscreens used to be splattered with insects so numerous were they. It doesn't happen now. Where have they gone?

Synthetic everything

The Cult is introducing genetically-modified versions of trees, plants and insects including a Gates-funded project to unleash hundreds of millions of genetically-modified, lab-altered and patented male mosquitoes to mate with wild mosquitoes and induce genetic flaws that cause them to die out. Clinically-insane Gates-funded Japanese researchers have developed mosquitos that spread vaccine and are dubbed 'flying vaccinators'. Gates is funding the modification of weather patterns in part to sell the myth that this is caused by carbon dioxide and he's funding geoengineering of the skies to change the atmosphere. Some of this came to light with the Gates-backed plan to release tonnes of chalk into the atmosphere to 'deflect the Sun and cool the planet'. Funny how they do this while the heating effect of the Sun is not factored into climate projections focussed on carbon dioxide. The reason is that they want to reduce carbon dioxide (so don't mention the Sun), but at the same time they do want to reduce the impact of the Sun which is so essential to human life and health. I have mentioned the sun-cholesterol-vitamin D connection as they demonise the Sun with warnings about skin cancer (caused by the chemicals in sun cream they tell you to splash on). They come from the other end of the process with statin drugs to reduce cholesterol that turns sunlight into vitamin D. A lack of vitamin D leads to a long list of health effects and how vitamin D levels must have fallen with people confined to their homes over 'Covid'. Gates is funding other forms of geoengineering and most importantly chemtrails which are dropping heavy metals, aluminium and self-replicating nanotechnology onto the Earth which is killing the natural world. See *Everything You Need To Know, But Have Never Been Told* for the detailed background to this.

Every human system is being targeted for deletion by a force that's not human. The Wetiko Cult has embarked on the process of transforming the human body from biological to synthetic biological as I have explained. Biological is being replaced by the artificial and synthetic – Archontic 'countermimicry' – right across human society. The plan eventually is to dispense with the human body altogether

and absorb human consciousness – which it wouldn't really be by then – into cyberspace (the simulation which is Wetiko/Yaldabaoth). Preparations for that are already happening if people would care to look. The alternative media rightly warns about globalism and 'the globalists', but this is far bigger than that and represents the end of the human race as we know it. The 'bad copy' of prime reality that Gnostics describe was a bad copy of harmony, wonder and beauty to start with before Wetiko/Yaldabaoth set out to change the simulated 'copy' into something very different. The process was slow to start with. Entrapped humans in the simulation timeline were not technologically aware and they had to be brought up to intellectual speed while being suppressed spiritually to the point where they could build their own prison while having no idea they were doing so. We have now reached that stage where technological intellect has the potential to destroy us and that's why events are moving so fast. Central American shaman Don Juan Matus said:

Think for a moment, and tell me how you would explain the contradictions between the intelligence of man the engineer and the stupidity of his systems of belief, or the stupidity of his contradictory behaviour. Sorcerers believe that the predators have given us our systems of beliefs, our ideas of good and evil; our social mores. They are the ones who set up our dreams of success or failure. They have given us covetousness, greed, and cowardice. It is the predator who makes us complacent, routinary, and egomaniacal.

In order to keep us obedient and meek and weak, the predators engaged themselves in a stupendous manoeuvre – stupendous, of course, from the point of view of a fighting strategist; a horrendous manoeuvre from the point of those who suffer it. They gave us their mind. The predators' mind is baroque, contradictory, morose, filled with the fear of being discovered any minute now.

For 'predators' see Wetiko, Archons, Yaldabaoth, Jinn, and all the other versions of the same phenomenon in cultures and religions all over the world. The theme is always the same because it's true and it's real. We have reached the point where we have to deal with it. The question is – how?

Don't fight – walk away

I thought I'd use a controversial subheading to get things moving in terms of our response to global fascism. What do you mean 'don't fight'? What do you mean 'walk away'? We've got to fight. We can't walk away. Well, it depends what we mean by fight and walk away. If fighting means physical combat we are playing Wetiko's game and falling for its trap. It wants us to get angry, aggressive, and direct hate and hostility at the enemy we think we must fight. Every war, every battle, every conflict, has been fought with Wetiko leading both sides. It's what it does. Wetiko wants a fight, anywhere, any place. Just hit me, son, so I can hit you back. Wetiko hits Wetiko and Wetiko hits Wetiko in return. I am very forthright as you can see in exposing Wetikos of the Cult, but I don't hate them. I refuse to hate them. It's what they want. What you hate you become. What you *fight* you become. Wokers, 'anti-haters' and 'anti-fascists' prove this every time they reach for their keyboards or don their balaclavas. By walk away I mean to disengage from Wetiko which includes ceasing to cooperate with its tyranny. Paul Levy says of Wetiko:

The way to 'defeat' evil is not to try to destroy it (for then, in playing evil's game, we have already lost), but rather, to find the invulnerable place within ourselves where evil is unable to vanquish us – this is to truly 'win' our battle with evil.

Wetiko is everywhere in human society and it's been on steroids since the 'Covid' hoax. Every shouting match over wearing masks has Wetiko wearing a mask and Wetiko not wearing one. It's an electrical circuit of push and resist, push and resist, with Wetiko pushing *and* resisting. Each polarity is Wetiko empowering itself. Dictionary definitions of 'resist' include 'opposing, refusing to accept or comply with' and the word to focus on is 'opposing'. What form does this take – setting police cars alight or 'refusing to accept or comply with'? The former is Wetiko opposing Wetiko while the other points the way forward. This is the difference between those aggressively demanding that government fascism must be obeyed who stand in stark contrast to the great majority of Pushbackers. We saw this clearly with a march by thousands of Pushbackers against lockdown in London followed days later by a Woker-hijacked

protest in Bristol in which police cars were set on fire. Masks were virtually absent in London and widespread in Bristol. Wetiko wants lockdown on every level of society and infuses its aggression to police it through its unknowing stooges. Lockdown protesters are the ones with the smiling faces and the hugs, The two blatantly obvious states of being – getting more obvious by the day – are the result of Wokers and their like becoming ever more influenced by the simulation Field of Wetiko and Pushbackers ever more influenced by The Field of a far higher vibration beyond the simulation. Wetiko can't invade the heart which is where most lockdown opponents are coming from. It's the heart that allows them to see through the lies to the truth in ways I will be highlighting.

Renegade Minds know that calmness is the place from which wisdom comes. You won't find wisdom in a hissing fit and wisdom is what we need in abundance right now. Calmness is not weakness – you don't have to scream at the top of your voice to be strong. Calmness is indeed a sign of strength. 'No' means I'm not doing it. NOOOO!!! doesn't mean you're not doing it even more. Volume does not advance 'No – I'm not doing it'. You are just not doing it. Wetiko possessed and influenced don't know how to deal with that. Wetiko wants a fight and we should not give it one. What it needs more than anything is our *cooperation* and we should not give that either. Mass rallies and marches are great in that they are a visual representation of feeling, but if it ends there they are irrelevant. You demand that Wetikos act differently? Well, they're not going to are they? They are Wetikos. We don't need to waste our time demanding that something doesn't happen when that will make no difference. We need to delete the means that *allows* it to happen. This, invariably, is our cooperation. You can demand a child stop firing a peashooter at the dog or you can refuse to buy the peashooter. If you provide the means you are cooperating with the dog being smacked on the nose with a pea. How can the authorities enforce mask-wearing if millions in a country refuse? What if the 74 million Pushbackers that voted for Trump in 2020 refused to wear masks, close their businesses or stay in their homes. It would be unenforceable. The

few control the many through the compliance of the many and that's always been the dynamic be it 'Covid' regulations or the Roman Empire. I know people can find it intimidating to say no to authority or stand out in a crowd for being the only one with a face on display; but it has to be done or it's over. I hope I've made clear in this book that where this is going will be far more intimidating than standing up now and saying 'No' – I will not cooperate with my own enslavement and that of my children. There might be consequences for some initially, although not so if enough do the same. The question that must be addressed is what is going to happen if we don't? It is time to be strong and unyieldingly so. No means no. Not here and there, but *everywhere* and *always*. I have refused to wear a mask and obey all the other nonsense. I will not comply with tyranny. I repeat: Fascism is not imposed by fascists – there are never enough of them. Fascism is imposed by the population acquiescing to fascism. *I will not do it.* I will die first, or my body will. Living meekly under fascism is a form of death anyway, the death of the spirit that Martin Luther King described.

Making things happen

We must not despair. This is not over till it's over and it's far from that. The 'fat lady' must refuse to sing. The longer the 'Covid' hoax has dragged on and impacted on more lives we have seen an awakening of phenomenal numbers of people worldwide to the realisation that what they have believed all their lives is not how the world really is. Research published by the system-serving University of Bristol and King's College London in February, 2021, concluded: 'One in every 11 people in Britain say they trust David Icke's take on the coronavirus pandemic.' It will be more by now and we have gathering numbers to build on. We must urgently progress from seeing the scam to ceasing to cooperate with it. Prominent German lawyer Reiner Fuellmich, also licenced to practice law in America, is doing a magnificent job taking the legal route to bring the psychopaths to justice through a second Nuremberg tribunal for crimes against humanity. Fuellmich has an impressive record of

beating the elite in court and he formed the German Corona Investigative Committee to pursue civil charges against the main perpetrators with a view to triggering criminal charges. Most importantly he has grasped the foundation of the hoax – the PCR test not testing for the ‘virus’ – and Christian Drosten is therefore on his charge sheet along with Gates frontman Tedros at the World Health Organization. Major players must be not be allowed to inflict their horrors on the human race without being brought to book. A life sentence must follow for Bill Gates and the rest of them. A group of researchers has also indicted the government of Norway for crimes against humanity with copies sent to the police and the International Criminal Court. The lawsuit cites participation in an internationally-planned false pandemic and violation of international law and human rights, the European Commission’s definition of human rights by coercive rules, Nuremberg and Hague rules on fundamental human rights, and the Norwegian constitution. We must take the initiative from hereon and not just complain, protest and react.

There are practical ways to support vital mass non-cooperation. Organising in numbers is one. Lockdown marches in London in the spring in 2021 were mass non-cooperation that the authorities could not stop. There were too many people. Hundreds of thousands walked the London streets in the centre of the road for mile after mile while the Face-Nappies could only look on. They were determined, but calm, and just *did it* with no histrionics and lots of smiles. The police were impotent. Others are organising group shopping without masks for mutual support and imagine if that was happening all over. Policing it would be impossible. If the store refuses to serve people in these circumstances they would be faced with a long line of trolleys full of goods standing on their own and everything would have to be returned to the shelves. How would they cope with that if it kept happening? I am talking here about moving on from complaining to being pro-active; from watching things happen to making things happen. I include in this our relationship with the police. The behaviour of many Face-Nappies

has been disgraceful and anyone who thinks they would never find concentration camp guards in the ‘enlightened’ modern era have had that myth busted big-time. The period and setting may change – Wetikos never do. I watched film footage from a London march in which a police thug viciously kicked a protestor on the floor who had done nothing. His fellow Face-Nappies stood in a ring protecting him. What he did was a criminal assault and with a crowd far outnumbering the police this can no longer be allowed to happen unchallenged. I get it when people chant ‘shame on you’ in these circumstances, but that is no longer enough. They *have* no shame those who do this. Crowds needs to start making a citizen’s arrest of the police who commit criminal offences and brutally attack innocent people and defenceless women. A citizen’s arrest can be made under section 24A of the UK Police and Criminal Evidence (PACE) Act of 1984 and you will find something similar in other countries. I prefer to call it a Common Law arrest rather than citizen’s for reasons I will come to shortly. Anyone can arrest a person committing an indictable offence or if they have reasonable grounds to suspect they are committing an indictable offence. On both counts the attack by the police thug would have fallen into this category. A citizen’s arrest can be made to stop someone:

- Causing physical injury to himself or any other person
- Suffering physical injury
- Causing loss of or damage to property
- Making off before a constable can assume responsibility for him

A citizen’s arrest may also be made to prevent a breach of the peace under Common Law and if they believe a breach of the peace will happen or anything related to harm likely to be done or already done in their presence. This is the way to go I think – the Common Law version. If police know that the crowd and members of the public will no longer be standing and watching while they commit

their thuggery and crimes they will think twice about acting like Brownshirts and Blackshirts.

Common Law – common sense

Mention of Common Law is very important. Most people think the law is the law as in one law. This is not the case. There are two bodies of law, Common Law and Statute Law, and they are not the same. Common Law is founded on the simple premise of do no harm. It does not recognise victimless crimes in which no harm is done while Statute Law does. There is a Statute Law against almost everything. So what is Statute Law? Amazingly it's the law of the sea that was brought ashore by the Cult to override the law of the land which is Common Law. They had no right to do this and as always they did it anyway. They had to. They could not impose their will on the people through Common Law which only applies to do no harm. How could you stitch up the fine detail of people's lives with that? Instead they took the law of the sea, or Admiralty Law, and applied it to the population. Statute Law refers to all the laws spewing out of governments and their agencies including all the fascist laws and regulations relating to 'Covid'. The key point to make is that Statute Law is *contract law*. It only applies between *contracting* corporations. Most police officers don't even know this. They have to be kept in the dark, too. Long ago when merchants and their sailing ships began to trade with different countries a contractual law was developed called Admiralty Law and other names. Again it only applied to *contracts* agreed between *corporate* entities. If there is no agreed contract the law of the sea had no jurisdiction *and that still applies to its new alias of Statute Law*. The problem for the Cult when the law of the sea was brought ashore was an obvious one. People were not corporations and neither were government entities. To overcome the latter they made governments and all associated organisations corporations. All the institutions are *private corporations* and I mean governments and their agencies, local councils, police, courts, military, US states, the whole lot. Go to the

Dun and Bradstreet corporate listings website for confirmation that they are all corporations. You are arrested by a private corporation called the police by someone who is really a private security guard and they take you to court which is another private corporation.

Neither have jurisdiction over you unless you consent and *contract* with them. This is why you hear the mantra about law enforcement policing by *consent* of the people. In truth the people 'consent' only in theory through monumental trickery.

Okay, the Cult overcame the corporate law problem by making governments and institutions corporate entities; but what about people? They are not corporations are they? Ah ... well in a sense, and *only* a sense, they are. Not people exactly – the illusion of people. The Cult creates a corporation in the name of everyone at the time that their birth certificate is issued. Note birth/ *berth* certificate and when you go to court under the law of the sea on land you stand in a *dock*. These are throwbacks to the origin. My Common Law name is David Vaughan Icke. The name of the corporation created by the government when I was born is called Mr David Vaughan Icke usually written in capitals as MR DAVID VAUGHAN ICKE. That is not me, the living, breathing man. It is a fictitious corporate entity. The trick is to make you think that David Vaughan Icke and MR DAVID VAUGHAN ICKE are the same thing. *They are not*. When police charge you and take you to court they are prosecuting the corporate entity and not the living, breathing, man or woman. They have to trick you into identifying as the corporate entity and contracting with them. Otherwise they have no jurisdiction. They do this through a language known as legalese. Lawful and legal are not the same either. Lawful relates to Common Law and legal relates to Statute Law. Legalese is the language of Statue Law which uses terms that mean one thing to the public and another in legalese. Notice that when a police officer tells someone why they are being charged he or she will say at the end: 'Do you understand?' To the public that means 'Do you comprehend?' In legalese it means 'Do you stand under me?' Do you stand under my authority? If you say

yes to the question you are unknowingly agreeing to give them jurisdiction over you in a contract between two corporate entities.

