SYSTEM

ADMIN & SECURTY HANDBOOK

New Horizons in IT: From Linux Beginner to Sysadmin in Steps.

BY WILLIAM QIU

TABLE OF CONTENTS

CHAPTER 0: INTRODUCTION

00 What is System Administration?

System administration is the backbone of modern IT infrastructure. It's a critical discipline that ensures computer systems function smoothly, securely, and efficiently. System administrators, or sysadmins, are the guardians and facilitators of the digital landscape, responsible for managing, configuring, and maintaining the systems that power businesses, organizations, and networks. They set up hardware, install and configure software, manage server and network security, and solve technical problems that arise.

Good system administration is essential. It ensures that systems are not just operational, but optimized. It prevents downtime, protects against data breaches, and supports the continuous growth and transformation of the IT environment. System administration is as much about foreseeing potential issues and planning for the future as it is about managing the present.

In this chapter, we explore the field of system administration through a series of questions and answers designed to shed light on its importance, the role of Linux in this domain, the significance of data, and the types of companies that most benefit from robust system administration.

Frequent asked Q&A on System Administration

Q: What encompasses the role of a system administrator?
A: The role of a system administrator includes installing, supporting, and maintaining servers or other computer systems, and planning for and responding to service outages and other problems. They also script or automate routine tasks to improve system efficiency and reduce human error.

Q: Why is there a need for system administrators in an organization?
A: Organizations need system administrators to ensure that the IT infrastructure upon which their business processes rely is stable and secure. Sysadmins are key players in managing and preventing disruptions to the critical services that keep the digital aspects of a company running.

Q: What advantages does Linux offer for system administration?
A: Linux offers flexibility, security, stability, and control. Being open-source, it allows for extensive customization to meet specific needs, which is a significant advantage in system administration.

Q&A on System Administration continued..

Q: Why is data management a critical part of system administration?
A: Effective data management ensures that data is accurate, available, and secure. System administrators implement data backup and recovery plans, ensuring that critical data is never lost and always accessible when needed.

Q: Which organizations typically require dedicated system administrators?
A: While almost every organization can benefit from dedicated system administrators, those with large IT infrastructures, such as tech companies, financial institutions, healthcare providers, and educational institutions, particularly need them to manage their complex systems.

Q: What does open-source mean?
A: Open-source refers to software whose source code is made freely available for anyone to use, modify, and distribute. It is developed in a collaborative manner, often by a community of developers that contribute to its design and improvement. Open-source software is built on the principles of collaboration, transparency, and community-oriented development.

Q: What can we do with open-source resources?
A: Open-source resources provide a wealth of opportunities for learning, development, and innovation. With open-source software, individuals and organizations can:

Customize Solutions: Tailor the software to meet specific needs or solve unique challenges, which is particularly valuable for businesses that require specialized functionality.

Enhance Collaboration: Contribute to the development of the software, submit bug reports, suggest features, and collaborate with a global community.

Ensure Security: Review and audit the source code for security vulnerabilities, which can lead to more secure software as issues are identified and addressed by the community.

Reduce Costs: Avoid the high costs associated with proprietary software, as open-source software can be used and distributed without licensing fees.

Foster Innovation: Use the software as a foundation for new projects or products, accelerating innovation by building on existing, proven software solutions.

It is a good practice to constantly update your system and apps with:

TIPS

```
sudo apt-get update && sudo apt-get upgrade -y
```

CHAPTER 1: HYPERVISOR AND VIRTUAL MACHINE CONFIGURATION

01

Objective

This chapter aims to guide the reader through the process of selecting a hypervisor, configuring virtual machines (VMs), and installing an operating system (OS). The rationale behind each choice and step will be elaborated to provide a deeper understanding of the system administration and security implications

Scope

We will cover the selection of VirtualBox as a hypervisor, detailed steps for VM configuration, and the Debian OS installation process. The chapter will conclude with a rationale for the choices made.

Tools and Technologies Used

- Hypervisor: Oracle VM VirtualBox
- Operating System: Debian Linux
- Other Tools: Python (for troubleshooting)

Step-by-Step Guide

Selecting a Hypervisor

- Reason for Selection: Oracle VM VirtualBox was chosen for its ease of use, compatibility with various operating systems, and extensive documentation. Its familiarity also reduces the learning curve, allowing more focus on the configuration and security aspects.

VM Configuration

- Download VirtualBox: Navigate to VirtualBox's official website and download the latest version compatible with your operating system.
- Install VirtualBox: Follow the installation prompts, ensuring you install all necessary dependencies.

Configure the Virtual Machine:

- Allocate 8GB of RAM to ensure smooth operation.
- Assign 100GB of hard disk space to avoid storage issues.
- Network Configuration: Set up bridged or NAT networking based on your connectivity needs.

