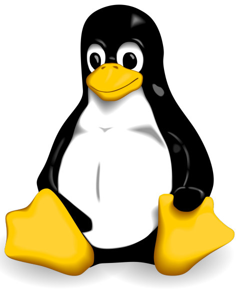


Linux Guide

10th edition March 2022

Foreword

This guide stems from the notes I have been taking while studying and working with Linux.
It contains useful information about standards and tools for Linux system administration, as well as a good amount of topics from the certification exams LPIC-1 (Linux Professional Institute Certification level 1), LPIC-2, RHCSA (Red Hat Certified System Administrator), RHCE (Red Hat Certified Engineer), and CEH (Certified Ethical Hacker). Unless otherwise specified, the shell of reference is Bash.
This is an independent publication and is not affiliated with LPI, Red Hat, EC-Council, or any other organization. You are free to use and share the whole guide or any single page, provided that you distribute them unmodified and not for profit.
This document has been composed with Apache OpenOffice.

Happy Linux hacking,

Daniele Raffo

Version history

1st edition	May 2013	9th edition	January 2021
2nd edition	September 2014	10th edition	March 2022
3rd edition	July 2015		
4th edition	June 2016		
5th edition	September 2017		
6th edition	August 2018		
7th edition	May 2019		
8th edition	January 2020		

Bibliography and suggested readings

- Evi Nemeth et al., *UNIX and Linux System Administration Handbook*, O'Reilly
- Rebecca Thomas et al., *Advanced Programmer's Guide to Unix System V*, McGraw-Hill
- Christoph Braun, *Unix System Security Essentials*, Addison-Wesley
- Mendel Cooper, *Advanced Bash-Scripting Guide*, http://tldp.org/LDP/abs/html
- Ellen Siever et al., *Linux in a Nutshell*, O'Reilly, http://archive.oreilly.com/linux/cmd
- Bruce Barnett, *The Grymoire*, http://www.grymoire.com/Unix
- Colin Barschel, *Unix Toolbox*, http://cb.vu/unixtoolbox.xhtml
- Adam Haeder et al., *LPI Linux Certification in a Nutshell*, O'Reilly
- Heinrich W. Klöpping et al., *The LPIC-2 Exam Prep*, http://lpic2.unix.nl
- Michael Jang, *RHCSA/RHCE Red Hat Linux Certification Study Guide*, McGraw-Hill
- Asghar Ghori, *RHCSA & RHCE RHEL 7: Training and Exam Preparation Guide*, Lightning Source Inc.
- Linus Torvalds' Linux documentation, https://github.com/torvalds/linux/tree/master/Documentation
- The Linux Documentation Project guides, https://www.tldp.org/guides.html
- RHEL manuals, https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux
- Linux man pages, https://www.kernel.org/doc/man-pages
- CentOS 7 man pages, https://www.unix.com/man-page-centos-repository.php
- A-Z index of Bash command line, http://ss64.com/bash
- GNU software manuals, http://www.gnu.org/manual
- Shell command line snippets, http://www.commandlinefu.com
- Bash command line snippets, http://www.bashoneliners.com
- RAM management in Linux, http://www.linuxatemyram.com
- Linux performance, http://www.brendangregg.com/linuxperf.html
- Regular expressions tester and cheat sheet, http://www.regextester.com
- Bash pitfalls, http://mywiki.wooledge.org/BashPitfalls
- Install instructions for any command, https://command-not-found.com

Index

Logical Volume Management (LVM) introduces an abstraction between physical and logical storage, allowing a more versatile use of filesystems. LVM uses the Linux device mapper feature (`/dev/mapper`).

Disks, partitions, and RAID devices are made of **Physical Volumes**, which are grouped into a **Volume Group**.
A Volume Group is divided into small fixed-size chunks called Physical Extents, which are mapped 1-to-1 to Logical Extents.
Logical Extents are grouped into **Logical Volumes**, on which filesystems are created.

How to create a Logical Volume

1.	Add a new disk to the machine	
2.	`lsblk`	Verify that the new disk is recognized e.g. as `/dev/sda`
3.	`fdisk /dev/sda`	Create a new partition (of type 0x8E = Linux LVM) on the new disk. This is not necessary but recommended, because other OSes might not recognize the LVM header and see the whole unpartitioned disk as empty
4.	`pvcreate /dev/sda1`	Initialize the Physical Volume to be used with LVM
5.	`vgcreate -s 8M myvg0 /dev/sda1`	Create a Volume Group and define the size of Physical Extents to 8 Mb (default value is 4 Mb)
or	`vgextend myvg0 /dev/sda1`	or add the Physical Volume to an existing Volume Group
6.	`lvcreate -L 1024M -n mylv myvg0`	Create a Logical Volume
7.	`mkfs -t ext3 /dev/myvg0/mylv`	Create a filesystem on the Logical Volume
8.	`mount /dev/myvg0/mylv /mnt/mystuff`	Mount the Logical Volume

How to increase the size of a Logical Volume (operation possible only if the underlying filesystem allows it)

1.	Add a new disk to the machine, to provide the extra disk space	
2.	`pvcreate /dev/sdc`	Initialize the Physical Volume
3.	`vgextend myvg0 /dev/sdc`	Add the Physical Volume to an existing Volume Group
or		
1.	Increase the size of an existing virtual disk (already initialized as PV)	
2.	`partprobe`	Notify the kernel of the new disk size
3.	`pvresize /dev/sdc`	Accommodate the Physical Volume to the new size
Then:		
4.	`lvextend -L 2048M /dev/myvg0/mylv`	Extend the Logical Volume by 2 Gb
or	`lvresize -L+2048M /dev/myvg0/mylv`	
or	`lvresize -l+100%FREE /dev/myvg/mylv`	or extend the Logical Volume taking all free space
5.	`resize2fs /dev/myvg0/mylv` (ext) `xfs_growfs /dev/myvg0/mylv` (XFS)	Extend the filesystem. Alternatively, use `lvresize -r` on the previous step

How to reduce the size of a Logical Volume (operation possible only if the underlying filesystem allows it)

1.	`resize2fs /dev/myvg0/mylv 900M`	Shrink the filesystem to 900 Mb
2.	`lvreduce -L 900M /dev/myvg0/mylv`	Shrink the Logical Volume to 900 Mb
or	`lvresize -L 900M /dev/myvg0/mylv`	

How to snapshot and backup a Logical Volume

1.	`lvcreate -s -L 1024M -n mysnap /dev/myvg0/mylv`	Create the snapshot like a Logical Volume
2.	`tar cvzf mysnap.tar.gz mysnap`	Backup the snapshot with any backup tool
3.	`lvremove /dev/mvvg0/mysnap`	Delete the snapshot

LVM - commands

PV commands		VG commands		LV commands	
pvs	Report information about Physical Volumes	vgs	Report information about Volume Groups	lvs	Report information about Logical Volumes
pvscan	Scan all disks for Physical Volumes	vgscan	Scan all disks for Volume Groups	lvscan	Scan all disks for Logical Volumes
pvdisplay	Display Physical Volume attributes	vgdisplay	Display Volume Group attributes	lvdisplay	Display Logical Volume attributes
pvck	Check Physical Volume metadata	vgck	Check Volume Group metadata		
pvcreate	Initialize a disk or partition for use with LVM	vgcreate	Create a Volume Group using Physical Volumes	lvcreate	Create a Logical Volume in a Volume Group
pvchange	Change Physical Volume attributes	vgchange	Change Volume Group attributes	lvchange	Change Logical Volume attributes
pvremove	Remove a Physical Volume	vgremove	Remove a Volume Group	lvremove	Remove a Logical Volume
		vgextend	Add a Physical Volume to a Volume Group	lvextend	Increase the size of a Logical Volume
		vgreduce	Remove a Physical Volume from a Volume Group	lvreduce	Reduce the size of a Logical Volume
pvresize	Modify the size of a Physical Volume			lvresize	Modify the size of a Logical Volume
		vgmerge	Merge two Volume Groups		
		vgsplit	Split two Volume Groups		
		vgimport	Import a Volume Group into a system		
		vgexport	Export a Volume Group from a system		
pvmove	Move the Logical Extents on a Physical Volume to wherever there are available Physical Extents (within the Volume Group) and then put the Physical Volume offline				

LVM global commands	
dmsetup *command*	Perform low-level LVM operations
lvm *command*	Perform LVM operations. May also be used as an interactive tool
lvmsar	LVM system activity reporter. Unsupported on LVM2
lvmdiskscan	Scan the system for disks and partitions usable by LVM
lvmconfig	Show the current LVM disk configuration

/dev/mapper/*vgname*-*lvname* /dev/*vgname*/*lvname*	Mapping of Logical Volumes in the filesystem
/etc/lvm/archive/	Directory containing Volume Groups metadata backups

Boot sequence (older systems)	
POST (Power-On Self Test)	Low-level check of PC hardware.
BIOS (Basic I/O System)	Detection of disks and hardware.
Chain loader GRUB	GRUB stage 1 is loaded from the MBR and executes GRUB stage 2 from filesystem. GRUB chooses which OS to boot on. The chain loader hands over to the boot sector of the partition on which resides the OS. The chain loader also mounts `initrd`, an initial ramdisk (typically a compressed ext2 filesystem) to be used as the initial root device during kernel boot; this makes possible to load kernel modules that recognize hard drives hardware and that are hence needed to mount the real root filesystem. Afterwards, the system runs `/linuxrc` with PID 1. (From Linux 2.6.13 onwards, the system instead loads into memory `initramfs`, a cpio-compressed image, and unpacks it into an instance of tmpfs in RAM. The kernel then executes `/init` from within the image.)
Linux kernel	Kernel decompression into memory. Kernel execution. Detection of devices. The real root filesystem is mounted on `/` in place of the initial ramdisk.
init	Execution of `init`, the first process (PID 1). The system tries to execute in order `/sbin/init`, `/etc/init`, `/bin/init`, and `/bin/sh`; if none of these succeeds, the kernel panics.
Startup	The system loads startup scripts and runlevel scripts.
Login	If in text mode, `init` calls the `getty` process, which runs the `login` command that asks the user for login and password. If in graphical mode, the X Display Manager starts the X Server.

Boot sequence (modern systems)	
POST (Power-On Self Test)	Low-level check of PC hardware.
BIOS (Basic I/O System)	Detection of disks and hardware.
GRUB 2	GRUB 2 is loaded from the MBR. It prompts the user to select a Linux kernel; the corresponding kernel image is then executed.
Linux kernel	Kernel decompression into memory. Kernel executes `systemd`.
systemd	Execution of `systemd` (PID 1). Mount of filesystems and swap partitions, start of low-level services (`sysinit.target`). Setting of various timers, paths, and sockets (`basic.target`). Loading of `default.target`, which is either `multi-user.target` i.e. console login in text mode or `graphical.target` i.e. graphical login.

Information about the boot process can be found in the manpages `man 7 boot` and `man 7 bootup`.

Modern systems use **UEFI (Unified Extensible Firmware Interface)** instead of BIOS. UEFI does not use the MBR boot code; it has knowledge of partition table and filesystems, and stores its application files required for launch in an EFI System Partition, usually formatted as FAT32.
After the POST, the system loads the UEFI firmware which initializes the hardware required for booting, then reads its Boot Manager data to determine which UEFI application to launch. The launched UEFI application may then launch another application, e.g. the kernel and initramfs in case of a boot loader like GRUB.

`efivar`	Manipulate UEFI variables
`efibootmgr`	Manipulate the UEFI Boot Manager
`efibootdump`	Display UEFI boot options

Startup sequence	Debian	Red Hat
At startup `/sbin/init` executes all instructions on `/etc/inittab`. This script at first switches to the default runlevel...	`id:2:initdefault:`	`id:5:initdefault:`
... then it runs the following script (same for all runlevels) which configures peripheral hardware, applies kernel parameters, sets hostname, and provides disks initialization...	`/etc/init.d/rcS`	`/etc/rc.d/rc.sysinit` or `/etc/rc.sysinit`
... and then, for runlevel *N*, it calls the script `/etc/init.d/rc` *`N`* (i.e. with the runlevel number as parameter) which launches all services and daemons specified in the following startup directories:	`/etc/rc`*`N`*`.d/`	`/etc/rc.d/rc`*`N`*`.d/`

The startup directories contain symlinks to the init scripts in `/etc/init.d/` which are executed in numerical order. Links starting with K are called with argument `stop`, links starting with S are called with argument `start`.

```
lrwxrwxrwx.  1 root root   14 Feb 11 22:32 K88sssd -> ../init.d/sssd
lrwxrwxrwx.  1 root root   15 Nov 28 14:50 K89rdisc -> ../init.d/rdisc
lrwxrwxrwx.  1 root root   17 Nov 28 15:01 S01sysstat -> ../init.d/sysstat
lrwxrwxrwx.  1 root root   18 Nov 28 14:54 S05cgconfig -> ../init.d/cgconfig
lrwxrwxrwx.  1 root root   16 Nov 28 14:52 S07iscsid -> ../init.d/iscsid
lrwxrwxrwx.  1 root root   18 Nov 28 14:42 S08iptables -> ../init.d/iptables
```

The last script to be run is `S99local -> ../init.d/rc.local`; therefore, an easy way to run a specific program upon boot is to call it from this script file.

`/etc/init.d/boot.local`		runs only at boot time, not when switching runlevel.
`/etc/init.d/before.local`	(SUSE)	runs only at boot time, before the scripts in the startup directories.
`/etc/init.d/after.local`	(SUSE)	runs only at boot time, after the scripts in the startup directories.

To add or remove services at boot sequence:	`update-rc.d` *`service`* `defaults`
	`update-rc.d -f` *`service`* `remove`
		`chkconfig --add` *`service`*
		`chkconfig --del` *`service`*

When adding or removing a service at boot, startup directories will be updated by creating or deleting symlinks for the default runlevels: `K` symlinks for runlevels 0 1 6, and `S` symlinks for runlevels 2 3 4 5.
Service will be run via the `xinetd` super server.

Linux Standard Base (LSB)

The Linux Standard Base defines a format to specify default values on an init script `/etc/init.d/foo`:

```
### BEGIN INIT INFO
# Provides: foo
# Required-Start: bar
# Defalt-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Description: Service Foo init script
### END INIT INFO
```

Default runlevels and `S`/`K` symlinks values can also be specified as such:

```
# chkconfig: 2345 85 15
# description: Foo service
```

`/etc/init/start-ttys.conf` (Red Hat)
Start the specified number of terminals at bootup via `getty`, which manages physical or virtual terminals (TTYs)

`/etc/sysconfig/init` (Red Hat)
Control appearance and functioning of the system during bootup

`/etc/machine-id` (Red Hat)
Randomly-generated machine ID.
The machine ID can be safely regenerated by deleting this file and then running the command `systemd-machine-id-setup`

`/etc/securetty`
List of TTYs from which the root user is allowed to login

`/etc/issue`
Message printed before the login prompt. Can contain these escape codes:

`\b`	Baudrate of line	`\o`	Domain name
`\d`	Date	`\r`	OS release number
`\s`	System name and OS	`\t`	Time
`\l`	Terminal device line	`\u`	Number of users logged in
`\m`	Machine architecture identifier	`\U`	"*n* users" logged in
`\n`	Nodename aka hostname	`\v`	OS version and build date

`/etc/issue.net`
Message printed before the login prompt on a remote session

`/etc/motd`
Message Of The Day, printed after a successful login, but before execution of the login shell

`/etc/nologin`
If this file exists, `login` and `sshd` deny login to all unprivileged users. Useful when doing system maintenance

`/etc/login.defs`
Definition of default values (UID and GID ranges, mail directory, account validity, password encryption method, etc.) for user account creation

`/var/log/secure` (Red Hat)
`/var/log/auth.log` (Debian)
Logfile containing user logins (both successful and failed) and authentication mechanisms

`/var/log/pwdfail`
Logfile containing failed authentication attempts

To prevent a specific user to log in, their shell can be set either as:

`/bin/false`
user is forced to exit immediately

`/sbin/nologin`
user is prompted a message and forced to exit; the message is "This account is currently not available" or the contents of file `/etc/nologin.txt` if it exists

	Runlevel (SysV)	Target (Systemd)	Debian	Red Hat
		`halt.target`	System halt, no power off	
	0	**`poweroff.target`**	Shutdown	
	1	**`rescue.target`**	Single user / maintenance mode	
default runlevels	**2**		Multi-user mode (default)	Multi-user mode without network
	3	**`multi-user.target`**	Multi-user mode	Multi-user mode with network
	4		Multi-user mode	Unused, for custom use
	5	**`graphical.target`**	Multi-user mode	Multi-user mode with network and X (default)
	6	**`reboot.target`**	Reboot	
	S	**`emergency.target`**	Single user / maintenance mode with no mounted filesystems and no running services (usually accessed through runlevel 1)	
		`default.target`		System will always boot to this target; this is a symlink to **`multi-user.target`** or **`graphical.target`**

Systemd's target `runlevel`*`n`*`.target` emulates a SysV's runlevel *n*.

`runlevel`
`who -r`
	Display the previous and the current runlevel

`init` *`runlevel`*
`telinit` *`runlevel`*
	Change to *runlevel*

`systemctl get-default`	Get the default target
`systemctl set-default` *`target`*	Set *target* as the default target
`systemctl isolate` *`target`*	Change to *target*
`systemctl emergency`	Change to maintenance single-user mode with only `/root` filesystem mounted
`systemctl rescue`	Change to maintenance single-user mode with only local filesystems mounted
`systemctl -t target`	List targets

To boot on the desired target a machine whose default target has become invalid (e.g. is symlinked to `reboot.target`), edit the GRUB 2 line and append `systemd.unit=`*`desired`*`.target` to the kernel parameters.

`shutdown -h now`
`halt`
`poweroff`
`init 0`
`telinit 0`
`systemctl isolate shutdown.target`
	Shut down the system. Depending on the system, it will be either halted or powered off

`shutdown -r now`
`reboot`
`init 6`
`telinit 6`
`systemctl isolate reboot.target`
	Reboot the system

`shutdown`	Shut down the system securely: all logged in users are notified via a message to their terminal, and login is disabled. Can only be run by the root user
`shutdown -a`	Non-root users that are listed in `/etc/shutdown.allow` can use this command to shut down the system
`shutdown -h 16:00` *`message`*	Schedule a shutdown for 4 PM and send a warning message to all logged in users
`shutdown -f`	Skip `fsck` on reboot
`shutdown -F`	Force `fsck` on reboot
`shutdown -c`	Cancel a shutdown that has been already initiated

SysV service management

/etc/init.d/*service operation* service *service operation* rc*service operation*	 (Red Hat) (SUSE)	Perform the specified operation (start, stop, status, etc.) on the specified service
update-rc.d *service* defaults chkconfig --add *service*	(Debian) (Red Hat)	Add a service at boot
update-rc.d -f *service* remove chkconfig --del *service*	(Debian) (Red Hat)	Remove a service at boot
update-rc.d -f *service* \\ start 30 2 3 4 5 . stop 70 0 1 6 .		Add a service on the default runlevels; creates S30 symlinks for starting the service and K70 symlinks for stopping it
chkconfig --levels 245 *service* on		Add the service on runlevels 2 4 5
chkconfig *service* on		Add the service on default runlevels
chkconfig *service* off		Remove the service on default runlevels
chkconfig *service*		Check if the service is enabled on the current runlevel
chkconfig *service* reset		Reset the on/off state of the service for all runlevels to whatever the LSB specifies in the init script
chkconfig *service* resetpriorities		Reset the start/stop priorities of the service for all runlevels to whatever the LSB specifies in the init script
chkconfig --list *service*		Display current configuration of service (its status and the runlevels in which it is active)
chkconfig chkconfig --list		List all active services and their current configuration
ls /etc/rc*n*.d	(Debian)	List services started on runlevel *n*

Supported service operations		
start	Start the service	Mandatory
stop	Stop the service	
restart	Restart the service (stop, then start)	
status	Display daemon PID and execution status	
force-reload	Reload configuration if service supports it, otherwise restart	
condrestart try-restart	Restart the service only if already running	Optional
reload	Reload the service configuration	

Systemd service management

`systemctl` *`operation service`*	Perform the specified operation (`start`, `stop`, `status`, etc.) on the specified service (unit file)

`systemctl enable` *`service`*	Add the service on the current target
`systemctl disable` *`service`*	Remove the service on the current target
`systemctl is-enabled` *`service`*	Check if the service is enabled on the current target
`systemctl mask` *`service`*	Mask the service on the current target. This prevents the service to be enabled or started
`systemctl unmask` *`service`*	Unmask the service on the current target

`systemctl list-unit-files --type=service`	List all active services and their current configuration
`systemctl`	List loaded and active units
`systemctl --all`	List all units, including inactive ones

/etc/inittab

```
# The default runlevel.
id:2:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change of runlevel.
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
# Normally not reached, but fall through in case of emergency.
z6:6:respawn:/sbin/sulogin

# /sbin/getty invocations for the runlevels.
# Id field must be the same as the last characters of the device (after "tty").
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
```

`/etc/inittab` describes which processes are started at bootup and during normal operation; it is read and executed by `init` at bootup.
All its entries have the form ***id***:***runlevels***:***action***:***process***.

id	1-4 characters, identifies uniquely an entry. For gettys and other login processes it should be equal to the suffix of the corresponding tty	
runlevels	Runlevels for which the specified action must be performed. If empty, action is performed on all runlevels	
action	`respawn`	Process will be restarted when it terminates
	`wait`	Process is started at the specified runlevel and `init` will wait for its termination (i.e. execution of further lines of `/etc/inittab` stops until the process exits)
	`once`	Process is executed once at the specified runlevel
	`boot`	Process is executed at system boot. Runlevels field is ignored
	`bootwait`	Process is executed at system boot and `init` will wait for its termination. Runlevels field is ignored
	`off`	Does nothing
	`ondemand`	Process is executed when an on-demand runlevel (A, B, C) is called
	`initdefault`	Specifies the default runlevel to boot on. Process field is ignored
	`sysinit`	Process is executed at system boot, before any `boot` or `bootwait` entries. Runlevels field is ignored
	`powerfail`	Process is executed when power goes down and a UPS kicks in. `init` will not wait for its termination
	`powerwait`	Process is executed when power goes down and a UPS kicks in. `init` will wait for its termination
	`powerfailnow`	Process is executed when power is down and the UPS battery is almost empty
	`powerokwait`	Process is executed when power has been restored from UPS
	`ctrlaltdel`	Process is executed when `init` receives a SIGINT via CTRL ALT DEL
	`kbdrequest`	Process is executed when a special key combination is pressed on console
process	Process to execute. If prepended by a `+`, `utmp` and `wtmp` accounting will not be performed	

Directory	Content
/bin	Essential command binaries for all users
/boot	Bootloader files (OS loader, kernel images, initrd, etc.)
/dev	Virtual filesystem containing device nodes to devices and partitions
/etc	System configuration files and scripts
/home	Home directories for users
/lib	Libraries for the binaries in `/bin` and `/sbin`, kernel modules
/lost+found	Storage directory for recovered files in this partition
/media	Mount points for removable media
/mnt	Mount points for temporary filesystems
/net	Access to directory tree on different external NFS servers
/opt	Optional, large add-on application software packages
/proc	Virtual filesystem providing kernel and processes information
/root	Home directory for the root user
/run	Runtime variable data; replaces `/var/run`
/sbin	Essential system binaries, system administration commands
/srv	Data for services provided by the system
/sys	Virtual filesystem providing information about hotplug hardware devices
/tmp	Temporary files; deleted at reboot
/usr	User utilities and applications
/usr/bin	Non-essential command binaries for all users
/usr/include	C header files
/usr/lib	Libraries for the binaries in `/usr/bin` and `/usr/sbin`
/usr/local	Software installed locally
/usr/local/bin	Local software binaries
/usr/local/games	Local game binaries
/usr/local/include	Local C header files
/usr/local/lib	Local libraries for the binaries in `/usr/local/bin` and `/usr/local/sbin`
/usr/local/man	Local man pages
/usr/local/sbin	Local system binaries
/usr/local/share	Local architecture-independent hierarchy
/usr/local/src	Local source code
/usr/sbin	Non-essential system binaries (daemons and services)
/usr/share	Architecture-independent files (icons, fonts, documentation, etc.)
/usr/share/doc	Package-specific documentation not included in man pages
/usr/share/man	Man pages
/usr/share/info	Documentation in Info format
/usr/src	Source code for the current OS
/var	Variable files (logs, caches, mail spools, etc.)
/var/log	Logfiles
/var/opt	Variable files for the application software installed in `/opt`
/var/spool	Queued items to be processed (mail messages, cron jobs, print jobs, etc.)
/var/tmp	Temporary files that need to be stored for a longer time; preserved between reboots

The manpage `man hier` contains information about filesystem hierarchy.

The **superblock** contains information relative to the filesystem e.g. filesystem type, size, status, metadata structures.

The **Master Boot Record (MBR)** is a 512-byte program located in the first sector of the hard disk; it contains information about hard disk partitions and has the duty of loading the OS.
MBR has the following limits:
- max 4 primary partitions per hard disk, or 3 primary partitions + 1 extended partition; partitions numbered from 1 to 4
- max 11 logical partitions (inside the extended partition) per hard disk; partitions numbered from 5 to 15
- max disk size is 2 Tb

On recent systems, the MBR is replaced by the **GUID Partition Table (GPT)**. GPT does not differentiate between primary, extended, or logical partitions; furthermore, it practically has no limits on number and size of partitions.

When a partition is initialized, a **UUID (Universal Unique Identifier)**, which is a 128-bit hash number, is associated to it.

Almost all modern filesystems use **journaling**; in a journaling filesystem, the journal logs changes before committing them to the filesystem, which ensures faster recovery and less risk of corruption in case of a crash.

FUSE (Filesystem in Userspace) is an interface for userspace programs to export a filesystem to the Linux kernel, and is particularly useful for virtual file systems.