This is a confidence trick in every way. Contracts have to be agreed between informed parties and if you don't know that David Vaughan Icke is agreeing to be the corporation MR DAVID VAUGHAN ICKE you cannot knowingly agree to contract. They are deceiving you and another way they do this is to ask for proof of identity. You usually show them a driving licence or other document on which your corporate name is written. In doing so you are accepting that you are that corporate entity when you are not. Referring to yourself as a 'person' or 'citizen' is also identifying with your corporate fiction which is why I made the Common Law point about the citizen's arrest. If you are approached by a police officer you identify yourself immediately as a living, breathing, man or woman and say 'I do not consent, I do not contract with you and I do not understand' or stand under their authority. I have a Common Law birth certificate as a living man and these are available at no charge from commonlawcourt.com. Businesses registered under the Statute Law system means that its laws apply. There are, however, ways to run a business under Common Law. Remember all 'Covid' laws and regulations are Statute Law – the law of *contracts* and you do not have to contract. This doesn't mean that you can kill someone and get away with it. Common Law says do no harm and that applies to physical harm, financial harm etc. Police are employees of private corporations and there needs to be a new system of non-corporate Common Law constables operating outside the Statute Law system. If you go to davidicke.com and put Common Law into the search engine you will find videos that explain Common Law in much greater detail. It is definitely a road we should walk.

With all my heart

I have heard people say that we are in a spiritual war. I don't like the term 'war' with its Wetiko dynamic, but I know what they mean. Sweep aside all the bodily forms and we are in a situation in which two states of consciousness are seeking very different realities.

Wetiko wants upheaval, chaos, fear, suffering, conflict and control. The other wants love, peace, harmony, fairness and freedom. That's where we are. We should not fall for the idea that Wetiko is all-powerful and there's nothing we can do. Wetiko is not all-powerful. It's a joke, pathetic. It doesn't have to be, but it has made that choice for now. A handful of times over the years when I have felt the presence of its frequency I have allowed it to attach briefly so I could consciously observe its nature. The experience is not pleasant, the energy is heavy and dark, but the ease with which you can kick it back out the door shows that its real power is in persuading us that it has power. It's all a con. Wetiko is a con. It's a trickster and not a power that can control us if we unleash our own. The con is founded on manipulating humanity to give its power to Wetiko which recycles it back to present the illusion that it has power when its power is *ours* that we gave away. This happens on an energetic level and plays out in the world of the seen as humanity giving its power to Wetiko authority which uses that power to control the population when the power is only the power the population has handed over. How could it be any other way for billions to be controlled by a relative few? I have had experiences with people possessed by Wetiko and again you can kick its arse if you do it with an open heart. Oh yes – the *heart* which can transform the world of perceived 'matter'.

We are receiver-transmitters and processors of information, but what information and where from? Information is processed into perception in three main areas – the brain, the heart and the belly. These relate to thinking, knowing, and emotion. Wetiko wants us to be head and belly people which means we think within the confines of the Matrix simulation and low-vibrational emotional reaction scrambles balance and perception. A few minutes on social media and you see how emotion is the dominant force. Woke is all emotion and is therefore thought-free and fact-free. Our heart is something different. It *knows* while the head *thinks* and has to try to work it out because it doesn't know. The human energy field has seven prime vortexes which connect us with wider reality ([Fig 23](#)). Chakra means

'wheels of light' in the Sanskrit language of ancient India. The main ones are: The crown chakra on top of the head; brow (or 'third eye') chakra in the centre of the forehead; throat chakra; heart chakra in the centre of the chest; solar plexus chakra below the sternum; sacral chakra beneath the navel; and base chakra at the bottom of the spine. Each one has a particular function or functions. We feel anxiety and nervousness in the belly where the sacral chakra is located and this processes emotion that can affect the colon to give people 'the shits' or make them 'shit scared' when they are nervous. Chakras all play an important role, but the Mr and Mrs Big is the heart chakra which sits at the centre of the seven, above the chakras that connect us to the 'physical' and below those that connect with higher realms (or at least should). Here in the heart chakra we feel love, empathy and compassion – 'My heart goes out to you'. Those with closed hearts become literally 'heart-less' in their attitudes and behaviour (see Bill Gates). Native Americans portrayed Wetiko with what Paul Levy calls a 'frigid, icy heart, devoid of mercy' (see Bill Gates).

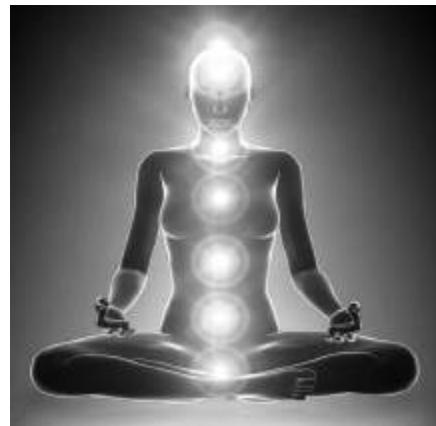


Figure 23: The chakra system which interpenetrates the human energy field. The heart chakra is the governor – or should be.

Wetiko trembles at the thought of heart energy which it cannot infiltrate. The frequency is too high. What it seeks to do instead is close the heart chakra vortex to block its perceptual and energetic influence. Psychopaths have 'hearts of stone' and emotionally-damaged people have 'heartache' and 'broken hearts'. The astonishing amount of heart disease is related to heart chakra

disruption with its fundamental connection to the ‘physical’ heart. Dr Tom Cowan has written an outstanding book challenging the belief that the heart is a pump and making the connection between the ‘physical’ and spiritual heart. Rudolph Steiner who was way ahead of his time said the same about the fallacy that the heart is a pump. *What?* The heart is not a pump? That’s crazy, right? Everybody knows that. Read Cowan’s *Human Heart, Cosmic Heart* and you will realise that the very idea of the heart as a pump is ridiculous when you see the evidence. How does blood in the feet so far from the heart get pumped horizontally up the body by the heart?? Cowan explains in the book the real reason why blood moves as it does. Our ‘physical’ heart is used to symbolise love when the source is really the heart vortex or spiritual heart which is our most powerful energetic connection to ‘out there’ expanded consciousness. That’s why we feel *knowing* – intuitive knowing – in the centre of the chest. Knowing doesn’t come from a process of thoughts leading to a conclusion. It is there in an instant all in one go. Our heart knows because of its connection to levels of awareness that *do* know. This is the meaning and source of intuition – intuitive *knowing*.

For the last more than 30 years of uncovering the global game and the nature of reality my heart has been my constant antenna for truth and accuracy. An American intelligence insider once said that I had quoted a disinformor in one of my books and yet I had only quoted the part that was true. He asked: ‘How do you do that?’ By using my heart antenna was the answer and anyone can do it. Heart-centred is how we are meant to be. With a closed heart chakra we withdraw into a closed mind and the bubble of five-sense reality. If you take a moment to focus your attention on the centre of your chest, picture a spinning wheel of light and see it opening and expanding. You will feel it happening, too, and perceptions of the heart like joy and love as the heart impacts on the mind as they interact. The more the chakra opens the more you will feel expressions of heart consciousness and as the process continues, and becomes part of you, insights and knowings will follow. An open

heart is connected to that level of awareness that knows all is *One*. You will see from its perspective that the fault-lines that divide us are only illusions to control us. An open heart does not process the illusions of race, creed and sexuality except as brief experiences for a consciousness that is all. Our heart does not see division, only unity (Figs 24 and 25). There's something else, too. Our hearts love to laugh. Mark Twain's quote that says 'The human race has one really effective weapon, and that is laughter' is really a reference to the heart which loves to laugh with the joy of knowing the true nature of infinite reality and that all the madness of human society is an illusion of the mind. Twain also said: 'Against the assault of laughter nothing can stand.' This is so true of Wetiko and the Cult. Their insecurity demands that they be taken seriously and their power and authority acknowledged and feared. We should do nothing of the sort. We should not get aggressive or fearful which their insecurity so desires. We should laugh in their face. Even in their no-face as police come over in their face-nappies and expect to be taken seriously. They don't take themselves seriously looking like that so why should we? Laugh in the face of intimidation. Laugh in the face of tyranny. You will see by its reaction that you have pressed all of its buttons. Wetiko does not know what to do in the face of laughter or when its targets refuse to concede their joy to fear. We have seen many examples during the 'Covid' hoax when people have expressed their energetic power and the string puppets of Wetiko retreat with their tail limp between their knees. Laugh – the world is bloody mad after all and if it's a choice between laughter and tears I know which way I'm going.



Figure 24: Head consciousness without the heart sees division and everything apart from everything else.

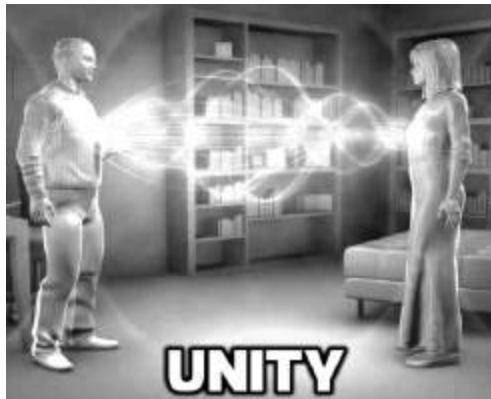


Figure 25: Heart consciousness sees everything as One.

Vaccines' and the soul

The foundation of Wetiko/Archon control of humans is the separation of incarnate five-sense mind from the infinite 'I' and closing the heart chakra where the True 'I' lives during a human life. The goal has been to achieve complete separation in both cases. I was interested therefore to read an account by a French energetic healer of what she said she experienced with a patient who had been given the 'Covid' vaccine. Genuine energy healers can sense information and consciousness fields at different levels of being which are referred to as 'subtle bodies'. She described treating the patient who later returned after having, without the healer's knowledge, two doses of the 'Covid vaccine'. The healer said:

I noticed immediately the change, very heavy energy emanating from [the] subtle bodies. The scariest thing was when I was working on the heart chakra, I connected with her soul: it was detached from the physical body, it had no contact and it was, as if it was floating in a state of total confusion: a damage to the consciousness that loses contact with the physical body, i.e. with our biological machine, there is no longer any communication between them.

I continued the treatment by sending light to the heart chakra, the soul of the person, but it seemed that the soul could no longer receive any light, frequency or energy. It was a very powerful experience for me. Then I understood that this substance is indeed used to detach consciousness so that this consciousness can no longer interact through this body that it possesses in life, where there is no longer any contact, no frequency, no light, no more energetic balance or mind.

This would create a human that is rudderless and at the extreme almost zombie-like operating with a fractional state of consciousness at the mercy of Wetiko. I was especially intrigued by what the healer said in the light of the prediction by the highly-informed Rudolf Steiner more than a hundred years ago. He said:

In the future, we will eliminate the soul with medicine. Under the pretext of a 'healthy point of view', there will be a vaccine by which the human body will be treated as soon as possible directly at birth, so that the human being cannot develop the thought of the existence of soul and Spirit. To materialistic doctors will be entrusted the task of removing the soul of humanity.

As today, people are vaccinated against this disease or that disease, so in the future, children will be vaccinated with a substance that can be produced precisely in such a way that people, thanks to this vaccination, will be immune to being subjected to the 'madness' of spiritual life. He would be extremely smart, but he would not develop a conscience, and that is the true goal of some materialistic circles.

Steiner said the vaccine would detach the physical body from the etheric body (subtle bodies) and 'once the etheric body is detached the relationship between the universe and the etheric body would become extremely unstable, and man would become an automaton'. He said 'the physical body of man must be polished on this Earth by spiritual will – so the vaccine becomes a kind of aryanique (Wetiko) force' and 'man can no longer get rid of a given materialistic feeling'. Humans would then, he said, become 'materialistic of constitution and can no longer rise to the spiritual'. I have been writing for years about DNA being a receiver-transmitter of information that connects us to other levels of reality and these 'vaccines' changing DNA can be likened to changing an antenna and what it can transmit and receive. Such a disconnection would clearly lead to changes in personality and perception. Steiner further predicted the arrival of AI. Big Pharma 'Covid vaccine' makers, expressions of Wetiko, are testing their DNA-manipulating evil on children as I write with a view to giving the 'vaccine' to babies. If it's a soul-body disconnecter – and I say that it is or can be – every child would be disconnected from 'soul' at birth and the 'vaccine' would create a closed system in which spiritual guidance from the greater self would play no part. This has been the ambition of Wetiko all

along. A Pentagon video from 2005 was leaked of a presentation explaining the development of vaccines to change behaviour by their effect on the brain. Those that believe this is not happening with the ‘Covid’ genetically-modifying procedure masquerading as a ‘vaccine’ should make an urgent appointment with Naivety Anonymous. Klaus Schwab wrote in 2018:

Neurotechnologies enable us to better influence consciousness and thought and to understand many activities of the brain. They include decoding what we are thinking in fine levels of detail through new chemicals and interventions that can influence our brains to correct for errors or enhance functionality.

The plan is clear and only the heart can stop it. With every heart that opens, every mind that awakens, Wetiko is weakened. Heart and love are far more powerful than head and hate and so nothing like a majority is needed to turn this around.

Beyond the Phantom

Our heart is the prime target of Wetiko and so it must be the answer to Wetiko. We *are* our heart which is part of one heart, the infinite heart. Our heart is where the true self lives in a human life behind firewalls of five-sense illusion when an imposter takes its place – *Phantom Self*; but our heart waits patiently to be set free any time we choose to see beyond the Phantom, beyond Wetiko. A Wetikoed Phantom Self can wreak mass death and destruction while the love of forever is locked away in its heart. The time is here to unleash its power and let it sweep away the fear and despair that is Wetiko. Heart consciousness does not seek manipulated, censored, advantage for its belief or religion, its activism and desires. As an expression of the One it treats all as One with the same rights to freedom and opinion. Our heart demands fairness for itself no more than for others. From this unity of heart we can come together in mutual support and transform this Wetikoed world into what reality is meant to be – a place of love, joy, happiness, fairness, justice and freedom. Wetiko has another agenda and that’s why the world is as

it is, but enough of this nonsense. Wetiko can't stay where hearts are open and it works so hard to keep them closed. Fear is its currency and its food source and love in its true sense has no fear. Why would love have fear when it knows it is *All That Is, Has Been, And Ever Can Be* on an eternal exploration of all possibility? Love in this true sense is not the physical attraction that passes for love. This can be an expression of it, yes, but Infinite Love, a love without condition, goes far deeper to the core of all being. It is the core of all being. Infinite reality was born from love beyond the illusions of the simulation. Love infinitely expressed is the knowing that all is One and the swiftly-passing experience of separation is a temporary hallucination. You cannot disconnect from Oneness; you can only perceive that you have and withdraw from its influence. This is the most important of all perception trickery by the mind parasite that is Wetiko and the foundation of all its potential for manipulation.

If we open our hearts, open the sluice gates of the mind, and redefine self-identity amazing things start to happen. Consciousness expands or contracts in accordance with self-identity. When true self is recognised as infinite awareness and label self – Phantom Self – is seen as only a series of brief experiences life is transformed. Consciousness expands to the extent that self-identity expands and everything changes. You see unity, not division, the picture, not the pixels. From this we can play the long game. No more is an experience something in and of itself, but a fleeting moment in the eternity of forever. Suddenly people in uniform and dark suits are no longer intimidating. Doing what your heart knows to be right is no longer intimidating and consequences for those actions take on the same nature of a brief experience that passes in the blink of an infinite eye. Intimidation is all in the mind. Beyond the mind there is no intimidation.

An open heart does not consider consequences for what it knows to be right. To do so would be to consider not doing what it knows to be right and for a heart in its power that is never an option. The Renegade Mind is really the Renegade Heart. Consideration of consequences will always provide a getaway car for the mind and

the heart doesn't want one. What is right in the light of what we face today is to stop cooperating with Wetiko in all its forms and to do it without fear or compromise. You cannot compromise with tyranny when tyranny always demands more until it has everything. Life is your perception and you are your destiny. Change your perception and you change your life. Change collective perception and we change the world.

*Come on people ... One human family, One heart, One goal ...
FREEEEEDOM!*

We must settle for nothing less.