Operating System Installation

- Download Debian ISO: Obtain the Debian installation media from Debian's official website.
- Create a New VM: In VirtualBox, create a new VM and select the downloaded Debian ISO as the startup disk.
- Follow Installation Prompts: Proceed with the Debian installation, selecting default options for simplicity.

CHAPTER 1: HYPERVISOR AND VIRTUAL MACHINE CONFIGURATION

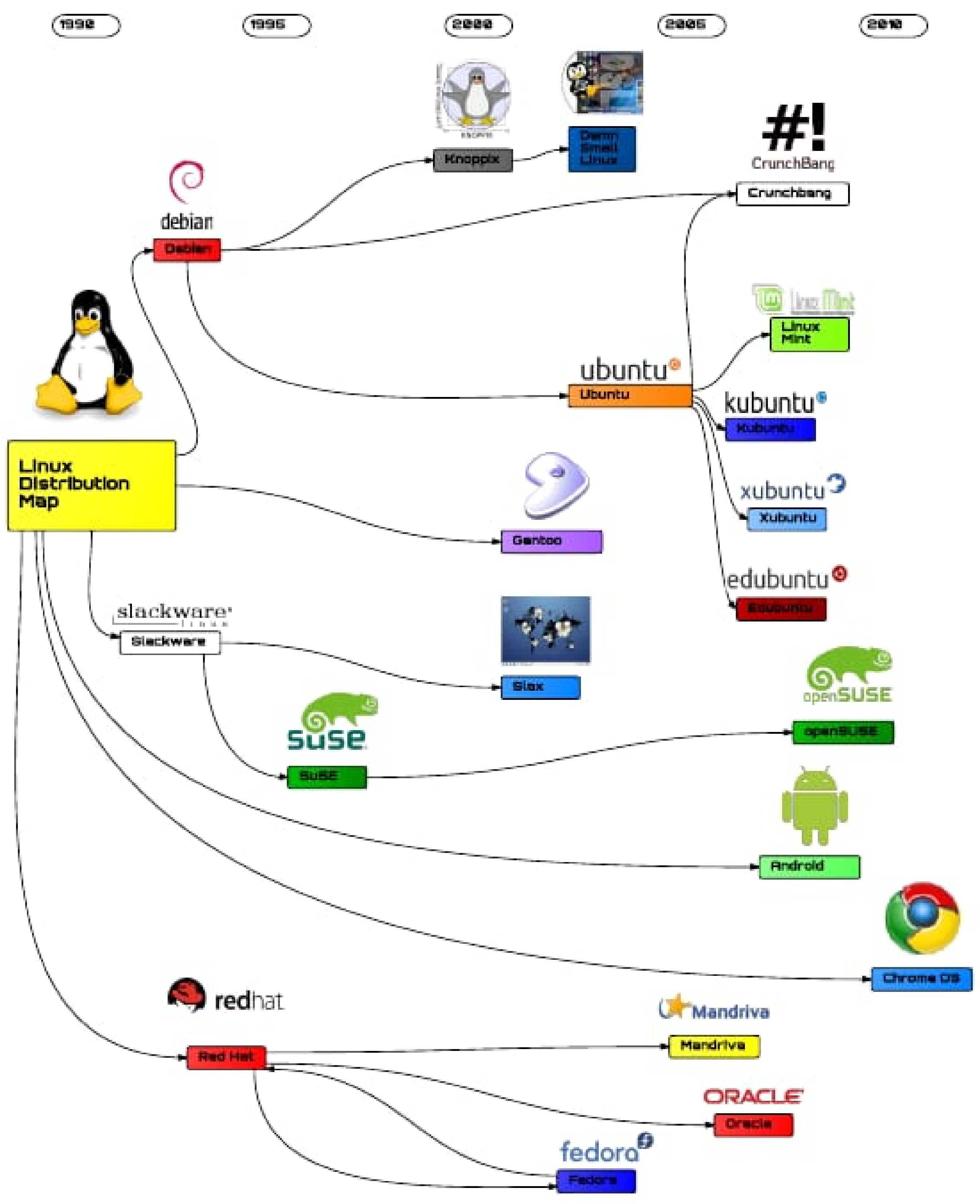
01 Troubleshooting Installation Issues

Missing Python Core: If the installation alerts to missing dependencies like Python Core, resolve this by installing the missing packages via command line:

```
>_
    pip install pywin32
```

Conclusion

Virtualization is a cornerstone of modern system administration, offering a flexible and efficient environment for testing and deployment. This chapter provided a foundation in VM configuration and OS installation, crucial for aspiring system administrators.



CHAPTER 2: SUDO AND SSHD CONFIGURATION

02

Objective

The purpose of this chapter is to explore the process of configuring sudo privileges for users and setting up Secure Shell Daemon (sshd) for secure remote access. These configurations are essential for managing permissions and ensuring secure administration of Linux systems.