`fdisk /dev/sda`	Disk partitioning interactive tool
`fdisk -l /dev/sda`	List the partition table of device `/dev/sda`
`parted`	Disk partitioning interactive tool
`sfdisk /dev/sda`	Disk partitioning non-interactive tool
`cfdisk`	Disk partitioning tool with text-based UI
`gparted` `gnome-disks`	Disk partitioning tool with GUI
`partprobe` *device* `hdparm -z` *device*	Notify the OS about partition table changes. Otherwise, the changes will take place only after reboot
`blockdev --getbsz /dev/sda1`	Get the block size of the specified partition
`wipefs` *device*	List all visible filesystems and their signatures' offsets
`wipefs -a` *device*	Erase filesystem or raid signatures (magic strings i.e. metadata) from the device to make the filesystem invisible from `blkid`
`file -s /dev/sda`	Show information about device `/dev/sda` e.g. whether it uses MBR
`blkid /dev/sda1`	Print the UUID of the specified partition
`blkid -L /boot`	Print the UUID of the specified partition, given its label
`blkid -U 652b786e-b87f-49d2-af23-8087ced0c667`	Print the name of the specified partition, given its UUID
`findfs UUID=652b786e-b87f-49d2-af23-8087ced0c667`	Print the name of the specified partition, given its UUID
`findfs LABEL=/boot`	Print the name of the specified partition, given its label
`e2label /dev/sda1`	Print the label of the specified partition

`mkfs -t` *`fstype device`*	Create a filesystem of the specified type on a partition (i.e. format the partition).
	mkfs is a wrapper utility for the actual filesystem-specific maker commands:
	`mkfs.ext2` aka `mke2fs`
	`mkfs.ext3` aka `mke3fs`
	`mkfs.ext4`
	`mkfs.msdos` aka `mkdosfs`
	`mkfs.ntfs` aka `mkntfs`
	`mkfs.reiserfs` aka `mkreiserfs`
	`mkfs.jfs`
	`mkfs.xfs`

`mkfs -t ext2 /dev/sda`
`mkfs.ext2 /dev/sda`
`mke2fs /dev/sda`
	Create an ext2 filesystem on `/dev/sda`

`mke2fs -j /dev/sda`
`mkfs.ext3 /dev/sda`
`mke3fs /dev/sda`
	Create an ext3 filesystem (ext2 with journaling) on `/dev/sda`

`mkfs -t msdos /dev/sda`
`mkfs.msdos /dev/sda`
`mkdosfs /dev/sda`
	Create a MS-DOS filesystem on `/dev/sda`

`fsck` *`device`*	Check and repair a Linux filesystem. The filesystem must be unmounted; running fsck on a mounted filesystem, even if mounted read-only, risks damaging it.
	Corrupted files will be placed into the `/lost+found` directory of the partition.
	fsck is a wrapper utility for the actual filesystem-specific checker commands, e.g.:
	`fsck.ext2` aka `e2fsck`
	`fsck.ext3` aka `e2fsck`
	`fsck.ext4` aka `e2fsck`
	`fsck.msdos`
	`fsck.vfat`
	`fsck.cramfs`
	`fsck.minix`

`fsck`
`fsck -As`
	Check and repair serially all filesystems listed in `/etc/fstab`

`fsck -f /dev/sda1`	Force a filesystem check on `/dev/sda1` even if fsck thinks it is not necessary
`fsck -y /dev/sda1`	During filesystem repair, do not ask questions and assume that the answer is always yes
`fsck -n /dev/sda1`	Perform safely a filesystem check on a mounted filesystem, only reporting errors without trying to correct them. Not recommended as the results will not be accurate, and some types of filesystem do not even support this option
`fsck.ext2 -c /dev/sda1`	Check an ext2 filesystem, running the `badblocks` command to mark all bad blocks and add them to the bad block inode, so that they will not be allocated to files or directories
`touch /forcefsck` (Red Hat)	Force a filesystem check after next reboot

`mount`
`cat /proc/mounts`
`cat /etc/mtab`
	Display the currently mounted filesystems, and their mount options.
	The commands `mount` and `umount` maintain in `/etc/mtab` a database of currently mounted filesystems, but `/proc/mounts` is authoritative

`mount -a`	Mount all devices listed in `/etc/fstab`, except those indicated as `noauto`
`mount -t ext3 /dev/sda /mnt`	Mount a Linux-formatted disk. The mount point (directory) must exist
`mount -t msdos /dev/fd0 /mnt`	Mount a MS-DOS filesystem floppy disk to mount point `/mnt`
`mount /dev/fd0`	Mount a floppy disk. `/etc/fstab` must contain an entry for `/dev/fd0`
`mount -o remount,rw /`	Remount the root directory as read-write, supposing it was mounted read-only. Useful to change flags (in this case, read-only to read-write) for a mounted filesystem that cannot be unmounted at the moment
`mount -o nolock 10.7.7.7:/export/ /mnt/nfs`	Mount a NFS share without running NFS daemons. Useful during system recovery
`mount -t iso9660 -o ro,loop=/dev/loop0 cd.img /mnt/cdrom`	Mount a CD-ROM ISO9660 image file like a CD-ROM (via the loop device)

`umount /dev/fd0`
`umount /mnt`
	Unmount a floppy disk that was mounted on `/mnt` (device must not be busy)

`umount -l /dev/fd0`	Unmount the floppy disk as soon as it is not in use anymore
`mountpoint /mnt`	Tell if a directory is a mount point
`findmnt`	List all mounted filesystems

`findmnt /dev/sda`
`findmnt /mnt`
	Find the filesystem, given a specified device or mount point

`eject /dev/fd0`
`eject /mnt`
	Eject a removable media device

Partition types					
0x00	Empty	0x4e	QNX4.x 2nd part	0xa8	Darwin UFS
0x01	FAT12	0x4f	QNX4.x 3rd part	0xa9	NetBSD
0x02	XENIX root	0x50	OnTrack DM	0xab	Darwin boot
0x03	XENIX usr	0x51	OnTrack DM6 Aux1	0xaf	HFS / HFS+
0x04	FAT16 < 32Mb	0x52	CP/M	0xb7	BSDI fs
0x05	Extended	0x53	OnTrack DM6 Aux3	0xb8	BSDI swap
0x06	FAT16	0x54	OnTrackDM6	0xbb	Boot Wizard hidden
0x07	HPFS / NTFS / exFAT	0x55	EZ-Drive	0xbe	Solaris boot
0x08	AIX	0x56	Golden Bow	0xbf	Solaris
0x09	AIX bootable	0x5c	Priam Edisk	0xc1	DRDOS/sec (FAT-12)
0x0a	OS/2 Boot Manager	0x61	SpeedStor	0xc4	DRDOS/sec (FAT-16 < 32Mb)
0x0b	W95 FAT32	0x63	GNU HURD or SysV	0xc6	DRDOS/sec (FAT-16)
0x0c	W95 FAT32 (LBA)	0x64	Novell Netware 286	0xc7	Syrinx
0x0e	W95 FAT16 (LBA)	0x65	Novell Netware 386	0xda	Non-FS data
0x0f	W95 extended (LBA)	0x70	DiskSecure Multi-Boot	0xdb	CP/M, CTOS, etc.
0x10	OPUS	0x75	PC/IX	0xde	Dell Utility
0x11	Hidden FAT12	0x80	Old Minix	0xdf	BootIt
0x12	Compaq diagnostics	0x81	Minix / old Linux	0xe1	DOS access
0x14	Hidden FAT16 < 32Mb	0x82	Linux swap / Solaris	0xe3	DOS R/O
0x16	Hidden FAT16	0x83	Linux	0xe4	SpeedStor
0x17	Hidden HPFS/NTFS	0x84	OS/2 hidden C: drive	0xeb	BeOS fs
0x18	AST SmartSleep	0x85	Linux extended	0xee	GPT
0x1b	Hidden W95 FAT32	0x86	NTFS volume set	0xef	EFI (FAT-12/16/32)
0x1c	Hidden W95 FAT32 (LBA)	0x87	NTFS volume set	0xf0	Linux/PA-RISC boot
0x1e	Hidden W95 FAT16 (LBA)	0x88	Linux plaintext	0xf1	SpeedStor
0x24	NEC DOS	0x8e	Linux LVM	0xf4	SpeedStor
0x27	Hidden NTFS WinRE	0x93	Amoeba	0xf2	DOS secondary
0x39	Plan 9	0x94	Amoeba BBT	0xfb	VMware VMFS
0x3c	PartitionMagic recovery	0x9f	BSD/OS	0xfc	VMware VMKCORE
0x40	Venix 80286	0xa0	IBM Thinkpad hibernation	0xfd	Linux raid autodetect
0x41	PPC PReP Boot	0xa5	FreeBSD	0xfe	LANstep
0x42	SFS	0xa6	OpenBSD	0xff	BBT
0x4d	QNX4.x	0xa7	NeXTSTEP		

The command `sfdisk -T` prints the above list of partition IDs and names.

Most used Linux-supported filesystems	
ext2	The oldest Linux ext filesystem, without journaling
ext3	ext2 with journaling
ext4	Linux journaling filesystem, an upgrade from ext3
Reiserfs	Journaling filesystem
XFS	Journaling filesystem, developed by SGI. Offers more performance and scalability than ext4
JFS	Journaling filesystem, developed by IBM
Btrfs	B-tree filesystem, developed by Oracle
msdos	DOS filesystem, supporting only 8-char filenames
umsdos	Extended DOS filesystem used by Linux, compatible with DOS
fat32	MS Windows FAT filesystem
vfat	Extended DOS filesystem, with support for long filenames
ntfs	Replacement for fat32 and vfat filesystems
minix	Native filesystem of the MINIX OS
iso9660	CD-ROM filesystem
cramfs	Compressed RAM disk
nfs	Network filesystem, used to access files on remote machines
SMB	Server Message Block, used to mount MS Windows network shares
proc	Pseudo filesystem, used as an interface to kernel data structures
swap	Pseudo filesystem, Linux swap area

The **swap** space is an area on disk (a file or a partition) used as a RAM extension. When there is not enough free physical RAM for a process, inactive pages in memory are temporarily **swapped out** of memory to disk, to later be **swapped in** to memory when RAM resources are available again. If both RAM and swap space become nearly full, the system may get clogged by spending all the time paging blocks of memory back and forth between RAM and swap (**thrashing**).
The amount of RAM plus the swap is defined as the **virtual memory**.

In Linux, a swap partition is usually preferred over a swap file. While a swap file can be resized more easily, it cannot be used for hibernation; this because the system must first locate the swap file's header, but in order to do so the filesystem containing the swap file must be mounted, and journaled filesystems such as ext3 or ext4 cannot be mounted during resume from disk. Also, in older Linux versions a swap partition used to have faster disk access and less fragmentation than a swap file, although the difference is negligible nowadays.
The swap partition is listed as filesystem type 0x82; however, it is not a filesystem, but a raw addressable memory space with no structure. For this reason it does not appear in the output of `mount` or `df` commands.
A swap partition can be created via any partitioning tool e.g. `fdisk`.

`dd if=/dev/zero of=/swapfile bs=1024 count=512000`	Create a 512-Mb swap file
`mkswap /swapfile`	Initialize a (already created) swap file or partition
`swapon /swapfile`	Enable a swap file or partition, thus telling the kernel that it can use it now
`swapoff /swapfile`	Disable a swap file or partition

`swapon -s`
`cat /proc/swaps`
`cat /proc/meminfo`
`free`
`top`
	Show the sizes of total and used swap areas

How to extend a LVM swap partition

1.	`lvs`	Determine the name of the swap Logical Volume
2.	`swapoff /dev/volgroup0/swap_lv`	Turn off the swap volume
3.	`lvresize -L+1G /dev/volgroup0/swap_lv`	Extend the swap volume with an additional 1 Gb of space
4.	`mkswap /dev/volgroup0/swap_lv`	Format the swap volume
5.	`swapon /dev/volgroup0/swap_lv`	Turn on the swap volume

/etc/fstab

```
# <filesystem>      <mount point>      <type>    <options>                              <dump> <pass>

/dev/sda2           /                  ext2      defaults                                    0  1
/dev/sdb1           /home              ext2      defaults                                    1  2
/dev/cdrom          /media/cdrom       auto      ro,noauto,user,exec                         0  0
/dev/fd0            /media/floppy      auto      rw,noauto,user,sync                         0  0
proc                /proc              proc      defaults                                    0  0
/dev/hda1           swap               swap      pri=42                                      0  0
nfsserver:/dirs     /mnt               nfs       intr                                        0  0
//smbserver/jdoe    /shares/jdoe       cifs      auto,credentials=/etc/smbcreds              0  0
LABEL=/boot         /boot              ext2      defaults                                    0  0
UUID=652b786e-b87f-49d2-af23-8087ced0c667  /test  ext4  errors=remount-ro,noatime            0  0
```

`/etc/fstab` contains information about filesystems, including all filesystems that must be automatically mounted at bootup.		
filesystem	Device or partition. The filesystem can be identified either by its name, label, or UUID	
mount point	Directory on which the partition will be mounted	
type	Filesystem type, or `auto` if detected automatically	
options	`defaults`	Use the default options. The default options depend on the filesystem type and can be found via the command: `tune2fs -l` *`device`* `| grep "Default mount options"` Most common default options: `rw, suid, dev, auto, nouser, exec, async`
	`ro`	Mount read-only
	`rw`	Mount read-write (default)
	`suid`	Permit SUID and SGID bit operations (default)
	`nosuid`	Do not permit SUID and SGID bit operations
	`dev`	Interpret block special devices on the filesystem (default)
	`nodev`	Do not interpret block special devices on the filesystem
	`auto`	Mount automatically at bootup, or when command `mount -a` is given (default)
	`noauto`	Mount only if explicitly demanded
	`user`	Partition can be mounted by any user
	`nouser`	Partition can be mounted only by the root user (default)
	`exec`	Binaries contained on the partition can be executed (default)
	`noexec`	Binaries contained on the partition cannot be executed
	`sync`	Write files immediately to the partition
	`async`	Buffer write operations and commit them at once later, or when device is unmounted (default)
	`noatime`	Do not update atime (access time) information for files. This results in a performance improvement because the system does not need anymore to do filesystem writes for files which are just being read
	`nodiratime`	Do not update atime (access time) information for directories
	`acl`	Support ACLs on files contained in the partition
	`context="`*`context`*`"`	Apply a specific SELinux context to the mount
	Other specific options apply to specific partition types (e.g. NFS or Samba)	
dump	Options for the `dump` backup utility. `0` = do not backup	
pass	Order in which the filesystem must be checked by `fsck`. `0` = do not check	

`df`
Report filesystem disk space usage

`df -h`
Report filesystem disk space usage in human-readable output

`df directory`
Shows on which device the specified *directory* is mounted

`du directory`
Report disk usage, as the size of each file contained in *directory*, in Kb

`du -s directory`
Show the total sum of the sizes of all files contained in *directory*

`du -h directory`
Report disk usage in human-readable output

`du -hs * | sort -hr`
Print out all files and directories in the current directory, ordered by size (the largest first), in human-readable output

`du -a /path | sort -nr | head`
Print out the 10 biggest files and directories under *path*

```
find /path -type f -exec du -Sh {} + \
| sort -hr | head
```
Print out the 10 biggest files under *path*

`ncdu`
Disk usage analyzer with ncurses UI

`duf`
Disk usage analyzer that shows the results in a table format

`resize2fs options device size`
Resize an ext2/ext3/ext4 filesystem

`lsblk`
List information about all available block devices

`lsscsi`
List information about all SCSI devices

`sync`
Flush the buffer and commit all pending writes.
To improve performance of Linux filesystems, many write operations are buffered in RAM and written at once; writes are done in any case before unmount, reboot, or shutdown

`chroot /path/to/newrootdir command`
Run a command in a chroot jail (i.e. in a new root directory). The command process will be unable to access files outside the chroot jail

`chroot /mnt/sysimage`
Start a shell with `/mnt/sysimage` as filesystem root.
Useful during system recovery when the machine has been booted from a removable media; this device is defined as the filesystem root and often needs to be changed to perform operations on the machine

`mknod /dev/sda`
Create a directory allocating the proper inode.
Useful if experiencing filesystem problems during system recovery

`losetup options`
Associate/disassociate a loop device with a regular file or block device, or query the status of a loop device

`multipath options device`
Detect and aggregate multiple I/O paths (SAN connections) to a device

`blkdiscard options device`
Discard device sectors, wiping the data they contain. Useful for SSDs

tune2fs *options device*	Adjust tunable filesystem parameters on ext2/ext3/ext4 filesystems
tune2fs -l /dev/sda1	List the contents of the filesystem superblock
tune2fs -j /dev/sda1	Add a journal to this ext2 filesystem, making it an ext3
tune2fs -m 1 /dev/sda1	Reserve 1% of the partition size to privileged processes. This space (5% by default) is reserved to avoid filesystem fragmentation and to allow privileged processes to continue to run correctly even when the partition is full
tune2fs -C 7 /dev/sda1	Set the mount count of the filesystem to 7
tune2fs -c 20 /dev/sda1	Set the filesystem to be checked by fsck after 20 mounts
tune2fs -i 15d /dev/sda1	Set the filesystem to be checked by fsck each 15 days
tune2fs -l /dev/sda1 | \\ grep "Default mount options"	Print the default mount options for the filesystem

Both mount-count-dependent and time-dependent checking are enabled by default for all hard drives on Linux, to avoid the risk of filesystem corruption going unnoticed.

dumpe2fs *options device*	Dump ext2/ext3/ext4 filesystem information
dumpe2fs -h /dev/sda1	Display filesystem's superblock information (number of mounts, last checks, UUID, etc.)
dumpe2fs /dev/sda1 | grep -i superblock	Display locations of superblock (primary and backup) of filesystem
dumpe2fs -b /dev/sda1	Display blocks that are marked as bad in the filesystem
debugfs *device*	Interactive ext2/ext3/ext4 filesystem debugger
debugfs -w /dev/sda1	Debug `/dev/sda1` in read-write mode (by default, debugfs accesses the device in read-only mode)
e2freefrag /dev/sda1	Report free space fragmentation on a ext2/ext3/ext4 filesystem
filefrag *file*	Display the number of extents into which a file is fragmented
e4defrag -c *directory*	Report filesystem fragmentation
hdparm	Get or set drive parameters for SATA/IDE devices
hdparm -g /dev/hda	Display drive geometry (cylinders, heads, sectors) of `/dev/hda`
hdparm -i /dev/hda	Display identification information for `/dev/hda`
hdparm -tT /dev/hda	Perform disk read benchmarks on the `/dev/hda` drive
hdparm -p 12 /dev/hda	Reprogram IDE interface chipset of `/dev/hda` to mode 4. Warning: using an unsupported mode can cause filesystem corruption
sdparm	Access drive parameters for SCSI devices

Many hard drives feature the **Self-Monitoring, Analysis and Reporting Technology (SMART)** whose purpose is to monitor the reliability of the drive, predict drive failures, and carry out different types of drive self-tests.
The `smartd` daemon attempts to poll this information from all drives every 30 minutes, logging all data to syslog.

smartctl -a /dev/sda	Print SMART information for drive `/dev/sda`
smartctl -s off /dev/sda	Disable SMART monitoring and log collection for drive `/dev/sda`
smartctl -t long /dev/sda	Begin an extended SMART self-test on drive `/dev/sda`

`xfs_growfs` *options mountpoint*	Expand an XFS filesystem. XFS does not support the opposite operation (shrink the filesystem)

`xfs_info /dev/sda1`
`xfs_growfs -n /dev/sda1`
	Print XFS filesystem geometry

`xfs_check` *options device*	Check XFS filesystem consistency
`xfs_repair` *options device*	Repair a damaged or corrupt XFS filesystem
`xfs_db -c frag -r` *device*	Display the level of fragmentation of a XFS filesystem
`xfs_fsr` *device*	Defragment a XFS filesystem
`xfsdump -v silent -f /dev/tape /`	Dump the root of a XFS filesystem to tape, with the lowest verbosity. Incremental and resumed dumps are stored in the inventory database `/var/lib/xfsdump/inventory`
`xfsdump -J - / | xfsrestore -J - /new`	Copy the contents of a XFS filesystem to another directory, without updating the inventory database
`xfsrestore -f /dev/tape /`	Restore a XFS filesystem from tape

`reiserfstune` *options device*	Adjust tunable filesystem parameters on ReiserFS filesystem
`debugreiserfs` *device*	Interactive ReiserFS filesystem debugger

`mkisofs -r -o cdrom.img data/`	Create a CD-ROM image with a ISO9660 filesystem from the contents of the target directory. This command also enables Rock Ridge extension (which contains the original file information for MS Windows 8.3 filenames e.g. permissions, filename) and sets all content on CD to be publicly readable, instead of inheriting the permissions from the original files. Other filesystems used for CD-ROMs are UDF (Universal Disk Format) and HFS (Hierarchical File System). Other CD-ROM filesystem extensions are MS Joliet (to create CD-ROMs more MS Windows compatible) and El Torito (to create bootable CD-ROMs)
`mkudffs /dev/hda`	Create a UDF filesystem
`udffsck /dev/hda`	Check a UDF filesystem
`wrudf /dev/hda`	Maintain a UDF filesystem. Provides an interactive shell
`cdrwtool -d /dev/sr0` *operation*	Manage a CD-RW drive (e.g. disk format, read/write speed)

AutoFS is a client-side service that allows automounting of filesystems, even for nonprivileged users.
AutoFS is composed of the `autofs` kernel module that monitors specific directories for attempts to access them; in this case, the kernel module signals the `automount` userspace daemon, which mounts the directory when it needs to be accessed and unmounts it when is no longer accessed.
Mounts managed by AutoFS should not be mounted/unmounted manually or via `/etc/fstab`, to avoid inconsistencies.

AutoFS configuration files	
`/etc/sysconfig/autofs`	AutoFS configuration file.

`/etc/auto.master`	Master map file for AutoFS.
	Each line is an indirect map, and each map file stores the configuration for the automounting of the subdirectory.
	The `-hosts` map tells AutoFS to mount/unmount automatically any export from the NFS server *nfsserver* when the directory `/net/`*`nfsserver`*`/` is accessed.
	
	`# mount point   map               options`
	`/net            -hosts`
	`/-              /etc/auto.direct`
	`/misc           /etc/auto.misc`
	`/home           /etc/auto.home    --timeout=60`

AutoFS map files	

`/etc/auto.direct`	Direct map file for automounting of a NFS share.
	
	`# dir      filesystem`
	`/mydir     nfsserver1.foo.org:/myshare`

`/etc/auto.misc`	Indirect map file for automounting of directory `/misc`.
	
	`# subdir   options                           filesystem`
	`public     -ro,soft,intr                     ftp.example.org:/pub`
	`cd         -fstype=iso9660,ro,nosuid,nodev   :/dev/cdrom`

`/etc/auto.home`	Indirect map file for automounting of directory `/home` on a NFS share.
	The `*` wildcard matches any subdirectory the system attempts to access, and the `&` variable takes the value of the match.
	
	`# subdir   options                           filesystem`
	`*          -rw,soft,intr                     nfsserver2.bar.org:/home/&`

RAID levels		
Level	**Description**	**Storage capacity**
RAID 0	Striping (data is written across all member disks). High I/O but no redundancy	Sum of the capacity of member disks
RAID 1	Mirroring (data is mirrored on all disks). High redundancy but high cost	Capacity of the smaller member disk
RAID 4	Parity (for fault tolerance) on a single disk. I/O bottleneck unless coupled to write-back caching	Sum of the capacity of member disks, minus one
RAID 5	Parity distributed across all disks. Can sustain one disk crash	Sum of the capacity of member disks, minus one
RAID 6	Double parity distributed across all disks. Can sustain two disk crashes	Sum of the capacity of member disks, minus two
RAID 10 (1+0)	Striping + mirroring. High redundancy but high cost	Capacity of the smaller member disk
Linear RAID	Data written sequentially across all disks. No redundancy	Sum of the capacity of member disks

```
mdadm -C /dev/md0 -l 5 \
-n 3 /dev/sdb1 /dev/sdc1 /dev/sdd1 \
-x 1 /dev/sde1
```

Create a RAID 5 array from three partitions and a spare.
Partitions type must be set to 0xFD.
Once the RAID device has been created, it must be formatted e.g. via `mke2fs -j /dev/md0`

```
mdadm --manage /dev/md0 -f /dev/sdd1
```

Mark a drive as faulty, before removing it

```
mdadm --manage /dev/md0 -r /dev/sdd1
```

Remove a drive from the RAID array.
The faulty drive can then be physically removed

```
mdadm --manage /dev/md0 -a /dev/sdd1
```

Add a drive to the RAID array.
To be run after the faulty drive has been physically replaced

```
mdadm --misc -Q /dev/sdd1
```

Display information about a device

```
mdadm --misc -D /dev/md0
```

Display detailed information about the RAID array

```
mdadm --misc -o /dev/md0
```

Mark the RAID array as read-only

```
mdadm --misc -w /dev/md0
```

Mark the RAID array as read & write

```
/etc/mdadm.conf
```

Configuration file for the `mdadm` command

```
DEVICE /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
ARRAY /dev/md0 level=raid5 num-devices=3
  UUID=0098af43:812203fa:e665b421:002f5e42
  devices=/dev/sdb1,/dev/sdc1,/dev/sdd1,/dev/sde1
```

```
cat /proc/mdstat
```

Display information about RAID arrays and devices

LILO **(Linux Loader)**		Obsolete. Small bootloader that can be placed in the MBR or the boot sector of a partition. The configuration file is `/etc/lilo.conf` (run `/sbin/lilo` afterwards to validate changes).
Syslinux	**SYSLINUX**	Able to boot from FAT and NTFS filesystems e.g. floppy disks and USB drives. Used for boot floppy disks, rescue floppy disks, and Live USBs.

	ISOLINUX	Able to boot from CD-ROM ISO 9660 filesystems.
		Used for Live CDs and bootable install CDs.
		
		The CD must contain the following files:
		`isolinux/isolinux.bin` ISOLINUX image, from the SYSLINUX distro
		`boot/isolinux/isolinux.cfg` ISOLINUX configuration
		`images/` Floppy images to boot
		`kernel/memdisk`
		
		and can be burnt with the command:
		`mkisofs -o output.iso -b isolinux/isolinux.bin -c isolinux/boot.cat \`
		`-no-emul-boot -boot-load-size 4 -boot-info-table` *`cd_root_dir`*

	PXELINUX	Able to boot from PXE (Pre-boot eXecution Environment). PXE uses DHCP or BOOTP to enable basic networking, then uses TFTP to download a bootstrap program that loads and configures the kernel.
		Used for Linux installations from a central server or network boot of diskless workstations.
		
		The boot TFTP server must contain the following files:
		`/tftpboot/pxelinux.0` PXELINUX image, from the SYSLINUX distribution
		`/tftpboot/pxelinux.cfg/` Directory containing a configuration file for each machine. A machine with Ethernet MAC address 88:99:AA:BB:CC:DD and IP address 192.0.2.91 (C000025B in hexadecimal) will search for its configuration filename in this order:
		`01-88-99-aa-bb-cc-dd`
		`C000025B`
		`C000025`
		`C00002`
		`C0000`
		`C000`
		`C00`
		`C0`
		`C`
		`default`

	EXTLINUX	General-purpose bootloader like LILO or GRUB. Now merged with SYSLINUX.

GRUB (Grand Unified Bootloader) is the standard boot manager on Linux distributions.
The latest version is **GRUB 2**; the previous version is known as **GRUB Legacy** or **GRUB 1**.

In GRUB 2, the GRUB bootstrap code i.e. GRUB Stage 1 (446 bytes) is stored in the 512-byte MBR; the MBR also contains the partition table (64 bytes) and the boot signature (2 bytes).
Once booted, GRUB Stage 1 locates and executes GRUB Stage 1.5.
GRUB Stage 1.5 contains common filesystem drivers necessary to locate GRUB Stage 2.
GRUB Stage 2 accesses the GRUB 2 configuration and commands stored in `/boot/grub2`. It loads one of the available Linux kernels into RAM and passes control to it.

`/boot/grub/grub.cfg` or `/boot/grub2/grub.cfg`   GRUB 2 configuration file

```
# Linux Red Hat
menuentry "Fedora 2.6.32" {   # Menu item to show on GRUB bootmenu
set root=(hd0,1)              # root filesystem is /dev/hda1
linux /vmlinuz-2.6.32 ro root=/dev/hda5 mem=2048M
initrd /initrd-2.6.32
}

# Linux Debian
menuentry "Debian 2.6.36-experimental" {
set root=(hd0,1)
linux (hd0,1)/bzImage-2.6.36-experimental ro root=/dev/hda6
}

# MS Windows
menuentry "MS Windows" {
set root=(hd0,2)
chainloader +1
}
```

The GRUB 2 configuration file must not be edited manually. Instead, it is necessary to edit the files in `/etc/grub.d/` (scripts that will be run in order) and the file `/etc/default/grub` (configuration file for menu display settings), then run `update-grub` (Debian) or `grub2-mkconfig` (Red Hat) which will recreate this configuration file.