Postscript

The big scare story as the book goes to press is the ‘Indian’ variant and the world is being deluged with propaganda about the ‘Covid catastrophe’ in India which mirrors in its lies and misrepresentations what happened in Italy before the first lockdown in 2020.

The *New York Post* published a picture of someone who had ‘collapsed in the street from Covid’ in India in April, 2021, which was actually taken during a gas leak in May, 2020. Same old, same old. Media articles in mid-February were asking why India had been so untouched by ‘Covid’ and then as their vaccine rollout gathered pace the alleged ‘cases’ began to rapidly increase. Indian ‘Covid vaccine’ maker Bharat Biotech was funded into existence by the Bill and Melinda Gates Foundation (the pair announced their divorce in May, 2021, which is a pity because they so deserve each other). The Indian ‘Covid crisis’ was ramped up by the media to terrify the world and prepare people for submission to still more restrictions. The scam that worked the first time was being repeated only with far more people seeing through the deceit. Davidicke.com and Ickonic.com have sought to tell the true story of what is happening by talking to people living through the Indian nightmare which has nothing to do with ‘Covid’. We posted a letter from ‘Alisha’ in Pune who told a very different story to government and media mendacity. She said scenes of dying people and overwhelmed hospitals were designed to hide what was really happening – genocide and starvation. Alisha said that millions had already died of starvation during the ongoing lockdowns while government and media were lying and making it look like the ‘virus’:

Restaurants, shops, gyms, theatres, basically everything is shut. The cities are ghost towns. Even so-called 'essential' businesses are only open till 11am in the morning. You basically have just an hour to buy food and then your time is up.

Inter-state travel and even inter-district travel is banned. The cops wait at all major crossroads to question why you are traveling outdoors or to fine you if you are not wearing a mask.

The medical community here is also complicit in genocide, lying about hospitals being full and turning away people with genuine illnesses, who need immediate care. They have even created a shortage of oxygen cylinders.

This is the classic Cult modus operandi played out in every country. Alisha said that people who would not have a PCR test not testing for the 'virus' were being denied hospital treatment. She said the people hit hardest were migrant workers and those in rural areas. Most businesses employed migrant workers and with everything closed there were no jobs, no income and no food. As a result millions were dying of starvation or malnutrition. All this was happening under Prime Minister Narendra Modi, a 100-percent asset of the Cult, and it emphasises yet again the scale of pure anti-human evil we are dealing with. Australia banned its people from returning home from India with penalties for trying to do so of up to five years in jail and a fine of £37,000. The manufactured 'Covid' crisis in India was being prepared to justify further fascism in the West. Obvious connections could be seen between the Indian 'vaccine' programme and increased 'cases' and this became a common theme. The Seychelles, the most per capita 'Covid vaccinated' population in the world, went back into lockdown after a 'surge of cases'.

Long ago the truly evil Monsanto agricultural biotechnology corporation with its big connections to Bill Gates devastated Indian farming with genetically-modified crops. Human rights activist Gurcharan Singh highlighted the efforts by the Indian government to complete the job by destroying the food supply to hundreds of millions with 'Covid' lockdowns. He said that 415 million people at the bottom of the disgusting caste system (still going whatever they say) were below the poverty line and struggled to feed themselves every year. Now the government was imposing lockdown at just the

time to destroy the harvest. This deliberate policy was leading to mass starvation. People may reel back at the suggestion that a government would do that, but Wetiko-controlled ‘leaders’ are capable of any level of evil. In fact what is described in India is in the process of being instigated worldwide. The food chain and food supply are being targeted at every level to cause world hunger and thus control. Bill Gates is not the biggest owner of farmland in America for no reason and destroying access to food aids both the depopulation agenda and the plan for synthetic ‘food’ already being funded into existence by Gates. Add to this the coming hyper-inflation from the suicidal creation of fake ‘money’ in response to ‘Covid’ and the breakdown of container shipping systems and you have a cocktail that can only lead one way and is meant to. The Cult plan is to crash the entire system to ‘build back better’ with the Great Reset.

'Vaccine' transmission

Reports from all over the world continue to emerge of women suffering menstrual and fertility problems after having the fake ‘vaccine’ and of the non-‘vaccinated’ having similar problems when interacting with the ‘vaccinated’. There are far too many for ‘coincidence’ to be credible. We’ve had menopausal women getting periods, others having periods stop or not stopping for weeks, passing clots, sometimes the lining of the uterus, breast irregularities, and miscarriages (which increased by 400 percent in parts of the United States). Non-‘vaccinated’ men and children have suffered blood clots and nose bleeding after interaction with the ‘vaccinated’. Babies have died from the effects of breast milk from a ‘vaccinated’ mother. Awake doctors – the small minority – speculated on the cause of non-‘vaccinated’ suffering the same effects as the ‘vaccinated’. Was it nanotechnology in the synthetic substance transmitting frequencies or was it a straight chemical bioweapon that was being transmitted between people? I am not saying that some kind of chemical transmission is not one possible answer, but the foundation of all that the Cult does is frequency and

this is fertile ground for understanding how transmission can happen. American doctor Carrie Madej, an internal medicine physician and osteopath, has been practicing for the last 20 years, teaching medical students, and she says attending different meetings where the agenda for humanity was discussed. Madej, who operates out of Georgia, did not dismiss other possible forms of transmission, but she focused on frequency in search of an explanation for transmission. She said the Moderna and Pfizer 'vaccines' contained nano-lipid particles as a key component. This was a brand new technology never before used on humanity. 'They're using a nanotechnology which is pretty much little tiny computer bits ... nanobots or hydrogel.' Inside the 'vaccines' was 'this sci-fi kind of substance' which suppressed immune checkpoints to get into the cell. I referred to this earlier as the 'Trojan horse' technique that tricks the cell into opening a gateway for the self-replicating synthetic material and while the immune system is artificially suppressed the body has no defences. Madej said the substance served many purposes including an on-demand ability to 'deliver the payload' and using the nano 'computer bits' as biosensors in the body. 'It actually has the ability to accumulate data from your body, like your breathing, your respiration, thoughts, emotions, all kinds of things.'

She said the technology obviously has the ability to operate through Wi-Fi and transmit and receive energy, messages, frequencies or impulses. 'Just imagine you're getting this new substance in you and it can react to things all around you, the 5G, your smart device, your phones.' We had something completely foreign in the human body that had never been launched large scale at a time when we were seeing 5G going into schools and hospitals (plus the Musk satellites) and she believed the 'vaccine' transmission had something to do with this: '... if these people have this inside of them ... it can act like an antenna and actually transmit it outwardly as well.' The synthetic substance produced its own voltage and so it could have that kind of effect. This fits with my own contention that the nano receiver-transmitters are designed to connect people to the

Smart Grid and break the receiver-transmitter connection to expanded consciousness. That would explain the French energy healer's experience of the disconnection of body from 'soul' with those who have had the 'vaccine'. The nanobots, self-replicating inside the body, would also transmit the synthetic frequency which could be picked up through close interaction by those who have not been 'vaccinated'. Madej speculated that perhaps it was 5G and increased levels of other radiation that was causing the symptoms directly although interestingly she said that non-'vaccinated' patients had shown improvement when they were away from the 'vaccinated' person they had interacted with. It must be remembered that you can control frequency and energy with your mind and you can consciously create energetic barriers or bubbles with the mind to stop damaging frequencies from penetrating your field. American paediatrician Dr Larry Palevsky said the 'vaccine' was not a 'vaccine' and was never designed to protect from a 'viral' infection. He called it 'a massive, brilliant propaganda of genocide' because they didn't have to inject everyone to get the result they wanted. He said the content of the jabs was able to infuse any material into the brain, heart, lungs, kidneys, liver, sperm and female productive system. 'This is genocide; this is a weapon of mass destruction.' At the same time American colleges were banning students from attending if they didn't have this life-changing and potentially life-ending 'vaccine'. Class action lawsuits must follow when the consequences of this college fascism come to light. As the book was going to press came reports about fertility effects on sperm in 'vaccinated' men which would absolutely fit with what I have been saying and hospitals continued to fill with 'vaccine' reactions. Another question is what about transmission via blood transfusions? The NHS has extended blood donation restrictions from seven days after a 'Covid vaccination' to 28 days after even a sore arm reaction.

I said in the spring of 2020 that the then touted 'Covid vaccine' would be ongoing each year like the flu jab. A year later Pfizer CEO, the appalling Albert Bourla, said people would 'likely' need a 'booster dose' of the 'vaccine' within 12 months of getting 'fully

'vaccinated' and then a yearly shot. 'Variants will play a key role', he said confirming the point. Johnson & Johnson CEO Alex Gorsky also took time out from his 'vaccine' disaster to say that people may need to be vaccinated against 'Covid-19' each year. UK Health Secretary, the psychopath Matt Hancock, said additional 'boosters' would be available in the autumn of 2021. This is the trap of the 'vaccine passport'. The public will have to accept every last 'vaccine' they introduce, including for the fake 'variants', or it would cease to be valid. The only other way in some cases would be continuous testing with a test not testing for the 'virus' and what is on the swabs constantly pushed up your noise towards the brain every time?

'Vaccines' changing behaviour

I mentioned in the body of the book how I believed we would see gathering behaviour changes in the 'vaccinated' and I am already hearing such comments from the non-'vaccinated' describing behaviour changes in friends, loved ones and work colleagues. This will only increase as the self-replicating synthetic material and nanoparticles expand in body and brain. An article in the *Guardian* in 2016 detailed research at the University of Virginia in Charlottesville which developed a new method for controlling brain circuits associated with complex animal behaviour. The method, dubbed 'magnetogenetics', involves genetically-engineering a protein called ferritin, which stores and releases iron, to create a magnetised substance – 'Magneto' – that can activate specific groups of nerve cells from a distance. This is claimed to be an advance on other methods of brain activity manipulation known as optogenetics and chemogenetics (the Cult has been developing methods of brain control for a long time). The ferritin technique is said to be non-invasive and able to activate neurons 'rapidly and reversibly'. In other words, human thought and perception. The article said that earlier studies revealed how nerve cell proteins 'activated by heat and mechanical pressure can be genetically engineered so that they become sensitive to radio waves and magnetic fields, by attaching them to an iron-storing protein called ferritin, or to inorganic

paramagnetic particles'. Sensitive to radio waves and magnetic fields? You mean like 5G, 6G and 7G? This is the human-AI Smart Grid hive mind we are talking about. The *Guardian* article said:

... the researchers injected Magneto into the striatum of freely behaving mice, a deep brain structure containing dopamine-producing neurons that are involved in reward and motivation, and then placed the animals into an apparatus split into magnetised and non-magnetised sections.

Mice expressing Magneto spent far more time in the magnetised areas than mice that did not, because activation of the protein caused the striatal neurons expressing it to release dopamine, so that the mice found being in those areas rewarding. This shows that Magneto can remotely control the firing of neurons deep within the brain, and also control complex behaviours.

Make no mistake this basic methodology will be part of the 'Covid vaccine' cocktail and using magnetics to change brain function through electromagnetic field frequency activation. The Pentagon is developing a 'Covid vaccine' using ferritin. Magnetics would explain changes in behaviour and why videos are appearing across the Internet as I write showing how magnets stick to the skin at the point of the 'vaccine' shot. Once people take these 'vaccines' anything becomes possible in terms of brain function and illness which will be blamed on 'Covid-19' and 'variants'. Magnetic field manipulation would further explain why the non-'vaccinated' are reporting the same symptoms as the 'vaccinated' they interact with and why those symptoms are reported to decrease when not in their company. Interestingly 'Magneto', a 'mutant', is a character in the Marvel Comic *X-Men* stories with the ability to manipulate magnetic fields and he believes that mutants should fight back against their human oppressors by any means necessary. The character was born Erik Lehnsherr to a Jewish family in Germany.

Cult-controlled courts

The European Court of Human Rights opened the door for mandatory 'Covid-19 vaccines' across the continent when it ruled in a Czech Republic dispute over childhood immunisation that legally

enforced vaccination could be ‘necessary in a democratic society’. The 17 judges decided that compulsory vaccinations did not breach human rights law. On the face of it the judgement was so inverted you gasp for air. If not having a vaccine infused into your body is not a human right then what is? Ah, but they said human rights law which has been specifically written to delete all human rights at the behest of the state (the Cult). Article 8 of the European Convention on Human Rights relates to the right to a private life. The crucial word here is ‘*except*’:

There shall be no interference by a public authority with the exercise of this right EXCEPT such as is in accordance with the law and is necessary in a democratic society in the interests of national security, public safety or the economic wellbeing of the country, for the prevention of disorder or crime, for the protection of health or morals, or for the protection of the rights and freedoms of others [My emphasis].

No interference *except* in accordance with the law means there *are* no ‘human rights’ *except* what EU governments decide you can have at their behest. ‘As is necessary in a democratic society’ explains that reference in the judgement and ‘in the interests of national security, public safety or the economic well-being of the country, for the prevention of disorder or crime, for the protection of health or morals, or for the protection of the rights and freedoms of others’ gives the EU a coach and horses to ride through ‘human rights’ and scatter them in all directions. The judiciary is not a check and balance on government extremism; it is a vehicle to enforce it. This judgement was almost laughably predictable when the last thing the Cult wanted was a decision that went against mandatory vaccination. Judges rule over and over again to benefit the system of which they are a part. Vaccination disputes that come before them are invariably delivered in favour of doctors and authorities representing the view of the state which owns the judiciary. Oh, yes, and we have even had calls to stop putting ‘Covid-19’ on death certificates within 28 days of a ‘positive test’ because it is claimed the practice makes the ‘vaccine’ appear not to work. They are laughing at you.

The scale of madness, inhumanity and things to come was highlighted when those not ‘vaccinated’ for ‘Covid’ were refused evacuation from the Caribbean island of St Vincent during massive volcanic eruptions. Cruise ships taking residents to the safety of another island allowed only the ‘vaccinated’ to board and the rest were left to their fate. Even in life and death situations like this we see ‘Covid’ stripping people of their most basic human instincts and the insanity is even more extreme when you think that fake ‘vaccine’-makers are not even claiming their body-manipulating concoctions stop ‘infection’ and ‘transmission’ of a ‘virus’ that doesn’t exist. St Vincent Prime Minister Ralph Gonsalves said: ‘The chief medical officer will be identifying the persons already vaccinated so that we can get them on the ship.’ Note again the power of the chief medical officer who, like Whitty in the UK, will be answering to the World Health Organization. This is the Cult network structure that has overridden politicians who ‘follow the science’ which means doing what WHO-controlled ‘medical officers’ and ‘science advisers’ tell them. Gonsalves even said that residents who were ‘vaccinated’ after the order so they could board the ships would still be refused entry due to possible side effects such as ‘wooziness in the head’. The good news is that if they were woozy enough in the head they could qualify to be prime minister of St Vincent.

Microchipping freedom

The European judgement will be used at some point to justify moves to enforce the ‘Covid’ DNA-manipulating procedure. Sandra Ro, CEO of the Global Blockchain Business Council, told a World Economic Forum event that she hoped ‘vaccine passports’ would help to ‘drive forced consent and standardisation’ of global digital identity schemes: ‘I’m hoping with the desire and global demand for some sort of vaccine passport – so that people can get travelling and working again – [it] will drive forced consent, standardisation, and frankly, cooperation across the world.’ The lady is either not very bright, or thoroughly mendacious, to use the term ‘forced consent’.