Scope

This chapter will detail the steps to create a user with sudo access, generate SSH keys for secure communication, and configure the sshd service. Practical examples and command lines used during the project will be included to provide a clear guide for implementation.

Tools and Technologies Used

- Operating System: Debian Linux
- Key Commands: useradd, usermod, sudo, ssh-keygen, systemctl

Step-by-Step Guide

Configuring sudo Access

1.Create a New User:

```
sudo useradd -m -s /bin/bash <username>
```

Replace <username> with the desired username.

2.Create a New User:

```
sudo usermod -aG sudo <username>
```

3. Verify sudo Access:

Switch to the new user and attempt to run a command with sudo to verify access.

Setting Up SSH Keys

1.Generate SSH Key Pair:

```
ssh-keygen -t rsa -b 4096
```

Follow the prompts to generate a public and private key pair.

02 Step-by-Step Guide continued...

Deploy Public Key:

Place the public key in the .ssh/authorized_keys file of the user needing remote access.

Configuring sshd:

Edit sshd_config File:

Edit **'/etc/ssh/sshd_config'**, setting '**PasswordAuthentication no'** to disable password authentication.

Restart sshd Service:

```
>_
sudo systemctl restart sshd
```

Verify Configuration:

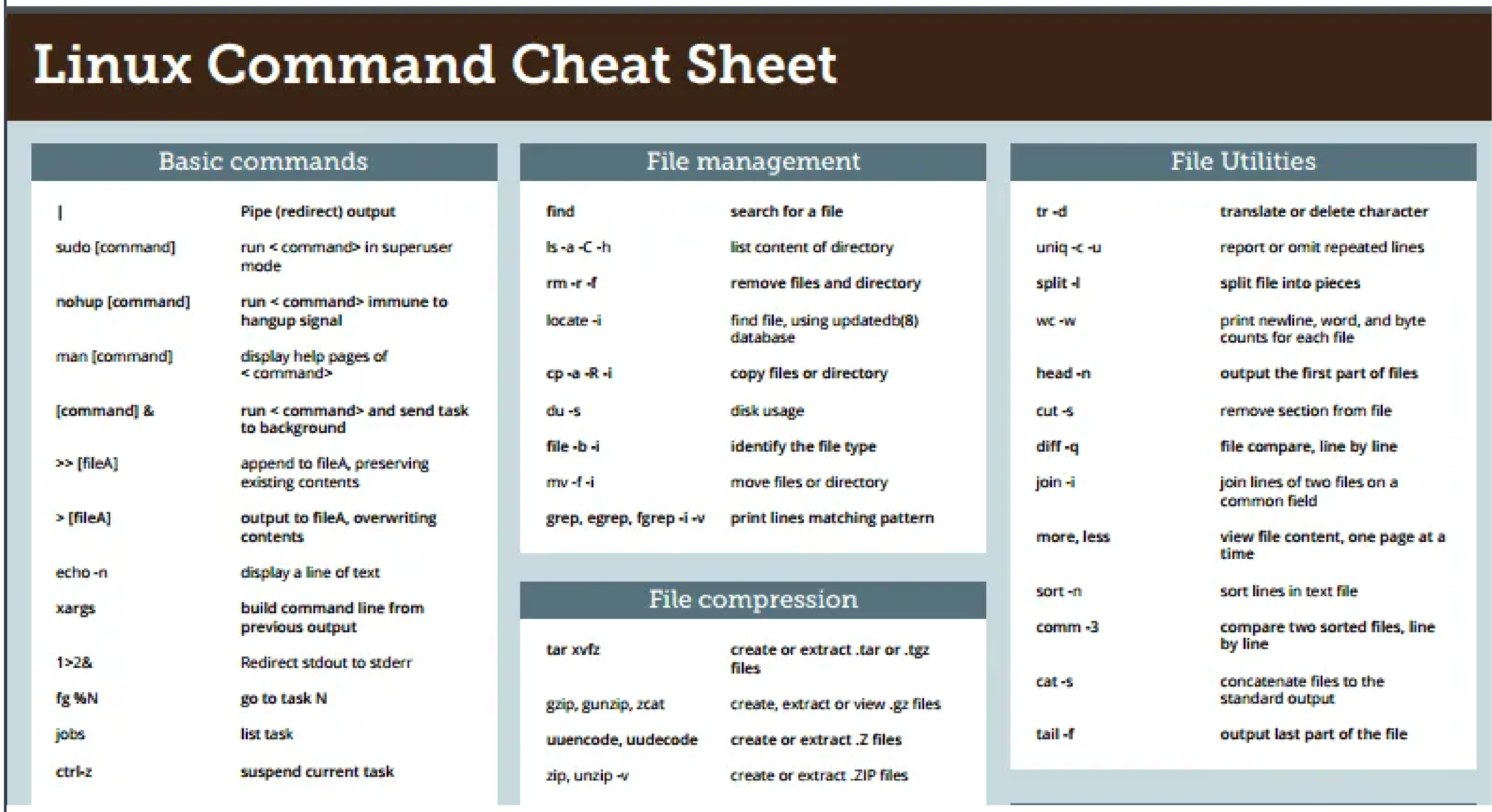
Attempt to SSH into the system using the private key. Password authentication should be disabled.