GRUB 2 configuration - Common kernel parameters	
`root=`	Specify the location of the filesystem root. This is a required parameter
`ro`	Mount read-only on boot
`quiet`	Disable non-critical kernel messages during boot
`debug`	Enable kernel debugging
`splash`	Show splash image
`single`	Boot in single-user mode (runlevel 1)
`emergency`	Emergency mode: after the kernel is booted, run `sulogin` (single-user login) which asks for the root password for system maintenance, then run a Bash shell. Does not load `init` or any daemon or configuration setting
`init=/bin/bash`	Run a Bash shell (may also be any other executable) instead of `init`

The GRUB menu, presented at startup, allows to choose the OS or kernel to boot:

ENTER	Boot the currently selected GRUB entry
C	Get a GRUB command line
E	Edit the selected GRUB entry (e.g. to edit kernel parameters in order to boot in single-user emergency mode, or to change IRQ or I/O port of a device driver compiled in the kernel)
B	Boot the currently selected GRUB entry. This is usually done after finishing modifying the entry
P	Bring up the GRUB password prompt. Necessary if a GRUB password has been set

`grub2-mkconfig -o /boot/grub2/grub.cfg`
`grub2-mkconfig -o /boot/efi/EFI/centos/grub.cfg`
	(BIOS)
	(EFI)
		Regenerate GRUB configuration file

`grub-install /dev/sda`	Install GRUB on first SATA drive
`grub`	Access the GRUB shell
`grub2-set-default 1`	Set GRUB to automatically boot the second entry in the GRUB menu
`grub2-editenv list`	Display the current GRUB menu entry that is automatically booted

`/boot/grub/device.map` — This file can be created to map Linux device filenames to BIOS drives

```
(fd0)   /dev/fd0
(hd0)   /dev/hda
```

`/boot/grub/menu.lst` or `/boot/grub/grub.conf` **GRUB Legacy configuration file**

```
timeout 10    # Boot the default kernel after 10 seconds
default 0     # Default kernel is 0

# Section 0: Linux boot
title   Debian     # Menu item to show on GRUB bootmenu
root    (hd0,0)    # root filesystem is /dev/hda1
kernel  /boot/vmlinuz-2.6.24-19-generic root=/dev/hda1 ro quiet splash
initrd  /boot/initrd.img-2.6.24-19-generic

# Section 1: MS Windows boot
title   Microsoft Windows XP
root    (hd0,1)    # root filesystem is /dev/hda2
savedefault
makeactive         # set the active flag on this partition
chainloader +1     # read 1 sector from start of partition and run

# Section 2: Firmware/BIOS update from floppy disk
title   Firmware update
kernel /memdisk    # boot a floppy disk image
initrd /floppy-img-7.7.7
```

GRUB Legacy shell commands	
blocklist *file*	Print the block list notation of a file
boot	Boot the loaded OS
cat *file*	Show the contents of a file
chainloader *file*	Chainload another bootloader
cmp *file1 file2*	Compare two files
configfile *file*	Load a configuration file
debug	Toggle debugging mode
displayapm	Display APM BIOS information
displaymem	Display memory configuration
embed *stage device*	Embed Stage 1.5 in the device
find *file*	Find a file
fstest	Toggle filesystem test mode
geometry *drive*	Print information on a drive geometry
halt	Shut down the system
help *command*	Show help for a command, or the available commands
impsprobe	Probe the Intel Multiprocessor Specification
initrd *file*	Load an initial ramdisk image file
install *options*	Install GRUB (deprecated; `setup` should be used instead)
ioprobe *drive*	Probe I/O ports used for a drive
kernel *file*	Load a kernel
lock	Lock a GRUB menu entry
makeactive	Set active partition on root disk to GRUB's root device
map *drive1 drive2*	Map a drive to another drive
md5crypt	Encrypt a password in MD5 format
module *file*	Load a kernel module
modulenounzip *file*	Load a kernel module without decompressing it
pause *message*	Print a message and wait for a key press
quit	Quit the GRUB shell
reboot	Reboot the system
read *address*	Read a 32-bit value from memory and print it
root *device*	Set the current root device
rootnoverify *device*	Set the current root device without mounting it
savedefault	Save current menu entry as the default entry
setup *device*	Install GRUB automatically on the device
testload *file*	Test the filesystem code on a file
testvbe *mode*	Test a VESA BIOS EXTENSION mode
uppermem *kbytes*	Set the upper memory size (only for old machines)
vbeprobe *mode*	Probe a VESA BIOS EXTENSION mode

dpkg and apt

dpkg is the low-level package manager for Debian.
It uses the DEB package format, which is compressed with ar.

dpkg -i *package*.deb	Install a package file
dpkg -r *package*	Remove a package
dpkg -l	List installed packages and their state
dpkg -L *package*	List the content of an installed package
dpkg -c *package*.deb	List the content of a package file
dpkg -S *file*	Show the package containing a specific file
dpkg-reconfigure *package*	Reconfigure a package

apt is the high-level package manager for Debian.
High-level package managers are able to install remote packages and automatically solve dependencies.

apt-get install *package*	Install a package
apt-get remove *package*	Remove a package
apt-get upgrade	Upgrade all installed packages
apt-get dist-upgrade	Upgrade all installed packages and handle dependencies with new versions
apt-get source *package*	Get the source code for a package
apt-get check	Check for broken dependencies and update package cache
apt-get install -f	Fix broken dependencies
apt-get update	Update information on available packages
apt-cache search *package*	Search for a package
apt-cache depends *package*	Show package dependencies
apt-cache show *package*	Show package records
apt-cache showpkg *package*	Show information about a package
apt-file update	Update information about package contents
apt-file list *package*	List the content of an uninstalled package
apt-file search *file*	Show which package provides a specific file
apt-key add *keyfile*	Add a key to the list of keys used to authenticate packages
apt-cdrom add	Add a CD-ROM to the sources list
cat /etc/apt/sources.list	Print list of available repositories

alien -i *package*.rpm	Convert an RPM package to DEB and install it. Warning: might break the package database system

dselect	Package manager with text interface, front-end to dpkg. Obsolete
aptitude	Package manager with ncurses UI, front-end to apt
synaptic	Package manager with Gtk+ UI, front-end to apt

rpm is the low-level package manager for Red Hat.
It uses the RPM package format, which is cpio-compressed.

rpm -i *package*.rpm rpm -i ftp://*host*/*package*.rpm rpm -i http://*host*/*package*.rpm	Install a package file
rpm -e *package*	Remove a package
rpm -U *package*.rpm	Upgrade a package (and remove old versions)
rpm -F *package*.rpm	Upgrade a package (only if an old version is already installed)
rpm -qa	List installed packages and their state
rpm -qa --last	List installed packages and their installation date, from newest to oldest
rpm -ql *package*	List the content of an installed package
rpm -qpl *package*.rpm	List the content of a package file
rpm -qf *file*	Show the package containing a specific file
rpm -V *package*	Verify an installed package
rpm -i *package*.src.rpm	Install a package source file
rpm -ba *package*.spec	Compile a package source file
rpm -U --root */path package*.rpm	Install a package using an alternative root directory (this is useful e.g. if the system has been booted from a removable media)

rpm2cpio *package*.rpm	Convert an RPM package to a cpio archive
createrepo *directory*	Create an XML file of repository metadata from the set of RPMs contained in *directory*

pirut	Package manager with GUI. Obsolete

whohas *package*	Query multiple package lists to display which version of *package* is available for different distros

dnf and yum

`dnf` is the default high-level package manager for RHEL 8.
It mainly maintains the same CLI options as its predecessor `yum`, which was the default high-level package manager from RHEL 5 to RHEL 7.

`dnf install` *`package`*	Install a package

`dnf install` *`package`*`.rpm`
`dnf localinstall` *`package`*`.rpm`
	Install a package file

`dnf remove` *`package`*	Remove a package
`dnf update` *`package`*	Update an installed package
`dnf update`	Update all installed packages

`dnf upgrade`
`dnf update --obsoletes`
	Update all installed packages and remove obsolete packages

`dnf swap` *`packageout packagein`*	Replace a package with another
`dnf list`	List all installed and available packages
`dnf list` *`searchterm`*	List installed and available packages matching the search term
`dnf list installed`	List installed packages
`dnf list available`	List packages available for install
`dnf search` *`searchterm`*	Search for packages that match the search term in the package name or summary
`dnf search all` *`searchterm`*	Search for packages that match the search term in the package name, summary, or description
`dnf deplist` *`package`*	Show package dependencies (recursively)
`dnf list` *`package`*	Show package records
`dnf info` *`package`*	Show information about a package

`dnf history`
`dnf history list`
	Show the installation history (installs, updates, etc.)

`dnf history list` *`n`*	Show item *n* of the installation history
`dnf history info` *`n`*	Show detailed information on item *n* of the installation history (begin and end times, packages altered, etc.)

`dnf history package` *`package`*
`dnf history list package` *`package`*
	Show the installation history about a package

`dnf whatprovides` *`file`*
`dnf provides` *`file`*
	Show which package provides a specific file

`dnf` *`cmd`* `--disablerepo="*" --enablerepo="`*`repo`*`"`	Execute the specified `dnf` command but only with a specific repository *repo* enabled

`dnf repolist`
`cat /etc/yum.repos.d/*.repo`
	Print list of available repositories

`dnf clean all`
`rm -rf /var/cache/dnf`
	Delete temporary files for repositories

`yumdownloader --resolve` *`package`*	Download package and all its dependencies
`yumdownloader --urls` *`package`*	Show URLs that would be downloaded
`yum-complete-transaction`	Try to complete unfinished or aborted package installations
`repoquery --tree-requires` *`package`*	Show a tree with all dependencies of *package*

Configuration of a yum repository	
`[fedora]`	Repository ID
`name=Fedora $releasever - $basearch`	Repository name

`baseurl=http://download.fedoraproject.org/pub/fedora/\`
`        linux/releases/$releasever/Everything/$basearch/os/`
`        http://foo.org/linux/$releasever/$basearch/os/`
`        http://bar.org/linux/$releasever/$basearch/os/`
	List of URLs to the repository's repodata directory. Can be any of these types:
	`file:///` local file
	`file://` NFS
	`http://` HTTP
	`https://` HTTPS
	`ftp://` FTP

`enabled=1`	Whether this repository is enabled
`gpgcheck=1`	Whether to perform a GPG signature check on the packages downloaded from this repository
`failovermethod=priority`	Makes yum try the baseurls in the order they are listed. By default, if more than one baseurl is specified, yum chooses one randomly

`metalink=https://mirrors.fedoraproject.org/metalink?\`
`repo=fedora-$releasever&arch=$basearch`
	URL to a metalink file that specifies the list of mirrors to use. Can be used with or in alternative to a baseurl

`gpgkey=file:///etc/pki/rpm-gpg/\`
`RPM-GPG-KEY-fedora-$releasever-$basearch`
	ASCII-armored GPG public key file of the repository

This repository configuration must be located in a repo file e.g. `/etc/yum.repos.d/fedora.repo`. The same repo file can contain multiple repository definitions.
The manpage `man yum.conf` lists all repository configuration options.

How to install a package on an offline machine

When installing a package on an offline machine, the machine is obviously unable to download the package dependencies. To solve this problem, first create an online machine identical to the offline machine, and with the smallest possible set of packages installed. Then proceed as described below.
On the online machine:

1. Install the package and all its dependencies in a local directory

```
mkdir /tmp/repo
yum --downloadonly --downloaddir=/tmp/repo install package
```

2. Create a local yum repository from the contents of the local directory

```
createrepo /tmp/repo
chown -R root:root /tmp/repo
chmod -R 755 /tmp/repo
```

3. Transfer the directory `/tmp/repo` from the online machine to the offline machine

On the offline machine:

4. Create a yum repo file `/etc/yum.repos.d/local.repo` for the new repository

```
[local]
name=Local
baseurl=file:///tmp/repo
enabled=1
gpgcheck=0
protect=1
```

5. Install the package from the local repository

```
yum install package
```

Other package managers

`snap` is a distro-independent software packaging and deployment system created by Canonical.
Programs are distributed as self-contained packages called **snaps**, and usually made available through the Snap Store.
Snaps run in a sandbox, for security reasons.
The snap file format is a single compressed SquashFS filesystem.

`snap install` *snap*	Install a *snap*
`snap info` *snap*	Get information about *snap*
`snap find "`*searchterm*`"`	Find a snap

`opkg` is a lightweight package manager used on embedded Linux systems. It is included in the OpenEmbedded and OpenWrt projects.

`opkg install` *package*	Install a package
`opkg remove` *package*	Uninstall a package

`pacman` is Arch Linux's default package manager.

`pacman -S` *package*	Install a package
`pacman -R` *package*	Uninstall a package

`npm` is the package manager for the JavaScript runtime environment Node.js.

`npm install` *package*	Install a package
`npm uninstall` *package*	Uninstall a package

Homebrew is a package manager originally for Apple macOS, but it has been ported on Linux as well.

`brew install` *package*	Install a package
`brew uninstall` *package* `brew remove` *package*	Uninstall a package

dd	Tool to copy data, byte by byte, from a file or block device. It should not be used on a mounted block device, because of write cache issues
dd if=/dev/sda of=/dev/sdb cat /dev/sda > /dev/sdb	Copy the content of one hard disk over another
dd if=/dev/sda1 of=sda1.img	Generate the image file of a partition
dd if=/dev/cdrom of=cdrom.iso bs=2048	Create an ISO file from a CD-ROM, using a block size transfer of 2 Kb
dd if=install.iso of=/dev/sdc bs=512k	Write an installation ISO file to a device (e.g. a USB thumb drive)
ddrescue	Data recovery tool. Like dd, but with high tolerance for read errors
testdisk	Data recovery tool. Recovers data from a deleted or corrupted partition
photorec	Data recovery tool. Recovers graphical image and video files from media such as digital cameras and CD-ROMs
ext3grep	Data recovery tool. Recovers deleted files from a EXT3 filesystem
extundelete	Data recovery tool. Recovers deleted files from a EXT3 or EXT4 filesystem
ext4magic	Data recovery tool. Recovers deleted files from a EXT3 or EXT4 filesystem
ntfsundelete	Data recovery tool. Recovers deleted files from a NTFS filesystem
scalpel	Data recovery tool. Recovers data from a disk image or a raw block device
rsync	Tool for local and remote file synchronization. For all copies after the first, copies only the blocks that have changed, making it a very fast and bandwidth-efficient backup solution
rsync -rzv /home /tmp/bak rsync -rzv /home/ /tmp/bak/home	Synchronize the content of the home directory with the temporary backup directory; use recursion, compression, and verbosity
rsync -avz /home root@10.0.0.7:/bak/	Synchronize the content of the home directory with the backup directory on the remote server via SSH; use archive mode (operates recursively and preserves owner, group, permissions, timestamps, and symlinks)
rclone	Tool for cloud storage management. Backups, restores, mirrors, and migrates files from and to a large number of cloud providers (e.g. Ceph, Amazon S3, ownCloud, Google Drive, Microsoft Azure, OpenStack Swift)
burp	Backup and restore program

`/dev/st0`	First SCSI tape device
`/dev/nst0`	First SCSI tape device (no-rewind device file)

`mt`	Utility for magnetic tapes
`mt -f /dev/nst0 asf 3`	Position the tape at the start of the 3rd file

`mtx`	Utility for tape libraries
`mtx -f /dev/sg1 status`	Display status of tape library
`mtx -f /dev/sg1 load 3`	Load tape from slot 3 to drive 0
`mtx -f /dev/sg1 unload`	Unload tape from drive 0 to original slot
`mtx -f /dev/sg1 transfer 3 4`	Transfer tape from slot 3 to slot 4
`mtx -f /dev/sg1 inventory`	Force robot to rescan all slots and drives
`mtx -f /dev/sg1 inquiry`	Inquiry about SCSI media device (Medium Changer = tape library)

cpio	ls | cpio -o > *archive*.cpio ls | cpio -oF *archive*.cpio	Create a cpio archive of all files in the current directory
	find /home/ | cpio -o > *archive*.cpio	Create a cpio archive of all users' home directories
	cpio -id < *archive*.cpio	Extract all files, recreating the directory structure
	cpio -i -t < *archive*.cpio	List the contents of a cpio archive file
gzip	gzip *file*	Compress a file with gzip
	gzip < *file* > *file*.gz	Compress a file with gzip, leaving the original file into place
	gunzip *file*.gz	Decompress a gzip-compressed file
	gunzip -tv *file*.gz	Test the integrity of a gzip-compressed file
	zcat *file*.gz	Read a gzip-compressed text file
	zgrep *pattern* *file*.gz	grep for a gzip-compressed text file
	zless *file*.gz	less for a gzip-compressed text file
	zmore *file*.gz	more for a gzip-compressed text file
	pigz *file*	Parallel, multicore-optimized gzip
bzip2	bzip2 *file*	Compress a file with bzip2
	bunzip2 *file*.bz2	Decompress a bzip2-compressed file
	bzcat *file*.bz2	Read a bzip2-compressed text file
7-Zip	7z a -t7z *archive*.7z *dir/*	Create a 7-Zip archive (has the highest compression ratio)
xz	xz *file*	Compress a file with xz
	unxz *file*.xz xz -d *file*.xz	Decompress a xz-compressed file
	xzcat *file*.xz	Read a xz-compressed file
LZMA	lzma *file* xz --format=lzma *file*	Compress a file with LZMA
	unlzma *file*.lzma xz --format=lzma -d *file*.lzma	Decompress a LZMA-compressed file
	lzcat *file*.lzma xz --format=lzma --d --stdout *file*.lzma	Read a LZMA-compressed file
rar	rar a *archive*.rar *dir/*	Create a RAR archive
	unrar x *archive*.rar	Extract a RAR archive
tar	tar cf *archive*.tar *dir/*	Create a tarred archive
	tar czf *archive*.tar.gz *dir/*	Create a tarred gzip-compressed archive
	tar xzf *archive*.tar.gz	Extract a tarred gzip-compressed archive
	tar cjf *archive*.tar.bz2 *dir/*	Create a tarred bzip2-compressed archive
	tar xjf *archive*.tar.bz2	Extract a tarred bzip2-compressed archive
	tar cJf *archive*.tar.xz *dir/*	Create a tarred xz-compressed archive
	tar xJf *archive*.tar.xz	Extract a tarred xz-compressed archive
	tar tf *archive*.tar	List the contents of a tarred archive
star	star -c -f=*archive*.star *dir/*	Create a star archive
	star -x -f=*archive*.star	Extract a star archive

`man` *command*	Show the **manpage** (manual page) for *command*
`man` *n* *command*	Show section *n* of the *command* manpage
`man man`	Show information about manpages' sections: 1 - Executable programs or shell commands 2 - System calls (functions provided by the kernel) 3 - Library calls (functions within program libraries) 4 - Special files 5 - File formats and conventions 6 - Games 7 - Miscellaneous 8 - System administration commands (for root only) 9 - Kernel routines
`man` *n* `intro`	Show an introduction to the contents of section *n*
`mandb`	Generate or refresh the search database for manpage entries. This must be done after installing new packages, in order to obtain results from `apropos` or `man -k`
`whatis` *command*	Show the manpage's short description for *command*
`apropos` *keyword* `man -k` *keyword*	Show the commands whose manpage's short description matches *keyword*. Inverse of the `whatis` command
`apropos -r` *regex* `man -k` *regex*	Show the commands whose manpage's short description matches *regex*
`man -K` *regex*	Show the commands whose manpage's full text matches *regex*
`info` *command*	Show the **Info** documentation for *command*
`catman` *section*	Create or update **cat pages**, a kind of preformatted manual pages (obsolete)
`help`	Show the list of available shell commands and functions
`help` *command*	Show help about a shell command or function
`command -v` *command*	Show the full path or alias for *command*. If no path is shown, *command* is a shell built-in
`dnf whatprovides /usr/share/man/mann/`*command*`.`*n*`.gz` (Red Hat)	Find which package provides section *n* of the *command* manpage
`dnf install man-pages` (Red Hat)	Install a large number of manpages from the Linux Documentation Project

`cat` *file*	Print a text file
`cat` *file1 file2* `>` *file3*	Concatenate text files
`cat` *file1* `>` *file2* `>` *file2* `<` *file1* `cat`	Copy *file1* to *file2*. The `cat` command is able to operate on binary streams as well and therefore this command also works on binary files (e.g. JPG images)
`cat >` *file* `<<`*EOF* *line 1* *line 2* *line 3* *EOF*	Create a **Here Document**, storing the lines entered in input to *file*. *EOF* can be any text
command `<<< '`*string*`'`	Create a **Here String**, passing *string* as input to *command*
`cat -etv <<< '`*string*`'`	Print *string*, showing all invisible characters
`bat`	Improved version of `cat` with syntax highlighting, auto paging, and other features
`tac` *file*	Print or concatenate text files in opposite order line-wise, from last line to first line
`rev` *file*	Print a text file with every line reversed character-wise, from last char to first char
`head` *file* `head -n 10` *file*	Print the first 10 lines of a text file
`tail` *file* `tail -n 10` *file*	Print the last 10 lines of a text file
`tail -f` *file*	Output appended data as the text file grows. Useful to read a logfile in real-time
`tail -n +1` *file1 file2 file3*	Print each file with a filename header
`multitail -i` *file1* `-i` *file2*	`tail` for multiple files at the same time (ncurses UI)
`column` *file*	Format a text file into columns
`pr` *file*	Format a text file for a printer
`fmt -w 75` *file*	Format a text file so that each line has a max width of 75 characters
`fold -w40` *file*	Wrap each line of a text file to 40 characters
`nl` *file*	Prepend line numbers to a text file
`expand` *file*	Convert tabs into spaces
`unexpand` *file*	Convert spaces into tabs
`join` *file1 file2*	Join lines of two text files on a common field
`paste` *file1 file2*	Merge lines of text files
`split -l 1` *file*	Split a text file into 1-line files; these will be named `xaa`, `xab`, `xac`, etc.
`sort` *file*	Sort alphabetically the lines of a text file
`shuf` *file*	Shuffle randomly the lines of a text file
`wc` *file*	Print the number of lines, words, and bytes of a text file
`uniq` *file*	Print unique lines of a text file, printing consecutive identical lines only once
`uniq -u` *file*	Print lines of a text file that occur only once and are not adjacent to identical lines
`diff` *file1 file2*	Compare two text files line by line and print the differences
`diff-so-fancy`	Improved version of `diff` with better readability

cut -d: -f3 *file*	Cut the lines of a file, considering : as the delimiter and printing only the 3rd field
cut -d: -f1 /etc/passwd	Print the list of local user accounts in the system
cut -c3-50 *file*	Print character 3 to 50 of each line of a file
sed 's/foo/bar/' *file*	Stream Editor: Replace the first occurrence on a line of "foo" with "bar" in *file*, and print on stdout the result
sed -i 's/foo/bar/' *file*	Replace "foo" with "bar", overwriting the results in *file*
sed 's/foo/bar/g' *file*	Replace all occurrences of "foo" with "bar"
sed '0,/foo/s//bar/' *file*	Replace only the first line match
sed -n '7,13p' *file*	Print line 7 to 13 of a text file
sed -n '$p' *file* sed '$!d' *file*	Print the last line of a text file
sed "s/foo/$var/" *file*	Replace "foo" with the value of variable $var. The double quotes are necessary for variable expansion
tr a-z A-Z <*file* tr [:lower:] [:upper:] <*file*	Translate characters: Convert all lowercase into uppercase in a text file
tr -d 0-9 <*file* tr -d [:digit:] <*file*	Delete all digits from a text file
tr '[a-zA-Z]' '[n-za-mN-ZA-M]' <*file*	Perform ROT13 encoding (i.e. decoding) of a text file
awk	Interpreter for the AWK programming language, designed for text processing and data extraction
rpl *oldstring newstring file*	Replace strings in a file
grep foo *file*	Print the lines of a file containing "foo"
grep -v foo *file*	Print the lines of a file not containing "foo"
grep -e foo -e bar *file* grep -E 'foo|bar' *file*	Print the lines of a file containing "foo" or "bar"
grep -v -e foo -e bar *file*	Print the lines of a file containing neither "foo" nor "bar"
grep -E *regex file* egrep *regex file*	Print the lines of a file matching the given Extended Regex
tail -f *file* | grep --line-buffered foo tail -f *file* | stdbuf -o0 grep foo	Output appended data as the text file grows, printing only the lines containing "foo"
look *string file*	Display lines from *file* (which must be sorted) beginning with the specified *string*. If *file* is not specified, /usr/share/dict/words is used
strings *file*	Show all printable character sequences at least 4-characters long contained in *file*

uuencode *file*	Encode a file using only printing ASCII characters. Used to transmit a binary file over a medium that supports only ASCII data, e.g. e-mail
uuencode -m *file*	Encode a file to Base64
uudecode *file*	Decode a file
base64 *file*	Encode a file to Base64
base64 -d *file*	Decode a file from Base64
od *file*	Dump a file into octal (or other formats)
hexdump *options file*	Dump a file into hexadecimal (or octal, decimal, ASCII)
xxd *options file*	Convert a file from binary to hexadecimal (i.e create a hex dump), or vice versa
cmp *file1 file2*	Compare two files byte by byte and print the differences. Like `diff`, but it operates on binary files
tidy	Correct and tidy up the markup of HTML, XHTML, and XML files
tidy -asxml -xml -indent -wrap 2000 \ -quiet --hide-comments yes *file.xml*	Strip out comments from an XML file
json_verify < *file.json*	Validate the syntax of a JSON file
json_reformat < *file.json*	Pretty format a JSON file
jq	JSON processor
fx	JSON viewer and processing tool
pandoc *options file*	Convert a file from a markup format to another (e.g. HTML, LaTeX, RTF, Markdown, OpenDocument XML, ODT, DOCX, PDF, EPUB)
antiword *file.doc*	Show text and images from a MS Word document
catdoc *file.doc*	Output plaintext from a MS Word document

`^`	Beginning of a line
`$`	End of a line
`\< \>`	Word boundaries (beginning of line, end of line, space, or punctuation mark)
`.`	Any character except newline
`[abc]`	Any of the characters specified
`[a-z]`	Any of the characters in the specified range
`[^abc]`	Any character except those specified
`*`	Zero or more times the preceding regex
`+`	One or more times the preceding regex
`?`	Zero or one time the preceding regex
`{5}`	Exactly 5 times the preceding regex
`{5,}`	5 times or more the preceding regex
`{,10}`	At most 10 times the preceding regex
`{5,10}`	Between 5 and 10 times the preceding regex
`|`	The regex either before or after the vertical bar
`( )`	Grouping, to be used for back-references. `\1` expands to the 1st match, `\2` to the 2nd, etc. up to `\9`

The symbols above are used in POSIX EREs (Extended Regular Expressions).
In POSIX BREs (Basic Regular Expressions), the symbols `? + { | ( )` need to be escaped; this is done by adding a backslash character `\` in front of them.

cp *file file2*	Copy a file
cp *file dir/*	Copy a file to a directory
cp -ar */dir1/.* */dir2/*	Copy a directory recursively
mv *file file2*	Rename a file
mv *file dir/*	Move a file to a directory
rm *file*	Delete a file

Common options:
-i Prompt before overwriting/deleting files (interactive)
-f Don't ask before overwriting/deleting files (force)

pv *file* > *file2*	Copy a file, monitoring the progress of data through a pipe
rename *str1 str2 file*	Rename a file, replacing in the filename the first occurrence of string *str1* with *str2*
rename .htm .html *.htm	Rename all .htm files to .html
unlink *file*	Remove the hard link to a file (equivalent to rm)
touch *file*	Change access timestamp and modify timestamp of a file as now. If the file does not exist, it is created
truncate -s *size file*	Shrink or extend a file to the specified size. If the file is larger than the specified size, it is truncated; if the file is shorter, the extra space is filled with zeros
mktemp	Create a temporary file or directory, using tmp.XXXXXXXXXX as filename template
fdupes *dir*	Examine a directory for duplicate files in it. To consider files a duplicate, it first compares file sizes and MD5 signatures, then file contents byte-by-byte
shred /dev/hda	Securely wipe the contents of a device
shred -u *file*	Securely delete a file

File-naming wildcards (globbing)	
*	Matches zero or more characters
?	Matches one character
[abc]	Matches a, b, or c
[!abc]	Matches any character except a, b, or c
[a-z]	Matches any character between a and z

Brace expansion	
cp foo.{txt,bak}	Copy file "foo.txt" to "foo.bak"

touch foo_{a,b,c}
touch foo_{a..c}
	Create files "foo_a", "foo_b", "foo_c"

Directory management

cd *directory*	Change to the specified directory
cd -	Change to the previously used directory
pwd	Print the current working directory
ls dir vdir	List the contents of the current directory
ls -d */	List only directories contained on the current directory
ls -lap --sort=v	List files, sorted by version number
mkdir *dir*	Create a directory
mkdir -m 755 *dir*	Create a directory with mode 755
mkdir -p */dir1/dir2/dir3*	Create a directory, creating also the parent directories if they don't exist
rmdir *dir*	Delete a directory (which must be empty)
tree	List directories and their contents in hierarchical format
dirs	Display the directory stack (i.e. the list of remembered directories)
pushd *dir*	Add *dir* to the top of the directory stack and make it the current working directory
popd	Remove the top directory from the directory stack and change to the new top directory
dirname *file*	Output the directory path in which *file* is located, stripping any non-directory suffix from the filename
realpath *file*	Output the resolved absolute path of *file*

Bash directory shortcuts	
.	Current directory
..	Parent directory
~	Home directory of current user
~*user*	Home directory of *user*
~-	Previously used directory

lsof	List all open files
lsof -u *user*	List all files currently open by *user*
lsof -i	List open files and their sockets (equivalent to netstat -ap)
lsof -i :80	List connections of local processes on port 80
lsof -iTCP:70-90	List connections of local processes on TCP ports between 70 and 90
lsof -i@10.0.0.3	List connections of local processes to remote host 10.0.0.3
lsof -i@10.0.0.3:80	List connections of local processes to remote host 10.0.0.3 on port 80
lsof -c mysqld	List all files opened by mysqld, the MySQL daemon
lsof *file*	List all processes using a specific *file*
lsof +L1	List open files with a link count smaller than 1 i.e. that have been unlinked. These files are not accessible but take up disk space. A process holding such a file prevents the system from deleting it (thus freeing disk space), until the process is killed or restarted
fuser	Show the name of processes using a specific file, directory, or socket
fuser -v *file*	Show the name of the process using *file*
fuser -v -n tcp 443	Show the name of the process running on port 443
lslocks	List information about all currently held file locks
lslk	List information about all locks currently held on files with local inodes
tmpwatch	Remove files which have not been accessed for some time
stat *file*	Display file or filesystem status
stat -c %A *file*	Display file permissions
stat -c %s *file*	Display file size, in bytes
crc32 *file*	Calculate the CRC-32 checksum of *file*. This is only used for error detection in transmission and storage, not to detect malicious modifications to the file (because the CRC-32 checksum is not a cryptographic hash)

In Linux, everything is (displayed as) a file. **File descriptors** are unique identifiers for any I/O resource e.g. a file, pipe, or network socket; they are automatically associated to any process launched.