You do not ‘consent’ if you are forced – you *submit*. She was describing what the plan has been all along and that’s to enforce a digital identity on every human without which they could not function. ‘Vaccine passports’ are opening the door and are far from the end goal. A digital identity would allow you to be tracked in everything you do in cyberspace and this is the same technique used by Cult-owned China to enforce its social credit system of total control. The ultimate ‘passport’ is planned to be a microchip as my books have warned for nearly 30 years. Those nice people at the Pentagon working for the Cult-controlled Defense Advanced Research Projects Agency (DARPA) claimed in April, 2021, they have developed a microchip inserted under the skin to detect ‘asymptomatic Covid-19 infection’ before it becomes an outbreak and a ‘revolutionary filter’ that can remove the ‘virus’ from the blood when attached to a dialysis machine. The only problems with this are that the ‘virus’ does not exist and people transmitting the ‘virus’ with no symptoms is brain-numbing bullshit. This is, of course, not a ruse to get people to be microchipped for very different reasons. DARPA also said it was producing a one-stop ‘vaccine’ for the ‘virus’ and all ‘variants’. One of the most sinister organisations on Planet Earth is doing this? Better have it then. These people are insane because Wetiko that possesses them is insane.

Researchers from the Salk Institute in California announced they have created an embryo that is part human and part monkey. My books going back to the 1990s have exposed experiments in top secret underground facilities in the United States where humans are being crossed with animal and non-human ‘extraterrestrial’ species. They are now easing that long-developed capability into the public arena and there is much more to come given we are dealing with psychiatric basket cases. Talking of which – Elon Musk’s scientists at Neuralink trained a monkey to play Pong and other puzzles on a computer screen using a joystick and when the monkey made the correct move a metal tube squirted banana smoothie into his mouth which is the basic technique for training humans into unquestioning compliance. Two Neuralink chips were in the monkey’s skull and

more than 2,000 wires ‘fanned out’ into its brain. Eventually the monkey played a video game purely with its brain waves. Psychopathic narcissist Musk said the ‘breakthrough’ was a step towards putting Neuralink chips into human skulls and merging minds with artificial intelligence. *Exactly.* This man is so dark and Cult to his DNA.

World Economic Fascism (WEF)

The World Economic Forum is telling you the plan by the statements made at its many and various events. Cult-owned fascist YouTube CEO Susan Wojcicki spoke at the 2021 WEF Global Technology Governance Summit (see the name) in which 40 governments and 150 companies met to ensure ‘the responsible design and deployment of emerging technologies’. Orwellian translation: ‘Ensuring the design and deployment of long-planned technologies will advance the Cult agenda for control and censorship.’ Freedom-destroyer and Nuremberg-bound Wojcicki expressed support for tech platforms like hers to censor content that is ‘technically legal but could be harmful’. Who decides what is ‘harmful’? She does and they do. ‘Harmful’ will be whatever the Cult doesn’t want people to see and we have legislation proposed by the UK government that would censor content on the basis of ‘harm’ no matter if the information is fair, legal and provably true. Make that *especially* if it is fair, legal and provably true. Wojcicki called for a global coalition to be formed to enforce content moderation standards through automated censorship. This is a woman and mega-censor so self-deluded that she shamelessly accepted a ‘free expression’ award – *Wojcicki* – in an event sponsored by her own *YouTube*. They have no shame and no self-awareness.

You know that ‘Covid’ is a scam and Wojcicki a Cult operative when YouTube is censoring medical and scientific opinion purely on the grounds of whether it supports or opposes the Cult ‘Covid’ narrative. Florida governor Ron DeSantis compiled an expert panel with four professors of medicine from Harvard, Oxford, and Stanford Universities who spoke against forcing children and

vaccinated people to wear masks. They also said there was no proof that lockdowns reduced spread or death rates of 'Covid-19'. Cult-gofer Wojcicki and her YouTube deleted the panel video 'because it included content that contradicts the consensus of local and global health authorities regarding the efficacy of masks to prevent the spread of Covid-19'. This 'consensus' refers to what the Cult tells the World Health Organization to say and the WHO tells 'local health authorities' to do. Wojcicki knows this, of course. The panellists pointed out that censorship of scientific debate was responsible for deaths from many causes, but Wojcicki couldn't care less. She would not dare go against what she is told and as a disgrace to humanity she wouldn't want to anyway. The UK government is seeking to pass a fascist 'Online Safety Bill' to specifically target with massive fines and other means non-censored video and social media platforms to make them censor 'lawful but harmful' content like the Cult-owned Facebook, Twitter, Google and YouTube. What is 'lawful but harmful' would be decided by the fascist Blair-created Ofcom.

Another WEF obsession is a cyber-attack on the financial system and this is clearly what the Cult has planned to take down the bank accounts of everyone – except theirs. Those that think they have enough money for the Cult agenda not to matter to them have got a big lesson coming if they continue to ignore what is staring them in the face. The World Economic Forum, funded by Gates and fronted by Klaus Schwab, announced it would be running a 'simulation' with the Russian government and global banks of just such an attack called Cyber Polygon 2021. What they simulate – as with the 'Covid' Event 201 – they plan to instigate. The WEF is involved in a project with the Cult-owned Carnegie Endowment for International Peace called the WEF-Carnegie Cyber Policy Initiative which seeks to merge Wall Street banks, 'regulators' (I love it) and intelligence agencies to 'prevent' (arrange and allow) a cyber-attack that would bring down the global financial system as long planned by those that control the WEF and the Carnegie operation. The Carnegie Endowment for International Peace sent an instruction to First World

War US President Woodrow Wilson not to let the war end before society had been irreversibly transformed.

The Wuhan lab diversion

As I close, the Cult-controlled authorities and lapdog media are systematically pushing ‘the virus was released from the Wuhan lab’ narrative. There are two versions – it happened by accident and it happened on purpose. Both are nonsense. The perceived existence of the never-shown-to-exist ‘virus’ is vital to sell the impression that there is actually an infective agent to deal with and to allow the endless potential for terrifying the population with ‘variants’ of a ‘virus’ that does not exist. The authorities at the time of writing are going with the ‘by accident’ while the alternative media is promoting the ‘on purpose’. Cable news host Tucker Carlson who has questioned aspects of lockdown and ‘vaccine’ compulsion has bought the Wuhan lab story. ‘Everyone now agrees’ he said. Well, I don’t and many others don’t and the question is *why* does the system and its media suddenly ‘agree’? When the media moves as one unit with a narrative it is always a lie – witness the hour by hour mendacity of the ‘Covid’ era. Why would this Cult-owned combination which has unleashed lies like machine gun fire suddenly ‘agree’ to tell the truth??

Much of the alternative media is buying the lie because it fits the conspiracy narrative, but it’s the *wrong* conspiracy. The real conspiracy is that *there is no virus* and that is what the Cult is desperate to hide. The idea that the ‘virus’ was released by accident is ludicrous when the whole ‘Covid’ hoax was clearly long-planned and waiting to be played out as it was so fast in accordance with the Rockefeller document and Event 201. So they prepared everything in detail over decades and then sat around strumming their fingers waiting for an ‘accidental’ release from a bio-lab? *What??* It’s crazy. Then there’s the ‘on purpose’ claim. You want to circulate a ‘deadly virus’ and hide the fact that you’ve done so and you release it down the street from the highest-level bio-lab in China? I repeat – *What??*

You would release it far from that lab to stop any association being made. But, no, we'll do it in a place where the connection was certain to be made. Why would you need to scam 'cases' and 'deaths' and pay hospitals to diagnose 'Covid-19' if you had a real 'virus'? What are sections of the alternative media doing believing this crap? Where were all the mass deaths in Wuhan from a 'deadly pathogen' when the recovery to normal life after the initial propaganda was dramatic in speed? Why isn't the 'deadly pathogen' now circulating all over China with bodies in the street? Once again we have the technique of tell them what they want to hear and they will likely believe it. The alternative media has its 'conspiracy' and with Carlson it fits with his 'China is the danger' narrative over years. China *is* a danger as a global Cult operations centre, but not for this reason. The Wuhan lab story also has the potential to instigate conflict with China when at some stage the plan is to trigger a Problem-Reaction-Solution confrontation with the West. Question everything – *everything* – and especially when the media agrees on a common party line.

Third wave ... fourth wave ... fifth wave ...

As the book went into production the world was being set up for more lockdowns and a 'third wave' supported by invented 'variants' that were increasing all the time and will continue to do so in public statements and computer programs, but not in reality. India became the new Italy in the 'Covid' propaganda campaign and we were told to be frightened of the new 'Indian strain'. Somehow I couldn't find it within myself to do so. A document produced for the UK government entitled 'Summary of further modelling of easing of restrictions – Roadmap Step 2' declared that a third wave was inevitable (of course when it's in the script) and it would be the fault of children and those who refuse the health-destroying fake 'Covid vaccine'. One of the computer models involved came from the Cult-owned *Imperial College* and the other from Warwick University which I wouldn't trust to tell me the date in a calendar factory. The document states that both models presumed extremely high uptake

of the ‘Covid vaccines’ and didn’t allow for ‘variants’. The document states: ‘The resurgence is a result of some people (mostly children) being ineligible for vaccination; others choosing not to receive the vaccine; and others being vaccinated but not perfectly protected.’ The mendacity takes the breath away. Okay, blame those with a brain who won’t take the DNA-modifying shots and put more pressure on children to have it as ‘trials’ were underway involving children as young as six months with parents who give insanity a bad name. Massive pressure is being put on the young to have the fake ‘vaccine’ and child age consent limits have been systematically lowered around the world to stop parents intervening. Most extraordinary about the document was its claim that the ‘third wave’ would be driven by ‘the resurgence in both hospitalisations and deaths … dominated by *those that have received two doses of the vaccine*, comprising around 60-70% of the wave respectively’. The predicted peak of the ‘third wave’ suggested 300 deaths per day with 250 of them *fully ‘vaccinated’ people*. How many more lies do acquiescers need to be told before they see the obvious? Those who took the jab to ‘protect themselves’ are projected to be those who mostly get sick and die? So what’s in the ‘vaccine’? The document went on:

It is possible that a summer of low prevalence could be followed by substantial increases in incidence over the following autumn and winter. Low prevalence in late summer should not be taken as an indication that SARS-CoV-2 has retreated or that the population has high enough levels of immunity to prevent another wave.

They are telling you the script and while many British people believed ‘Covid’ restrictions would end in the summer of 2021 the government was preparing for them to be ongoing. Authorities were awarding contracts for ‘Covid marshals’ to police the restrictions with contracts starting in July, 2021, and going through to January 31st, 2022, and the government was advertising for ‘Media Buying Services’ to secure media propaganda slots worth a potential £320 million for ‘Covid-19 campaigns’ with a contract not ending until March, 2022. The recipient – via a list of other front companies – was reported to be American media marketing giant Omnicom Group

Inc. While money is no object for ‘Covid’ the UK waiting list for all other treatment – including life-threatening conditions – passed 4.5 million. Meantime the Cult is seeking to control all official ‘inquiries’ to block revelations about what has really been happening and why. It must not be allowed to – we need Nuremberg jury trials in every country. The cover-up doesn’t get more obvious than appointing ultra-Zionist professor Philip Zelikow to oversee two dozen US virologists, public health officials, clinicians, former government officials and four American ‘charitable foundations’ to ‘learn the lessons’ of the ‘Covid’ debacle. The personnel will be those that created and perpetuated the ‘Covid’ lies while Zelikow is the former executive director of the 9/11 Commission who ensured that the truth about those attacks never came out and produced a report that must be among the most mendacious and manipulative documents ever written – see *The Trigger* for the detailed exposure of the almost unimaginable 9/11 story in which Sabbatians can be found at every level.

Passive no more

People are increasingly challenging the authorities with amazing numbers of people taking to the streets in London well beyond the ability of the Face-Nappies to stop them. Instead the Nappies choose situations away from the mass crowds to target, intimidate, and seek to promote the impression of ‘violent protestors’. One such incident happened in London’s Hyde Park. Hundreds of thousands walking through the streets in protest against ‘Covid’ fascism were ignored by the Cult-owned BBC and most of the rest of the mainstream media, but they delighted in reporting how police were injured in ‘clashes with protestors’. The truth was that a group of people gathered in Hyde Park at the end of one march when most had gone home and they were peacefully having a good time with music and chat. Face-Nappies who couldn’t deal with the full-march crowd then waded in with their batons and got more than they bargained for. Instead of just standing for this criminal brutality the crowd used their numerical superiority to push the Face-Nappies out of the

park. Eventually the Nappies turned and ran. Unfortunately two or three idiots in the crowd threw drink cans striking two officers which gave the media and the government the image they wanted to discredit the 99.9999 percent who were peaceful. The idiots walked straight into the trap and we must always be aware of potential agent provocateurs used by the authorities to discredit their targets.

This response from the crowd – the can people apart – must be a turning point when the public no longer stand by while the innocent are arrested and brutally attacked by the Face-Nappies. That doesn't mean to be violent, that's the last thing we need. We'll leave the violence to the Face-Nappies and government. But it does mean that when the Face-Nappies use violence against peaceful people the numerical superiority is employed to stop them and make citizen's arrests or Common Law arrests for a breach of the peace. The time for being passive in the face of fascism is over.

We are the many, they are the few, and we need to make that count before there is no freedom left and our children and grandchildren face an ongoing fascist nightmare.

COME ON PEOPLE – IT'S TIME.

One final thought ...

The power of love
A force from above
Cleaning my soul
Flame on burn desire
Love with tongues of fire
Purge the soul
Make love your goal

I'll protect you from the hooded claw
Keep the vampires from your door
When the chips are down I'll be around
With my undying, death-defying
Love for you

Envy will hurt itself
Let yourself be beautiful
Sparkling love, flowers
And pearls and pretty girls
Love is like an energy
Rushin' rushin' inside of me

This time we go sublime
Lovers entwine, divine, divine,
Love is danger, love is pleasure
Love is pure – the only treasure

I'm so in love with you
Purge the soul
Make love your goal

The power of love
A force from above
Cleaning my soul
The power of love
A force from above
A sky-scraping dove

Flame on burn desire
Love with tongues of fire
Purge the soul
Make love your goal

Frankie Goes To Hollywood

APPENDIX

Cowan-Kaufman-Morell Statement on Virus Isolation (SOVI)

Isolation: The action of isolating; the fact or condition of being isolated or standing alone; separation from other things or persons; solitariness

Oxford English Dictionary

The controversy over whether the SARS-CoV-2 virus has ever been isolated or purified continues. However, using the above definition, common sense, the laws of logic and the dictates of science, any unbiased person must come to the conclusion that the SARS-CoV-2 virus has never been isolated or purified. As a result, no confirmation of the virus' existence can be found. The logical, common sense, and scientific consequences of this fact are:

- the structure and composition of something not shown to exist can't be known, including the presence, structure, and function of any hypothetical spike or other proteins;
- the genetic sequence of something that has never been found can't be known;
- "variants" of something that hasn't been shown to exist can't be known;
- it's impossible to demonstrate that SARS-CoV-2 causes a disease called Covid-19.

In as concise terms as possible, here's the proper way to isolate, characterize and demonstrate a new virus. First, one takes samples (blood, sputum, secretions) from many people (e.g. 500) with symptoms which are unique and specific enough to characterize an illness. Without mixing these samples with ANY tissue or products that also contain genetic material, the virologist macerates, filters and ultracentrifuges i.e. *purifies* the specimen. This common virology technique, done for decades to isolate bacteriophages¹ and so-called giant viruses in every virology lab, then allows the virologist to demonstrate with electron microscopy thousands of identically sized and shaped particles. These particles are the isolated and purified virus.