Discussion

Configuring sudo and sshd properly is critical for system security and management. It ensures that only authorized users can execute privileged commands and access the system remotely, reducing the risk of unauthorized access or configuration changes.

Conclusion

This chapter provided a guide to essential security configurations for Linux systems, focusing on user privileges and secure remote access. Mastery of these configurations is vital for system administrators aiming to secure their environments effectively.



Linux Command Cheat Sheet

Basic commands	
|	Pipe (redirect) output
sudo [command]	run < command> in superuser mode
nohup [command]	run < command> immune to hangup signal
man [command]	display help pages of < command>
[command] &	run < command> and send task to background
>> [fileA]	append to fileA, preserving existing contents
> [fileA]	output to fileA, overwriting contents
echo -n	display a line of text
xargs	build command line from previous output
1>2&	Redirect stdout to stderr
fg %N	go to task N
jobs	list task
ctrl-z	suspend current task

File management	
find	search for a file
ls -a -C -h	list content of directory
rm -r -f	remove files and directory
locate -i	find file, using updatedb(8) database
cp -a -R -i	copy files or directory
du -s	disk usage
file -b -i	identify the file type
mv -f -i	move files or directory
grep, egrep, fgrep -i -v	print lines matching pattern

File compression	
tar xvfz	create or extract .tar or .tgz files
gzip, gunzip, zcat	create, extract or view .gz files
uuencode, uudecode	create or extract .Z files
zip, unzip -v	create or extract .ZIP files

File Utilities	
tr -d	translate or delete character
uniq -c -u	report or omit repeated lines
split -l	split file into pieces
wc -w	print newline, word, and byte counts for each file
head -n	output the first part of files
cut -s	remove section from file
diff -q	file compare, line by line
join -i	join lines of two files on a common field
more, less	view file content, one page at a time
sort -n	sort lines in text file
comm -3	compare two sorted files, line by line
cat -s	concatenate files to the standard output
tail -f	output last part of the file

CHAPTER 3: ZFS ON LINUX

03

Objective

This chapter aims to introduce the ZFS filesystem, focusing on its installation and configuration on Linux systems. ZFS is known for its robust data integrity, storage efficiency, and advanced features like snapshotting and replication, which are crucial for data management and disaster recovery.

Scope

We will cover the process of installing ZFS on a Debian Linux system, configuring storage pools, and understanding key ZFS features. The chapter includes practical command examples to help readers effectively set up and manage ZFS on their systems.

Tools and Technologies Used

- Operating System: Debian Bookworm (or your specific Linux distribution)
- Software: ZFS on Linux
- Commands and Tools: apt, zpool, zfs

Step-by-Step Guide

Installing ZFS

1. Prepare the System:

Update your system's package list to ensure all packages are up to date.

```
sudo apt update
```

2. Install ZFS:

If you're using Debian, ZFS might be available in the backports repository. Install ZFS by running:

```
sudo apt install -t stable-backports zfsutils-linux
```

Configuring ZFS Storage Pools

1.Create a ZFS Pool:

Assuming you have two or more disks available for ZFS, create a mirrored pool for redundancy.

```
sudo zpool create mypool mirror /dev/sda /dev/sdb
```

03 Step-by-Step Guide continued...

2.Verify Pool Creation:

Check the status of the newly created pool to ensure it's functioning correctly.

```
sudo zpool status mypool
```

Exploring ZFS Features

1.Snapshots:

Create a snapshot of a filesystem for backup or restoration purposes.

```
sudo zfs snapshot mypool/mydataset@mysnapshot
```

2.Compression:

Enable compression to save space on your ZFS filesystem.

```
sudo zfs set compression=on mypool/mydataset
```

Discussion

ZFS's advanced features like snapshotting, compression, and data integrity checks provide a powerful toolset for managing storage in Linux environments. However, it's important to carefully plan your ZFS setup and consider factors like hardware requirements and pool layout for optimal performance and reliability.

Conclusion

This chapter provided an introduction to ZFS on Linux, from installation to basic configuration and usage. ZFS's comprehensive feature set makes it an excellent choice for managing complex storage needs, offering significant advantages in data integrity, scalability, and efficiency.