Standard POSIX file descriptors

#	Name	Type	Default device	Device file
0	Standard input (stdin)	Input text stream	Keyboard	`/dev/stdin`
1	Standard output (stdout)	Output text stream	Terminal	`/dev/stdout`
2	Standard error (stderr)	Output text stream	Terminal	`/dev/stderr`

`mail user@email < file`	Redirect *file* to the stdin of command `mail` (in this case, send via e-mail the contents of *file* to the email address *user@email*). Redirection is handled by the shell, not by the command invoked. The space after the redirection operator can be omitted

`ls > file`
`ls 1> file`
	Redirect the stdout of command `ls` to *file* (in this case, write on *file* the contents of the current directory). This overwrites *file* if it already exists, unless the Bash noclobber option is set (via `set -o noclobber`)

`ls >| file`	Redirect the stdout of command `ls` to *file*, even if noclobber is set

`ls >> file`
`ls 1>> file`
	Append the stdout of command `ls` to *file*

`ls 2> file`	Redirect the stderr of command `ls` to *file* (in this case, write any error encountered by the command `ls` to *file*)
`ls 2>> file`	Append the stderr of command `ls` to *file*
`ls 2> /dev/null`	Silence any error coming from the command `ls`

`cat <file1 >file2`
`<file1 cat >file2`
`<file1 >file2 cat`
	Redirect *file1* to the stdin and *file2* to the stdout of the command `cat` (in this case, copy *file1* to *file2*).
	`cat >file2 <file1` also works, but is not recommended because it truncates *file2* if for any reason *file1* cannot be opened

`cat /etc/passwd | wc -l`	Pipe the stdout of command `cat` to the stdin of command `wc` (in this case, print the number of accounts in the system). Piped commands run concurrently

`echo "$(sort file)" >file`
``echo "`sort file`" >file``
`sort file | sponge file`
	Sort the contents of *file* and write the output to the file itself.
	`sort file > file` would not produce the desired result, because the stdout destination is created (and therefore the content of the preexisting *file* is deleted) before the `sort` command is run

`ls 2>&1`	Redirect stderr of command `ls` to stdout
`ls >file 2>&1`	Redirect both stdout and stderr of command `ls` to *file*. Commands `ls &> file` and `ls >& file` also work on some systems, but are not recommended because they are not POSIX standard
`>file`	Create an empty file. If the file exists, its content will be deleted
`tee file`	Read from stdin and write both to stdout and *file*
`tee -a file`	Read from stdin and append both to stdout and *file*
`ls | tee file`	Write the contents of the current directory to screen and to *file* at the same time

`stdbuf` *option command*	Run *command* with modified stdin, stdout, or stderr buffering
`sponge` *file*	Read from stdin and write to *file*, absorbing all input before opening the output file for writing
`ifne`	Run a command only if stdout is not empty
`find . -name core | ifne mail root`	If there is a file named "core" in the current directory, send it via e-mail to the root user

read and echo

```
read MYVAR
```

Read a variable from standard input

```
read -n 8 MYVAR
```

Read only max 8 characters from standard input

```
read -t 60 MYVAR
```

Read a variable from standard input, timing out after one minute

```
read -s MYVAR
```

Read a variable from standard input without echoing to terminal (silent mode)

```
while read -r line
do
   echo "Hello $line"
done < file
```

Process a text file line by line, reading from *file*, and output the lines.
If *file* is `/dev/stdin`, reads from standard input instead

```
while read line
do
   for word in $line
   do
      echo "Hello $word"
   done
done < file
```

Process a text file containing multiple words in each line, and output the words

```
while IFS=$'\t' read -r -a array
do
   echo "${array[0]}"
   echo "${array[1]}"
   echo "${array[2]}"
done < file
```

Process a text file containing three words per line separated by a tab, and output the words. Example of input file:

```
aaaa    bbb     ccc
dd      eeeee   ff
ggg     hhh     iiii
```

```
echo $MYVAR
```

Print a variable on screen

```
echo -n "message"
printf "message"
```

Print *message* onscreen without a trailing line feed

```
echo -e '\a'
```

Produce an alert sound (BEL sequence)

```
echo .*
```

Resolve globs, printing all files whose name begins with a dot in the current dir

```
echo rm -f .*
```

Resolve globs, expanding the filenames and printing the actual `rm` command that would have been executed

```
pv -qL10 <<< "message"
```

Print *message* onscreen, one character at a time

Any application, program, script, or service that runs on the system is a **process**. Processes whose parent is a shell are called **jobs**.
Signals are used for inter-process communication. Each process has a unique **PID (Process ID)** and a **PPID (Parent Process ID)**; when a process spawns a child, the process PID is assigned to the child's PPID.
The process with PID 1 (`init` or `systemd`) is the ancestor of all processes and is unkillable; its death causes a kernel panic.
The parent process of an orphaned child is set to the nearest ancestor process of the child that marked itself as a subreaper, or the process with PID 1 if there is no such ancestor subreaper process.
A **zombie** process is a process that has terminated execution but whose parent, for some reason, failed to reap. When a child process dies, its status becomes EXIT_ZOMBIE and a SIGCHLD is sent to the parent. The parent should then call the `wait()` system call to read the dead process' exit status and other information; until that moment, the child process remains a zombie.
Zombie processes do not take up system resources and are usually not a problem, but may be a symptom that the parent program was sloppily coded. To eliminate a zombie, terminate its parent by sending it a SIGKILL.

cgroups (control groups) are a feature of the Linux kernel allowing the organization of processes into hierarchical groups for monitoring and rate limiting purposes. Many projects (e.g. systemd, Docker, and Kubernetes) use it.

`ps -ef` (UNIX options) `ps aux` (BSD options)	List all processes
`pstree` *`PID`*	Display all processes in hierarchical format. The process tree is rooted at *PID*, or at `init` if *PID* is omitted
`pidof` *`processname`*	Show PIDs of processes with name *processname*
`pidof -s` *`processname`*	Show PID of process with name *processname*, returning a single result

`pgrep sshd`
`ps -ef | grep "[s]shd"`
	Show processes whose name is "sshd"

`pgrep -u root sshd`	Show processes whose name is "sshd" and are owned by root
`pmap` *`PID`*	Display the memory map of process *PID*
`jobs`	List all jobs
CTRL Z	Suspend a job, putting it in the stopped state (send a SIGTSTP)
`bg %`*`n`*	Put job #*n* in the background (send a SIGCONT)
`fg %`*`n`*	Resume job #*n* in the foreground and make it the current job (send a SIGCONT)
`kill %`*`n`*	Kill job #*n*
`disown %`*`n`*	Remove job #*n* from the table of active jobs
`disown -h %`*`n`*	Prevent job #*n* from receiving a SIGHUP if the shell receives that signal
`:(){ :|:& };:`	**Fork bomb**: starts a process that continually replicates itself, slowing down or crashing the system because of resource starvation. Dangerous!

`(` *`command`* `)& pid=$!; \`
`sleep` *`n`*`; kill -9 $pid`
	Run *command* and kill it after *n* seconds

To each process is associated a niceness value: the higher the niceness, the lower the priority.
The niceness value ranges from -20 to 19, and a newly created process has a default niceness of 0.
Unprivileged users can modify a process' niceness only within the range from 1 to 19.

`nice -n -5` *`command`*	Start *command* with a niceness of -5. If niceness is omitted, a default value of 10 is used
`renice -5` *`command`*	Change the niceness of a running *command* to -5
`snice`	Change the niceness of a process. Obsolete

Most frequently used signals

Signal number	Signal name	Effect
1	SIGHUP	Used by many daemons to signal them to reload their configuration
2	SIGINT	Interrupt, stop
9	SIGKILL	Kill unconditionally (this signal cannot be ignored)
15	SIGTERM	Terminate gracefully
17	SIGCHLD	Child stopped or terminated
18	SIGCONT	Continue execution
20	SIGTSTP	Stop execution

The manpage `man 7 signal` lists all signal numbers and names.

kill -l	List all available signal names
kill -l *n*	Print the name of signal number *n*
kill -9 1138	Send a signal 9 (SIGKILL) to process 1138, hence killing it
kill -s SIGCHLD *PPID*	Eliminate a zombie process by sending its parent (*PPID*) a SIGCHLD
killall -9 sshd	Kill processes whose name is "sshd"
pkill -9 -u root sshd	Kill processes whose name is "sshd" and are owned by root
pkill -9 -u *user*	Kill all processes owned by *user*, forcing the user to log out
skill	Send a signal to a process or show process status. Obsolete
xkill	Kill a process by its X GUI resource. Pops up a cursor to select a window
nohup *script.sh*	Prevent a process from terminating (receiving a SIGHUP) when its parent Bash dies. When a Bash shell is terminated cleanly via `exit`, its jobs become child of the Bash's parent and continue running. When a Bash shell is killed instead, it issues a SIGHUP to its children which terminate execution
trap *action condition*	Trap a signal
strace *command*	Trace the execution of *command*, intercepting and printing system calls called by a process and signals received by a process
ipcs	Show IPC facilities information (shared memory, message queues, and semaphores)

`top`	Monitor processes in real-time
`htop`	Monitor processes in real-time (ncurses UI)
`iotop`	Display I/O usage by processes in the system
`atop`	Advanced system monitor that displays the load on CPU, RAM, disk, and network
`powertop`	Power consumption and power management diagnosis tool
`uptime`	Show how long the system has been up, how many users are connected, and the system load averages for the past 1, 5, and 15 minutes
`sar`	Show reports about system activity (including reboots). Reports are generated from data collected via the cron job `sysstat` and stored in `/var/log/sa/s`*`n`*, where *n* is the day of the month

`sar -f /var/log/sa/sa13 \`
`-s 06:00:00 -e 09:00:00`
	Show reports for system activity from 6 to 9 AM on the 13th of the month

`sar -u` *`n m`*	Show real-time CPU activity, every *n* seconds for *m* times
`sar -n DEV`	Show real-time network activity (received and transmitted packets per second)
`sysbench`	Multi-threaded benchmark tool able to monitor different OS parameters: file I/O, scheduler, memory allocation, thread implementation, databases
`inxi`	Debugging tool to rapidly and easily gather system information and configuration
`stress-ng`	Tool for CPU and RAM stress tests
`collectd`	System statistics collector
`sensors`	Print sensor chips information (e.g. temperature)
`psensor`	GUI client tool for monitoring hardware sensors (e.g. temperature, fan speed) of a remote server
`psensor-server`	HTTP server for `psensor`
`corefreqd`	Daemon for CoreFreq, a CPU monitoring tool with BIOS-like functionalities
`corefreq-cli`	CoreFreq client
`sysmon`	Monitor for system events. Developed by Sysinternals
`conky`	System monitor widget GUI with integration for audio player, email, and news
`gkrellm`	System monitor widget GUI

There exist more complete resource monitoring solutions for a Linux environment, e.g. Munin, Zabbix, Centreon, and Nagios (system and network monitor and alert tools), MRTG and Cacti (network monitors), and Netdata (real-time performance and health monitor).

`vmstat` Print a report about virtual memory statistics: processes, memory, paging, block I/O, traps, disks, and CPU activity

`iostat` Print a report about CPU utilization, device utilization, and network filesystem.
The first report shows statistics since the system boot; subsequent reports will show statistics since the previous report

`mpstat` Print a report about processor activities

`vmstat` *n m*
`iostat` *n m*
`mpstat` *n m*
Print the relevant report every *n* seconds for *m* times

Output of command `vmstat`

```
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 296724 267120 3393400    0    0    17    56    0    3  2  2 95  1  0
```

procs	**r**	Number of runnable processes (running or waiting for run time)	
	b	Number of processes in uninterruptible sleep	
memory	**swpd**	Virtual memory used (swap)	in Kb
	free	Free memory (idle)	
	buff	Memory used as buffers	
	cache	Memory used as cache	
swap	**si**	Memory swapped in from disk	in Kb/second
	so	Memory swapped out to disk	
io	**bi**	Blocks received in from a block device	in blocks/second
	bo	Blocks sent out to a block device	
system	**in**	Number of interrupts	per second
	cs	Number of context switches	
cpu	**us**	Time spent running user code (non-kernel)	in percentage of total CPU time
	sy	Time spent running system code (kernel)	
	id	Time spent idle	
	wa	Time spent waiting for I/O	
	st	Time stolen from a virtual machine	

`free` Show the amount of free and used memory in the system

Output of command `free`

```
              total        used        free      shared  buff/cache   available
Mem:       16344088     2273312    11531400      776228     2539376    12935112
Swap:       1048572           0     1048572
```

```
             total       used       free     shared    buffers     cached
Mem:       1504544    1491098      13021          0      91112     764542
-/+ buffers/cache:     635212     869498
Swap:      2047686       7667    2040019
```

Mem	**total**	Total configured amount of memory
	used	Used memory
	free	Unused memory
	shared	Memory used by tmpfs, 0 if not available
	buff/cache	Memory used by kernel buffers, page cache, and slabs
	available	Memory available for new applications (without using swap) *
-/+ buffers/cache	**used**	Memory used by kernel buffers
	free	Memory available for new applications (without using swap) *
Swap	**total**	Total configured amount of swap space
	used	Used swap space
	free	Free swap space *

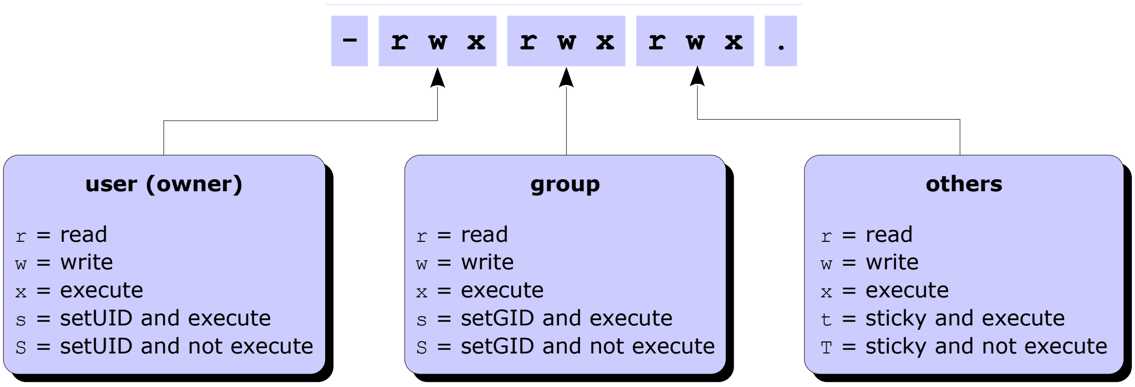
* These are the true values indicating the free system resources available.

All values are in kB, unless unit options are used:

`-b`	Show values in bytes
`-k`	Show values in kilobytes (kB)
`-m`	Show values in megabytes (MB)
`-g`	Show values in gigabytes (GB)
`-h`	Show values in human-readable format, i.e. scaled to the shortest 3-digit unit and displaying the unit
`--si`	Show values according to the International Systems of Units (powers of 1000 instead of powers of 1024)

Performance Co-Pilot (PCP) is an open source framework and toolkit for monitoring and analyzing system performance, either live or historical.

`pcp`	Run a PCP command or summarize an installation
`pminfo`	Display information about performance metrics
`pmdumptext`	Dump performance metrics to a text table
`pmrep`	Performance metrics reporter
`pmstat`	Periodically print a one-line summary of system performance at the highest level
`pmval` `pmevent`	Print values of a performance metric



user (owner)

`r` = read
`w` = write
`x` = execute
`s` = setUID and execute
`S` = setUID and not execute

group

`r` = read
`w` = write
`x` = execute
`s` = setGID and execute
`S` = setGID and not execute

others

`r` = read
`w` = write
`x` = execute
`t` = sticky and execute
`T` = sticky and not execute

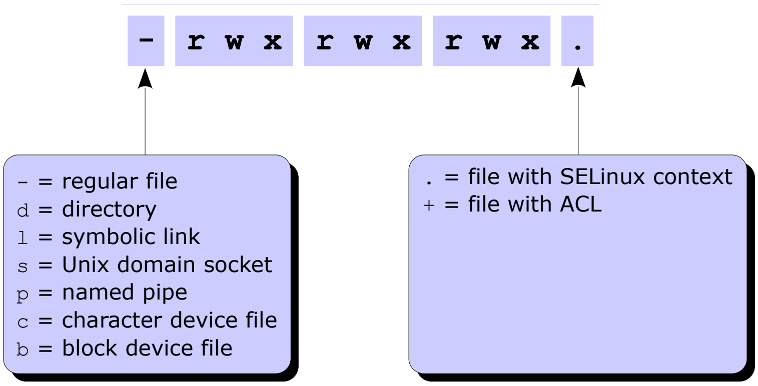
Permission	Octal value	Command	Effect on file	Effect on directory
Read	user: 400	`chmod u+r`	Can open and read the file	Can list directory content
	group: 40	`chmod g+r`		
	others: 4	`chmod o+r`		
Write	user: 200	`chmod u+w`	Can modify the file	Can create, delete, and rename files in the directory
	group: 20	`chmod g+w`		
	others: 2	`chmod o+w`		
Execute	user: 100	`chmod u+x`	Can execute the file (binary or script)	Can enter the directory, and search files within (by accessing a file's inode)
	group: 10	`chmod g+x`		
	others: 1	`chmod o+x`		
SetUID (SUID)	4000	`chmod u+s`	Executable is run with the privileges of the file's owner	No effect
SetGID (SGID)	2000	`chmod g+s`	Executable is run with the privileges of the file's group	All new files and subdirectories inherit the directory's group ID
Sticky	1000	`chmod +t`	No effect	Files inside the directory can be deleted or moved only by the file's owner

File permissions are not cumulative; the most specific permission (i.e. user vs group, and group vs others) applies.

`chmod 711` *file* `chmod u=rwx,go=x` *file*	Set read, write, and execute permission to user; set execute permission to group and others
`chmod u+wx` *file*	Add write and execute permission to user
`chmod -x` *file*	Remove execute permission from everybody (user, group, and others)
`chmod -R g+x` */path*	Set the group execute bit recursively on *path* and every directory and file underneath

`find` */path* `-type d \`
`-exec chmod g+x {} \;`
	Set the group execute bit recursively on *path* and every directory, but not file, underneath

`chown` *user file*	Change the owner of the file to *user*
`chown` *user*`:`*group file*	Change the owner of the file to *user*, and group ownership of the file to *group*
`chown :`*group file* `chgrp` *group file*	Change group ownership of the file to *group*
`umask 022`	Set the permission mask to 022, hence masking write permission for group and others. Linux default permissions are 0666 for files and 0777 for directories. These base permissions are ANDed with the inverted umask value to calculate the final permissions of a newly created file or directory



- = regular file
d = directory
l = symbolic link
s = Unix domain socket
p = named pipe
c = character device file
b = block device file

. = file with SELinux context
+ = file with ACL

Attribute	Effect
`a`	File can only be opened in append mode for writing
`A`	When file is accessed, its atime record is not modified. This reduces I/O operations
`c`	File is automatically compressed on the fly on disk by the kernel. Reading it produces uncompressed data
`C`	For filesystems which perform copy-on-write, file is not subject to copy-on-write updates
`d`	File will not be backed up by the `dump` program
`D`	When directory is modified, changes are written synchronously on disk. Equivalent to `dirsync` mount option
`e`	File is using extents for mapping the blocks on disk
`E`	Compression error on a compressed file. This attribute is used by experimental compression patches
`h`	File stores its blocks in units of filesystem blocksize instead of in units of sectors. This means that the file is (or was) larger than 2 Tb
`i`	File is immutable i.e. cannot be deleted, modified, renamed, linked, or changed permissions
`I`	Directory is being indexed using hashed trees
`j`	All file data is written to the ext3 or ext4 journal before being written to the file itself
`N`	File has data stored inline within the inode itself
`s`	File will be securely wiped by zeroing when deleted
`S`	When file is modified, changes are written synchronously on disk. Equivalent to the `sync` mount option
`t`	For filesystems with support for tail-merging, file will not have EOF partial block fragment merged with other files. This is necessary for some applications e.g. LILO
`T`	Directory is the top of directory hierarchies for the purpose of the Orlov block allocator
`u`	File can be undeleted after being deleted
`X`	Raw contents of compressed file can be accessed directly. This attribute is used by experimental compression patches
`Z`	Compressed file is dirty. This attribute is used by experimental compression patches

`chattr +`*`attribute file`*	Add a file or directory attribute
`chattr -`*`attribute file`*	Remove a file or directory attribute
`chattr =`*`attribute file`*	Set a file or directory attribute, removing all other attributes
`lsattr `*`file`*	List file or directory attributes

Timestamp	Value tracked	Displayed via
mtime	Time of last **modification** to file contents (data itself)	`ls -l`
ctime	Time of last **change** to file contents or file metadata (owner, group, or permissions)	`ls -lc`
atime	Time of last **access** to file for reading contents	`ls -lu`

The POSIX standard does not define a timestamp for file **creation**. Some filesystems (e.g. ext4, JFS, Btrfs) store this value, but currently there is no Linux kernel API to access it.

Access Control Lists (ACLs) provide a fine-grained set of permissions that can be applied to files and directories.
An **access ACL** is set on an individual file or directory; a **default ACL** is set on a directory, and applies to all files and subdirectories created inside it that don't have an access ACL.
The final permissions are the intersection of the ACL with the chmod/umask value.
A partition must have been mounted with the `acl` option in order to support ACLs on files.

setfacl -m u:*user*:*permissions file*	Set an access ACL on a file for a user
setfacl -m g:*group*:*permissions file*	Set an access ACL on a file for a group
setfacl -m m:*permissions file*	Set the effective rights mask on a file
setfacl -m o:*permissions file*	Set the permissions on a file for other users
setfacl -x u:*user file*	Remove an access ACL from a file for a user
setfacl -x g:*group file*	Remove an access ACL from a file for a group

The *permissions* are standard Unix permissions specified as any combination of `r w x`.

setfacl -m d:u:*user*:*permissions dir* setfacl -d -m u:*user*:*permissions dir*	Same as above, but set a default ACL instead of an access ACL. This applies to all commands above

getfacl *file*	Display the access (and default, if any) ACL for a file
getfacl *file1* | setfacl --set-file=- *file2*	Copy the ACL of *file1* and apply it to *file2*
getfacl --access *dir* | setfacl -d -M- *dir*	Copy the access ACL of a directory and set it as default ACL

chacl *options*	Change an ACL. This command exists to provide compatibility with IRIX

man acl	Show the manpage about ACLs

An **inode** is a structure containing all file metadata: file type, permissions, owner, group, size, number of links, attributes, access/change/modification/deletion times, ACLs, and address where the actual file content (data) is stored. An inode does not contain the name of the file; this information is stored in the directory where the file is located (i.e. referenced).
A directory contains a list of mappings between filenames and inodes.

In Linux, two types of links exist: **hard links** and **symbolic links** (aka **soft links**).

The **link count** of a file is the total number of hard links to that file (i.e. to that file's inode). By default, files have a link count of 1, and directories have a link count of 2 (the directory itself, and the `.` link inside the directory). The link count of a directory is increased by one for each subdirectory (because of the `..` parent link inside the subdirectory).
Once a file has no hard links pointing to it, the file is deleted, provided that no process holds the file open for reading.

	Hard link	**Symbolic link**
Definition	A link to an already existing inode	A path to a filename; a shortcut
Command to create it	`ln` *`file hardlink`*	`ln -s` *`file symlink`*
Link is still valid if the original file is moved or deleted	Yes (because the link still references the inode to which the original file pointed)	No (because the path now references a non-existent file)
Can link to a file in another filesystem	No (because inode numbers make sense only within a determinate filesystem)	Yes
Can link to a directory	No	Yes
Link permissions	Reflect the original file's permissions, even when these are changed	`rwxrwxrwx`
Link attributes	`-` (regular file)	`l` (symbolic link)
Inode number	The same as the original file	A different inode number (since it's a different file)

`ls -i`	Show a listing of the directory with the inode number for each file
`ls -l`	Show a listing of the directory with the link count for each file
`df -i`	Report filesystem inode usage
`find / -inum` *`n`*	Find all files linked to the same inode *n*
`find / -samefile` *`file`*	Find all files linked to the same inode as *file*

`find /path -name "foo*"`
`find /path -name "foo*" -print`
	Find all files and dirs, in the directory tree rooted at */path*, whose name starts with "foo"

`find / -name "foo*" -exec chmod 700 {} \;`	Find all files and dirs whose name start with "foo" and apply permission 700 to all of them
`find / -name "foo*" -ok chmod 700 {} \;`	Find all files and dirs whose name start with "foo" and apply permission 700 to all of them, asking for confirmation
`find / -size +128M`	Find all files larger than 128 Mb
`find / -type f -ctime +10`	Find all files last changed more than 10 days ago
`find / -type f -perm -4000`	Find all files with SUID set (a possible security risk, because a shell with SUID root is a backdoor)
`find / -type f -newermt "May 4 2:55" -delete`	Find and delete all files newer than the specified timestamp. Using `-delete` is preferable to using `-exec rm {} \;`
`find . -type f -print -exec cat {} \;`	Print all files, in the current directory and under, prepending them with a filename header
`find . \! -name "*.gz" -type f -exec gzip {} \;`	Find all files, in the current directory and under, which do not have the `gz` extension, and compress them

`find / -xdev -type f -size +100M \`
`-exec ls -lah {} \;`
	Find all files larger than 100 Mb in the current filesystem only and display detailed information about them

`locate file`
`slocate file`
	Locate *file* by searching the file index `/etc/updatedb.conf`, not by actually walking the filesystem. The search is fast but will only held results relative to the last rebuild of the file index

`updatedb`	Rebuild the file index
`type command` (Bash) `whence command` (KornShell, Z shell) `where command` (Z shell)	Determine if a command is a program, a built-in (i.e. an internal feature of the shell), an alias, or a function
`which command`	Locate a binary executable command within the PATH
`which -a command`	Locate all matches of a command, not only the first one
`whereis command`	Locate the binary, source, and manpage files for a command
`whereis -b command`	Locate the binary files for a command
`whereis -s command`	Locate the source files for a command
`whereis -m command`	Locate the manpage files for a command
`file file`	Analyze the content of a file or directory, and display the kind of file (e.g. executable, text file, program text, swap file)

`history`	Show the history of command lines executed up to this moment. Commands prepended by a space will be executed but will not show up in the history. After the user logs out from Bash, history is saved into `~/.bash_history`
`!`*n*	Execute command number *n* in the command line history
`!!`	Execute again the last executed command
`history -c`	Clear the command line history
`history -d` *n*	Delete command number *n* from the command line history
`export HISTSIZE=`*m*	Set the command line history to contain only the *m* past commands
`fc`	Edit and execute again the last executed command
`fc -l -`*n*	List the last *n* commands in the command line history

`alias ls='ls -lap'`	Set up an alias for the `ls` command
`alias`	Show all defined aliases
`unalias ls`	Remove the alias for the `ls` command
`unalias -a`	Remove all defined aliases
`\ls` `/bin/ls`	Run the non-aliased version of the `ls` command

`rm ./-rf`	Delete a file called "-rf". To operate on a file whose name begins with a special character, specify the file path (either relative or absolute)

Almost all Linux commands accept the option `-v` (verbose), and some commands also accept the options `-vv` or `-vvv` (increasing levels of verbosity).

All Bash built-in commands, and all commands that respect the POSIX requirements, accept the option `--` which marks in the arguments the end of options and the start of operands:

`grep -- -i` *file*	Search for the string "-i" in *file*
`rm -- -rf`	Delete a file called "-rf"
`info -- cat --`	Display the Info entry for `--`

Shells		
`sh`	**Thompson Shell**	The first Unix shell, now obsolete. Simple command interpreter, not designed for scripting.
`sh`	**Bourne Shell**	Default shell for Version 7 Unix, now obsolete. On current Linux systems, `/bin/sh` is a symlink to the default shell.
`bash`	**Bash (Bourne Again Shell)**	GNU replacement for the Bourne Shell. Default shell for most Linux distributions.
`csh`	**C shell**	Shell originally designed for BSD.
`tcsh`	**tcsh**	C shell with additional features.
`ksh`	**KornShell**	Shell based on the Bourne Shell, with improvements.
`zsh`	**Z shell**	Shell based on the Bourne Shell, with improvements.
`ash`	**Almquist shell**	Lightweight shell, was the default shell for BSD.
`dash`	**Debian Almquist shell**	Port of Almquist shell for Debian.
`fish`	**Friendly interactive shell**	Interactive, user-friendly shell.

The scope of **variables** is the current shell only, while **environment variables** are visible within the current shell as well as within all subshells and Bash child processes spawned by the shell.
Environment variables are set in `/etc/environment` in the form *`variable=value`*.
Conventionally, variable names are lowercase while environment variable names are uppercase.