These identical particles are then checked for uniformity by physical and/or microscopic techniques. Once the purity is determined, the particles may be further characterized. This would include examining the structure, morphology, and chemical composition of the particles. Next, their genetic makeup is characterized by extracting the genetic material directly from the purified particles and using genetic-sequencing techniques, such as Sanger sequencing, that have also been around for decades. Then one does an analysis to confirm that these uniform particles are exogenous (outside) in origin as a virus is conceptualized to be, and not the normal breakdown products of dead and dying tissues.² (As of May 2020, we know that virologists have no way to determine whether the particles they're seeing are viruses or just normal breakdown products of dead and dying tissues.)³

1 Isolation, characterization and analysis of bacteriophages from the haloalkaline lake Elmenteita, KenyaJuliah Khayeli Akhwale et al, PLOS One, Published: April 25, 2019.
<https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0215734> – accessed 2/15/21

2 "Extracellular Vesicles Derived From Apoptotic Cells: An Essential Link Between Death and Regeneration," Maojiao Li et al, Frontiers in Cell and Developmental Biology, 2020 October 2.
<https://www.frontiersin.org/articles/10.3389/fcell.2020.573511/full> – accessed 2/15/21

3 "The Role of Extracellular Vesicles as Allies of HIV, HCV and SARS Viruses," Flavia Giannessi, et al, *Viruses*, 2020 May

If we have come this far then we have fully isolated, characterized, and genetically sequenced an exogenous virus particle. However, we still have to show it is causally related to a disease. This is carried out by exposing a group of healthy subjects (animals are usually used) to this isolated, purified virus in the manner in which the disease is thought to be transmitted. If the animals get sick with the same disease, as confirmed by clinical and autopsy findings, one has now shown that the virus actually causes a disease. This demonstrates infectivity and transmission of an infectious agent.

None of these steps has even been attempted with the SARS-CoV-2 virus, nor have all these steps been successfully performed for any so-called pathogenic virus. Our research indicates that a single study showing these steps does not exist in the medical literature.

Instead, since 1954, virologists have taken unpurified samples from a relatively few people, often less than ten, with a similar disease. They then minimally process this sample and inoculate this unpurified sample onto tissue culture containing usually four to six other types of material – all of which contain identical genetic material as to what is called a “virus.” The tissue culture is starved and poisoned and naturally disintegrates into many types of particles, some of which contain genetic material. Against all common sense, logic, use of the English language and scientific integrity, this process is called “virus isolation.” This brew containing fragments of genetic material from many sources is then subjected to genetic analysis, which then creates in a computer-simulation process the alleged sequence of the alleged virus, a so-called *in silico* genome. At no time is an actual virus confirmed by electron microscopy. At no time is a genome extracted and sequenced from an actual virus. This is scientific fraud.

The observation that the unpurified specimen — inoculated onto tissue culture along with toxic antibiotics, bovine fetal tissue, amniotic fluid and other tissues — destroys the kidney tissue onto which it is inoculated is given as evidence of the virus' existence and pathogenicity. This is scientific fraud.

From now on, when anyone gives you a paper that suggests the SARS-CoV-2 virus has been isolated, please check the methods sections. If the researchers used Vero cells or any other culture method, you know that their process was not isolation. You will hear the following excuses for why actual isolation isn't done:

1. There were not enough virus particles found in samples from patients to analyze.
2. Viruses are intracellular parasites; they can't be found outside the cell in this manner.

If No. 1 is correct, and we can't find the virus in the sputum of sick people, then on what evidence do we think the virus is dangerous or even lethal? If No. 2 is correct, then how is the virus spread from person to person? We are told it emerges from the cell to infect others. Then why isn't it possible to find it?

Finally, questioning these virology techniques and conclusions is not some distraction or divisive issue. Shining the light on this truth is essential to stop this terrible fraud that humanity is confronting. For, as we now know, if the virus has never been isolated, sequenced or shown to cause illness, if the virus is imaginary, then why are we wearing masks, social distancing and putting the whole world into prison?

Finally, if pathogenic viruses don't exist, then what is going into those injectable devices erroneously called "vaccines," and what is their purpose? This scientific question is the most urgent and relevant one of our time.

We are correct. The SARS-CoV2 virus does not exist.

Sally Fallon Morell, MA

Dr. Thomas Cowan, MD

Dr. Andrew Kaufman, MD

Bibliography

- Alinsky, Saul:** *Rules for Radicals* (Vintage, 1989)
- Antelman, Rabbi Marvin:** *To Eliminate the Opiate* (Zahavia, 1974)
- Bastardi, Joe:** *The Climate Chronicles* (Relentless Thunder Press, 2018)
- Cowan, Tom:** *Human Heart, Cosmic Heart* (Chelsea Green Publishing, 2016)
- Cowan, Tom, and Fallon Morell, Sally:** *The Contagion Myth* (Skyhorse Publishing, 2020)
- Forbes, Jack D:** *Columbus And Other Cannibals – The Wetiko Disease of Exploitation, Imperialism, and Terrorism* (Seven Stories Press, 2008 – originally published in 1979)
- Gates, Bill:** *How to Avoid a Climate Disaster: The Solutions We Have and the Breakthroughs We Need* (Allen Lane, 2021)
- Huxley, Aldous:** *Brave New World* (Chatto & Windus, 1932)
- Köhnlein, Dr Claus, and Engelbrecht, Torsten:** *Virus Mania* (emu-Vertag, Lahnstein, 2020)
- Lanza, Robert, and Berman, Bob:** *Biocentrism* (BenBella Books, 2010)
- Lash, John Lamb:** *Not In His Image* (Chelsea Green Publishing, 2006)
- Lester, Dawn, and Parker, David:** *What Really Makes You Ill – Why everything you thought you knew about disease is wrong* (Independently Published, 2019)
- Levy, Paul:** *Dispelling Wetiko, Breaking the Spell of Evil* (North Atlantic Books, 2013)
- Marx, Karl:** *A World Without Jews* (Philosophical Library, first edition, 1959)
- Mullis, Kary:** *Dancing Naked in the Mine Field* (Bloomsbury, 1999)
- O'Brien, Cathy:** *Trance-Formation of America* (Reality Marketing, 1995)
- Scholem, Gershon:** *The Messianic Idea in Judaism* (Schocken Books, 1994)
- Schwab, Klaus, and Davis, Nicholas:** *Shaping the Future of the Fourth Industrial Revolution: A guide to building a better world* (Penguin Books, 2018)
- Schwab, Klaus:** *The Great Reset* (Agentur Schweiz, 2020)
- Sunstein, Cass and Thaler, Richard:** *Nudge: Improving Decisions About Health, Wealth, and Happiness* (Penguin, 2009)
- Swan, Shanna:** *Count Down: How Our Modern World Is Threatening Sperm Counts, Altering Male and Female Reproductive Development and Imperiling the Future of the Human Race* (Scribner, 2021)
- Tegmark, Max:** *Our Mathematical Universe: My Quest for the Ultimate Nature of Reality* (Penguin, 2015)
- Velikovsky, Immanuel:** *Worlds in Collision* (Paradigma, 2009)

Wilton, Robert: *The Last Days of the Romanovs* (Blurb, 2018, first published 1920)

Index

A

abusive relationships

blaming themselves, abused as [ref1](#)
children [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)
conspiracy theories [ref1](#)
domestic abuse [ref1](#), [ref2](#)
economic abuse and dependency [ref1](#)
isolation [ref1](#)
physical abuse [ref1](#)
psychological abuse [ref1](#)
signs of abuse [ref1](#)

addiction

alcoholism [ref1](#)
frequencies [ref1](#)
substance abuse [ref1](#), [ref2](#)
technology [ref1](#), [ref2](#), [ref3](#)

Adelson, Sheldon [ref1](#), [ref2](#), [ref3](#)

Agenda 21/Agenda 2030 (UN) [ref1](#), [ref2](#), [ref3](#), [ref4](#)

AIDs/HIV [ref1](#)

causal link between HIV and AIDs [ref1](#), [ref2](#)
retroviruses [ref1](#)
testing [ref1](#), [ref2](#)
trial-run for Covid-19, as [ref1](#), [ref2](#)
aliens/extraterrestrials [ref1](#), [ref2](#)
aluminium [ref1](#)
Amazon [ref1](#), [ref2](#), [ref3](#)

amplification cycles [ref1](#), [ref2](#)
anaphylactic shock [ref1](#), [ref2](#), [ref3](#), [ref4](#)
animals [ref1](#), [ref2](#), [ref3](#)
antibodies [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Antifa [ref1](#), [ref2](#), [ref3](#), [ref4](#)
antigens [ref1](#), [ref2](#)
anti-Semitism [ref1](#), [ref2](#), [ref3](#)
Archons [ref1](#), [ref2](#)
 consciousness [ref1](#), [ref2](#), [ref3](#)
 energy [ref1](#), [ref2](#), [ref3](#)
 ennoia [ref1](#)
 genetic manipulation [ref1](#), [ref2](#)
 inversion [ref1](#), [ref2](#), [ref3](#)
 lockdowns [ref1](#)
 money [ref1](#)
 radiation [ref1](#)
 religion [ref1](#), [ref2](#)
 technology [ref1](#), [ref2](#), [ref3](#)
 Wetiko factor [ref1](#), [ref2](#), [ref3](#), [ref4](#)
artificial intelligence (AI) [ref1](#)
army made up of robots [ref1](#), [ref2](#)
 Human 2.0 [ref1](#), [ref2](#)
 Internet [ref1](#)
 MHRA [ref1](#)
 Morgellons fibres [ref1](#), [ref2](#)
 Smart Grid [ref1](#)
 Wetiko factor [ref1](#)
asymptomatic, Covid-19 as [ref1](#), [ref2](#), [ref3](#)
aviation industry [ref1](#)

B

banking, finance and money [ref1](#), [ref2](#), [ref3](#)

2008 crisis [ref1](#), [ref2](#)

boom and bust [ref1](#)

cashless digital money systems [ref1](#)

central banks [ref1](#)

credit [ref1](#)

digital currency [ref1](#)

fractional reserve lending [ref1](#)

Great Reset [ref1](#)

guaranteed income [ref1](#), [ref2](#), [ref3](#)

Human 2.0 [ref1](#)

incomes, destruction of [ref1](#), [ref2](#)

interest [ref1](#)

one per cent [ref1](#), [ref2](#)

scams [ref1](#)

BBC [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

Becker-Phelps, Leslie [ref1](#)

Behavioural Insights Team (BIT) (Nudge Unit) [ref1](#), [ref2](#), [ref3](#)

behavioural scientists and psychologists, advice from [ref1](#), [ref2](#)

Bezos, Jeff [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Biden, Hunter [ref1](#)

Biden, Joe [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#), [ref11](#),
[ref12](#), [ref13](#), [ref14](#), [ref15](#), [ref16](#), [ref17](#)

Big Pharma

cholesterol [ref1](#)

health professionals [ref1](#), [ref2](#)

immunity from prosecution in US [ref1](#)

vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

Wetiko factor [ref1](#), [ref2](#)

WHO [ref1](#), [ref2](#), [ref3](#)

Bill and Melinda Gates Foundation [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#),
[ref7](#)

billionaires [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#) [ref10](#), [ref11](#)
bird flu (H5N1) [ref1](#)
Black Lives Matter (BLM) [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Blair, Tony [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
Brin, Sergei [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
British Empire [ref1](#)
Bush, George HW [ref1](#), [ref2](#)
Bush, George W [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Byrd, Robert [ref1](#)

C

Canada

Global Cult [ref1](#)
hate speech [ref1](#)
internment [ref1](#)
masks [ref1](#)
old people [ref1](#)
SARS-COV-2 [ref1](#)
satellites [ref1](#)
vaccines [ref1](#)
wearable technology [ref1](#)

Capitol Hill riot [ref1](#), [ref2](#)
agents provocateur [ref1](#)
Antifa [ref1](#)
Black Lives Matter (BLM) [ref1](#), [ref2](#)
QAnon [ref1](#)
security precautions, lack of [ref1](#), [ref2](#), [ref3](#)

carbon dioxide [ref1](#), [ref2](#)
care homes, deaths in [ref1](#), [ref2](#)
cashless digital money systems [ref1](#)
censorship [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

fact-checkers [ref1](#)
masks [ref1](#)
media [ref1](#), [ref2](#)
private messages [ref1](#)
social media [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
transgender persons [ref1](#)
vaccines [ref1](#), [ref2](#), [ref3](#)
Wokeness [ref1](#)

Centers for Disease Control (CDC) (United States) [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#), [ref11](#), [ref12](#), [ref13](#)

centralisation [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

chakras [ref1](#)

change agents [ref1](#), [ref2](#), [ref3](#)

chemtrails [ref1](#), [ref2](#), [ref3](#)

chief medical officers and scientific advisers [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

children *see also young people*

abuse [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)

care, taken into [ref1](#), [ref2](#), [ref3](#)

education [ref1](#), [ref2](#), [ref3](#), [ref4](#)

energy [ref1](#)

family courts [ref1](#)

hand sanitisers [ref1](#)

human sacrifice [ref1](#)

lockdowns [ref1](#), [ref2](#), [ref3](#)

masks [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

mental health [ref1](#)

old people [ref1](#)

parents, replacement of [ref1](#), [ref2](#)

Psyop (psychological operation), Covid as a [ref1](#), [ref2](#)

reframing [ref1](#)

smartphone addiction [ref1](#)

social distancing and isolation [ref1](#)
social media [ref1](#)
transgender persons [ref1](#), [ref2](#)
United States [ref1](#)
vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)
Wetiko factor [ref1](#)

China [ref1](#), [ref2](#), [ref3](#), [ref4](#)
anal swab tests [ref1](#)
Chinese Revolution [ref1](#), [ref2](#), [ref3](#)
digital currency [ref1](#)
Global Cult [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)
guaranteed income [ref1](#)
Imperial College [ref1](#)
Israel [ref1](#)
lockdown [ref1](#), [ref2](#)
masculinity crisis [ref1](#)
masks [ref1](#)
media [ref1](#)
origins of virus in China [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
pollution causing respiratory diseases [ref1](#)
Sabbatians [ref1](#), [ref2](#)
Smart Grid [ref1](#), [ref2](#)
social credit system [ref1](#)
testing [ref1](#), [ref2](#)
United States [ref1](#), [ref2](#)
vaccines [ref1](#), [ref2](#)
Wetiko factor [ref1](#)
wet market conspiracy [ref1](#)
Wuhan [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

cholesterol [ref1](#), [ref2](#)

Christianity [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
criticism [ref1](#)
cross, inversion of the [ref1](#)

Nag Hammadi texts [ref1](#), [ref2](#), [ref3](#)
Roman Catholic Church [ref1](#), [ref2](#)
Sabbatians [ref1](#), [ref2](#)
Satan [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Wokeness [ref1](#)

class [ref1](#), [ref2](#)

climate change hoax [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Agenda 21/Agenda 2030 [ref1](#), [ref2](#), [ref3](#)

carbon dioxide [ref1](#), [ref2](#)

Club of Rome [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

fear [ref1](#)

funding [ref1](#)

Global Cult [ref1](#)

green new deals [ref1](#)

green parties [ref1](#)

inversion [ref1](#)

perception, control of [ref1](#)

PICC [ref1](#)

reframing [ref1](#)

temperature, increases in [ref1](#)

United Nations [ref1](#), [ref2](#)

Wikipedia [ref1](#)

Wokeness [ref1](#), [ref2](#)

Clinton, Bill [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

Clinton, Hillary [ref1](#), [ref2](#), [ref3](#)

the cloud [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

Club of Rome and climate change hoax [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

cognitive therapy [ref1](#)

Cohn, Roy [ref1](#)

Common Law [ref1](#)

Admiralty Law [ref1](#)

arrests [ref1](#), [ref2](#)

contractual law, Statute Law as [ref1](#)
corporate entities, people as [ref1](#)
legalese [ref1](#)
sea, law of the [ref1](#)
Statute Law [ref1](#)

Common Purpose leadership programme [ref1](#), [ref2](#)
communism [ref1](#), [ref2](#)
co-morbidities [ref1](#)
computer-generated virus,
Covid-19 as [ref1](#), [ref2](#), [ref3](#)
computer models [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
connections [ref1](#), [ref2](#), [ref3](#), [ref4](#)
consciousness [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Archons [ref1](#), [ref2](#), [ref3](#)
expanded [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
experience [ref1](#)
heart [ref1](#)
infinity [ref1](#), [ref2](#)
religion [ref1](#), [ref2](#)
self-identity [ref1](#)
simulation thesis [ref1](#)
vaccines [ref1](#)
Wetiko factor [ref1](#), [ref2](#)

conspiracy theorists [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
contradictory rules [ref1](#)
contrails [ref1](#)
Corman-Drosten test [ref1](#), [ref2](#), [ref3](#), [ref4](#)
countermimicry [ref1](#), [ref2](#), [ref3](#)
Covid-19 vaccines *see* vaccines
Covidiots [ref1](#), [ref2](#)
Cowan, Tom [ref1](#), [ref2](#), [ref3](#), [ref4](#)
crimes against humanity [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

cyber-operations [ref1](#)

cyberwarfare [ref1](#)