CHAPTER 4: INTEGRATING NFS WITH ZFS FOR EFFICIENT STORAGE SHARING

04

Objective

The objective of this chapter is to demonstrate the process of integrating ZFS with NFS to create a robust, efficient storage sharing system. This combination leverages ZFS's advanced file system features with NFS's network sharing capabilities, providing a powerful solution for distributed environments.

Scope

We'll cover the configuration of ZFS pools and file systems, setting them up for sharing over the network using NFS. This includes steps for NFS export and mount configurations, alongside ZFS features optimization for network sharing.

Tools and Technologies Used

- Operating System: Debian Linux (or any compatible Linux distribution)
- Software/Technologies: ZFS, NFS (nfs-kernel-server package)
- Commands and Tools: zpool, zfs, exportfs, showmount

Step-by-Step Guide

Preparing ZFS for NFS Sharing

1.Create ZFS File System:

Assuming you have a ZFS pool named mypool, create a new ZFS file system intended for NFS sharing.

```
sudo zfs create mypool/sharedfs
```

2. Set ZFS Properties:

Adjust ZFS properties to optimize for NFS sharing, such as setting the appropriate record size or enabling compression.

```
sudo zfs set sharenfs=on mypool/sharedfs
```

Configuring NFS Server

1.Install NFS Kernel Server:

Ensure the NFS server package is installed.

```
sudo apt install nfs-kernel-server
```

04 Step-by-Step Guide continued...

2.Export ZFS File System:

Edit /etc/exports to add the ZFS file system with the appropriate permissions.

```
>_
    sudo zpool status mypool
```

3.Apply Export Settings:

Make the NFS daemon re-read the exports file and apply changes.

```
>_
    sudo exportfs -rav
```

Mounting NFS on Client

1.Install NFS Client Utilities:

On the client system, ensure NFS client utilities are installed.

```
>_
    sudo apt install nfs-common
```

2.Create Mount Point and Mount:

Create a directory to mount the NFS share and mount it.

```
>_
    sudo apt install nfs-common
```

Discussion

Integrating ZFS with NFS combines the strengths of both technologies, offering a scalable and reliable storage solution. This setup is ideal for environments where data integrity, performance, and network accessibility are crucial. Key considerations include network security settings, ZFS performance tuning, and NFS version compatibility.

Conclusion

This chapter provided a guide to setting up a shared storage solution using ZFS and NFS. The integration of these two technologies allows for efficient data sharing across networks, making it an excellent choice for organizations with distributed storage needs. By following the outlined steps, system administrators can leverage the combined benefits of ZFS and NFS for their storage architecture.

CHAPTER 5: CONTAINERIZATION WITH DOCKER AND WEB SERVING WITH NGINX

05

Objective

This chapter aims to explore containerization with Docker, focusing on deploying an Nginx web server within a Docker container. The use of Docker exemplifies modern deployment practices, offering lightweight, scalable, and isolated environments for applications.

Scope

We will guide readers through the installation of Docker, the process of pulling and running the Nginx image from Docker Hub, and basic configuration steps to serve content through Nginx within a Docker container.

Tools and Technologies Used

- Operating System: Debian Linux (or compatible Linux distributions)
- Software/Technologies: Docker, Nginx Docker Image
- Commands and Tools: docker, systemctl, curl

Step-by-Step Guide

Installing Docker

1.Update Package Index:

sudo apt update

```
>_
sudo apt update
```

2. Install Required Packages:

To allow apt to use a repository over HTTPS:

```
>_
sudo apt install apt-transport-https ca-certificates curl software-
properties-common gnupg2
```

3.Add Docker's Official GPG Key:

```
>_
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-
key add -
```

4. Set Up the Docker Repository:

```
>_
echo "deb [arch=amd64] https://download.docker.com/linux/debian
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list
```

Install Docker CE:

```
>_
sudo apt update
sudo apt install docker-ce
```

05 Step-by-Step Guide continued...

Running Nginx in a Docker Container

1.Pull the Nginx Image:

```
>_
    sudo docker pull nginx
```

2.Run Nginx Container:

Start an Nginx container and map port 80 to the container.

```
>_
    sudo docker run --name my-nginx -p 80:80 -d nginx
```

2.Verify Nginx is Running:

Access the web server by navigating to the server's IP address or domain in a web browser or using **'curl'**.

```
>_
    curl http://localhost
```

Customizing Nginx Configuration (Optional)

Access Container's Bash:

```
>_
    sudo docker exec -it my-nginx /bin/bash
```

Modify the Nginx configuration files as needed, then restart the Nginx process within the container for changes to take effect.

Discussion

Containerization with Docker provides a streamlined method for deploying web servers like Nginx. This approach simplifies configuration, enhances portability, and isolates the Nginx environment from the host system, reducing potential conflicts and easing scalability.