`set`	Display all variables
`env`	Display all environment variables
`readonly -p`	Display all variables that are read-only

`VAR=value`
`((VAR=value))`
`let "VAR=value"`
	Set the value of a variable.
	There must be no spaces around the `=` sign. It is possible to add space around `((` and `))`

`readonly VAR=value`	Set a variable making its value unchangeable

`set ${VAR:=value}`
`VAR=${VAR:-value}`
	Set a variable only if it is not already set (i.e. does not exist) or is null

`unset VAR`	Unset (i.e. delete) a variable
`export VAR`	Export a variable, making it an environment variable
`export VAR=value`	Set the value of a variable and export it
`envsubst < file`	Read a text file mentioning environment variables in the form `$VAR` and output it replacing each variable name with its value

`command $VAR`
`command ${VAR}HELLO`
`command "${VAR}"`
	Pass a variable as argument to *command*.
	If other characters follow the variable name, it is necessary to specify the boundaries of the variable name via `{}` to make it unambiguous.
	It is recommended to double quote the variable when referencing it, to prevent interpretation of special characters (except `\ $ `` ` ``) and word splitting (if the variable value contains whitespaces), which will cause unintended results

``VAR=`command` ``
`VAR=$(command)`
	Command substitution. Assigns to a variable the standard output resulting from *command* (which is executed in a subshell)

`echo ${VAR,,}`	Print a string variable in lowercase
`echo ${VAR:-message}`	If variable exists and is not null, print its value, otherwise print *message*
`echo ${VAR:+message}`	If variable exists and is not null, print *message*, otherwise print nothing
`ARRAY=(val1 val2 val3)`	Set an array (i.e. a variable containing multiple values), assigning the first three elements
`ARRAY[3]=val4`	Assign a fourth element in the previous array
`echo ${ARRAY[n]}`	Print the array element number *n*
`echo ${ARRAY[*]}`	Print all array elements
`TOKENS=($STRING)`	String tokenizer. Splits a string stored in the variable *STRING* into tokens, according to the content of the shell variable `$IFS`, and stores them as elements in the array *TOKENS*
`echo ${#STRING}`	Print a string's length (number of characters)

Shell variables - operations

```
VAR=$((5 + 37))
VAR=$[5 + 37]
VAR=$((VAR2 - 3))
VAR=`expr $VAR2 - 3`
```

Evaluate a numeric expression, assigning the result to another variable

```
[ $(($VAR % 2)) -eq 0 ] && command
```

Evaluate a numeric expression, running *command* if the variable is even

```
((VAR++))
((++VAR))
((VAR+=1))
((VAR=VAR+1))
```

Increase a variable by 1

```
for i in /path/*
do
   echo "Filename: $i"
done
```

Loop and operate through all the output tokens (in this case, files in the *path*). The construct `for i in $(ls /path/)` must not be used, because filenames containing particular characters (whitespaces, glob characters, hyphens etc.) will cause unintended results

Bash built-in variables	
$0	Script name
$*n*	*n*th argument passed to the script or function
$@	All arguments passed to the script or function; each argument is a separate word
$*	All arguments passed to the script or function, as a single word
$#	Number of arguments passed to the script or function
$?	Exit status of the last recently executed command
${PIPESTATUS[*n*]}	Exit status of the *n*th command in the executed pipeline
$$	PID of the script in which this variable is called
$!	PID of the last recently executed background command
$SHLVL	Deepness level of current shell, starting with 1
$IFS	Internal Field Separator; defines the token separators for strings, to perform word splitting after expansion. By default it has the value "space, tab, newline"
$RANDOM	Pseudorandom integer value between 0 and 32767

Bash shell event	Files run	
When a login shell is launched	/etc/profile /etc/profile.d/*.sh ~/.bash_profile ~/.bash_login ~/.profile	The shell executes the system-wide profile files, then the first of the 3 user files that exists and is readable
When a login shell exits	~/.bash_logout	
When a non-login shell is launched	/etc/bash.bashrc /etc/bashrc ~/.bashrc	

set -*option* set -o *longoption*	Enable a Bash option
set +*option* set +o *longoption*	Disable a Bash option
set -o	Show the status of all Bash options
set -v set -o verbose	Print shell input lines as they are read
set -x set -o xtrace	Print command traces before execution of each command (debug mode)
set -e set -o errexit	Exit the script immediately if a command fails. Recommended option
set -u set -o nounset	Treat expansion of unset variables as an error. This avoids unintended results

There are three ways to run a script with a specific Bash option enabled:
- Run the script with bash -*option script.sh*
- Specify the shebang line in the script as #!/bin/bash -*option*
- Add the command set -*option* at the beginning of the script

shopt	Display the list of all shell options with their current value (on or off)
shopt -s *shelloption*	Set (enable) a specific shell option
shopt -u *shelloption*	Unset (disable) a specific shell option

Bash shell scripts must start with the shebang line `#!/bin/bash` indicating the location of the script interpreter.

Script execution	
source *script.sh* . *script.sh*	Script execution takes place in the same shell. Variables defined and exported in the script are seen by the shell when the script exits
bash *script.sh* ./*script.sh* (file must be executable)	Script execution spawns a new shell

command &	Execute *command* in the background
command1; *command2*	Execute *command 1* and then *command 2*
command1 && *command2*	Execute *command 2* only if *command 1* executed successfully (exit status = 0)
command1 || *command2*	Execute *command 2* only if *command 1* did not execute successfully (exit status > 0)
(*command1* && *command2*)	Group commands together for evaluation priority
(*command*)	Run *command* in a subshell. This is used to isolate *command*'s effects, as variable assignments and other changes to the shell environment operated by *command* will not remain after *command* completes
exit	Terminate a script
exit *n*	Terminate a script with the specified exit status number *n*. By convention, a 0 exit status is used if the script executed successfully, a non-zero value otherwise
command || exit 1	(To be used inside a script.) Exit the script if *command* fails
/bin/true	Do nothing and return immediately a status code of 0 (indicating success)
/bin/false	Do nothing and return immediately a status code of 1 (indicating failure)

if *command*
then echo "Success"
else echo "Failure"
fi
	Run a command, then evaluate whether it exited successfully or failed

function *fname* { *commands* }
fname() { *commands* }
	Define a function. A function must be defined before it can be used in a Bash script. Argument number *n* is accessed in the body of the function via $*n*.
	An advantage of functions over aliases is that functions can be passed arguments

fname arg1 arg2 ...	Call a function
readonly -f *fname*	Mark an already defined function as read-only, preventing it to be redefined
typeset -f	Show functions defined in the current Bash session
readonly -p -f	Show functions which are read-only
shellcheck	Script analyzer and debugger
dialog	Display shell script (terminal) dialogs for user messages and input
zenity	Display GTK+ graphical dialogs for user messages and input

getopts Parse positional parameters in a shell script

getopts **syntax**	

while getopts abc:d: *OPT*
do
   case $*OPT* in
	Definition of accepted options

a)
          command_a
          exit 0
          ;;
	Matches option -a.
	Executes a command

b)
          command_b
          exit 0
          ;;	

c)
          command_c $OPTARG
          exit 0
          ;;
	Matches option -c *argument*.
	Executes a command with argument

d)
          command_d $OPTARG
          exit 0
          ;;	

*)
          default_command
          exit 1
          ;;
	Command to execute if none of above options applies

esac done	

`cat /etc/debian_version`
`cat /etc/fedora-release`
`cat /etc/redhat-release`
`cat /etc/lsb-release`
`lsb_release -a`
`cat /etc/os-release`
	(Debian)
	(Fedora)
	(Red Hat)
		Display Linux distribution name and version

`screenfetch`		Display detailed system information including Desktop Environment, Window Manager, Window Manager theme, screen resolution, etc.

Command execution

`watch command` Execute *command* every 2 seconds

`watch -d -n 1 command` Execute *command* every second, highlighting the differences in the output

`time command` Execute *command* and, at its completion, write to stderr timing statistics about the run: elapsed real time between invocation and termination, user CPU time, system CPU time

`timeout 30s command` Execute *command* and kill it after 30 seconds

`command | ts` Prepend a timestamp to each line of the output of *command*

`rlwrap command` Readline wrapper. Executes *command*, intercepting user input to provide line editing, history, and completion

`sleep 5` Pause for 5 seconds

`sleep $[($RANDOM % 60) + 1]s` Sleep for a random time between 1 and 60 seconds

`sleep infinity` Pause forever

`usleep 5000` Pause for 5000 microseconds

`yes` Output endlessly the string "y"

`yes string` Output endlessly *string*

`yes | fsck /dev/sda` Automatically answer yes every time `fsck` asks for confirmation before fixing errors

`script file` Generate a typescript of a terminal session.
Forks a subshell and starts recording on *file* everything that is printed on terminal; the typescript ends when the user exits the subshell

`expect` Dialogue with interactive programs according to a script, analyzing what can be expected from the interactive program and replying accordingly

`cmdtest` Tool for black box testing of Linux command line programs

`busybox` BusyBox, "the Swiss Army knife of Embedded Linux", an optimized multi-call binary which contains many Linux commands and utilities.
Useful for system recovery if Bash built-ins or common commands have become unusable or have been removed from the system

`busybox applet arguments` Execute *applet*, which operates as the homonym Linux command

`xargs command` Call *command* multiple times, one for each argument found on stdin

`ls foo* | xargs cat` Print via `cat` the content of every file whose name starts by "foo"

`parallel command` Run *command* in parallel.
This is used to operate on multiple inputs, similarly to `xargs`

`test "$MYVAR" operator "value" && command`
`[ "$MYVAR" operator "value" ] && command`
`if [ "$MYVAR" operator "value" ]; then command; fi`
	Perform a test; if it results true, *command* is executed

Test operators

Integer operators

`-eq` *value*	Equal to
`-ne` *value*	Not equal to
`-lt` *value*	Less than
`-le` *value*	Less than or equal to
`-gt` *value*	Greater than
`-ge` *value*	Greater than or equal to

Numeric operators

`=` *value*	Equal to
`!=` *value*	Not equal to
`<` *value*	Less than
`<=` *value*	Less than or equal to
`>` *value*	Greater than
`>=` *value*	Greater than or equal to

Expression operators

expression1 `-a` *expression2*	Logical AND
expression1 `-o` *expression2*	Logical OR
`!` *expression*	Logical NOT
`\(` *expression* `\)`	Priority

String operators

`-z`	Is zero length
`-n` or nothing	Is non-zero length
`=` *string* or `==` *string*	Is equal to
`!=` *string*	Is not equal to
`<` *string*	Is alphabetically before
`>` *string*	Is alphabetically after
`substr` *string pos len*	Substring
`index` *string chars*	Index of any chars in string
`length` *string*	String length
string `:` *regex* or `match` *string regex*	String matches regex

File operators

`-e` *file* or `-a` *file*	Exists
`-f` *file*	Is a regular file
`-d` *file*	Is a directory
`-b` *file*	Is a block special file
`-c` *file*	Is a character special file
`-r` *file*	Is readable
`-w` *file*	Is writable
`-x` *file*	Is executable
`-k` *file*	Is sticky
`-u` *file*	Is SUID
`-g` *file*	Is SGID
`-O` *file*	Is owned by the Effective UID
`-G` *file*	Is owned by the Effective GID
`-p` *file*	Is a named pipe (aka FIFO)
`-S` *file*	Is a socket
`-h` *file* or `-L` *file*	Is a symbolic link
`-s` *file*	Is non-zero length
`-N` *file*	Was modified since last read
file1 `-nt` *file2*	Is newer than
file1 `-ot` *file2*	Is older than
file1 `-ef` *file2*	Refer to same device and inode as

`expr "$MYVAR" = "39 + 3"`	Evaluate an expression (in this case, assigns the value 42 to the variable)
`expr` *string* `:` *regex*	Return the length of the substring matching the regex
`expr` *string* `: \(`*regex*`\)`	Return the substring matching the regex

Operators			
Mathematical operators		**Logical operators**	
`+`	Addition	`!`	Logical negation
`-`	Subtraction	`&&`	Logical AND
`*`	Multiplication	`||`	Logical OR
`/`	Division	**Bitwise operators**	
`%`	Remainder	`~`	Bitwise negation
`**`	Exponentiation	`&`	Bitwise AND
`++`	Pre/post increment	`|`	Bitwise OR
`--`	Pre/post decrement	`^`	Bitwise XOR
Assignment operators		`<<`	Left bitwise shift
`=`	Assignment	`>>`	Right bitwise shift
op`=`	Operation and assignment		

Tests

```
if [test 1]
then
   [command block 1]
elif [test 2]
then
   [command block 2]
else
   [command block 3]
fi
```

```
case $STRING in
   pattern1)
      [command block 1]
      ;;
   pattern2)
      [command block 2]
      ;;
   *)
      [command block default]
      ;;
esac
```

Loops

```
while [test]
do
   [command block]
done
```

The *command block* executes as long as *test* is true

```
until [test]
do
   [command block]
done
```

The *command block* executes as long as *test* is false

```
for item in [list]
do
   [command block]
done
```

The *command block* executes for each *item* in *list*

```
i=0
while [ $i -le 7 ]
do
   echo $i
   let i++
done
```

```
i=0
until [ $i -gt 7 ]
do
   echo $i
   let i++
done
```

```
for i in 0 1 2 3 4 5 6 7
do
   echo $i
done
```

```
for i in {0..7}
do
   echo $i
done
```

```
start=0
end=7
for i in $(seq $start $end)
do
   echo $i
done
```

```
start=0
end=7
for ((i = start; i <= end; i++))
do
   echo $i
done
```

Loop jumps

```
break
```

Exit a loop

```
continue
```

Jump to the next iteration

```
i=0
while true
do
   if [ $i -gt 7 ]; then break; fi
   echo $i
   let i++
done
```

```
i=-9
while [ $i -lt 7 ]
do
   let i++
   if [ $i -lt 0 ]; then continue; fi
   echo $i
done
```

`vi`	Vi, a text editor
`vim`	Vi Improved, an advanced text editor
`gvim`	Vim with GUI
`vimdiff` *`file1 file2`*	Compare two text files in Vim
`pico`	PIne COmposer, a simple text editor derived from Pine
`nano`	Simple text editor, GNU clone of Pico
`rnano`	Restricted version of Nano: does not allow the user to access the filesystem (except for files specified as argument) or to run a command shell
`emacs`	GUI text editor
`gedit`	GUI text editor
`ed`	Line-oriented text editor
`hexedit`	Hexadecimal and ASCII editor
`more`	Text pager (obsolete)
`less`	Text pager
`most`	Text pager with advanced features (screen split, binary viewer, etc.)

`g`	Go to the first line in the file
`n``g`	Go to line number *n*
`G`	Go to the last line in the file
`F`	Go to the end of the file, and move forward automatically as the file grows
CTRL C	Stop moving forward
`-N`	Show line numbers
`-n`	Don't show line numbers
`=`	Show information about the file
CTRL G	Show current and total line number, byte, and percentage of the file read
`/`*`pattern`*	Search *pattern* forward
`?`*`pattern`*	Search *pattern* backwards
`&`*`pattern`*	Display only lines matching *pattern*
`n`	Search next occurrences forward
`N`	Search next occurrences backwards
`:n`	When reading multiple files, go to the next file
`:p`	When reading multiple files, go to the previous file
`R`	Repaint the screen
`V`	Show version number
`h`	Help
`q`	Quit

`less +`*`command file`*	Open *file* for reading, applying *command* (see list above)
`less +F --follow-name `*`file`*	Move forward, attempting periodically to reopen *file* by name; useful to keep reading a logfile that is being rotated. Note that, by default, `less` continues to read the original input file even if it has been renamed

ESC	Go to Command mode		
i	Insert text before cursor	and go to Insert mode	
I	Insert text after line		
a	Append text after cursor		
A	Append text after line		
v	Go to Visual mode, character-wise	then use the arrow keys to select a block of text	
V	Go to Visual mode, line-wise		
d	Delete selected block	gu	Switch selected block to lowercase
y	Copy (yank) selected block into buffer	gU	Switch selected block to uppercase
w	Move to next word	$	Move to end of line
b	Move to beginning of word	1G	Move to line 1 i.e. beginning of file
e	Move to end of word	G	Move to end of file
0	Move to beginning of line	z RETURN	Make current line the top line of the screen
CTRL G	Show current line and column number		
ma	Mark position "a". Marks a-z are local to current file, while marks A-Z are global to a specific file		
'a	Go to mark "a". If using a global mark, it also opens the specific file		
y'a	Copy (yank) from mark "a" to current line, into the buffer		
d'a	Delete from mark "a" to current line		
p	Paste buffer after current line	yy	Copy current line
P	Paste buffer before current line	yyp	Duplicate current line
x	Delete current character	D	Delete from current character to end of line
X	Delete before current character	dd	Delete current line
7dd	Delete 7 lines. Almost any command can be prepended by a number to repeat it that number of times		
u	Undo last command. Vi can undo the last command only, Vim is able to undo several commands		
.	Repeat last text-changing command		
/*string*	Search for *string* forward	n	Search for next match of *string*
?*string*	Search for *string* backwards	N	Search for previous match of *string*
:s/*s1*/*s2*/	Replace the first occurrence of *s1* with *s2* in the current line		
:s/*s1*/*s2*/g	Replace globally every occurrence of *s1* with *s2* in the current line		
:%s/*s1*/*s2*/g	Replace globally every occurrence of *s1* with *s2* in the whole file		
:%s/*s1*/*s2*/gc	Replace globally every occurrence of *s1* with *s2* in the whole file, asking for confirmation		
:5,40s/^/#/	Add a hash character at the beginning of each line, from line 5 to 40		
!!*program*	Replace line with output from *program*		
:r *file*	Read *file* and insert it after current line		
:X	Encrypt current document. Vi will automatically prompt for the password to encrypt and decrypt		
:w *file*	Write to *file*		
:wq :x ZZ	Save changes and quit		
:q	Quit (fails if there are unsaved changes)	:q!	Abandon all changes and quit

vi -R *file*	Open *file* in read-only mode
cat *file* | vi -	Open *file* in read-only mode; this is done from the shell, by having Vi read from stdin

Option	Effect
`ai`	Turn on auto indentation
`all`	Display all options
`ap`	Print a line after the commands `d c J m :s t u`
`aw`	Automatic write on commands `:n ! e# ^^ :rew ^} :tag`
`bf`	Discard control characters from input
`dir=`*`tmpdir`*	Set *tmpdir* as directory for temporary files
`eb`	Precede error messages with a bell
`ht=8`	Set terminal tab as 8 spaces
`ic`	Ignore case when searching
`lisp`	Modify brackets for Lisp compatibility
`list`	Show tabs and EOL characters
`set listchars=tab:>-`	Show tab as > for the first char and as - for the following chars
`magic`	Allow pattern matching with special characters
`mesg`	Enable UNIX terminal messaging
`nu`	Show line numbers
`opt`	Speed up output by eliminating automatic Return
`para=LIlPLPPPQPbpP`	Set macro to start paragraphs for `{ }` operators
`prompt`	Prompt `:` for command input
`re`	Simulate smart terminal on dumb terminal
`remap`	Accept macros within macros
`report`	Show the largest size of changes on status line
`ro`	Make file read-only
`scroll=12`	Set screen size as 12 lines
`shell=/bin/bash`	Set shell escape to `/bin/bash`
`showmode`	Show current mode on status line
`slow`	Postpone display updates during inserts
`sm`	Show matching parentheses when typing
`sw=8`	Set shift width to 8 characters
`tags=/usr/lib/tags`	Set path for files checked for tags
`term`	Print terminal type
`terse`	Print terse messages
`timeout`	Eliminate 1-second time limit for macros
`tl=3`	Set significance of tags beyond 3 characters (0 = all)
`ts=8`	Set tab stops to 8 for text input
`wa`	Inhibit normal checks before write commands
`warn`	Display the warning message "No write since last change"
`window=24`	Set text window as 24 lines
`wm=0`	Set automatic wraparound 0 spaces from right margin

`:set `*`option`*	turn on an *option*
`:set no`*`option`*	turn off an *option*
`:set `*`option`*` ?`	show the current value of *option*

Options can also be permanently set by including them in `~/.exrc` (Vi) or `~/.vimrc` (Vim)

```
SHOW DATABASES;
```

Show all existing databases

```
USE CompanyDatabase;
```

Select a database to use

```
SELECT DATABASE();
```

Show which database is currently selected

```
DROP DATABASE CompanyDatabase;
```

Delete a database

```
SHOW TABLES;
```

Show all tables from the selected database

```
CREATE TABLE customers (
cusid INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(32), lastname VARCHAR(32), dob DATE,
city VARCHAR(24), zipcode VARCHAR(5));

CREATE TABLE payments (
payid INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
date DATE, fee INT, bill VARCHAR(128), cusid INT,
CONSTRAINT FK1 FOREIGN KEY (cusid) REFERENCES customers(cusid));
```

Create tables

```
INSERT INTO customers (firstname,lastname,dob)
VALUES ('Arthur','Dent',1959-08-01), ('Trillian','',1971-03-19);
```

Insert new records in a table

```
DELETE FROM customers WHERE firstname LIKE 'Zaphod';
```

Delete some records in a table

```
UPDATE customers SET city = 'London' WHERE zipcode = 'L1 42HG';
```

Modify records in a table

```
CREATE INDEX lastname_index ON customers(lastname);
ALTER TABLE customers ADD INDEX lastname_index (lastname);
```

Create an index for faster searches

```
DESCRIBE customers;
```

Describe the columns of a table

```
SHOW CREATE TABLE customers;
```

Show the code used to create a table

```
SHOW INDEXES FROM customers;
```

Show primary key and indexes of a table

```
DROP TABLE customers;
```

Delete a table

```
ALTER TABLE customers MODIFY city VARCHAR(32);
```

Modify a column type

```
CREATE VIEW cust_view AS
SELECT * FROM customers WHERE city != 'London';
```

Create a view. Views are used similarly to tables

```
COMMIT;
```

Commit changes to the database

```
ROLLBACK;
```

Rollback the current transaction, canceling any changes done during it

```
START TRANSACTION;
BEGIN;
```

Disable autocommit for this transaction, until a COMMIT or ROLLBACK is issued

If no database has been selected for use, tables must be referenced by `databasename.tablename`.

`SELECT * FROM customers;`	Select all columns from the customers table
`SELECT firstname, lastname FROM customers LIMIT 5;`	Select first and last name of customers, showing 5 records only

`SELECT firstname, lastname FROM customers LIMIT 1000,5;`
`SELECT firstname, lastname FROM customers OFFSET 1000 LIMIT 5;`
	Select first and last name of customers, skipping the first 1000 records and showing 5 records only

`SELECT firstname, lastname FROM customers WHERE zipcode = 'L1 42HG';`	Select first and last name of customers whose zip code is "L1 42HG"
`SELECT firstname, lastname FROM customers WHERE zipcode IS NOT NULL;`	Select first and last name of customers with an existing zip code
`SELECT * FROM customers ORDER BY lastname, firstname;`	Select customers in alphabetical order by last name, then first name
`SELECT * FROM customers ORDER by zipcode DESC;`	Select customers, sorting them by zip code in reverse order

`SELECT firstname, lastname,`
`TIMESTAMPDIFF(YEAR,dob,CURRENT_DATE) AS age FROM customers;`
	Select first name, last name, and calculated age of customers

`SELECT DISTINCT city FROM customers;`	Show all cities, retrieving each unique output record only once
`SELECT city, COUNT(*) FROM customers GROUP BY city;`	Show all cities and the number of customers in each city. NULL values are not counted
`SELECT cusid, SUM(fee) FROM payments GROUP BY cusid;`	Show all fee payments grouped by customer ID, summed up

`SELECT cusid, AVG(fee) FROM payments GROUP BY cusid`
`HAVING AVG(fee)<50;`
	Show the average of fee payments grouped by customer ID, where this average is less than 50

`SELECT MAX(fee) FROM payments;`	Show the highest fee in the table
`SELECT COUNT(*) FROM customers;`	Show how many rows are in the table

`SELECT cusid FROM payments t1 WHERE fee =`
`(SELECT MAX(t2.fee) FROM payments t2 WHERE t1.cusid=t2.cusid);`
	Show the customer ID that pays the highest fee (via a subquery)

`SELECT @maxfee:=MAX(fee) FROM payments;`
`SELECT cusid FROM payments t1 WHERE fee = @maxfee;`
	Show the customer ID that pays the highest fee (via a user set variable)

`SELECT * FROM customers WHERE lastname IN (SELECT lastname`
`FROM customers GROUP BY lastname HAVING COUNT(lastname) > 1);`
	Show the customers which have same last name as other customers

`SELECT cusid FROM payments WHERE fee >`
`ALL (SELECT fee FROM payments WHERE cusid = 4242001;`
	Show the customer IDs that pay fees higher than the highest fee paid by customer ID 4242001

`SELECT * FROM customers WHERE firstname LIKE 'Trill%';`	Select customers whose first name matches the expression: `%` = zero or more chars `_` = a single char
`SELECT * FROM customers WHERE firstname REGEXP '^Art.*r$';`	Select customers whose first name matches the regex

`SELECT firstname, lastname FROM customers WHERE zipcode = 'L1 42HG'`
`UNION`
`SELECT firstname, lastname FROM customers WHERE cusid > 4242001;`
	Select customers that satisfy any of the two requirements

`SELECT firstname, lastname FROM customers WHERE zipcode = 'L1 42HG'`
`INTERSECT`
`SELECT firstname, lastname FROM customers WHERE cusid > 4242001;`
	Select customers that satisfy both of the two requirements

`SELECT firstname, lastname FROM customers WHERE zipcode = 'L1 42HG'`
`EXCEPT`
`SELECT firstname, lastname FROM customers WHERE cusid > 4242001;`
	Select customers that satisfy the first requirement but not the second

SQL	MySQL	Operation

SELECT customers.name, payments.bill
FROM customers, payments
WHERE customers.cusid = payments.cusid;
 	
SELECT customers.name, payments.bill
FROM customers NATURAL JOIN payments;
 	
SELECT customers.name, payments.bill
FROM customers JOIN payments
USING (cusid);
 	
SELECT customers.name, payments.bill
FROM customers JOIN payments
ON customers.cusid = payments.cusid;
	SELECT customers.name, payments.bill
	FROM customers
	[JOIN | INNER JOIN | CROSS JOIN]
	payments
	ON customers.cusid = payments.cusid;
	
	SELECT customers.name, payments.bill
	FROM customers
	[JOIN | INNER JOIN | CROSS JOIN]
	payments
	USING (cusid);
		Perform a **join** (aka **inner join**) of two tables to select data that are in a relationship

SELECT customers.name, payments.bill
FROM customers CROSS JOIN payments;
	SELECT customers.name, payments.bill
	FROM customers JOIN payments;
		Perform a **cross join** (aka **Cartesian product**) of two tables

SELECT customers.name, payments.bill
FROM customers LEFT JOIN payments
ON customers.cusid = payments.cusid;
		Perform a **left join** (aka **left outer join**) of two tables, returning records matching the join condition and also records in the left table with unmatched values in the right table

SELECT customers.name, payments.bill
FROM customers RIGHT JOIN payments
ON customers.cusid = payments.cusid;
		Perform a **right join** (aka **right outer join**) of two tables, returning records matching the join condition and also records in the right table with unmatched values in the left table

MySQL is the most used open source RDBMS (Relational Database Management System). It runs on TCP port 3306.
On RHEL 7 and later it is replaced by its fork **MariaDB**, but the names of the client and of most tools remain unchanged.

mysqld_safe	Start the MySQL server (mysqld) with safety features such as restarting the server if errors occur and logging runtime information to the error logfile. This is the recommended command
mysql_install_db (deprecated) mysqld --initialize	Initialize the MySQL data directory, create system tables, and set up an administrative account. To be run just after installing the MySQL server
mysql_secure_installation	Set password for root, remove anonymous users, disable remote root login, and remove test database. To be run just after installing the MySQL server
mysql -u root -p	Login to MySQL as root and prompt for the password
mysql -u root -p*password*	Login to MySQL as root with the specified password
mysql -u root -p -h *host* -P *port*	Login to the specified remote MySQL host and port
mysql -u root -p -eNB'SHOW DATABASES'	Run an SQL command via MySQL. Flags are: e Run in batch mode N Do not print table header B Do not print table decoration characters +-|
mysqldump -u root -p --all-databases > *dump.sql*	Backup all databases to a dump file
mysqldump -u root -p *db* > *dump.sql*	Backup a database to a dump file
mysqldump -u root -p --databases *db1 db2* > *dump.sql*	Backup multiple databases to a dump file
mysqldump -u root -p *db table1 table2* > *dump.sql*	Backup some tables of a database to a dump file
mysql -u root -p < *dump.sql*	Restore all databases from a dump file (which contains a complete dump of a MySQL server)
mysql -u root -p *db* < *dump.sql*	Restore a specific database from a dump file (which contains one database)
mysql_upgrade -u root -p	Check all tables in all databases for incompatibilities with the current version of MySQL
mysqlcheck	Perform table maintenance. Each table is locked while is being processed. Options are: --check Check table for errors (default) --analyze Analyze table --optimize Optimize table --repair Repair table; can fix almost all problems except unique keys that are not unique
mysqlcheck --check *db table*	Check the specified table of the specified database
mysqlcheck --check --databases *db1 db2*	Check the specified databases
mysqlcheck --check --all-databases	Check all databases

`mysqlslap`	Tool for MySQL stress tests
`mysqltuner.pl`	Review the current MySQL installation configuration for performances and stability
`mysqlreport` (obsolete)	Generate a user-friendly report of MySQL status values
`mytop`	Monitor MySQL processes and queries
`innotop`	Monitor MySQL InnoDB transactions

```
dbs="$(mysql -uroot -ppassword -Bse'SHOW DATABASES;')"
for db in $dbs
do
   [operation on $db]
done
```

Perform an operation on each database name

`SELECT Host, User FROM mysql.user;`	List all MySQL users
`CREATE USER 'user'@'localhost' IDENTIFIED BY 'p4ssw0rd';`	Create a MySQL local user and set their password
`DROP USER 'user'@'localhost';`	Delete a MySQL user

`SET PASSWORD FOR 'user'@'localhost' = PASSWORD('p4ssw0rd');`
`SET PASSWORD FOR 'user'@'localhost' = '*7E684A3DF6273CD1B6DE53';`
	Set a password for a MySQL user.
	The password can be specified either in plaintext or by its hash value

`SHOW GRANTS FOR 'user'@'localhost';`	Show permissions for a user
`GRANT ALL PRIVILEGES ON database.* TO 'user'@'localhost';`	Grant permissions to a user
`REVOKE ALL PRIVILEGES ON database.* FROM 'user'@'localhost';`	Revoke permissions from a user; must match the already granted permission on the same database or table