D

DARPA (Defense Advanced Research Projects Agency) [ref1](#)

deaths

care homes [ref1](#)

certificates [ref1](#), [ref2](#), [ref3](#), [ref4](#)

mortality rate [ref1](#)

post-mortems/autopsies [ref1](#)

recording [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

deceit

pyramid of deceit [ref1](#), [ref2](#)

sequence of deceit [ref1](#)

decoding [ref1](#), [ref2](#), [ref3](#)

dehumanisation [ref1](#), [ref2](#), [ref3](#)

Delphi technique [ref1](#)

democracy [ref1](#)

dependency [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Descartes, René [ref1](#)

DNA

numbers [ref1](#)

vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)

DNR (do not resuscitate)

orders [ref1](#)

domestic abuse [ref1](#), [ref2](#)

downgrading of Covid-19 [ref1](#)

Drosten, Christian [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

Duesberg, Peter [ref1](#), [ref2](#)

E

economic abuse [ref1](#)

Edmunds, John [ref1](#), [ref2](#)

education [ref1](#), [ref2](#), [ref3](#), [ref4](#)

electromagnetic spectrum [ref1](#), [ref2](#)

Enders, John [ref1](#)

energy

Archons [ref1](#), [ref2](#), [ref3](#)

children and young people [ref1](#)

consciousness [ref1](#)

decoding [ref1](#)

frequencies [ref1](#), [ref2](#), [ref3](#), [ref4](#)

heart [ref1](#)

human energy field [ref1](#)

source, humans as an energy [ref1](#), [ref2](#)

vaccines [ref1](#)

viruses [ref1](#)

ennoia [ref1](#)

Epstein, Jeffrey [ref1](#), [ref2](#)

eternal 'I' [ref1](#), [ref2](#)

ethylene oxide [ref1](#)

European Union [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Event [ref1](#) and Bill Gates [ref2](#)

exosomes, Covid-19 as natural defence mechanism called [ref1](#)

experience [ref1](#), [ref2](#)

Extinction Rebellion [ref1](#), [ref2](#)

F

Facebook

addiction [ref1](#), 448–50

Facebook

Archons [ref1](#)
censorship [ref1](#), [ref2](#), [ref3](#)
hate speech [ref1](#)
monopoly, as [ref1](#)
private messages, censorship of [ref1](#)
Sabbatians [ref1](#)
United States election fraud [ref1](#)
vaccines [ref1](#)
Wetiko factor [ref1](#)

fact-checkers [ref1](#)

Fauci, Anthony [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#),
[ref11](#), [ref12](#)

fear [ref1](#), [ref2](#), [ref3](#), [ref4](#)
climate change [ref1](#)
computer models [ref1](#)
conspiracy theories [ref1](#)
empty hospitals [ref1](#)
Italy [ref1](#), [ref2](#), [ref3](#)
lockdowns [ref1](#), [ref2](#), [ref3](#), [ref4](#)
masks [ref1](#), [ref2](#)
media [ref1](#), [ref2](#)
medical staff [ref1](#)
Psyop (psychological operation), Covid as a [ref1](#)
Wetiko factor [ref1](#), [ref2](#)

female infertility [ref1](#)

Fermi Paradox [ref1](#)

Ferguson, Neil [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

fertility, decline in [ref1](#)

The Field [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

finance *see banking, finance and money*

five-senses [ref1](#), [ref2](#)
Archons [ref1](#), [ref2](#), [ref3](#)

censorship [ref1](#)
consciousness, expansion of [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
decoding [ref1](#)
education [ref1](#), [ref2](#)
the Field [ref1](#), [ref2](#)
God, personification of [ref1](#)
infinity [ref1](#), [ref2](#)
media [ref1](#)
paranormal [ref1](#)
perceptual programming [ref1](#), [ref2](#)
Phantom Self [ref1](#)
pneuma not nous, using [ref1](#)
reincarnation [ref1](#)
self-identity [ref1](#)
Wetiko factor [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
5G [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)
Floyd, George and protests, killing of [ref1](#)
flu, re-labelling of [ref1](#), [ref2](#), [ref3](#)
food and water, control of [ref1](#), [ref2](#)
Freemasons [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
Frei, Rosemary [ref1](#)
frequencies
addictions [ref1](#)
Archons [ref1](#), [ref2](#), [ref3](#)
awareness [ref1](#)
chanting and mantras [ref1](#)
consciousness [ref1](#)
decoding [ref1](#), [ref2](#)
education [ref1](#)
electromagnetic (EMF) frequencies [ref1](#)
energy [ref1](#), [ref2](#), [ref3](#), [ref4](#)
fear [ref1](#)

the Field [ref1](#), [ref2](#) 5G [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)
five-senses [ref1](#), [ref2](#)
ghosts [ref1](#)
Gnostics [ref1](#)
hive-minds [ref1](#)
human, meaning of [ref1](#)
light [ref1](#), [ref2](#)
love [ref1](#), [ref2](#)
magnetism [ref1](#)
perception [ref1](#)
reality [ref1](#), [ref2](#), [ref3](#)
simulation [ref1](#)
terror [ref1](#)
vaccines [ref1](#)
Wetiko [ref1](#), [ref2](#), [ref3](#)
Fuellmich, Reiner [ref1](#), [ref2](#), [ref3](#)
furlough/rescue payments [ref1](#)

G

Gallo, Robert [ref1](#), [ref2](#), [ref3](#)

Gates, Bill

Archons [ref1](#), [ref2](#), [ref3](#)
climate change [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Daily Pass tracking system [ref1](#)
Epstein [ref1](#)
fascism [ref1](#)
five senses [ref1](#)
GAVI [ref1](#)
Great Reset [ref1](#)
GSK [ref1](#)
Imperial College [ref1](#), [ref2](#)
Johns Hopkins University [ref1](#), [ref2](#), [ref3](#)

lockdowns [ref1](#), [ref2](#)

masks [ref1](#)

Nuremberg trial, proposal for [ref1](#), [ref2](#)

Rockefellers [ref1](#), [ref2](#)

social distancing and isolation [ref1](#)

Sun, dimming the [ref1](#)

synthetic meat [ref1](#), [ref2](#)

vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

Wellcome Trust [ref1](#)

Wetiko factor [ref1](#), [ref2](#), [ref3](#)

WHO [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)

Wokeness [ref1](#)

World Economic Forum [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Gates, Melinda [ref1](#), [ref2](#), [ref3](#)

GAVI vaccine alliance [ref1](#)

genetics, manipulation of [ref1](#), [ref2](#), [ref3](#)

Germany [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#) *see also Nazi Germany*

Global Cult [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

anti-human, why Global Cult is [ref1](#)

Black Lives Matter (BLM) [ref1](#), [ref2](#), [ref3](#), [ref4](#)

China [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)

climate change hoax [ref1](#)

contradictory rules [ref1](#)

Covid-19 [ref1](#), [ref2](#), [ref3](#)

fascism [ref1](#)

geographical origins [ref1](#)

immigration [ref1](#)

Internet [ref1](#)

mainstream media [ref1](#), [ref2](#)

masks [ref1](#), [ref2](#)

monarchy [ref1](#)

non-human dimension [ref1](#)

perception [ref1](#)
political parties [ref1](#), [ref2](#)
pyramidal hierarchy [ref1](#), [ref2](#), [ref3](#)
reframing [ref1](#)
Sabbantian-Frankism [ref1](#), [ref2](#)
science, manipulation of [ref1](#)
spider and the web [ref1](#)
transgender persons [ref1](#)
vaccines [ref1](#)
who controls the Cult [ref1](#)
Wokeness [ref1](#), [ref2](#), [ref3](#), [ref4](#)

globalisation [ref1](#), [ref2](#)

Gnostics [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Google [ref1](#), [ref2](#), [ref3](#), [ref4](#)

government

- behavioural scientists and psychologists, advice from [ref1](#), [ref2](#)
- definition [ref1](#)
- Joint Biosecurity Centre (JBC) [ref1](#)
- people, abusive relationship with [ref1](#)

Great Reset [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

- fascism [ref1](#), [ref2](#), [ref3](#)
- financial system [ref1](#)
- Human 2.0 [ref1](#)
- water and food, control of [ref1](#)

green parties [ref1](#)

Griesz-Brisson, Margarite [ref1](#)

guaranteed income [ref1](#), [ref2](#), [ref3](#)

H

Hancock, Matt [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

hand sanitisers [ref1](#)

heart [ref1](#), [ref2](#)

hive-minds/groupthink ref1, ref2, ref3

holographs ref1, ref2, ref3, ref4

hospitals, empty ref1

human, meaning of ref1

Human 2.0 ref1

addiction to technology ref1

artificial intelligence (AI) ref1, ref2

elimination of Human 1.0 ref1

fertility, decline in ref1

Great Reset ref1

implantables ref1

money ref1

mRNA ref1

nanotechnology ref1

parents, replacement of ref1, ref2

Smart Grid, connection to ref1, ref2

synthetic biology ref1, ref2, ref3, ref4

testosterone levels, decrease in ref1

transgender = transhumanism ref1, ref2, ref3

vaccines ref1, ref2, ref3, ref4

human sacrifice ref1, ref2, ref3

Hunger Games Society ref1, ref2, ref3, ref4, ref5, ref6, ref7

Huxley, Aldous ref1, ref2, ref3

I

identity politics ref1, ref2, ref3

Illuminati ref1, ref2

illusory physical reality ref1

immigration ref1, ref2, ref3, ref4

Imperial College ref1, ref2, ref3, ref4, ref5, ref6

implantables ref1, ref2

incomes, destruction of [ref1](#), [ref2](#)

Infinite Awareness [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Internet [ref1](#), [ref2](#) *see also* social media
 artificial intelligence (AI) [ref1](#)
 independent journalism, lack of [ref1](#)
 Internet of Bodies (IoB) [ref1](#)

Internet of Everything (IoE) [ref1](#), [ref2](#)

Internet of Things (IoT) [ref1](#), [ref2](#)

lockdowns [ref1](#)

Psyop (psychological operation), Covid as a [ref1](#)
 trolls [ref1](#)

intersectionality [ref1](#)

inversion
 Archons [ref1](#), [ref2](#), [ref3](#)
 climate change hoax [ref1](#)
 energy [ref1](#)
 Judaism [ref1](#), [ref2](#), [ref3](#)
 symbolism [ref1](#)
 Wetiko factor [ref1](#)
 Wokeness [ref1](#), [ref2](#), [ref3](#)

Islam
 Archons [ref1](#)
 crypto-Jews [ref1](#)
 Islamic State [ref1](#), [ref2](#)
 Jinn and Djinn [ref1](#), [ref2](#), [ref3](#)
 Ottoman Empire [ref1](#)
 Wahhabism [ref1](#)

isolation *see* **social distancing and isolation**

Israel
 China [ref1](#)
 Cyber Intelligence Unit Beersheba complex [ref1](#)
 expansion of illegal settlements [ref1](#)

formation [ref1](#)
Global Cult [ref1](#)
Judaism [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
medical experiments, consent for [ref1](#)
Mossad [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Palestine-Israel conflict [ref1](#), [ref2](#), [ref3](#)
parents, replacement of [ref1](#)
Sabbatians [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
September 11, 2001, terrorist attacks on United States [ref1](#)
Silicon Valley [ref1](#)
Smart Grid [ref1](#), [ref2](#)
United States [ref1](#), [ref2](#)
vaccines [ref1](#)
Wetiko factor [ref1](#)

Italy

fear [ref1](#), [ref2](#), [ref3](#)
Lombardy [ref1](#), [ref2](#), [ref3](#)
vaccines [ref1](#)

J

Johns Hopkins University [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
Johnson, Boris [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)
Joint Biosecurity Centre (JBC) [ref1](#)

Judaism

anti-Semitism [ref1](#), [ref2](#), [ref3](#)
Archons [ref1](#), [ref2](#)
crypto-Jews [ref1](#)
inversion [ref1](#), [ref2](#), [ref3](#)
Israel [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Labour Party [ref1](#)
Nazi Germany [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Sabbatians [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Silicon Valley [ref1](#)
Torah [ref1](#)
United States [ref1](#), [ref2](#)
Zionists [ref1](#), [ref2](#), [ref3](#)

K

Kaufman, Andrew [ref1](#), [ref2](#), [ref3](#), [ref4](#)
knowledge [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
Koch's postulates [ref1](#)
Kurzweil, Ray [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
Kushner, Jared [ref1](#), [ref2](#)

L

Labour Party [ref1](#), [ref2](#)
Lanka, Stefan [ref1](#), [ref2](#)
Lateral Flow Device (LFD) [ref1](#)
Levy, Paul [ref1](#), [ref2](#), [ref3](#)
Life Program [ref1](#)
lockdowns [ref1](#), [ref2](#), [ref3](#)
 amplification tampering [ref1](#)
 Archons [ref1](#)
 Behavioural Insights Team [ref1](#)
 Black Lives Matter (BLM) [ref1](#)
 care homes, deaths in [ref1](#)
 children
 abuse [ref1](#), [ref2](#)
 mental health [ref1](#)
 China [ref1](#), [ref2](#)
 computer models [ref1](#)
 consequences [ref1](#), [ref2](#)
 dependency [ref1](#), [ref2](#), [ref3](#)

domestic abuse [ref1](#)
fall in cases [ref1](#)
fear [ref1](#), [ref2](#), [ref3](#), [ref4](#)
guaranteed income [ref1](#)
Hunger Games Society [ref1](#), [ref2](#), [ref3](#)
interaction, destroying [ref1](#)
Internet [ref1](#), [ref2](#)
overdoses [ref1](#)
perception [ref1](#)
police-military state [ref1](#), [ref2](#)
protests [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
psychopathic personality [ref1](#), [ref2](#), [ref3](#)
reporting/snitching, encouragement of [ref1](#), [ref2](#)
testing [ref1](#)
vaccines [ref1](#)
Wetiko factor [ref1](#)
WHO [ref1](#)
love [ref1](#), [ref2](#), [ref3](#)
Lucifer [ref1](#), [ref2](#), [ref3](#)

M

Madej, Carrie [ref1](#), [ref2](#)
Magufuli, John [ref1](#), [ref2](#)
mainstream media [ref1](#)
BBC [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)
censorship [ref1](#), [ref2](#)
China [ref1](#)
climate change hoax [ref1](#)
fear [ref1](#), [ref2](#)
Global Cult [ref1](#), [ref2](#)
independent journalism, lack of [ref1](#)
Ofcom [ref1](#), [ref2](#), [ref3](#)

perception [ref1](#), [ref2](#)
Psyop (psychological operation), Covid as a [ref1](#)
Sabbatians [ref1](#), [ref2](#)
social disapproval [ref1](#)
social distancing and isolation [ref1](#)
United States [ref1](#), [ref2](#)
vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Mao Zedong [ref1](#), [ref2](#), [ref3](#)

Marx and Marxism [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

masculinity [ref1](#)

masks/face coverings [ref1](#), [ref2](#), [ref3](#)

censorship [ref1](#)

children [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

China, made in [ref1](#)

dehumanisation [ref1](#), [ref2](#), [ref3](#)

fear [ref1](#), [ref2](#)

flu [ref1](#)

health professionals [ref1](#), [ref2](#), [ref3](#), [ref4](#)

isolation [ref1](#)

laughter [ref1](#)

mass non-cooperation [ref1](#)

microplastics, risk of [ref1](#)

mind control [ref1](#)

multiple masks [ref1](#)

oxygen deficiency [ref1](#), [ref2](#), [ref3](#)

police [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

pollution, as cause of plastic [ref1](#)