Conclusion

This chapter introduced the fundamental concepts of containerization using Docker and demonstrated how to deploy an Nginx web server within a Docker container. The simplicity and efficiency of Docker containers make them an ideal choice for developers and system administrators looking to deploy web applications quickly and reliably.

CHAPTER 6: PERFORMANCE AND SYSTEM INTROSPECTION

06

Objective

The objective of this chapter is to explore methods and tools for system performance monitoring and introspection. Understanding and optimizing system performance are crucial for maintaining a reliable and efficient IT infrastructure.

Scope

We will cover various tools and commands for performance monitoring and system introspection, including how to identify and analyze performance bottlenecks. Special attention will be given to practical examples using tools such as DTrace, eBPF, and SystemTap.

Tools and Technologies Used

- Operating System: Linux (various distributions as applicable)
- Software/Technologies: DTrace, eBPF (Enhanced Berkeley Packet Filter), SystemTap
- Commands and Tools: Various commands provided by the mentioned technologies, tailored to specific performance questions and investigations.

Step-by-Step Guide

Introduction to Tools

- **DTrace:** Dynamic tracing framework for troubleshooting kernel and application problems on production systems in real-time.
- **eBPF:** Provides the ability to run programs in the Linux kernel without changing kernel source code or loading kernel modules.
- **SystemTap:** Allows administrators and developers to write and execute scripts to deeply examine the activities of a running Linux system.

Performance Monitoring Examples

1.CPU Usage Analysis with eBPF:

To allow apt to use a repository over HTTPS:

```
>_
bpftrace -e 'tracepoint:sched:sched_process_exec { printf("%d %s\n", pid, comm); }'
```

Objective: Identify processes that are consuming significant CPU resources.

2. Network Bottlenecks with eBPF:

```
>_
bpftrace -e 'tracepoint:net:net_dev_xmit { printf("%s %d\n", args->dev->name, args->len); }'
```

Objective: Monitor network packet transmission times and identify potential bottlenecks.

06 Step-by-Step Guide continued...

3. Network Latency Analysis:

```
bpftrace -e 'kprobe:tcp_sendmsg { printf("%d %s\n", pid, comm); }'
```

Objective: Investigate latencies in client-server communication to ensure optimal performance.

Discussion

This chapter emphasizes the importance of continuous performance monitoring and system introspection. By utilizing powerful tools like DTrace, eBPF, and SystemTap, system administrators and developers can gain valuable insights into the inner workings of their systems, allowing for informed decision-making regarding performance optimizations and troubleshooting.

Conclusion

The exploration of performance monitoring and system introspection tools in this chapter provides a foundation for maintaining and improving the performance of IT systems. These tools offer the means to proactively identify and resolve issues, ensuring that systems run efficiently and reliably.

You can make you own Linux distro with LINUX From Scratch!

CHAPTER 7: IMPLEMENTING INTRUSION DETECTION AND PREVENTION SYSTEMS

07

Objective

This chapter aims to elucidate the process of selecting, implementing, and configuring IDS and IPS solutions to safeguard network environments. The focus will be on utilizing Zeek (formerly Bro) to monitor network activities and detect potential security threats.

Scope

We will explore the installation and basic configuration of Zeek, creating and deploying custom detection scripts, and analyzing logs to identify suspicious activities. Emphasis will be placed on the practical application of IDS/IPS for real-time network security monitoring and threat detection.

Tools and Technologies Used

- Operating System: Linux distributions (specific focus on Debian)
- Software/Technologies: Zeek IDS/IPS
- Commands and Tools: zeekctl, cat, custom Zeek scripts

Step-by-Step Guide

Installing Zeek on Debian Linux

1.Add Zeek Repository:

```
echo 'deb
http://download.opensuse.org/repositories/security:/zeek/Debian_10/
/' | sudo tee /etc/apt/sources.list.d/zeek.list
```

2. Import the Repository Key:

```
wget -qO -
https://download.opensuse.org/repositories/security:zeek/Debian_10/R
elease.key | sudo apt-key add -
```

3.Install Zeek:

Update your package index and install Zeek.

```
sudo apt update
sudo apt install zeek-lts
```

Configuring Zeek for Network Monitoring

1.Basic Configuration:

Configure Zeek to monitor network interfaces of interest.

```
sudo zeekctl config set Interface=eth0
```

07 Step-by-Step Guide continued...

2.Deploying Zeek:

Use zeekctl to start Zeek and apply your configuration.

```
sudo zeekctl deploy
```

Creating Custom Detection Scripts

Example Script: Create a script to detect SSH brute-force attempts.