`GRANT SELECT ON *.* TO 'john'@'localhost' IDENTIFIED BY 'p4ssw0rd';`
`GRANT SELECT ON *.* TO 'john'@'localhost' IDENTIFIED BY PASSWORD '*7E684A3DF6273CD1B6DE53';`
	Create a MySQL user and set their grants at the same time

`FLUSH PRIVILEGES;`	Reload and commit the grant tables; must be run after any GRANT command

`SELECT * INTO OUTFILE 'file.csv'`
`FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'`
`LINES TERMINATED BY '\n' FROM database.table;`
	Export a table to a CSV file

`USE database; SOURCE dump.sql;`	Restore a database from a dump file
`USE database; LOAD DATA LOCAL INFILE 'file' INTO TABLE table;`	Populate a table with data from a file (one record per line, values separated by tabs)

`SHOW CREATE TABLE table;`
`SHOW CREATE VIEW view;`
	Print the CREATE statement that created *table* or *view*

`DO SLEEP(n);`
`SELECT SLEEP(n);`
	Sleep for *n* seconds

`SET PROFILING=1;`	Enable profiling
`SHOW PROFILE;`	Show the profile of the last executed query, with detailed steps and their timing

`statement;`
`statement\g`
	Send an SQL statement to the server

`statement\G`	Display result in vertical format, showing each record in multiple rows
`SELECT /*!99999 comment*/ * FROM database.table;`	Insert a comment
`SELECT /*!v statement*/ * FROM database.table;`	The commented *statement* is executed only if MySQL is version *v* or higher
`\c`	Cancel current input
`\! command`	Run a shell command
`TEE logfile`	Log all I/O of the current MySQL session to the specified logfile

`SHOW VARIABLES;`
`SHOW SESSION VARIABLES;`
`SHOW LOCAL VARIABLES;`
	Print session variables (affecting current connection only)

`SHOW GLOBAL VARIABLES;`	Print global variables (affecting global operations on the server)
`SHOW VARIABLES LIKE '%query%';`	Print session variables that match the given pattern

`SHOW VARIABLES LIKE 'hostname';`
`SELECT @@hostname;`
	Print a session variable with the given name

`SET sort_buffer_size=10000;`
`SET SESSION sort_buffer_size=10000;`
`SET LOCAL sort_buffer_size=10000;`
`SET @@sort_buffer_size=10000;`
`SET @@session.sort_buffer_size=10000;`
`SET @@local.sort_buffer_size=10000;`
	Set a session variable

`SET GLOBAL sort_buffer_size=10000;`
`SET @@global.sort_buffer_size=10000;`
	Set a global variable

`SHOW STATUS;`
`SHOW SESSION STATUS;`
`SHOW LOCAL STATUS;`
	Print session status (concerning current connection only)

`SHOW GLOBAL STATUS;`	Print global status (concerning global operations on the server)
`SHOW STATUS LIKE '%wsrep%';`	Print session status values that match the given pattern
`SHOW WARNINGS;`	Print warnings, errors and notes resulting from the most recent statement in the current session that generated messages
`SHOW ERRORS;`	Print errors resulting from the most recent statement in the current session that generated messages
`SHOW TABLE STATUS;`	Print information about all tables of the current database e.g. engine (InnoDB or MyISAM), rows, indexes, data length
`SHOW ENGINE INNODB STATUS;`	Print statistics concerning the InnoDB engine

`SELECT * FROM information_schema.processlist;`
`SHOW FULL PROCESSLIST;`
	Print the list of threads running in the local session; if run as root, print the list of threads running on the system

`SELECT * FROM information_schema.processlist`
`WHERE user='you';`
	Print the list of threads running in the local session and all other logged in sessions

`SELECT VERSION();`	Print the version of the MySQL server

`SELECT CURDATE();`
`SELECT CURRENT_DATE;`
	Print the current date

`SELECT CURTIME();`
`SELECT CURRENT_TIME;`
	Print the current time

`SELECT NOW();`	Print the current date and time
`SELECT USER();`	Print the current user@hostname that is logged in

`INSTALL COMPONENT`
`'file://component_validate_password';`
	Install the Validate Password component

`SHOW VARIABLES LIKE 'validate_password%';`	Print the current settings of the Validate Password component

`UNINSTALL COMPONENT`
`'file://component_validate_password';`
	Uninstall the Validate Password component

`\s`	Print status information about server and current connection

```
SELECT table_schema AS "Name",
SUM(data_length+index_length)/1024/1024 AS "Size in Mb"
FROM information_schema.tables GROUP BY table_schema;
```

Display the sizes of all databases in the system (counting data + indexes)

```
SELECT table_schema AS "Name",
SUM(data_length+index_length)/1024/1024 AS "Size in Mb"
FROM information_schema.tables WHERE table_schema='database';
```

Display the size of *database*

```
SELECT table_name AS "Name",
ROUND(((data_length)/1024/1024),2) AS "Data size in Mb",
ROUND(((index_length)/1024/1024),2) AS "Index size in Mb"
FROM information_schema.TABLES WHERE table_schema='database'
ORDER BY table_name;
```

Display data and index size of all tables of *database*

```
SELECT table_name, table_rows
FROM information_schema.tables WHERE table_schema='database';
```

Print an estimate of the number of rows of each table of *database*

```
SELECT SUM(data_length+index_length)/1024/1024 AS "InnoDB Mb"
FROM information_schema.tables WHERE engine='InnoDB';
```

Display the amount of InnoDB data in all databases

```
SELECT table_name, engine
FROM information_schema.tables WHERE table_schema = 'database';
```

Print name and engine of all tables in *database*

```
SELECT CONCAT('KILL ',id,';')
FROM information_schema.processlist WHERE user='user'
INTO OUTFILE '/tmp/killuser'; SOURCE /tmp/killuser;
```

Kill all connections belonging to *user*

```
SELECT COUNT(1) SlaveThreadCount
FROM information_schema.processlist WHERE user='system user';
```

Distinguish between master and slave server; returns 0 on a master, >0 on a slave

```
SELECT ROUND(SUM(CHAR_LENGTH(field)<40)*100/COUNT(*),2)
FROM table;
```

Display the percentage of rows on which the string *field* is shorter than 40 chars

```
SELECT CHAR_LENGTH(field) AS Length, COUNT(*) AS Occurrences
FROM table GROUP BY CHAR_LENGTH(field);
```

Display all different lengths of string *field* and the number of times they occur

```
SELECT MAX(CHAR_LENGTH(field)) FROM table;
```

Display the longest string stored in *field*

```
SHOW FULL TABLES IN database WHERE table_type LIKE 'VIEW';
```

Display the list of views in *database*

```
SELECT "Table 1" AS `set`, t1.* FROM table1 t1 WHERE
ROW(t1.col1, t1.col2, t1.col3) NOT IN (SELECT * FROM table2)
UNION ALL
SELECT "Table 2" AS `set`, t2.* FROM table2 t2 WHERE
ROW(t2.col1, t2.col2, t2.col3) NOT IN (SELECT * FROM table1)
```

Display the differences between the contents of two tables *table1* and *table2* (assuming the tables are composed of 3 columns each)

How to resync a master-slave replication

1. On the master, on terminal 1:

   ```
   mysql -uroot -p
   RESET MASTER;
   FLUSH TABLES WITH READ LOCK;
   SHOW MASTER STATUS;
   ```

   Note the values of MASTER_LOG_FILE and MASTER_LOG_POS; these values will need to be copied on the slave

2. On the master, on terminal 2:

   ```
   mysqldump -uroot -p --all-databases > /path/to/dump.sql
   ```

   It is not necessary to wait until the dump completes

3. On the master, on terminal 1:

   ```
   UNLOCK TABLES;
   ```

4. Transfer the dump file from the master to the slave

5. On the slave:

   ```
   mysql -uroot -p
   STOP SLAVE;
   SOURCE /path/to/dump.sql;
   RESET SLAVE;
   CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin.nnnnnn', MASTER_LOG_POS=mm;
   START SLAVE;
   SHOW SLAVE STATUS;
   ```

How to recover the MySQL root password

1. Stop the MySQL server

2. Restart the MySQL server skipping the grant tables

   ```
   mysqld_safe --skip-grant-tables --skip-networking &
   ```

3 Connect to the MySQL server passwordlessly

   ```
   mysql -uroot
   ```

4. Reload the grant tables

   ```
   FLUSH PRIVILEGES;
   ```

5. Change the root password

   ```
   SET PASSWORD FOR 'root'@'localhost' = PASSWORD('newpassword');
   ```

6. Stop the MySQL server and restart it normally

PostgreSQL (aka **Postgres**) is an open source object-relational database. By default it listens for connections on TCP port 5432.

\list \l	List all databases
\list+ \l+	List all databases, displaying database size and description
\connect *database* \c *database*	Connect to *database*
\q	Quit

How to set up PostgreSQL with a database owned by a user

1.	Set up PostgreSQL	postgresql-setup initdb
2.	Change the password of the postgres shell user	passwd postgres
3	Create the *user* shell user	useradd *user*
4.	Switch to the postgres shell user and connect to PostgreSQL	su - postgres psql -U postgres
5.	In PostgreSQL, create the *user*	CREATE ROLE *user* WITH LOGIN; \password *user* \q
6.	Create a *database* owned by *user*	createdb -E utf8 -l C -T template0 *database* -O *user*
7.	Switch to the postgres shell user and connect to PostgreSQL	su - postgres psql -U postgres
8.	In PostgreSQL, grant the necessary privileges on *database* to *user*	GRANT ALL PRIVILEGES ON DATABASE *database* TO *user*; \q
9.	Verify that *user* can login to PostgreSQL	su - *user* psql -U *user* -W

The **X Window System** (aka **X11** or **X**) is a windowing system for Linux and UNIX-like OSes, providing a basic framework for GUI applications via a client-server model. The open source implementation is the **X.Org Server**.
A **display manager** provides a login screen to enter an X session and introduces the user to the **desktop environment** (e.g. GNOME, KDE, CDE, Enlightenment).

Display Manager		Configuration files		Display Manager greeting screen

`xdm`	X Display Manager	`/etc/x11/xdm/Xaccess`	Control of inbound requests from remote hosts	Defined in `/etc/x11/xdm/Xresources` by the line:
				`xlogin*greeting: \`
				`Debian GNU/Linux (CLIENTHOST)`

		`/etc/x11/xdm/Xresources`	Configuration settings for X applications and the login screen	
		`/etc/x11/xdm/Xservers`	Association of X displays with local X server software, or with X terminals via XDMCP	
		`/etc/x11/xdm/Xsession`	Script launched by xdm after login	
		`/etc/x11/xdm/Xsetup_0`	Script launched before the graphical login screen	
		`/etc/x11/xdm/xdm-config`	Association of all xdm configuration files	
`gdm`	GNOME Display Manager	`/etc/gdm/gdm.conf` or `/etc/gdm/custom.conf`		Configured via `gdmsetup`
`kdm`	KDE Display Manager	`/etc/kde/kdm/kdmrc`		Configured via `kdm_config`

`/etc/X11/xorg.conf` Configuration file for X

`~/.Xresources` Configuration settings for X applications, in the form `program*resource: value`

`$DISPLAY` Environment variable defining the display name of the X server, in the form `hostname:displaynumber.screennumber`

The following line in `/etc/inittab` instructs `init` to launch XDM at runlevel 5:

```
x:5:respawn:/usr/X11R6/bin/xdm -nodaemon
```

The following lines in `/etc/sysconfig/desktop` define GNOME as the default Display Environment and Display Manager:

```
desktop="gde"
displaymanager="gdm"
```

/etc/init.d/xdm start /etc/init.d/gdm start /etc/init.d/kdm start	Start the appropriate Display Manager
startx	Initialize an X session
xorgconfig (Debian) Xorg -configure (Red Hat)	Configure X (text mode)
xorgcfg (Debian) system-config-display (Red Hat)	Configure X (graphical mode)
xhost + 10.3.3.3 xhost - 10.3.3.3	Add or remove 10.3.3.3 to the list of hosts allowed making X connections to the local machine
switchdesk gde	Switch to the GDE Display Manager at runtime
X -version	Show which version of X is running
xdotool	X automation tool
xdotool getwindowfocus	Get the ID of the currently focused window (if run in command line, it is the terminal where this command is typed)
xdotool selectwindow	Pop up an X cursor and get the ID of the window selected by it
xdotool key --window 12345678 Return	Simulate a RETURN keystroke inside window ID 12345678
xprop	X property displayer. Pops up a cursor to select a window
xprop | grep WM_CLASS	Get process name and GUI application name of the selected window
xrandr	Configuration utility for the RandR (Resize and Rotate) X extension
xrandr -q	Show screen(s) size and resolution
xrandr --output eDP1 --right-of VGA1	Extend the screen on a VGA physical monitor situated to the left
xsel	Manipulate the X selection (primary, secondary, and clipboard)
xsel -b < *file*	Copy the contents of a file to the X clipboard
xsel -b -a < *file*	Append the contents of a file to the X clipboard
xsel -b -o	Output onscreen the contents of the X clipboard
xset	Configuration utility for X
xset r rate 200 50	Set key repeat delay to 200 ms and rate to 50 characters per second
xclip	X clipboard tool
cat *file* | xclip -i	Copy the contents of a file to the X clipboard
xautolock	Run a program in case of user inactivity during a X session
xdpyinfo	Display information about the X server
xwininfo	Display information about windows
xosview	Monitor able to display several system parameters (CPU usage, memory usage, load average, page swapping, interrupts, battery level, etc.)

`xmodmap`	Display and edit the keyboard modifier map and keymap table used by X applications
`loadkeys`	Load keyboard translation tables
`kbdcontrol`	Control and configure the keyboard
`xkbcomp` (Red Hat) `ckbcomp` (Ubuntu)	Compile a XKB keyboard description into a format understood by `kbdcontrol` and `loadkeys`
`mkfontdir`	Catalog the newly installed fonts in the new directory
`xset fp+ /usr/local/fonts`	Dynamically add new installed fonts in `/usr/local/fonts` to the X server
`xfs`	Start the X font server
`fc-cache`	Install fonts and build font information cache
`fc-scan` *`fontfile`*`.ttf`	Get information about a font

Main

BackSpace	ff08
Tab	ff09
Linefeed	ff0a
Clear	ff0b
Return	ff0d
Pause	ff13
Scroll_Lock	ff14
Sys_Req	ff15
Escape	ff1b
Delete	ffff

Cursor control

Home	ff50
Left	ff51
Up	ff52
Right	ff53
Down	ff54
Prior	ff55
Page_Up	ff55
Next	ff56
Page_Down	ff56
End	ff57
Begin	ff58

Misc functions

Select	ff60
Print	ff61
Execute	ff62
Insert	ff63
Undo	ff65
Redo	ff66
Menu	ff67
Find	ff68
Cancel	ff69
Help	ff6a
Break	ff6b
Mode_switch	ff7e
script_switch	ff7e
Num_Lock	ff7f

Modifiers

Shift_L	ffe1
Shift_R	ffe2
Control_L	ffe3
Control_R	ffe4
Caps_Lock	ffe5
Shift_Lock	ffe6
Meta_L	ffe7
Meta_R	ffe8
Alt_L	ffe9
Alt_R	ffea
Super_L	ffeb
Super_R	ffec
Hyper_L	ffed
Hyper_R	ffee

Latin 1

space	0020
exclam	0021
quotedbl	0022
numbersign	0023
dollar	0024
percent	0025
ampersand	0026
apostrophe	0027
quoteright	0027
parenleft	0028
parenright	0029
asterisk	002a
plus	002b
comma	002c
minus	002d
period	002e
slash	002f
0 - 9	0030 - 0039
colon	003a
semicolon	003b
less	003c
equal	003d
greater	003e
question	003f
at	0040
A - Z	0041 - 005a
bracketleft	005b
backslash	005c
bracketright	005d
asciicircum	005e
underscore	005f
grave	0060
quoteleft	0060
a - z	0061 - 007a
braceleft	007b
bar	007c
braceright	007d
asciitilde	007e
nobreakspace	00a0
exclamdown	00a1
cent	00a2
sterling	00a3
currency	00a4
yen	00a5
brokenbar	00a6
section	00a7
diaeresis	00a8
copyright	00a9
ordfeminine	00aa
guillemotleft	00ab
notsign	00ac
hyphen	00ad
registered	00ae
macron	00af
degree	00b0
plusminus	00b1
twosuperior	00b2
threesuperior	00b3
acute	00b4
mu	00b5
paragraph	00b6
periodcentered	00b7
cedilla	00b8
onesuperior	00b9
masculine	00ba
guillemotright	00bb
onequarter	00bc
onehalf	00bd
threequarters	00be
questiondown	00bf
Agrave	00c0
Aacute	00c1
Acircumflex	00c2
Atilde	00c3
Adiaeresis	00c4
Aring	00c5
AE	00c6
Ccedilla	00c7
Egrave	00c8
Eacute	00c9
Ecircumflex	00ca
Ediaeresis	00cb
Igrave	00cc
Iacute	00cd
Icircumflex	00ce
Idiaeresis	00cf
ETH	00d0
Eth	00d0
Ntilde	00d1
Ograve	00d2
Oacute	00d3
Ocircumflex	00d4
Otilde	00d5
Odiaeresis	00d6
multiply	00d7
Oslash	00d8
Ooblique	00d8
Ugrave	00d9
Uacute	00da
Ucircumflex	00db
Udiaeresis	00dc
Yacute	00dd
THORN	00de
Thorn	00de
ssharp	00df
agrave	00e0
aacute	00e1
acircumflex	00e2
atilde	00e3
adiaeresis	00e4
aring	00e5
ae	00e6
ccedilla	00e7
egrave	00e8
eacute	00e9
ecircumflex	00ea
ediaeresis	00eb
igrave	00ec
iacute	00ed
icircumflex	00ee
idiaeresis	00ef
eth	00f0
ntilde	00f1
ograve	00f2
oacute	00f3
ocircumflex	00f4
otilde	00f5
odiaeresis	00f6
division	00f7
oslash	00f8
ooblique	00f8
ugrave	00f9
uacute	00fa
ucircumflex	00fb
udiaeresis	00fc
yacute	00fd
thorn	00fe
ydiaeresis	00ff

Latin 2

Aogonek	01a1
breve	01a2
Lstroke	01a3
Lcaron	01a5
Sacute	01a6
Scaron	01a9
Scedilla	01aa
Tcaron	01ab
Zacute	01ac
Zcaron	01ae
Zabovedot	01af
aogonek	01b1
ogonek	01b2
lstroke	01b3
lcaron	01b5
sacute	01b6
caron	01b7
scaron	01b9
scedilla	01ba
tcaron	01bb
zacute	01bc
doubleacute	01bd
zcaron	01be
zabovedot	01bf
Racute	01c0
Abreve	01c3
Lacute	01c5
Cacute	01c6
Ccaron	01c8
Eogonek	01ca
Ecaron	01cc
Dcaron	01cf
Dstroke	01d0
Nacute	01d1
Ncaron	01d2
Odoubleacute	01d5
Rcaron	01d8
Uring	01d9
Udoubleacute	01db
Tcedilla	01de
racute	01e0
abreve	01e3
lacute	01e5
cacute	01e6
ccaron	01e8
eogonek	01ea
ecaron	01ec
dcaron	01ef
dstroke	01f0
nacute	01f1
ncaron	01f2
odoubleacute	01f5
rcaron	01f8
uring	01f9
udoubleacute	01fb
tcedilla	01fe
abovedot	01ff

This table is derived from `keysymdef.h`, which defines keysym codes (i.e. characters or functions associated with each key in the X Window System) as `XK_`*`key`* and its hex value. The *key* can be passed as argument to the `xdotool key` command.

/etc/passwd User accounts

```
root:x:0:0:/root:/bin/bash
bin:x:1:1:/bin:/bin/bash
jdoe:x:500:100:John Doe,,555-1234,,:/home/jdoe:/bin/bash
1    2 3   4   5                    6          7
```

1 Login name

2 Hashed password (obsolete), or `x` if password is in `/etc/shadow`

3 UID – User ID

4 GID – Default Group ID

5 GECOS field – Information about the user: Full name, Room number, Work phone, Home phone, Other

6 Home directory of the user

7 Login shell (if set to `/sbin/nologin` or `/bin/false`, user will be unable to log in)

/etc/shadow User passwords

```
root:$6$qk8JmJHf$X9GfOZ/i9LZP4Kldu6.D3cx2pXA:15537:0:99999:7:::
bin:*:15637:0:99999:7:::
jdoe:!$6$YOiH1otQ$KxeeUKHExK8e3jCUdw9Rxy3Wu53:15580:0:99999:7::15766:
1    2 a b        c                           3     4 5     6 78     9
```

1 Login name

2 Hashed password (`*` if account is disabled, `!` or `!!` if no password is set, prefixed by `!` if the account is locked).
Composed of the following subfields separated by `$`:
a Hashing algorithm: `1` = MD5, `2a` = Blowfish, `5` = SHA256, `6` = SHA512 (recommended)
b Random salt, up to 16 chars long. This is to thwart password cracking attempts based on rainbow tables
c String obtained by hashing the user's plaintext password concatenated to the stored salt

3 Date of last password change (in number of days since 1 January 1970)

4 Days before password may be changed; if 0, user can change the password at any time

5 Days after which password must be changed

6 Days before password expiration that user is warned

7 Days after password expiration that account is disabled

8 Date of account disabling (in number of days since 1 January 1970)

9 Reserved field

/etc/group Group accounts

```
root:x:0:root
jdoe:x:501
staff:x:530:jdoe,asmith
1     2 3   4
```

1 Group name

2 Encrypted password, or `x` if password is in `/etc/gshadow`

3 GID – Group ID

4 Group members (if this is not their Default Group)

/etc/gshadow Group passwords

```
root::root:root
jdoe:!::
staff:0cfz7IpLhW19i::root,jdoe
1     2            3 4
```

1 Group name

2 Encrypted password, or `!` if no password is set (default)

3 Group administrators

4 Group members

`/etc/shadow` and `/etc/gshadow` are mode 000 and therefore readable only by the root user.

useradd -m *user*	Create a user account, creating and populating their homedir from /etc/skel
useradd -mc "*Name Surname*" *user*	Create a user account, specifying their full name
useradd -ms /bin/ksh *user*	Create a user account, specifying their login shell
useradd -D	Show default values for user account creation, as specified in /etc/login.defs and /etc/default/useradd

usermod -c "*Name Surname*" *user*	Modify the GECOS field of a user account
usermod -L *user*	Lock a user account
usermod -U *user*	Unlock a user account

Most options for usermod and useradd are the same.

userdel -r *user*	Delete a user and their homedir
chfn *user*	Change the GECOS field of a user
chsh *user*	Change the login shell of a user
passwd *user*	Change the password of a user
passwd -l *user*	Lock a user account
passwd -S *user*	Show information about a user account: username, account status (L=locked, P=password, NP=no password), date of last password change, min age, max age, warning period, inactivity period in days
chage -E 2022-02-14 *user*	Change the password expiration date; account will be locked at that date
chage -d 13111 *user*	Change the date (in number of days since 1 January 1970) of last password change
chage -d 0 *user*	Force the user to change password at their next login
chage -M 30 *user*	Change the max number of days during which a password is valid
chage -m 7 *user*	Change the min number of days between password changes
chage -W 15 *user*	Change the number of days before password expiration that the user will be warned
chage -I 3 *user*	Change the number of days after password expiration before the account is locked
chage -l *user*	List password aging information for a user
chpasswd	Tool for batch update of passwords. Reads from stdin a list of *username*:*password*
vipw vigr	Edit manually /etc/passwd, /etc/shadow, /etc/group, or /etc/gshadow
adduser deluser	User-friendly front-end commands for user management
system-config-users (Red Hat)	GUI for user and group management

Group management

`groupadd` *group*	Create a group
`groupmod -n` *newname oldname*	Change a group name
`groupdel` *group*	Delete a group
`gpasswd` *group*	Set or change the password of a group
`gpasswd -a` *user group*	Add a user to a group
`gpasswd -d` *user group*	Delete a user from a group
`gpasswd -A` *user group*	Add a user to the list of administrators of the group
`addgroup` `delgroup`	User-friendly front-end commands for group management

On a system, every user is identified by a numeric **UID (User ID)**, and every group by a numeric **GID (Group ID)**.
UID 0 is assigned to the superuser.
UIDs from 0 to 99 should* be reserved for static allocation by the system and not be created by applications.
UIDs from 100 to 499 should* be reserved for dynamic allocation by the superuser and post-install scripts.
UIDs for user accounts start from 500 (Red Hat) or 1000 (SUSE, Debian).

* as recommended by the Linux Standard Base core specifications

A process has an effective, saved, and real UID and GID.

Effective UID	Used for most access checks, and as the owner for files created by the process. An unprivileged process can change its effective UID only to either its saved UID or its real UID.
Saved UID	Used when a process running with elevated privileges needs to temporarily lower its privileges. The process changes its effective UID (usually root) to an unprivileged one, and its privileged effective UID is copied to the saved UID. Later, the process can resume its elevated privileges by resetting its effective UID back to the saved UID.
Real UID	Used to identify the real owner of the process and affect the permissions for sending signals. An unprivileged process can signal another process only if the sender’s real or effective UID matches the receiver's real or saved UID. Child processes inherit the credentials from the parent, so they can signal each other.

`whoami`	Print your username (as effective UID)
`id`	Print your real and effective UID and GID, and the groups of which you are a member
`id` *`user`*	Print real and effective UID and GID, and group membership information, about *user*
`id -u`	Print your effective UID
`who`	Print the list of users logged into the system
`w`	Print the list of users logged into the system, and what they are doing
`users`	Print names of currently logged in users
`finger` *`user@host`*	Print information about *user* on *host*
`rwho`	Print information about currently logged in users for all hosts on the local network
`rusers`	Print names of currently logged in users for all hosts on the local network

sudo

Sudo is a mechanism that allows running a command as another user. Sudo access rights are defined in the sudoers files `/etc/sudoers` and `/etc/sudoers.d/*`; these files must be edited only via `visudo`.
Commands run by sudo users are logged via syslog on `/var/log/auth.log` (Debian) or `/var/log/secure` (Red Hat).

`sudo -u` *`user command`*	Run *command* as *user*

`sudo` *`command`*
`sudo -u root` *`command`*
	Run *command* as root

`sudo su -`
`sudo -i`
	Login on an interactive shell as root

`sudo -u` *`user`* `-s`	Login as *user* with a shell, even if the user's shell is `/sbin/nologin` or similar
`sudo -l`	List the allowed commands for the current user
`sudo !!`	Run again the last command, but this time as root

`sudoedit /etc/passwd`
`sudo -e /etc/passwd`
	Edit safely a file (in this case, `/etc/passwd`) according to security policies. It is recommended to have nonprivileged users run this command instead of sudoing text editors as root on protected files, because the text editor might spawn a shell, causing security issues

`visudo`	Edit safely the sudoers file
`visudo -c`	Check the sudoers file for syntax errors, unused aliases, etc.
`su` *`user`*	Run a shell as *user*

`su`
`su root`
	Run a shell as root

`su -c "fdisk -l"`	Pass a single command to the shell

`su -`
`su -l`
	Ensure that the spawned shell is a login shell, hence running login scripts and setting the correct environment variables. Recommended option

`gksudo -u root` *`command`*
`gksu -u root -l`
	GUI front-ends to `su` and `sudo` used to run an X Window command or application as root. Pops up a requester prompting the user for root's password

`runuser -u` *`user command`*	Run *command* as *user*. Can be launched only by root
`doas -u` *`user command`*	Run *command* as *user*. Simpler version of sudo; access rights are configured in the file `/etc/doas.conf`

`chvt` *`n`* CTRL ALT F*n*	Make `/dev/tty`*`n`* the foreground terminal
`clear` CTRL L	Clear the terminal screen
`tset` `reset`	Initialize the terminal
`vlock` `away`	Lock the virtual console (terminal)
`tty`	Print your terminal device (e.g. `/dev/tty1`, `/dev/pts/1`)
`stty`	Change or display terminal line settings
`stty -ixon`	Disable XON/XOFF flow control
`tmux`	Terminal multiplexer
`reptyr`	Attach an existing running program to a new terminal
`screen`	Screen manager that multiplexes a single virtual VT100/ANSI terminal between multiple processes or shells. When the connection to a terminal is lost (e.g. because the terminal is closed manually, the user logs out, or the remote SSH session goes into timeout), a SIGHUP is sent to the shell and from there to all running child processes which are therefore terminated. The `screen` command starts an interactive shell screen session, to which the user will be able to reattach later
`screen -S` *`sessionname`*	Start a screen session with the specified session name
`screen` *`command`*	Start the specified command in a screen session; session will end when the command exits
`screen -list`	Show the list of detached screen sessions

`screen -r` *`pid.tty.host`*
`screen -r` *`owner/pid.tty.host`*
	Resume a detached screen session

`screen -R`	Resume the last detached screen session
`screen -d -R` *`sessionname`*	Detach a remote screen session and reattach your current terminal to it
CTRL A	Send a command to the window manager: `0` ... `9` Switch between screen sessions `c` Create a new screen session `?` Show help

How to detach an already running job that was not started in a `screen` session
(this procedure detaches the job from its parent shell, so that the job will not be killed when the terminal is closed)

1.	CTRL Z	Suspend the job
2.	`bg`	Send the job to background
3.	`jobs`	Show the number (let us assume is *n*) of the backgrounded job
4.	`disown -h %`*`n`*	Mark job *n* so it will not receive a SIGHUP from its parent shell

or

1.	`screen`	Start a new screen session
2.	`reptyr` *`pid`*	Attach the job with process ID *pid* to the new terminal (screen session)

`write` *user*	Write interactively a message to the terminal of *user* (which must be logged in)
`echo "`*Message*`" | write` *user*	Write a message to the terminal of *user* (which must be logged in)
`wall`	Write interactively a message to the terminal of all logged in users
`echo "`*Message*`" | wall`	Write a message to the terminal of all logged in users
`talk` *user*	Open an interactive chat session with *user* (which must be logged in)
`mesg`	Display your current message permission status. Enabling/disabling the other users to send you messages is done by enabling/disabling the group write permission of your terminal device, which is owned by system group `tty`. The root user is always able to message users, regardless of their permission status

`mesg y`
`chmod g+w $(tty)`
	Allow the other users to message you via `write`, `wall`, and `talk`

`mesg n`
`chmod g-w $(tty)`
	Disallow the other users to message you via `write`, `wall`, and `talk`

`cron` is a job scheduler, allowing repeated execution of commands specified in crontab files.
The `crond` daemon checks the crontab files every minute and runs the command as the specified user at the specified times.
It is not necessary to restart `crond` after the modification of a crontab file, as the changes will be reloaded automatically.
On Systemd-based distros, Systemd timers are an alternative to cron jobs.