Psyop (psychological operation), Covid as a [ref1](#)

reframing [ref1](#), [ref2](#)

risk assessments, lack of [ref1](#), [ref2](#)

self-respect [ref1](#)

surgeons [ref1](#)

United States [ref1](#)
vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Wetiko factor [ref1](#)
'worms' [ref1](#)
The Matrix movies [ref1](#), [ref2](#), [ref3](#)
measles [ref1](#), [ref2](#)
media see mainstream media
Medicines and Healthcare products Regulatory Agency (MHRA)
[ref1](#), [ref2](#), [ref3](#), [ref4](#)
Mesopotamia [ref1](#)
messaging [ref1](#)
military-police state [ref1](#), [ref2](#), [ref3](#)
mind control [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#) *see also MKUltra*
MKUltra [ref1](#), [ref2](#), [ref3](#)
monarchy [ref1](#)
money *see banking, finance and money*
Montagnier, Luc [ref1](#), [ref2](#), [ref3](#)
Mooney, Bel [ref1](#)
Morgellons disease [ref1](#), [ref2](#)
mortality rate [ref1](#)
Mullis, Kary [ref1](#), [ref2](#), [ref3](#)
Musk, Elon [ref1](#)

N

Nag Hammadi texts [ref1](#), [ref2](#), [ref3](#)
nanotechnology [ref1](#), [ref2](#), [ref3](#)
narcissism [ref1](#)
Nazi Germany [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)
near-death experiences [ref1](#), [ref2](#)
Neocons [ref1](#), [ref2](#), [ref3](#)

Neuro-Linguistic Programming (NLP) and the Delphi technique
[ref1](#)

NHS (National Health Service)

amplification cycles [ref1](#)

Common Purpose [ref1](#), [ref2](#)

mind control [ref1](#)

NHS England [ref1](#)

saving the NHS [ref1](#), [ref2](#)

vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

whistle-blowers [ref1](#), [ref2](#), [ref3](#)

No-Problem-Reaction-Solution [ref1](#), [ref2](#), [ref3](#), [ref4](#)

non-human dimension of Global Cult [ref1](#)

nous [ref1](#)

numbers, reality as [ref1](#)

Nuremberg Codes [ref1](#), [ref2](#), [ref3](#)

Nuremberg-like tribunal, proposal for [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#),
[ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#), [ref11](#), [ref12](#)

Ø

Obama, Barack [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)

O'Brien, Cathy [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Ochel, Evita [ref1](#)

Ofcom [ref1](#), [ref2](#), [ref3](#)

old people [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Oneness [ref1](#), [ref2](#), [ref3](#)

Open Society Foundations (Soros) [ref1](#), [ref2](#), [ref3](#)

oxygen 406, 528–34

P

paedophilia [ref1](#), [ref2](#)

Page, Larry [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

Palestine-Israel conflict [ref1](#), [ref2](#), [ref3](#)

pandemic, definition of [ref1](#)

pandemic and health crisis scenarios/simulations [ref1](#), [ref2](#), [ref3](#), [ref4](#)

paranormal [ref1](#)

PCR tests [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

Pearl Harbor attacks, prior knowledge of [ref1](#)

Pelosi, Nancy [ref1](#), [ref2](#), [ref3](#)

perception [ref1](#), [ref2](#), [ref3](#), [ref4](#)

- climate change hoax [ref1](#)
- control [ref1](#), [ref2](#), [ref3](#)
- decoding [ref1](#), [ref2](#)
- enslavement [ref1](#)
- externally-delivered perceptions [ref1](#)
- five senses [ref1](#)
- human labels [ref1](#)
- media [ref1](#), [ref2](#)
- political parties [ref1](#), [ref2](#)
- Psyop (psychological operation), Covid as a [ref1](#)
- sale of perception [ref1](#)
- self-identity [ref1](#), [ref2](#)
- Wokeness [ref1](#)

Phantom Self [ref1](#), [ref2](#), [ref3](#)

pharmaceutical industry *see* **Big Pharma**

phthalates [ref1](#)

Plato's Allegory of the Cave [ref1](#), [ref2](#)

pneuma [ref1](#)

police

- Black Lives Matter (BLM) [ref1](#)
- brutality [ref1](#)
- citizen's arrests [ref1](#), [ref2](#)
- common law arrests [ref1](#), [ref2](#)

Common Purpose ref1
defunding ref1
lockdowns ref1, ref2
masks ref1, ref2, ref3, ref4
police-military state ref1, ref2, ref3
psychopathic personality ref1, ref2, ref3, ref4
reframing ref1
United States ref1, ref2, ref3, ref4
Wokeness ref1

polio ref1

political correctness ref1, ref2, ref3, ref4

political parties ref1, ref2, ref3, ref4

political puppets ref1

pollution ref1, ref2, ref3

post-mortems/autopsies ref1

Postage Stamp Consensus ref1, ref2

pre-emptive programming ref1

Problem-Reaction-Solution ref1, ref2, ref3, ref4, ref5, ref6, ref7, ref8

Project for the New American Century ref1, ref2, ref3, ref4

psychopathic personality ref1

- Archons ref1
- heart energy ref1
- lockdowns ref1, ref2, ref3
- police ref1, ref2, ref3, ref4
- recruitment ref1, ref2
- vaccines ref1
- wealth ref1
- Wetiko ref1, ref2

Psyop (psychological operation), Covid as a ref1, ref2, ref3, ref4, ref5

Pushbackers ref1, ref2, ref3, ref4

pyramid structure ref1, ref2, ref3, ref4

Q

QAnon Psyop [ref1](#), [ref2](#), [ref3](#)

R

racism *see also* **Black Lives**

Matter (BLM)

anti-racism industry [ref1](#)

class [ref1](#)

critical race theory [ref1](#)

culture [ref1](#)

intersectionality [ref1](#)

reverse racism [ref1](#)

white privilege [ref1](#), [ref2](#)

white supremacy [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Wokeness [ref1](#), [ref2](#), [ref3](#)

radiation [ref1](#), [ref2](#)

randomness, illusion of [ref1](#), [ref2](#), [ref3](#)

reality [ref1](#), [ref2](#), [ref3](#)

reframing [ref1](#), [ref2](#)

change agents [ref1](#), [ref2](#)

children [ref1](#)

climate change [ref1](#)

Common Purpose leadership programme [ref1](#), [ref2](#)

contradictory rules [ref1](#)

enforcers [ref1](#)

masks [ref1](#), [ref2](#)

NLP and the Delphi technique [ref1](#)

police [ref1](#)

Wetiko factor [ref1](#)

Wokeness [ref1](#), [ref2](#)

religion *see also* particular religions

alien invasions [ref1](#)

Archons [ref1](#), [ref2](#)

consciousness [ref1](#), [ref2](#)

control, system of [ref1](#), [ref2](#), [ref3](#)

criticism, prohibition on [ref1](#)

five senses [ref1](#)

good and evil, war between [ref1](#)

hidden non-human forces [ref1](#), [ref2](#)

Sabbatians [ref1](#)

save me syndrome [ref1](#)

Wetiko [ref1](#)

Wokeness [ref1](#)

repetition and mind control [ref1](#), [ref2](#), [ref3](#)

reporting/snitching, encouragement of [ref1](#), [ref2](#)

Reptilians/Grey entities [ref1](#)

rewiring the mind [ref1](#)

Rivers, Thomas Milton [ref1](#), [ref2](#)

Rockefeller family [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)

Rockefeller Foundation documents [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Roman Empire [ref1](#)

Rothschild family [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)

RT-PCR tests [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

Russia

collusion inquiry in US [ref1](#)

Russian Revolution [ref1](#), [ref2](#)

Sabbatians [ref1](#)

§

Sabbantian-Frankism [ref1](#), [ref2](#)

anti-Semitism [ref1](#), [ref2](#)

banking and finance [ref1](#), [ref2](#), [ref3](#)

China [ref1](#), [ref2](#)

Israel [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Judaism [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Lucifer [ref1](#)

media [ref1](#), [ref2](#)

Nazis [ref1](#), [ref2](#)

QAnon [ref1](#)

Rothschilds [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

Russia [ref1](#)

Saudi Arabia [ref1](#)

Silicon Valley [ref1](#)

Sumer [ref1](#)

United States [ref1](#), [ref2](#), [ref3](#)

Wetiko factor [ref1](#)

Wokeness [ref1](#), [ref2](#), [ref3](#)

SAGE (Scientific Advisory Group for Emergencies) [ref1](#), [ref2](#), [ref3](#),
[ref4](#)

SARS-1 [ref1](#)

SARs-CoV-2 [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

Satan/Satanism [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

satellites in low-orbit [ref1](#)

Saudi Arabia [ref1](#)

Save Me Syndrome [ref1](#)

scapegoating [ref1](#)

Schwab, Klaus [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#),
[ref11](#), [ref12](#)

science, manipulation of [ref1](#)

self-identity [ref1](#), [ref2](#), [ref3](#), [ref4](#)

self-respect, attacks on [ref1](#)

September 11, 2001, terrorist attacks on United States [ref1](#), [ref2](#),
[ref3](#), [ref4](#)

77th Brigade of UK military [ref1](#), [ref2](#), [ref3](#)

Silicon Valley/tech giants [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#) *see also*

Facebook

Israel [ref1](#)
Sabbatians [ref1](#)
technocracy [ref1](#)
Wetiko factor [ref1](#)
Wokeness [ref1](#)
simulation hypothesis [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Smart Grid [ref1](#), [ref2](#), [ref3](#)
artificial intelligence (AI) [ref1](#)
China [ref1](#), [ref2](#)
control centres [ref1](#)
the Field [ref1](#)
Great Reset [ref1](#)
Human 2.0 [ref1](#), [ref2](#)
Israel [ref1](#), [ref2](#)
vaccines [ref1](#)
Wetiko factor [ref1](#)
social disapproval [ref1](#)
social distancing and isolation [ref1](#), [ref2](#), [ref3](#)
abusive relationships [ref1](#), [ref2](#)
children [ref1](#)
flats and apartments [ref1](#)
heart issues [ref1](#)
hugs [ref1](#)
Internet [ref1](#)
masks [ref1](#)
media [ref1](#)
older people [ref1](#), [ref2](#)
one-metre (three feet) rule [ref1](#)
rewiring the mind [ref1](#)
simulation, universe as a [ref1](#)
SPI-B [ref1](#)
substance abuse [ref1](#)

suicide and self-harm [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
technology [ref1](#)
torture, as [ref1](#), [ref2](#)
two-metre (six feet) rule [ref1](#)
women [ref1](#)

social justice [ref1](#), [ref2](#), [ref3](#), [ref4](#)

social media *see also Facebook bans on alternative views* [ref1](#)
censorship [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
children [ref1](#)
emotion [ref1](#)
perception [ref1](#)
private messages [ref1](#)
Twitter [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
Wetiko factor [ref1](#)
YouTube [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Soros, George [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)

Spain [ref1](#)

SPI-B (Scientific Pandemic Insights Group on Behaviours) [ref1](#),
[ref2](#), [ref3](#), [ref4](#)

spider and the web [ref1](#), [ref2](#), [ref3](#), [ref4](#)

Starmer, Keir [ref1](#)

Statute Law [ref1](#)

Steiner, Rudolf [ref1](#), [ref2](#), [ref3](#)

Stockholm syndrome [ref1](#)

streptomycin [ref1](#)

suicide and self-harm [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

Sumer [ref1](#), [ref2](#)

Sunstein, Cass [ref1](#), [ref2](#), [ref3](#)

swine flu (H1N1) [ref1](#), [ref2](#), [ref3](#)

synchronicity [ref1](#)

synthetic biology [ref1](#), [ref2](#), [ref3](#), [ref4](#)

synthetic meat [ref1](#), [ref2](#)

T

technology *see also* **artificial intelligence (AI); Internet; social media addiction** [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Archons [ref1](#), [ref2](#)
the cloud [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
cyber-operations [ref1](#)
cyberwarfare [ref1](#)
radiation [ref1](#), [ref2](#)
social distancing and isolation [ref1](#)
technocracy [ref1](#)

Tedros Adhanom Ghebreyesus [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#),
[ref8](#), [ref9](#), [ref10](#), [ref11](#), [ref12](#), [ref13](#)

telepathy [ref1](#)

Tenpenny, Sherri [ref1](#)

Tesla, Nikola [ref1](#)

testosterone levels, decrease in [ref1](#)

testing for Covid-19 [ref1](#), [ref2](#)
 anal swab tests [ref1](#)
 cancer [ref1](#)
 China [ref1](#), [ref2](#), [ref3](#)
 Corman-Drosten test [ref1](#), [ref2](#), [ref3](#), [ref4](#)
 death certificates [ref1](#), [ref2](#)
 fraudulent testing [ref1](#)
 genetic material, amplification of [ref1](#)
 Lateral Flow Device (LFD) [ref1](#)
 PCR tests [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)
 vaccines [ref1](#), [ref2](#), [ref3](#)

Thunberg, Greta [ref1](#), [ref2](#), [ref3](#)

Totalitarian Tiptoe [ref1](#), [ref2](#), [ref3](#), [ref4](#)

transgender persons
 activism [ref1](#)
 artificial wombs [ref1](#)

censorship [ref1](#)
child abuse [ref1](#), [ref2](#)
Human 2.0 [ref1](#), [ref2](#), [ref3](#)
Wokeness [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
women, deletion of rights and status of [ref1](#), [ref2](#)
young persons [ref1](#)
travel restrictions [ref1](#)
Trudeau, Justin [ref1](#), [ref2](#), [ref3](#)
Trump, Donald [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#),
[ref11](#)
Twitter [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)

U

UKColumn [ref1](#), [ref2](#)
United Nations (UN) [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#) *see also Agenda 21/Agenda 2030 (UN)*
United States [ref1](#), [ref2](#)
 American Revolution [ref1](#)
 borders [ref1](#), [ref2](#)
 Capitol Hill riot [ref1](#), [ref2](#)
 children [ref1](#)
 China [ref1](#), [ref2](#)
 CIA [ref1](#), [ref2](#)
 Daily Pass tracking system [ref1](#)
 demographics by immigration, changes in [ref1](#)
 Democrats [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
 election fraud [ref1](#)
 far-right domestic terrorists, pushbackers as [ref1](#)
 Federal Reserve [ref1](#)
 flu/respiratory diseases statistics [ref1](#)
 Global Cult [ref1](#), [ref2](#)
 hand sanitisers, FDA warnings on [ref1](#)

immigration, effects of illegal [ref1](#)
impeachment [ref1](#)
Israel [ref1](#), [ref2](#)
Judaism [ref1](#), [ref2](#), [ref3](#)
lockdown [ref1](#)
masks [ref1](#)
mass media [ref1](#), [ref2](#)
nursing homes [ref1](#)
Pentagon [ref1](#), [ref2](#), [ref3](#), [ref4](#)
police [ref1](#), [ref2](#), [ref3](#), [ref4](#)
pushbackers [ref1](#)
Republicans [ref1](#), [ref2](#)
borders [ref1](#), [ref2](#)
Democrats [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Russia, inquiry into collusion with [ref1](#)
Sabbatians [ref1](#), [ref2](#), [ref3](#)
September 11, 2001, terrorist attacks [ref1](#), [ref2](#), [ref3](#), [ref4](#)
UFO sightings, release of information on [ref1](#)
vaccines [ref1](#)
white supremacy [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Woke Democrats [ref1](#), [ref2](#)