- Save the following content in /opt/zeek/share/zeek/site/local.zeek or a similar directory based on your installation:

```
@load policy/protocols/conn/known-services
@load policy/protocols/ssh/detect-bruteforcing

redef Notice::policy += {
  [$note=SSH::Password_Guessing, $action=Notice::ACTION_ALARM]
};
```

Deploy the script:

```
sudo zeekctl deploy
```

Analyzing Logs for Suspicious Activity

Review Connection Logs:

Look for unusual patterns or connections to unfamiliar services.

```
cat /opt/zeek/logs/current/conn.log | zeek-cut id.orig_h id.resp_h
id.resp_p service
```

Discussion

Implementing Zeek as an IDS/IPS solution provides a robust framework for network security monitoring. Custom scripts enhance Zeek's flexibility, allowing for tailored detection mechanisms that cater to specific network environments and threat landscapes.

Conclusion

This chapter has introduced Zeek IDS/IPS, showcasing its installation, configuration, and the development of custom detection scripts. Zeek's comprehensive network analysis capabilities make it an invaluable tool for detecting and preventing security threats in real time, thereby bolstering network defenses.

CHAPTER 8: STREAMLINING SYSTEM ADMINISTRATION WITH AUTOMATION

08

Objective

The goal of this chapter is to demonstrate the application of automation in system administration, specifically using Ansible to automate SSH configuration, system updates, and application installations. Automation simplifies repetitive tasks, ensures consistency, and enhances security across the IT infrastructure.

Scope

We'll cover the creation and execution of Ansible playbooks for automating SSH daemon configuration, patching systems, and deploying Nginx. This includes setting up Ansible, writing playbooks, and understanding the impact of automation on system administration efficiency and security.

Tools and Technologies Used

- Operating System: Ubuntu and Red Hat Linux distributions
- Software/Technologies: Ansible, SSH, Nginx
- Commands and Tools: ansible-playbook, ssh-keygen, apt, yum

Step-by-Step Guide

Setting Up Ansible

1.Install Ansible:

```
sudo apt update
sudo apt install ansible -y
```

2. Configure Ansible Inventory:

Add your managed nodes' IP addresses or hostnames to **'/etc/ansible/hosts'.**

Automating SSH Configuration

SSH Playbook: Create a playbook named ssh_config.yml with the following tasks:

- Disable password authentication.
- Add a specific public key to the authorized_keys file.

```
---
- hosts: all
  become: yes
  tasks:
    - name: Disable password authentication in sshd
      ansible.builtin.lineinfile:
        path: /etc/ssh/sshd_config
        regexp: '^#?PasswordAuthentication'
        line: 'PasswordAuthentication no'
      notify: restart sshd

    - name: Add authorized key
      ansible.builtin.authorized_key:
        user: 'username'
        state: present
        key: "{{ lookup('file', '/path/to/public/key') }}"
#Replace username and /path/to/public/key with your actual user and key path.
  handlers:
    - name: restart sshd
      ansible.builtin.service:
        name: sshd
        state: restarted
```

08 Step-by-Step Guide continued...

Patching Systems with Ansible

Update Playbook:

Write a playbook that checks the operating system type and applies updates accordingly.

```
---
- hosts: all
  become: yes
  tasks:
    - name: Update all packages on Debian/Ubuntu
      ansible.builtin.apt:
        upgrade: 'dist'
      when: ansible_os_family == "Debian"

    - name: Update all packages on RedHat/CentOS
      ansible.builtin.yum:
        name: '*'
        state: latest
      when: ansible_os_family == "RedHat"
```

Installing Nginx Using Ansible

```
---
- hosts: web_servers
#Ensure you define the web_servers group in your Ansible inventory.
  become: yes
  tasks:
    - name: Install Nginx
      ansible.builtin.package:
        name: nginx
        state: present

    - name: Start and enable Nginx
      ansible.builtin.service:
        name: nginx
        state: started
        enabled: yes
```

Discussion

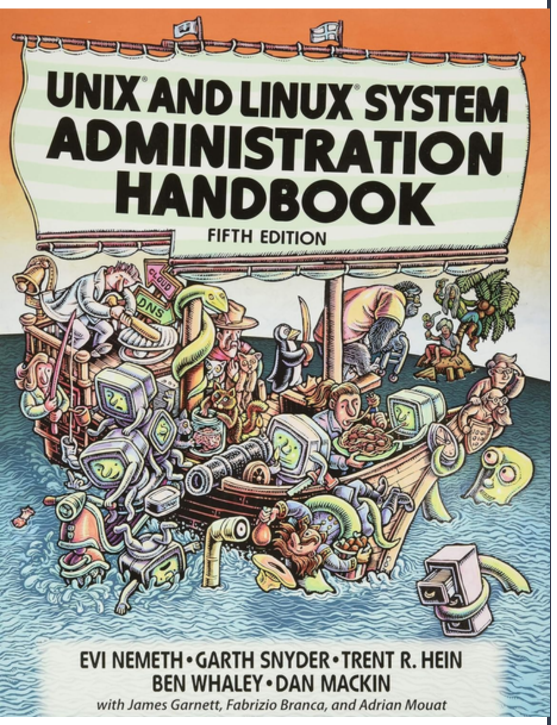
Automation with Ansible transforms the approach to system administration, reducing manual effort, mitigating human errors, and standardizing configurations across the IT landscape. This chapter emphasizes the practical applications of automation in key system administration tasks, showcasing Ansible's power and flexibility.