If `/etc/cron.allow` exists, only users listed therein can access the service.
If `/etc/cron.deny` exists, all users except those listed therein can access the service.
If none of these files exist, all users can access the service.

`/etc/crontab` `/etc/cron.d/*`	System-wide crontab files

`/etc/cron.hourly/`
`/etc/cron.daily/`
`/etc/cron.weekly/`
`/etc/cron.monthly/`
	Scripts placed in these directories will be automatically executed on the specified periods

`/var/spool/cron/`*`user`*	Crontab of *user*. This file has the same format as the system-wide crontab files, except that the "user" field is not present

`crontab -e`	Edit your user crontab file
`crontab -l`	List the contents of your crontab file
`crontab -e -u` *`user`*	Edit the crontab file of another *user* (command available only to the superuser)

/etc/crontab	
`# m  h   dom mon dow  user  command`	
` 25  6    *   *   1   root  /opt/script1.sh`	every Monday at 6:25 AM
`*/5 16    *   *   *   root  /opt/script2.sh`	from 4:00 to 4:55 PM every 5 minutes every day
`0,30 7   25  12   *   jdoe  /home/jdoe/foo.sh`	at 7:00 and 7:30 AM on 25th December
`  3 17    *   *  1-5  root  /bin/rm /tmp/abc.o`	at 5:03 PM every day, from Monday to Friday

m	minutes
h	hours
dom	day of month (1-31)
mon	month (1-12 or jan-dec)
dow	day of week (0-7 or sun-sat; 0=7=Sunday)
user	User as whom the command will be executed
command	Command that will be executed at the specified times

The `crond` daemon also runs anacron jobs, which allow execution of periodic jobs on a machine that is not always powered on, such as a laptop. Only the superuser can schedule anacron jobs, which have a granularity of one day (vs one minute for cron jobs).

`/var/spool/anacron/`*`jobid`*	Date of the last execution of the anacron job identified by *jobid*

/etc/anacrontab	
`# period  delay  job-identifier  command`	
`  7       10     cron.weekly     /opt/script3.sh`	If the job has not been run in the last 7 days, wait 10 minutes and then execute the command

period	period, in days, during which the command was not executed
delay	delay to wait, in minutes, before execution of the command
job-identifier	job identifier in anacron messages; should be unique for each anacron job
command	command that will be executed

`at` is used for scheduled execution of commands that must run only once. Execution of these commands is the duty of the `atd` daemon.

If `/etc/at.allow` exists, only users listed therein can access the service.
If `/etc/at.deny` exists, all users except those listed therein can access the service.
If none of these files exist, no user except the superuser can access the service.

`at 5:00pm tomorrow script.sh`
`at -f listofcommands.txt 5:00pm tomorrow`
`echo "rm file" | at now+2 minutes`
	Execute a command once at the specified time (absolute or relative)

`at -l`
`atq`
	List the scheduled jobs

`at -d 3`
`atrm 3`
	Remove job number 3 from the list

Systemd timers are an alternative to cron jobs.

`/etc/systemd/system/`	Directory containing timers and associated services
`systemctl list-timers`	List all active timers
`systemctl list-timers --all`	List all loaded timers, active and inactive

`batch`	Schedule execution of a command for when the system is not too charged. Reads a command from stdin and runs it when the system's load average falls below 0.8

`bc`	Calculator
`dc`	Calculator featuring unlimited precision arithmetic
`factor`	Find the prime factors of an integer
`units`	Convert quantities between different units
`seq`	Print a sequence of numbers

`seq -s*` *n* `| bc`	Calculate the factorial of *n*

`datamash`	Perform numeric operations, apply statistical functions, or change formatting on tabular data
`vd`	VisiData, an interactive tool to explore and operate on tabular data
`jp`	Draw simple plots from CSV or JSON data
`daff`	Compare tabular data and find the differences

`gnuplot`	Utility to plot 2D and 3D graphs

`in2csv`	Convert various tabular data formats into CSV. Part of the csvkit Python package
`sql2csv`	Execute SQL queries on a database and output the result as CSV
`csvclean`	Correct common syntax error on a CSV file
`csvcut`	Filter data on a CSV file
`csvgrep`	Find data in specific columns of a CSV file
`csvjoin`	Join CSV tables (similarly to SQL JOIN)
`csvsort`	Sort data on a CSV file
`csvstack`	Stack up rows from CSV files
`csvformat`	Convert a CSV file to another format
`csvjson`	Convert a CSV file to JSON format
`csvlook`	Render a CSV file in the terminal as a fixed width table (compatible with Markdown)
`csvpy`	Load a CSV file into a Python CLI shell
`csvsql`	Generate SQL queries from a CSV file and execute them on a database
`csvstat`	Print statistics for all columns of a CSV file

`q`	Execute SQL queries against CSV files
`textql`	Execute SQL queries against CSV files

`cc` C compiler

`gcc` GNU C and C++ compiler

`g++` GNU C++ compiler

`ld` GNU linker. Generates an executable file from object files created during compilation

`gasp` Preprocessor for assembly programs

`gdb` GNU debugger. Displays what is happening inside a program while it executes

`make` Utility for automatic compiling, re-compiling, and installation of multi-file programs.
It determines automatically which parts of a multi-file program need recompiling. The relationships about these parts, and the commands that must be used to update them, are described in a Makefile (`./Makefile` by default). The Makefile also describes targets that are going to be used as arguments to the `make` command to perform the desired action, e.g.:

- `all` Compile the whole program
- `install` Compile the program and install it, copying the executable file and all accessory files (libraries, manual, etc.) in their final destination directory for actual use
- `clean` Delete all temporary files in the current directory that are normally created by the compilation of the program, without deleting configuration files
- `dist` Create a distribution tarfile for the program

`shc` Shell script compiler, used to prevent a shell script from inspection or modification. It encrypts a shell script, generates C source code, and compiles the C code into a stripped binary executable file

`patch` Apply or remove a patch

`lsdiff` List the files which are modified in a patch

`magick`	ImageMagick, a versatile tool to edit, transform, and convert image files
`scrot`	Take a screenshot
`exiftool`	Read, write, modify, and delete Exif metadata in image files
`exiv2`	Read, write, modify, and delete Exif, IPTC, and XMP metadata in image files
`beep`	Produce a beep from the machine's speakers
`speaker-test`	Speaker test tone generator for the ALSA (Advanced Linux Sound Architecture) framework
`arecord`	Sound recorder for the ALSA soundcard driver
`aplay`	Sound player for the ALSA soundcard driver
`sox`	Sound eXchange, "the Swiss Army knife" to read and write audio files
`ncmpc`	`mpd` (Music Player Daemon) client with ncurses UI
`ncmpcpp`	`mpd` client with improved features with respect to `ncmpc`
`lsdvd`	List the contents of a DVD
`youtube-dl`	Download a video from YouTube

`cal`	Calendar
`banner`	Print a text in large letters made of the character #
`figlet`	Print a text in large letters, in a specific font
`toilet`	Print a text in large colorful letters, in a specific font
`lolcat`	Print a text in rainbow coloring
`jp2a`	Convert a JPG image into ASCII art
`tesseract`	OCR tool to extract text from an image
`aspell`	Spell checker
`dict`	Query dictionaries on remote machines via the DICT dictionary protocol
`fortune`	Print a random aphorism, like those found in fortune cookies
`cloc`	Count lines of source code
`nnn`	Terminal file manager
`ipcalc`	IP addresses calculator
`grepcidr`	IP addresses filter against CIDR specifications
`on_ac_power`	Return 0 (true) if machine is connected to AC power, 1 (false) if on battery. Useful for laptops
`pwgen`	Random password generator
`pwqgen`	Random password generator with controllable quality
`uuidgen`	UUID generator (random or time-based)
`haveged`	Random number generator using the HAVEGE (Hardware Volatile Entropy Gathering and Expansion) algorithm. Can be run as a daemon to automatically replenish `/dev/random` whenever the supply of random bits in the random device gets too low
`goaccess`	Real-time webserver log analyzer with ncurses UI. Also able to produce its output in HTML format
`gotty` *`command`*	Launch a CLI *command* and show the results in a web page (by running a web server on port 8080)
`gnome-terminal`	GNOME shell terminal GUI
`gnome-tweaks`	GNOME Tweak Tool GUI
`cool-retro-term`	Terminal emulator GUI that mimics old cathodic tube screens
`fsv`	File System Visualizer, a 3D file manager GUI. Open source clone of SGI's `fsn` for IRIX

Red Hat Linux	1995 - 2004	One of the first Linux distros to support ELF binaries.
Red Hat Enterprise Linux (RHEL)	2000 - present	Most used, and de facto standard, commercial Linux distro for servers in corporate environment. Initially based on Red Hat Linux.
Fedora	2003 - present	Upstream source for RHEL and CentOS / CentOS Stream.
CentOS	2004 - 2021	Free and community-supported Linux distro, downstream of RHEL until 2020, when Red Hat shifted development to **CentOS Stream** as upstream source for RHEL.
Rocky Linux	2021 - present	Successor to CentOS, created by the original founder of CentOS when Red Hat stopped its development.
AlmaLinux	2021 - present	Free and community-supported Linux distro, created by CloudLinux to replace CentOS when Red Hat stopped its development.
CloudLinux OS	2010 - present	Commercial Linux distro marketed to shared hosting providers and developed by CloudLinux. Based on CentOS.
Scientific Linux (SL)	2004 - present	Aimed at scientific environments (labs and universities) and developed originally by Fermilab, CERN, DESY, and ETH Zurich. Derived from RHEL.
Caldera Network Desktop	1995 - 2002	Early Linux distro based on Red Hat Linux. In 1997 it became **Caldera OpenLinux (COL)**.
MCC Interim Linux	1992 - 1996	First Linux distro for the general public, released by the University of Manchester.
Yggdrasil Linux/GNU/X (LGX)	1992 - 1995	The first Live CD Linux distro (i.e. usable without installation on the hard disk).
Softlanding Linux System (SLS)	1992 - 1993	First Linux distro to include the X Window System and an extended set of software packages.
Slackware	1993 - 2016	Created as a cleanup of SLS, with focus on design simplicity.
SUSE Linux	1994 - present	Based on Slackware, and similar to Red Hat Linux. In 2003 it became **SUSE Linux Enterprise (SLE)**.
openSUSE	2005 - present	Fork of SLE aimed at promoting free and open source software.
CRUX	2002 - present	Lightweight Linux distro aimed at experienced users. It uses a BSD-like package management system.
Arch Linux	2002 - present	Focused on design simplicity and minimalism. Inspired by CRUX.
Manjaro	2011 - present	Based on Arch Linux.
Garuda Linux	2020 - present	Based on Arch Linux.
EndeavourOS	2019 - present	Based on Arch Linux. Successor to **Antergos**.
Gentoo	2000 - present	Distro in which all programs' source code is compiled locally and is customized and optimized for the specific type of computer, resulting in improved performances. Originally called **Enoch Linux**.
Lightweight Portable Security (LPS)	2007 - 2021	Live CD Linux distro developed by the US Department of Defense and designed to serve as a secure network end node. Renamed **Trusted End Node Security (TENS)** in 2011.
Red Flag Linux	1999 - 2020	Linux distro developed in China.
Red Star OS	2008 - present	Official state OS of North Korea, bundled with government spyware. Its UI resembles Microsoft Windows XP (v1 and v2) or Apple macOS (v3 and v4).

Debian	1993 - present	Composed of free and open source software. One of the first Linux distros.
Ubuntu	2004 - present	The most known user-friendly distro, based on Debian. It spawned a number of derivative distros e.g. **Lubuntu** (lightweight distro with LXQt instead of GNOME), **Kubuntu** (with KDE), and **Xubuntu** (with Xfce).
Linux Mint	2006 - present	Based on Ubuntu, offers full multimedia support (codecs, etc).
Pop!_OS	2017 - present	Based on Ubuntu, offers full support for AMD and Nvidia GPUs. Built by computer manufacturer System76 and preinstalled on their systems.
elementary OS	2011 - present	Focused on immediate usability, with a UI resembling Apple macOS. Based on Ubuntu.
Zorin OS	2009 - present	Distro providing a UI that can be customized to resemble Microsoft Windows or Apple macOS. Based on Ubuntu.
Puppy Linux	2003 - 2020	Lightweight, user-friendly distro with minimal memory footprint.
Knoppix	2000 - present	Live CD distro, based on Debian.
Kali Linux	2013 - present	The de facto "hacker distro", designed for digital forensics and pentesting. Based on Debian. Rebuild of **BackTrack**, which was based on Knoppix.
Linux Mandrake	1998 - 2011	The first user-friendly Linux distro. Later merged with **Conectiva Linux** to become **Mandriva Linux**.
DemoLinux	1998 - 2001	One of the first Live CD Linux distros. Based initially on Linux Mandrake, then on Debian.
Devuan	2016 - present	Fork of Debian that uses init-like systems instead of systemd.
Damn Small Linux (DSL)	2005 - 2008	Designed to run on older hardware with minimal amounts of RAM. Distributed as a Live CD of about 50 MB in size. Based on Knoppix.
Tiny Core Linux (TCL)	2009 - present	Minimalist Linux distro, about 10 MB in size, based on BusyBox. Created by the developer of Damn Small Linux.
Bayanihan Linux	2003 - 2011	Linux desktop distro developed by the Philippine government. Based originally on Red Hat Linux and Fedora, then on Debian.
Pardus	2005 - present	Linux desktop distro developed by the Turkish government. Based on Debian.
Astra Linux	2011 - present	Linux distro developed and certified for use within Russian armed forces and intelligence agencies. Based on Debian.
Deepin	2004 - present	Linux distro used mostly in China, criticized for possible breaches of user privacy. Formerly known as **Hiweed Linux**. Based on Debian.
Tails	2009 - present	The Amnesic Incognito Live System. Distro focused on privacy and anonymity; runs from a Live USB/DVD, leaves no digital footprint on the machine, and connects to the Internet exclusively via Tor. Based on Debian. Successor to **Incognito**, which was based on Gentoo.
Whonix	2012 - present	Distro focused on privacy and anonymity; consists of a "Workstation" Debian VM and a "Tor Gateway" Debian VM. Previously called **TorBOX**. Based on **Kicksecure**, a hardened Debian derivative providing protection against malicious code through defense-in-depth.
Qubes OS	2012 - present	Security-focused single-user OS which implements Security by Isolation, running each application in a securely-isolated compartment called qube. A different VM (via Xen) is run for each different domain of trust.

This is a partial list of Linux distributions. More than one thousand Linux distros, either living or defunct, exist.

Locale environment variables	
`LANG` `LANGUAGE`	Language, stored in `/etc/default/locale`. When scripting, it is recommended to set `LANG=C` because this specifies the minimal locale environment for C translation, and guarantees a standard collation and formats for the execution of scripts
`LC_CTYPE`	Character classification and case conversion
`LC_NUMERIC`	Non-monetary numeric formats
`LC_TIME`	Date and time formats
`LC_COLLATE`	Alphabetical order
`LC_MONETARY`	Monetary formats
`LC_MESSAGES`	Language and encoding of system messages and user input
`LC_PAPER`	Paper size
`LC_NAME`	Personal name formats
`LC_ADDRESS`	Geographic address formats
`LC_TELEPHONE`	Telephone number formats
`LC_MEASUREMENT`	Measurement units (metric or others)
`LC_IDENTIFICATION`	Metadata about locale
`LC_ALL`	Special variable overriding all others
The values of these locale environment variables are in the format *`language_territory.encoding`* e.g. `en_US.UTF-8`. The list of supported locales is stored in `/usr/share/i18n/SUPPORTED`.	

`locale`	Show locale environment variables
`locale-gen it_IT.UTF-8`	Generate a locale (in this case IT) by compiling a list of locale definition files
`apt-get install manpages-it language-pack-it` (Debian)	Install a different locale (in this case IT); this affects system messages and manpages
`iconv -f ISO-8859-10` *`filein`* `-t UTF-8 >` *`fileout`*	Convert a text file from a character set to another
`recode cp1251..utf8` *`file`*	Convert a text file from a character set to another

ISO/IEC-8859 is a standard for 8-bit encoding of printable characters. The first 256 characters in ISO/IEC-8859-1 (Latin-1) are identical to those in Unicode.
UTF-8 encoding can represent every character in the Unicode set, and is the de facto standard for text containing characters with diacritics (which do not fit in the ASCII 7-bit set). It was designed for backward compatibility with ASCII. UTF-8 encodes a Unicode character into 8, 16, 24, or 32 bits, whatever necessary; a UTF-8 file containing only ASCII characters is identical to an ASCII file.

System time

`date`	Show current date and time

`date -d "9999 days ago"`
`date -d "1970/01/01 + 4242"`
	Calculate a date and show it

`date +"%F %H:%M:%S"`	Show current date in the format specified
`date +"%s"`	Show current date in Unix time format (i.e. the number of seconds elapsed since 00:00:00 1/1/1970)
`date -s "20210104 23:30:00"`	Set the date
`date 010423302021`	Set the date, in the format *MMDDhhmmYYYY*

`timedatectl`	Show current date and time

`timedatectl set-time 2021-01-04`
`timedatectl set-time 23:30`
	Set the date

`timedatectl list-timezones`	List all timezones

`zdump GMT`	Show current date and time in the GMT timezone

`tzselect`
`tzconfig`
`dpkg-reconfigure tzdata` (Debian)
`timedatectl set-timezone` *`tz`* (Red Hat)
	Set the timezone

`/etc/timezone` (Debian)	Timezone
`/etc/localtime` (Red Hat)	Timezone. This is a symlink to the appropriate timezone file in `/usr/share/zoneinfo/`

`hwclock --show`
`hwclock -r`
	Show the hardware clock

`hwclock --hctosys`
`hwclock -s`
	Set the system time from the hardware clock

`hwclock --systohc`
`hwclock -w`
	Set the hardware clock from system time

`hwclock --utc`	Indicate that the hardware clock is kept in Coordinated Universal Time
`hwclock --localtime`	Indicate that the hardware clock is kept in local time

NTP (Network Time Protocol) is used to keep a machine's clock in sync with Internet timeservers. It uses UDP port 123.

`ntpd`	NTP daemon
`ntpd -q`	Synchronize the time once and quit
`ntpd -g`	Force NTP to start even if clock is off by more than the panic threshold (1000 secs)
`ntpd -nqg`	Start NTP as a non-daemon, force synchronization of the clock, and quit. The NTP daemon must not be running when this command is launched
`ntpdc`	Query and modify the state of the NTP daemon
`ntpq -p` *timeserver*	Print the list of peers for the timeserver
`ntptrace`	Trace from where a NTP server gets its time, following the chain of NTP servers back to the primary time source
`ntpdate` *timeserver*	Synchronizes the clock with the specified timeserver
`ntpdate -b` *timeserver*	Brutally set the clock, without waiting for it to adjust slowly
`ntpdate -q` *timeserver*	Query the timeserver without setting the clock

The `ntpdate` command is deprecated; to synchronize the clock, `ntpd` should be used instead.

`chronyd`	Daemon for chrony, a versatile NTP client/server
`chronyc`	Command line interface for the chrony daemon

syslog

/etc/syslog.conf

```
# facility.level                  action
*.info;mail.none;authpriv.none   /var/log/messages
authpriv.*                       /var/log/secure
mail.*                           /var/log/maillog
*.alert                          root
*.emerg                          *
local5.*                         @10.7.7.7
local7.*                         /var/log/boot.log
```

Facility	**Level**	**Action**	
Creator of the message	Severity of the message	Destination of the message	

`auth` or `security`†
`authpriv`
`cron`
`daemon`
`kern`
`lpr`
`mail`
`mark` (for syslog internal use)
`news`
`syslog`
`user`
`uucp`
`local0` ... `local7` (custom)
	`emerg` or `panic`† (highest)
	`alert`
	`crit`
	`err` or `error`†
	`warning` or `warn`†
	`notice`
	`info`
	`debug` (lowest)
	
	`none` (facility disabled)
		`file`
		`@`*`host`*
		`user1``,`*`user2`*`,`*`user3`*
		`*`
			message is written into a log *file*
			message is sent to a logger server *host* (via UDP port 514)
			message is sent to the specified users' consoles
			message is sent to all logged in users' consoles

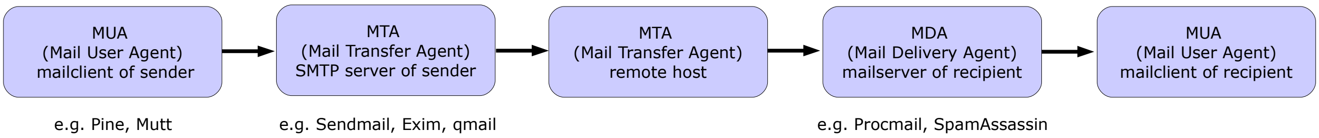
† = deprecated			

Facilities and levels are listed in the manpage `man 3 syslog`.

`syslogd` `rsyslogd` (Ubuntu 14)	Daemon logging events from user processes
`klogd`	Daemon logging events from kernel processes

`/var/log/messages`	Global system logfile
`/var/log/dmesg`	Kernel ring buffer information
`/var/log/kern.log`	Kernel log
`/var/log/boot.log`	Information logged during boot

`logger -p auth.info "`*`Message`*`"`	Send a message to syslog with facility "auth" and priority "info"
`logrotate`	Rotate logs. It gzips, renames, and eventually deletes old logfiles according to the configuration files `/etc/logrotate.conf` and `/etc/logrotate.d/*`. It is usually scheduled as a daily cron job



`~/.forward`	Mail address(es) to which forward the user's mail, or mail commands
`/etc/aliases` `/etc/mail/aliases`	Aliases database for users on the local machine. Each line has syntax *`alias`*`:` *`user`*
`/var/spool/mail/`*`user`*	Inbox for *`user`* on the local machine
`/var/log/mail.log` (Debian) `/var/log/maillog` (Red Hat)	Mail logs

`mail` `mailx`	Mailclient with advanced commands for non-interactive (batch) use
`pine`	Mailclient (obsolete)
`alpine`	Mailclient, a replacement for `pine`
`lsmbox`	List the number of messages in a mailbox
`swaks`	The Swiss Army's Knife SMTP, a flexible and scriptable SMTP test tool

`mailx -s "`*`Subject`*`" -S smtp="`*`mailserver`*`:25" \`
`user@domain.com` `<` *`messagefile`*
	Send a mail message to *user@domain.com* via an external SMTP server *mailserver*

`uuencode` *`binaryfile`* `| mail` *`user@domain.com`*	Send a binary file to *user@domain.com* (obsolete, not recommended because many mailclients will display the received attachment inline)
`mutt -a` *`binaryfile`* `--` *`user@domain.com`* `< /dev/null`	Send a binary file to *user@domain.com* using the Mutt MUA

Mailbox formats		
mbox	Each mail folder is a single file, storing multiple email messages. Advantages: universally supported; fast search inside a mail folder. Disadvantages: issues with file locking; possible mailbox corruption.	`$HOME/Mail/`*`folder`*
Maildir	Each mail folder is a directory, and contains the subdirectories `/cur`, `/new`, and `/tmp`. Each email message is stored in its own file with a unique filename ID. The process that delivers an email message writes it to a file in the `tmp/` directory, and then moves it to `new/`. The moving is commonly done by hard linking the file to `new/` and then unlinking the file from `tmp/`, which guarantees that a MUA will not see a partially written message as it never looks in `tmp/`. When the MUA finds mail messages in `new/` it moves them to `cur/`. Advantages: fast location/retrieval/deletion of a specific mail message; no file locking needed; can be used with NFS. Disadvantages: some filesystems may not efficiently handle a large number of small files; searching text inside all mail messages is slower.	`$HOME/Mail/`*`folder`*`/`

SMTP commands

```
220 smtp.example.com ESMTP Postfix      (server)
HELO xyz.linux.org                      (client)
250 Hello xyz.linux.org, glad to meet you
MAIL FROM: alice@linux.org
250 Ok
RCPT TO bob@foobar.com
250 Ok
RCPT TO carol@quux.net
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: Alice <alice@linux.org>
To: Bob <bob@foobar.com>
Cc: Carol <carol@quux.net>
Date: Wed, 13 August 2014 18:02:43 -0500
Subject: Test message

This is a test message.
.
250 OK id=1OjReS-0005kT-Jj
QUIT
221 Bye
```

`HELO xyz.linux.org`	Initiate the conversation and identify client host to server
`EHLO xyz.linux.org`	Like HELO, but tell server to use Extended SMTP
`MAIL FROM: alice@linux.org`	Specify mail sender
`RCPT TO: bob@foobar.com`	Specify mail recipient
`DATA`	Specify data to send. Ended with a dot on a single line
`QUIT`	Disconnect
`RSET`	
`HELP`	List all available commands
`NOOP`	Empty command
`VRFY alice@linux.org`	Verify the existence of an email address (this command should not be implemented, for security reasons)
`EXPN list@linux.org`	Tell the actual delivery address of aliases and mailing lists

SMTP response codes

first digit	**1**	Command accepted, but not processed until client sends confirmation
	2	Command successfully completed
	3	Command accepted, but not processed until client sends more information
	4	Command failed due to temporary errors
	5	Command failed due to permanent errors
second digit	**0**	Syntax error or command not implemented
	1	Informative response in reply to a request for information
	2	Connection response in reply to a data transmission
	5	Status response in reply to a mail transfer operation
third digit		Specifies further the response

211	System status or help reply
214	Help message
220	The server is ready
221	The server is ending the conversation
250	The requested action was completed
251	The specified user is not local, but the server will forward the mail message
354	Reply to the DATA command. After getting this, start sending the message body
421	The mail server will be shut down, try again later
450	The mailbox that you are trying to reach is busy, try again later
451	The requested action was not done because some error occurred in the mail server
452	The requested action was not done because the mail server ran out of system storage
500	The last command contained a syntax error or the command line was too long
501	The parameters or arguments in the last command contained a syntax error
502	The last command is not implemented in the mail server
503	The last command was sent out of sequence
504	One of the parameters in the last command is not implemented by the server
550	The mailbox that you are trying to reach cannot be found or you do not have access rights
551	The specified user is not local, so part of message text will contain a forwarding address
552	The mailbox that you are trying to reach has run out of space, try again later
553	The mail address that you specified was not syntactically correct
554	The mail transaction has failed for unknown causes

Sendmail is an MTA distributed as a monolithic binary file.
Previous versions used to run SUID `root`, which caused many security problems; recent versions run SGID `smmsp`, the group that has write access on the mail queue.
Sendmail uses `smrsh`, a restricted shell, to run some external programs.