V

vaccines [ref1](#), [ref2](#), [ref3](#)
adverse reactions [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Africa [ref1](#)
anaphylactic shock [ref1](#), [ref2](#), [ref3](#), [ref4](#)
animals [ref1](#), [ref2](#)
anti-vax movement [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
AstraZeneca/Oxford [ref1](#), [ref2](#), [ref3](#), [ref4](#)
autoimmune diseases, rise in [ref1](#), [ref2](#)
Big Pharma [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#)

bioweapon, as real [ref1](#), [ref2](#)
black and ethnic minority communities [ref1](#)
blood clots [ref1](#), [ref2](#)
Brain Computer Interface (BCI) [ref1](#)
care homes, deaths in [ref1](#)
censorship [ref1](#), [ref2](#), [ref3](#)
chief medical officers and scientific advisers, financial interests of
[ref1](#), [ref2](#)
children [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#), [ref10](#)
China [ref1](#), [ref2](#)
clinical trials [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
compensation [ref1](#)
compulsory vaccinations [ref1](#), [ref2](#), [ref3](#)
computer programs [ref1](#)
consciousness [ref1](#)
cover-ups [ref1](#)
creation before Covid [ref1](#)
cytokine storm [ref1](#)
deaths and illnesses caused by vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
definition [ref1](#)
developing countries [ref1](#)
digital tattoos [ref1](#)
DNA-manipulation [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#),
[ref10](#)
emergency approval [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
female infertility [ref1](#)
funding [ref1](#)
genetic suicide [ref1](#)
Global Cult [ref1](#)
heart chakras [ref1](#)
hesitancy [ref1](#)
Human 2.0 [ref1](#), [ref2](#), [ref3](#), [ref4](#)
immunity from prosecution [ref1](#), [ref2](#), [ref3](#)

implantable technology [ref1](#)
Israel [ref1](#)
Johnson & Johnson [ref1](#), [ref2](#), [ref3](#), [ref4](#)
lockdowns [ref1](#)
long-term effects [ref1](#)
mainstream media [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
masks [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Medicines and Healthcare products Regulatory Agency (MHRA)
[ref1](#), [ref2](#)
messaging [ref1](#)
Moderna [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
mRNA vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)
nanotechnology [ref1](#), [ref2](#)
NHS [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
older people [ref1](#), [ref2](#)
operating system [ref1](#)
passports [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Pfizer/BioNTech [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
polyethylene glycol [ref1](#)
pregnant women [ref1](#)
psychopathic personality [ref1](#)
races, targeting different [ref1](#)
reverse transcription [ref1](#)
Smart Grid [ref1](#)
social distancing [ref1](#)
social media [ref1](#)
sterility [ref1](#)
synthetic material, introduction of [ref1](#)
tests [ref1](#), [ref2](#), [ref3](#)
travel restrictions [ref1](#)
variants [ref1](#), [ref2](#)
viruses, existence of [ref1](#)
whistle-blowing [ref1](#)

WHO [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Wokeness [ref1](#)
working, vaccine as [ref1](#)
young people [ref1](#)
Vallance, Patrick [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#)
variants [ref1](#), [ref2](#), [ref3](#)
vegans [ref1](#)
ventilators [ref1](#), [ref2](#)
virology [ref1](#), [ref2](#)
virtual reality [ref1](#), [ref2](#), [ref3](#)
viruses, existence of [ref1](#)
visual reality [ref1](#), [ref2](#)
vitamin D [ref1](#), [ref2](#)
von Braun, Wernher [ref1](#), [ref2](#)

W

war-zone hospital myths [ref1](#)
waveforms [ref1](#), [ref2](#)
wealth [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#) [ref10](#), [ref11](#)
wet market conspiracy [ref1](#)
Wetiko factor [ref1](#)
alcoholism and drug addiction [ref1](#)
anti-human, why Global Cult is [ref1](#)
Archons [ref1](#), [ref2](#), [ref3](#), [ref4](#)
artificial intelligence (AI) [ref1](#)
Big Pharma [ref1](#), [ref2](#)
children [ref1](#)
China [ref1](#)
consciousness [ref1](#), [ref2](#)
education [ref1](#)
Facebook [ref1](#)

fear [ref1](#), [ref2](#)
frequency [ref1](#), [ref2](#)
Gates [ref1](#), [ref2](#)
Global Cult [ref1](#), [ref2](#)
heart [ref1](#), [ref2](#)
lockdowns [ref1](#)
masks [ref1](#)
Native American concept [ref1](#)
psychopathic personality [ref1](#), [ref2](#)
reframing/retraining programmes [ref1](#)
religion [ref1](#)
Silicon Valley [ref1](#)
Smart Grid [ref1](#)
smartphone addiction [ref1](#), [ref2](#)
social media [ref1](#)
war [ref1](#), [ref2](#)
WHO [ref1](#)
Wokeness [ref1](#), [ref2](#), [ref3](#)
Yaldabaoth [ref1](#), [ref2](#), [ref3](#), [ref4](#)
whistle-blowing [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
white privilege [ref1](#), [ref2](#)
white supremacy [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
Whitty, Christopher [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#),
[ref10](#)
'who benefits' [ref1](#)
Wi-Fi [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Wikipedia [ref1](#), [ref2](#)
Wojcicki, Susan [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#)
Wokeness
Antifa [ref1](#), [ref2](#), [ref3](#), [ref4](#)
anti-Semitism [ref1](#)
billionaire social justice warriors [ref1](#), [ref2](#), [ref3](#)

Capitol Hill riot [ref1](#), [ref2](#)
censorship [ref1](#)
Christianity [ref1](#)
climate change hoax [ref1](#), [ref2](#)
culture [ref1](#)
education, control of [ref1](#)
emotion [ref1](#)
facts [ref1](#)
fascism [ref1](#), [ref2](#), [ref3](#)
Global Cult [ref1](#), [ref2](#), [ref3](#), [ref4](#)
group-think [ref1](#)
immigration [ref1](#)
indigenous people, solidarity with [ref1](#)
inversion [ref1](#), [ref2](#), [ref3](#)
left, hijacking the [ref1](#), [ref2](#)
Marxism [ref1](#), [ref2](#), [ref3](#)
mind control [ref1](#)
New Woke [ref1](#)
Old Woke [ref1](#)
Oneness [ref1](#)
perceptual programming [ref1](#)
 Phantom Self [ref1](#)
police [ref1](#)
defunding the [ref1](#)
reframing [ref1](#)
public institutions [ref1](#)
Pushbackers [ref1](#), [ref2](#), [ref3](#)
racism [ref1](#), [ref2](#), [ref3](#)
reframing [ref1](#), [ref2](#)
religion, as [ref1](#)
Sabbatians [ref1](#), [ref2](#), [ref3](#)
Silicon Valley [ref1](#)
social justice [ref1](#), [ref2](#), [ref3](#), [ref4](#)

transgender [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)
United States [ref1](#), [ref2](#)
vaccines [ref1](#)
Wetiko factor [ref1](#), [ref2](#), [ref3](#)
young people [ref1](#), [ref2](#), [ref3](#)
women, deletion of rights and status of [ref1](#), [ref2](#)
World Economic Forum (WEF) [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#),
[ref8](#), [ref9](#)
World Health Organization (WHO) [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#),
[ref7](#), [ref8](#), [ref9](#)
AIDs/HIV [ref1](#)
amplification cycles [ref1](#)
Big Pharma [ref1](#), [ref2](#), [ref3](#)
cooperation in health emergencies [ref1](#)
creation [ref1](#), [ref2](#)
fatality rate [ref1](#)
funding [ref1](#), [ref2](#), [ref3](#)
Gates [ref1](#)
Internet [ref1](#)
lockdown [ref1](#)
vaccines [ref1](#), [ref2](#), [ref3](#), [ref4](#)
Wetiko factor [ref1](#)
world number 1 (masses) [ref1](#), [ref2](#)
world number 2 [ref1](#)
Wuhan [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#) [ref8](#)

Y

Yaldabaoth [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#)
Yeadon, Michael [ref1](#), [ref2](#), [ref3](#), [ref4](#)
young people *see also children* addiction to technology [ref1](#)
Human 2.0 [ref1](#)
vaccines [ref1](#), [ref2](#)

Wokeness [ref1](#), [ref2](#), [ref3](#)

YouTube [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#)

WHO 548

Z

Zaks, Tal [ref1](#)

Zionism [ref1](#), [ref2](#), [ref3](#)

Zuckerberg, Mark [ref1](#), [ref2](#), [ref3](#), [ref4](#), [ref5](#), [ref6](#), [ref7](#), [ref8](#), [ref9](#),
[ref10](#), [ref11](#), [ref12](#)

Zulus [ref1](#)

ICKONIC

THE ALTERNATIVE

Ickonic is something that has been a dream of mine for the last 5 years. growing up around alternative information I have always had a natural interest in what is going on in the World and what could I do to make it better. Across the range of subjects and positions of influence occupied mainly by people who don't strive to make things better it's the Media that I have always found the most frustrating and fascinating. Mainly because if the Media did their Jobs properly then so much of the negative things happening in the World simply would not be able to happen, because they would be exposed within a heartbeat.

Free Press and the Opportunities that the internet could have given would mean that the Media are able to expose things like never before and hold people to account for their actions. As we all know there are 'Untouchables' that walk among us, people the Media simply won't touch, expose or investigate and that leads to the dark underworlds that infest the establishment the World over. Well I say enough, it's time for something different, a different kind of Media, where no one is off limits from exposing and investigating. All we're interested in at Ickonic is the truth of what is really going on in the World on whichever subject we're covering.

We hope you enjoy what we have created and take something away from the platform, we aim to deliver information that's informative and most importantly self-empowering, you're not a little person, you're part of something much bigger than that and its time we as a collective race began to understand that and look to the future as ours to take.

It's time...

Jaymie Icke - Founder Ickonic Alternative Media.

SIGN UP NOW AT ICKONIC.COM

DAVID ICKE

THE ANSWER



We live in extraordinary times with billions bewildered and seeking answers for what is happening. David Icke, the man who has been proved right again and again, has spent 30 years uncovering the truth behind world affairs and in a stream of previous books he predicted current events.

The Answer will change your every perception of life and the world and set you free of the illusions that control human society. There is nothing more vital for our collective freedom than humanity becoming aware of what is in this book.

Available now at davidicke.com.

THE **TRIGGER**

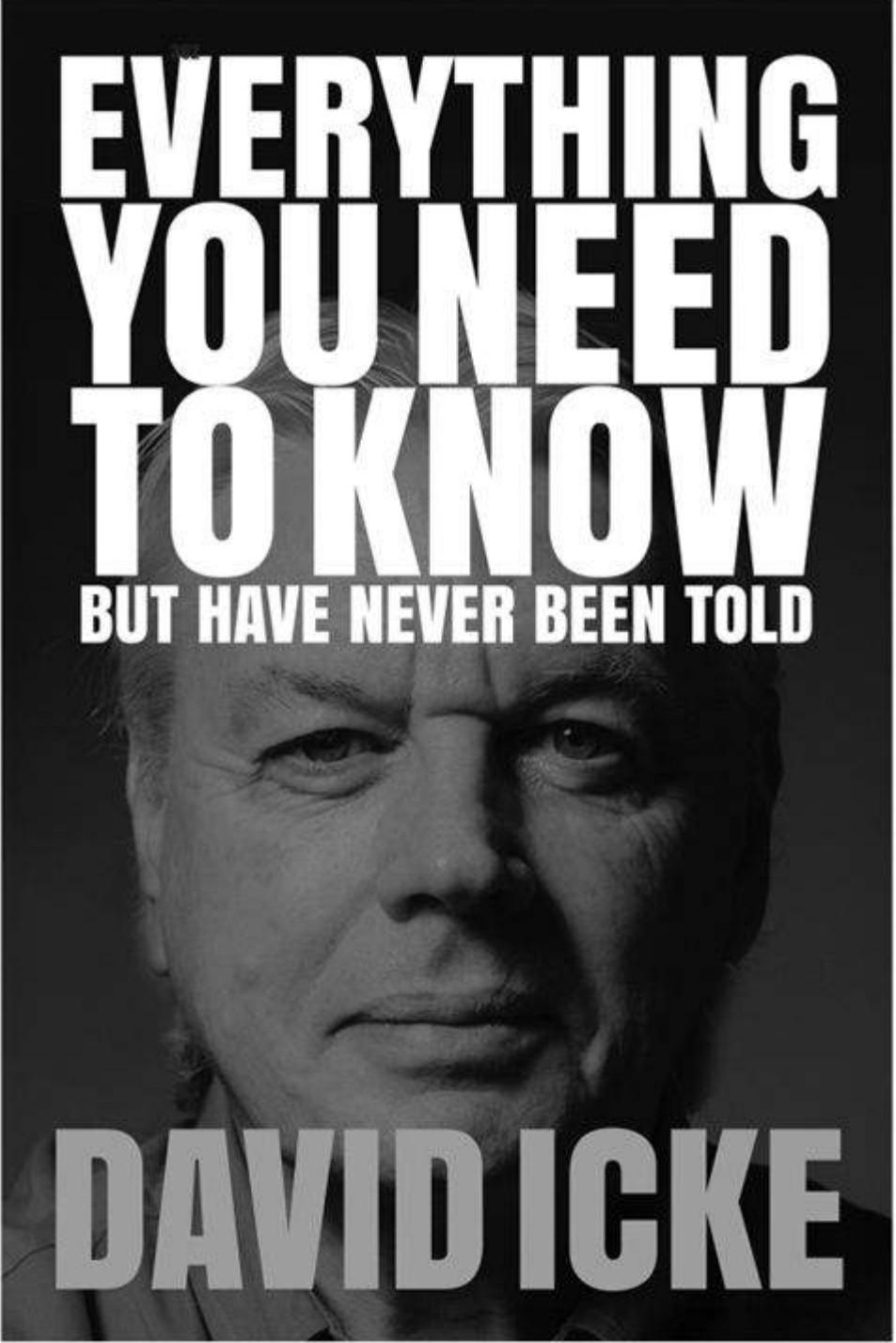
THE LIE THAT CHANGED THE WORLD
- WHO REALLY DID IT AND WHY



DAVID ICKE

EVERYTHING YOU NEED TO KNOW

BUT HAVE NEVER BEEN TOLD



DAVID ICKE

DAVIDICKE.COM



**DAVID ICKE STORE
LATEST NEWS ARTICLES
DAVID ICKE VIDEOS
WEEKLY DOT-CONNECTOR PODCASTS
LIVE EVENTS**
WWW.DAVIDICKE.COM

THE LIFE STORY OF DAVID ICKE

RENEGADE

/'ren-i.gəd/

noun

A person who behaves in a rebelliously unconventional manner.

THE FEATURE LENGTH FILM



AVAILABLE NOW AT DAVIDICKE.COM

2 NEW BOOKS
BY NEIL HAGUE

ORION'S DOOR

SYMBOLS OF CONSCIOUSNESS & BLUEPRINTS OF CONTROL
- THE STORY OF ORION'S INFLUENCE OVER HUMANITY

CUTTING EDGE VISIONARY ART
& UNIQUE ILLUSTRATED BOOKS

NEIL HAGUE

FOR
BOOKS, PRINTS & T-SHIRTS

VISIT:

NEILHAGUEBOOKS.COM

OR NEILHAGUE.COM

DR. COVID UNIVERSE

ADVENTURES
IN CLOWNLAND



NEIL HAGUE

Before you go ...

For more detail, background and evidence about the subjects in *Perceptions of a Renegade Mind* – and so much more – see my others books including *And The Truth Shall Set You Free; The Biggest Secret; Children of the Matrix; The David Icke Guide to the Global Conspiracy; Tales from the Time Loop; The Perception Deception; Remember Who You Are; Human Race Get Off Your Knees; Phantom Self; Everything You Need To Know But Have Never Been Told, The Trigger and The Answer.*

You can subscribe to the fantastic new Ickonic media platform where there are many hundreds of hours of cutting-edge information in videos, documentaries and series across a whole range of subjects which are added to every week. This includes my 90 minute breakdown of the week's news every Friday to explain *why* events are happening and to what end.