Conclusion

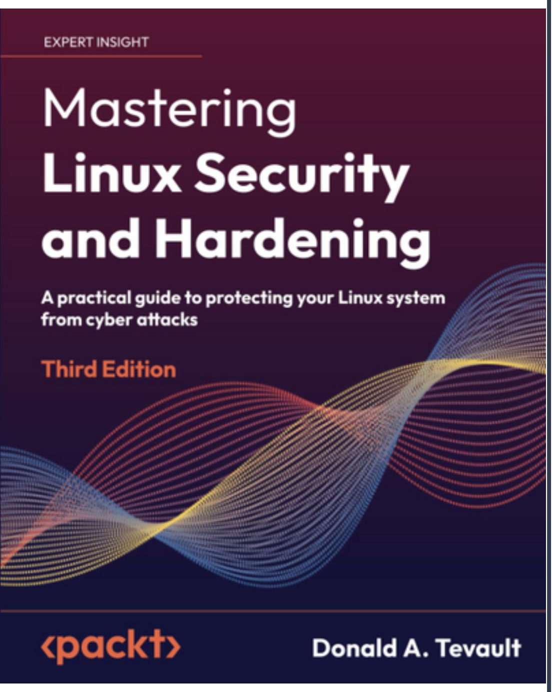
This chapter provided an overview of using automation to enhance system administration processes, with Ansible serving as the central tool for automating SSH configurations, system updates, and Nginx installations. Adopting automation practices like those described here can significantly improve the efficiency, consistency, and security of managing IT infrastructures.

REFERENCES

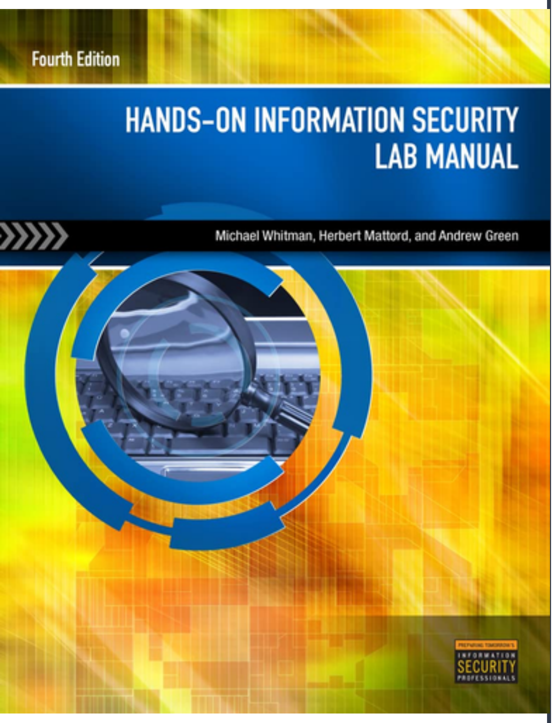
09


UNIX AND LINUX SYSTEM ADMINISTRATION HANDBOOK
FIFTH EDITION
EVI NEMETH • GARTH SNYDER • TRENT R. HEIN
BEN WHALEY • DAN MACKIN
with James Garnett, Fabrizio Branca, and Adrian Mouat

Nemeth, E., Snyder, G., Hein, T. R., Whaley, B., & Macklin, D. (2017). Unix and Linux Administration Handbook, 5th Edition. Addison-Wesley. ISBN-10: 0-13-427755-4.


EXPERT INSIGHT
Mastering
Linux Security and Hardening
A practical guide to protecting your Linux system from cyber attacks
Third Edition
‹packt›
Donald A. Tevault

Tevault, D. A. (2020). Mastering Linux Security and Hardening: Protect your Linux systems from intruders, malware attacks, and other cyber threats, 2nd Edition. Packt Publishing. ISBN-10: 1838981772.


Fourth Edition
HANDS-ON INFORMATION SECURITY LAB MANUAL
Michael Whitman, Herbert Mattord, and Andrew Green

Whitman, M., et al. (2014). Hands-on Information Security Lab Manual, 4th Edition. Cengage Learning.ISBN-10: 1285167570