Configuration files (must not be edited by hand):

`/etc/mail/`	`submit.cf`	Sendmail local mail transfer configuration file
	`sendmail.cf`	Sendmail MTA configuration file

`m4 /etc/mail/submit.mc > /etc/mail/submit.cf`	Generate a `.cf` configuration file from an editable `.mc` text file

Database files (must not be edited by hand):

`/etc/mail/`	`access.db`	Access control file to allow or deny access to systems or users
	`local-host-names.db`	List of domains that must be considered as local accounts
	`virtusertable.db`	Map for local accounts, used to distribute incoming email
	`mailertable.db`	Routing table, used to dispatch emails from remote systems
	`domaintable.db`	Domain table, used for transitions from an old domain to a new one
	`genericstable.db`	Map for local accounts, used to specify a different sender for outgoing mail
	`genericsdomain.db`	Local FQDN

`makemap hash /etc/mail/access.db < /etc/mail/access`	Generate a `.db` database file from an editable text file

Temporary mailqueue files (where *nnn* is the Message ID):

`/var/spool/mqueue/`	`df`*nnn*	Mail body
	`qf`*nnn*	Message envelope with headers and routing information
	`Qf`*nnn*	Message envelope if abandoned
	`hf`*nnn*	Message envelope if held / quarantined by a milter (mail filter)
	`tf`*nnn*	Temporary file
	`lf`*nnn*	Lock file
	`nf`*nnn*	Backup file
	`xf`*nnn*	Transcript of delivery attempts

`newaliases`
`sendmail -bi`
	Update the aliases database. Must be run after any change to `/etc/aliases`

`mailq`
`sendmail -bp`
	Examine the mail queue

`sendmail -bt`	Run Sendmail in test mode
`sendmail -q`	Force a queue run

`hoststat`	Print statistics about remote hosts usage
`purgestat`	Clear statistics about remote host usage
`mailstats`	Print statistics about the mailserver
`praliases`	Display email aliases

Exim is a free MTA, distributed under open source GPL license.

`/etc/exim.conf` `/usr/local/etc/exim/configure` (FreeBSD)	Exim4 configuration file

`exim4 -bp`	Examine the mail queue
`exim4 -M` *messageID*	Attempt delivery of message
`exim4 -Mrm` *messageID*	Remove a message from the mail queue
`exim4 -Mvh` *messageID*	See the headers of a message in the mail queue
`exim4 -Mvb` *messageID*	See the body of a message in the mail queue
`exim4 -Mvc` *messageID*	See a message in the mail queue
`exim4 -qf` *domain*	Force a queue run of all queued messages for a *domain*
`exim4 -Rff` *domain*	Attempt delivery of all queued messages for a *domain*
`exim4 -bV`	Show version and other info

`exinext`	Give the times of the next queue run
`exigrep`	Search through Exim logfiles
`exicyclog`	Rotate Exim logfiles

Postfix is a fast, secure, easy to configure, open source MTA intended as a replacement for Sendmail. It is implemented as a set of small helper daemons, most of which run in a chroot jail with low privileges. The main ones are:

`master`	Postfix master daemon, always running; starts the other daemons when necessary
`nqmgr`	Queue manager for incoming and outgoing mail, always running
`smtpd`	SMTP daemon for incoming mail
`smtp`	SMTP daemon for outgoing mail
`bounce`	Manager of bounce messages
`cleanup`	Daemon that verifies the syntax of outgoing messages before they are handed to the queue manager
`local`	Daemon that handles local mail delivery
`virtual`	Daemon that handles mail delivery to virtual users

`/var/spool/postfix/`	`incoming`	Incoming queue. All new mail entering the Postfix queue is written here by the cleanup daemon. Under normal conditions this queue is nearly empty
	`active`	Active queue. Contains messages ready to be sent. The queue manager places messages here from the incoming queue as soon as they are available
	`deferred`	Deferred queue. A message is placed here when all its deliverable recipients are delivered, and delivery failed for some recipients for a transient reason. The queue manager scans this queue periodically and puts some messages back into the active queue to retry sending
	`bounce`	Message delivery status report about why mail is bounced (non-delivered mail)
	`defer`	Message delivery status report about why mail is delayed (non-delivered mail)
	`trace`	Message delivery status report (delivered mail)

`postfix reload`	Reload configuration
`postconf -e 'mydomain = example.org'`	Edit a setting in the Postfix configuration
`postconf -l`	List supported mailbox lock methods
`postconf -m`	List supported database types
`postconf -v`	Increase logfile verbosity
`postmap` *`dbtype:textfile`*	Manage Postfix lookup tables, creating a hashed map file of database type *dbtype* from *textfile*
`postmap hash:/etc/postfix/transport`	Regenerate the transport database
`postalias`	Convert `/etc/aliases` into the aliases database file `/etc/aliases.db`
`postsuper`	Operate on the mail queue
`postqueue`	Unprivileged mail queue manager

Postfix - configuration

`/etc/postfix/main.cf`	**Postfix main configuration file**
`mydomain = example.org`	This system's domain
`myorigin = $mydomain`	Domain from which all sent mail will appear to originate
`myhostname = foobar.$mydomain`	This system's hostname
`inet_interfaces = all`	Network interface addresses that this system receives mail on. Value can also be `localhost`, `all`, or `loopback-only`
`proxy_interfaces = 1.2.3.4`	Network interface addresses that this system receives mail on by means of a proxy or NAT unit
`mynetworks = 10.3.3.0/24 !10.3.3.66`	Networks the SMTP clients are allowed to connect from

`mydestination = $myhostname, localhost,`
`   $mydomain, example.com,`
`   hash:/etc/postfix/otherdomains`
	Domains for which Postfix will accept received mail.
	Value can also be a lookup database file e.g. a hashed map

`relayhost = 10.6.6.6`	Relay host to which Postfix should send all mail for delivery, instead of consulting DNS MX records
`relay_domains = $mydestination`	Sources and destinations for which mail will be relayed. Can be empty if Postfix is not intended to be a mail relay

`virtual_alias_domains = virtualex.org`
`virtual_alias_maps = /etc/postfix/virtual`
 	
or
 	
`virtual_alias_domains = hash:/etc/postfix/virtual`
	Set up Postfix to handle mail for virtual domains too.
	The `/etc/postfix/virtual` file is a hashed map, each line of the file containing the virtual domain email address and the destination real domain email address:
	`jdoe@virtualex.org    john.doe@example.org`
	`ksmith@virtualex.org  kim.smith`
	`@virtualex.org        root`
	The `@virtualex.org` in the last line is a catch-all specifying that all other email messages to the virtual domain are delivered to the root user on the real domain

`mailbox_command = /usr/bin/procmail`	Use Procmail as MDA
A line beginning with whitespace or tab is a continuation of the previous line. A line beginning with a `#` is a comment. A `#` not placed at the beginning of a line is not a comment delimiter.	

`/etc/postfix/master.cf` **Postfix master daemon configuration file**

```
# service type private unpriv chroot wakeup maxproc command + args
smtp      inet n       -      -      -      -       smtpd
pickup    fifo n       -      -      60     1       pickup
cleanup   unix n       -      -      -      0       cleanup
qmgr      fifo n       -      -      300    1       qmgr
rewrite   unix -       -      -      -      -       trivial-rewrite
bounce    unix -       -      -      -      0       bounce
defer     unix -       -      -      -      0       bounce
flush     unix n       -      -      1000?  0       flush
smtp      unix -       -      -      -      -       smtp
showq     unix n       -      -      -      -       showq
error     unix -       -      -      -      -       error
local     unix -       n      n      -      -       local
virtual   unix -       n      n      -      -       virtual
lmtp      unix -       -      n      -      -       lmtp
```

service	Name of the service
type	Transport mechanism used by the service
private	Whether the service is accessible only by Postfix daemons and not by the whole system. Default is yes
unprivileged	Whether the service is unprivileged i.e. not running as root. Default is yes
chroot	Whether the service is chrooted. Default is yes
wakeup	How often the service needs to be woken up by the master daemon. Default is never
maxproc	Max number of simultaneous processes providing the service. Default is 50
command	Command used to start the service
The – indicates that an option is set to its default value.	

Procmail is a regex-based MDA whose main purpose is to preprocess and sort incoming email messages.
It is able to work both with the standard mbox format and the Maildir format.
To have all email processed by Procmail, create a `~/.forward` file with the following content:
`"|exec /usr/local/bin/procmail || exit 75"`

`/etc/procmailrc`	System-wide recipes
`~/.procmailrc`	User's recipes

`procmail -h`	List all Procmail flags for recipes

`formail`	Utility for email filtering and editing
`lockfile`	Utility for mailbox file locking
`mailstat`	Utility for generation of reports from Procmail logs

`/etc/procmailrc` and `~/.procmailrc` **Procmail recipes**	

`PATH=$HOME/bin:/usr/bin:/bin:/usr/sbin:/sbin`
`MAILDIR=$HOME/Mail`
`DEFAULT=$MAILDIR/Inbox`
`LOGFILE=$HOME/.procmaillog`
	Common parameters, nonspecific to Procmail

`:0h:` or `:0:`
`* ^From: .*(alice|bob)@foobar\.org`
`$DEFAULT`
	Flag: match headers (default) and use file locking (highly recommended when writing to a file or a mailbox in mbox format)
	Condition: match the header specifying the sender address
	Destination: default mailfolder

`:0:`
`* ^From: .*owner@listserv\.com`
`* ^Subject:.*Linux`
`$MAILDIR/Geekstuff1`
	Conditions: match sender address and subject headers
	Destination: specified mailfolder, in mbox format

`:0`
`* ^From: .*owner@listserv\.com`
`* ^Subject:.*Linux`
`$MAILDIR/Geekstuff2/`
	Flag: file locking not necessary because using Maildir format
	Conditions: match sender address and subject headers
	Destination: specified mailfolder, in Maildir format

`# Blacklisted by SpamAssassin`
`:0`
`* ^X-Spam-Status: Yes`
`/dev/null`
	Flag: file locking not necessary because blackholing to `/dev/null`
	Condition: match SpamAssassin's specific header
	Destination: delete the message

`:0B:`
`* hacking`
`$MAILDIR/Geekstuff`
	Flag: match body of message instead of headers

`:0HB:`
`* hacking`
`$MAILDIR/Geekstuff`
	Flag: match either headers or body of message

`:0:`
`* > 256000`
`| /root/myprogram`
	Condition: match messages larger than 256 Kb
	Destination: pipe message through the specified program

`:0fw`
`* ^From: .*@foobar\.org`
`| /root/myprogram`
	Flags: use the pipe as a filter (modifying the message), and have Procmail wait that the filter finished processing the message

`:0c`
`* ^Subject:.*administration`
`! secretary@domain.com`
 	
`:0:`
`$MAILDIR/Forwarded`
	Flag: copy the message and proceed with next recipe
	Destination: forward to specified email address, and (this is ordered by the next recipe) save in the specified mailfolder

Courier is an MTA that provides modules for ESMTP, IMAP, POP3, webmail, and mailing list services in a single framework. To use Courier, it is necessary first to launch the `courier-authlib` service, then launch the desired mail service e.g. `courier-imap` for the IMAP service.

`/usr/lib/courier-imap/etc/`
or
`/etc/courier/`
	`imapd`	Courier IMAP daemon configuration

	`imapd-ssl`	Courier IMAPS daemon configuration
	`pop3d`	Courier POP3 daemon configuration
	`pop3d-ssl`	Courier POP3S daemon configuration

`/usr/lib/courier-imap/share/`	Directory for public and private keys

`mkimapdcert`	Generate a certificate for the IMAPS service
`mkpop3dcert`	Generate a certificate for the POP3 service
`makealiases`	Create system aliases in `/usr/lib/courier/etc/aliases.dat`, which is made by processing a `/usr/lib/courier/etc/aliases/system` text file:

```
root          : postmaster
mailer-daemon : postmaster
MAILER-DAEMON : postmaster
uucp          : postmaster
postmaster    : admin
```

`/usr/lib/courier-imap/etc/pop3d`	**Courier POP configuration file**
`ADDRESS=0`	Address on which to listen. 0 means all addresses
`PORT=127.0.0.1.900,192.168.0.1.900`	Port number on which connections are accepted. In this case, accept connections on port 900 on IP addresses 127.0.0.1 and 192.168.0.1
`POP3AUTH="LOGIN CRAM-MD5 CRAM-SHA1"`	POP authentication advertising SASL (Simple Authentication and Security Layer) capability, with CRAM-MD5 and CRAM-SHA1
`POP3AUTH_TLS="LOGIN PLAIN"`	Also advertise SASL PLAIN if SSL is enabled
`MAXDAEMONS=40`	Maximum number of POP3 servers started
`MAXPERIP=4`	Maximum number of connections to accept from the same IP address
`PIDFILE=/var/run/courier/pop3d.pid`	PID file
`TCPDOPTS="-nodnslookup -noidentlookup"`	Miscellaneous `couriertcpd` options. Should not be changed
`LOGGEROPTS="-name=pop3d"`	Options for `courierlogger`
`POP3_PROXY=0`	Enable or disable proxying
`PROXY_HOSTNAME=myproxy`	Override value from `gethostname()` when checking if a proxy connection is required
`DEFDOMAIN="@example.com"`	Optional default domain. If the username does not contain the first character of `DEFDOMAIN`, then it is appended to the username. If `DEFDOMAIN` and `DOMAINSEP` are both set, then `DEFDOMAIN` is appended only if the username does not contain any character from `DOMAINSEP`
`POP3DSTART=YES`	Flag intended to be read by the system startup script
`MAILDIRPATH=Maildir`	Maildir directory

/usr/lib/courier-imap/etc/imapd	Courier IMAP configuration file
ADDRESS=0	Address on which to listen. 0 means all addresses
PORT=127.0.0.1.900,192.168.0.1.900	Port number on which connections are accepted. In this case, accept connections on port 900 on IP addresses 127.0.0.1 and 192.168.0.1
AUTHSERVICE143=imap	Authenticate using a different service parameter depending on the connection's port. This only works with authentication modules that use the service parameter, such as PAM
MAXDAEMONS=40	Maximum number of IMAP servers started
MAXPERIP=20	Maximum number of connections to accept from the same IP address
PIDFILE=/var/run/courier/imapd.pid	PID file for couriertcpd
TCPDOPTS="-nodnslookup -noidentlookup"	Miscellaneous couriertcpd options. Should not be changed
LOGGEROPTS="-name=imapd"	Options for courierlogger
DEFDOMAIN="@example.com"	Optional default domain. If the username does not contain the first character of DEFDOMAIN, then it is appended to the username. If DEFDOMAIN and DOMAINSEP are both set, then DEFDOMAIN is appended only if the username does not contain any character from DOMAINSEP

IMAP_CAPABILITY="IMAP4rev1 UIDPLUS \
CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT \
THREAD=REFERENCES SORT QUOTA IDLE"
	Specifies what most of the response should be to the CAPABILITY command

IMAP_KEYWORDS=1	Enable or disable custom IMAP keywords. Possible values are: 0 disable keywords 1 enable keywords 2 enable keywords with a slower algorithm
IMAP_ACL=1	Enable or disable IMAP ACL extension
SMAP_CAPABILITY=SMAP1	Enable the experimental Simple Mail Access Protocol extensions
IMAP_PROXY=0	Enable or disable proxying
IMAP_PROXY_FOREIGN=0	Proxying to non-Courier servers. Resends the CAPABILITY command after logging in to remote server. May not work with all IMAP clients
IMAP_IDLE_TIMEOUT=60	How often, in seconds, the server should poll for changes to the folder while in IDLE mode
IMAP_CHECK_ALL_FOLDERS=0	Enable or disable server check for mail in every folder
IMAP_UMASK=022	Set the umask of the server process. This value is passed to the umask command. Mostly useful for shared folders, where file permissions of the messages may be important
IMAP_ULIMITD=131072	Set the upper limit of the size of the data segment of the server process, in Kb. This value is passed to the ulimit -d command. Used as an additional safety check to stop potential DoS attacks that exploit memory leaks to exhaust all the available RAM on the server
IMAP_USELOCKS=1	Enable or disable dot-locking to support concurrent multiple access to the same folder. Strongly recommended when using shared folders
IMAP_SHAREDINDEXFILE=\ /etc/courier/shared/index	Index of all accessible folders. This setting should normally not be changed
IMAP_TRASHFOLDERNAME=Trash	Trash folder
IMAP_EMPTYTRASH=Trash:7,Sent:30	Purge folders i.e. delete all messages from the specified folders after the specified number of days
IMAP_MOVE_EXPUNGE_TO_TRASH=0	Enable or disable moving expunged messages to the trash folder (instead of directly deleting them)
HEADERFROM=X-IMAP-Sender	Save the return address ($SENDER) in the X-IMAP-Sender mail header. This header is added to the sent message, but not in the copy of the message saved in the folder
MAILDIRPATH=Maildir	Mail directory

Dovecot is an open source, security-hardened, fast, and efficient IMAP and POP3 server.
It implements its own high-performance dbox mailbox format. By default, it uses PAM authentication.
The script `mkcert.sh` can be used to create self-signed SSL certificates.

`/etc/dovecot.conf`	**Dovecot configuration file**
`base_dir = /var/run/dovecot/`	Base directory where to store runtime data
`protocols = imaps pop3s`	Protocols to serve. If Dovecot should use `dovecot-auth`, this can be set to `none`
`listen = *, [::]`	Network interfaces on which to accept connections. In this case, listen to all IPv4 and IPv6 interfaces
`disable_plaintext_auth = yes`	If yes, disable LOGIN command and all other plaintext authentications unless SSL/TLS is used (LOGINDISABLED capability)
`shutdown_clients = yes`	If yes, kill all IMAP and POP3 processes when Dovecot master process shuts down; if no, Dovecot can be upgraded without forcing existing client connections to close
`log_path = /dev/stderr`	Log file to use for error messages, instead of sending them to syslog. In this case, log to stderr
`info_log_path = /dev/stderr`	Log file to use for informational and debug messages. Default value is the same as `log_path`
`syslog_facility = mail`	Syslog facility to use, if logging to syslog
`login_dir = /var/run/dovecot/login`	Directory where the authentication process places authentication UNIX sockets. The login process needs to be able to connect to these sockets
`login_chroot = yes`	Chroot login process to the `login_dir`
`login_user = dovecot`	User for the login process and for access control in the authentication process. This is not the user that will access mail messages
`login_process_size = 64`	Maximum login process size, in Mb
`login_process_per_connection = yes`	If yes, each login is processed in its own process (more secure); if no, each login process processes multiple connections (faster)
`login_processes_count = 3`	Number of login processes to keep for listening for new connections
`login_max_processes_count = 128`	Maximum number of login processes to create
`login_max_connections = 256`	Maximum number of connections allowed per each login process. This setting is used only if `login_process_per_connection = no`; once the limit is reached, the process notifies master so that it can create a new login process
`login_greeting = Dovecot ready.`	Greeting message for clients

`login_trusted_networks = \`
`10.7.7.0/24 10.8.8.0/24`
	Trusted network ranges (usually IMAP proxy servers).
	Connections from these IP addresses are allowed to override their IP addresses and ports, for logging and authentication checks.
	`disable_plaintext_auth` is also ignored for these networks

`mbox_read_locks = fcntl`
`mbox_write_locks = dotlock fcntl`
	Locking methods to use for locking mailboxes in mbox format.
	Possible values are:
	`dotlock` Create *`mailbox`*`.lock` file; oldest and NSF-safe method
	`dotlock_try` Same as `dotlock`, but skip if failing
	`fcntl` Recommended; works with NFS too if `lockd` is used
	`flock` May not exist in all systems; doesn't work with NFS
	`lockf` May not exist in all systems; doesn't work with NFS

`maildir_stat_dirs = no`	Option for mailboxes in Maildir format. If no (default), the LIST command returns all entries in the mail directory beginning with a dot; if yes, returns only entries which are directories

`dbox_rotate_size = 2048`
`dbox_rotate_min_size = 16`
	Maximum and minimum file size, in Kb, of a mailbox in dbox format until it is rotated

`!include /etc/dovecot/conf.d/*.conf`	Include configuration file
`!include_try /etc/dovecot/extra.conf`	Include optional configuration file, and do not report an error if file is not found

`/etc/dovecot.conf`	**Dovecot configuration file**

`mail_location = \`
`mbox:~/mail:INBOX=/var/spool/mail/%u`
or
`mail_location = maildir:~/Maildir`
	Mailbox location, in mbox or Maildir format. Variables:
	`%u` username
	`%n` user part in *user@domain*, same as `%u` if there is no domain
	`%d` domain part in *user@domain*, empty if there is no domain
	`%h` home directory

`namespace shared {`	Definition of a shared namespace, for accessing other users' mailboxes that have been shared. Private namespaces are for users' personal emails. Public namespaces are for shared mailboxes managed by root user
`   separator = /`	Hierarchy separator to use. It should be the same for all namespaces, and depends on the underlying mail storage format
`   prefix = shared/%%u/`	Prefix required to access this namespace; must be different for each. In this case, mailboxes are visible under `shared/`*`user@domain`*`/`; the variables `%%n`, `%%d`, and `%%u` are expanded to the destination user

`   location = maildir:%%h/Maildir:\`
`      INDEX=~/Maildir/shared/%%u`
	Mailbox location for other users' mailboxes; it is in the same format as `mail_location` which is also the default for it.
	`%`*`variable`* and `~/` expand to the logged in user's data;
	`%%`*`variable`* expands to the destination user's data

`   inbox = no`	Define whether this namespace contains the INBOX. Note that there can be only one INBOX across all namespaces
`   hidden = no`	Define whether the namespace is hidden i.e. not advertised to clients via NAMESPACE extension
`   subscriptions = no`	Namespace handles its own subscriptions; if set to no, the parent namespace handles them and Dovecot uses the default namespace for saving subscriptions. If `prefix` is empty, this should be set to yes
`   list = children`	Show the mailboxes under this namespace with LIST command, making the namespace visible for clients that do not support the NAMESPACE extension. In this case, lists child mailboxes but hide the namespace prefix; list the namespace only if there are visible shared mailboxes
`}`	

`mail_uid = 666`
`mail_gid = 666`
	UID and GID used to access mail messages

`mail_privileged_group = mail`	Group to enable temporarily for privileged operations. Currently this is used only with INBOX when its initial creation or a dotlocking fails
`mail_access_groups = tmpmail`	Supplementary groups to with grant access for mail processes. Used typically to set up access to shared mailboxes
`lock_method = fcntl`	Locking method for index files. Can be `fcntl`, `flock`, or `dotlock`

`first_valid_uid = 500`
`last_valid_uid = 0`
	Valid UID range for users; default is 500 and above. This makes sure that users cannot login as daemons or other system users.
	Denying root login is hardcoded to Dovecot and cannot be bypassed

`first_valid_gid = 1`
`last_valid_gid = 0`
	Valid GID range for users; default is non-root.
	Users with invalid primary GID are not allowed to login

`max_mail_processes = 512`	Maximum number of running mail processes. When this limit is reached, new users are not allowed to login
`mail_process_size = 256`	Maximum mail process size, in Mb
`valid_chroot_dirs =`	List of directories under which chrooting is allowed for mail processes
`mail_chroot =`	Default chroot directory for mail processes. Usually not needed as Dovecot does not allow users to access files outside their mail directory
`mailbox_idle_check_interval = 30`	Minimum time, in seconds, to wait between mailbox checks. When the IDLE command is running, mailbox is checked periodically for new mails or other changes

`/etc/dovecot.conf`	**Dovecot configuration file**
`protocol pop3 {`	Block with options for the POP3 protocol
`listen = *:110`	Network interfaces on which to accept POP3 connections
`login_executable = /usr/libexec/dovecot/pop3-login`	Location of the POP3 login executable
`mail_executable = /usr/libexec/dovecot/pop3`	Location of the POP3 mail executable
`pop3_no_flag_updates = no`	If set to no, do not try to set mail messages non-recent or seen with POP3 sessions, to reduce disk I/O. With Maildir format do not move files from `new/` to `cur/`; with mbox format do not write `Status-` headers
`pop3_lock_session = no`	Defines whether to keep the mailbox locked for the whole POP3 session
`pop3_uidl_format = %08Xu%08Xv`	POP3 UIDL (Unique Mail Identifier) format to use
`}`	
`protocol imap {`	Block with options for the IMAP protocol

`listen = *:143`
`ssl_listen = *:993`
	Network interfaces on which to accept IMAP and IMAPS connections

`login_executable = /usr/libexec/dovecot/imap-login`	Location of the IMAP login executable
`mail_executable = /usr/libexec/dovecot/imap`	Location of the IMAP mail executable
`mail_max_userip_connections = 10`	Maximum number of IMAP connections allowed for a user from each IP address
`imap_idle_notify_interval = 120`	Waiting time, in seconds, between "OK Still here" notifications when client is IDLE
`}`	
`ssl = yes`	SSL/TLS support. Possible values are `yes`, `no`, `required`
`ssl_cert_file = /etc/ssl/certs/dovecot-cert.pem`	Location of the SSL certificate
`ssl_key_file = /etc/ssl/private/dovecot-key.pem`	Location of private key
`ssl_key_password = p4ssw0rd`	Password of private key, if it is password-protected. Since `/etc/dovecot.conf` is usually world-readable, it is better to place this setting into a root-owned 0600 file instead and include it via the setting `!include_try /etc/dovecot/dovecot-passwd.conf`. Alternatively, Dovecot can be started with `dovecot -p p4ssw0rd`
`ssl_ca_file = /etc/dovecot/cafile.pem`	List of trusted SSL certificate authorities. This file contains CA certificates followed by CRLs
`ssl_verify_client_cert = yes`	Request client to send a certificate
`ssl_cipher_list = ALL:!LOW:!SSLv2`	List of SSL ciphers to use
`verbose_ssl = yes`	Show protocol level SSL errors

/etc/dovecot.conf	Dovecot configuration file
auth_executable = /usr/libexec/dovecot/dovecot-auth	Location of the authentication executable
auth_process_size = 256	Max authentication process size, in Mb
auth_username_chars = abcde … VWXYZ01234567890.-_@	List of allowed characters in the username. If the username entered by the user contains a character not listed in here, the login automatically fails. This is to prevent a user exploiting any potential quote-escaping vulnerabilities with SQL/LDAP databases
auth_realms =	List of realms for SASL authentication mechanisms that need them. If empty, multiple realms are not supported
auth_default_realm = example.org	Default realm/domain to use if none was specified
auth_anonymous_username = anonymous	Username to assign to users logging in with ANONYMOUS SASL mechanism
auth_verbose = no	Defines whether to log unsuccessful authentication attempts and the reasons why they failed
auth_debug = no	Define whether to enable more verbose logging (e.g. SQL queries) for debugging purposes
auth_failure_delay = 2	Delay before replying to failed authentications, in seconds
auth default {	
mechanisms = plain login cram-md5	Accepted authentication mechanisms

passdb passwd-file {
      args = /etc/dovecot.deny
      deny = yes
   }
	Deny login to the users listed in /etc/dovecot.deny
	(this file contains one user per line)

passdb pam {
      args = cache_key=%u%r dovecot
   }
	PAM authentication block.
	Enables authentication matching (username and remote IP address) for PAM

passdb passwd {
      blocking = yes
      args =
   }
	System users e.g. NSS or /etc/passwd

passdb shadow {
      blocking = yes
      args =
   }
	Shadow passwords for system users, e.g. NSS or /etc/passwd

passdb bsdauth {
      cache_key = %u
      args =
   }
	PAM-like authentication for OpenBSD

passdb sql {
      args = /etc/dovecot/dovecot-sql.conf
   }
	SQL database

passdb ldap {
      args = /etc/dovecot/dovecot-ldap.conf
   }
	LDAP database

socket listen {
      master {
         path = /var/run/dovecot/auth-master
         mode = 0600
         user =
         group =
      }
      client {
         path = /var/run/dovecot/auth-client
         mode = 0660
      }
   }
	Export the authentication interface to other programs.
	Master socket provides access to userdb information, and is typically used to give Dovecot's local delivery agent access to userdb so it can find mailbox locations. The default user/group is the one who started dovecot-auth (i.e. root).
	The client socket is generally safe to export to everyone. Typical use is to export it to the SMTP server so it can do SMTP AUTH lookups using it

}	

FTP (File Transfer Protocol) is a client-server unencrypted protocol for file transfer. Secure alternatives are **FTPS** (FTP secured with SSL/TLS) and **SFTP (SSH File Transfer Protocol)**.

FTP can operate either in active or in passive mode:

Active mode (default)
1. Client connects to FTP server on port 21 (control channel) and sends second unprivileged port number
2. Server acknowledges
3. Server connects from port 20 (data channel) to client's second unprivileged port number
4. Client acknowledges

Passive mode (more protocol-compliant, because it is the client, not the server, that initiates the second connection)
1. Client connects to FTP server on port 21 and requests passive mode via the PASV command
2. Server acknowledges and sends unprivileged port number via the PORT command
3. Client connects to server's unprivileged port number
4. Server acknowledges

`ftp`	Standard FTP client
`lftp`	Sophisticated FTP client with support for HTTP and BitTorrent

Very Secure FTP is a hardened and high-performance FTP implementation. The `vsftpd` daemon operates with multiple processes that run as a non-privileged user in a chrooted jail.

`/etc/vsftpd/vsftpd.conf`	**Very Secure FTP server configuration file**
`listen=NO`	Run `vsftpd` in standalone mode (i.e. not via inetd)?
`local_enable=YES`	Allow local system users (i.e. in `/etc/passwd`) to log in?
`chroot_local_user=YES`	Chroot local users in their home directory?
`write_enable=YES`	Allow FTP commands that write on the filesystem (i.e. STOR, DELE, RNFR, RNTO, MKD, RMD, APPE, and SITE)?
`anonymous_enable=YES`	Allow anonymous logins? If yes, `anonymous` and `ftp` are accepted as logins
`anon_root=/var/ftp/pub`	Directory to go after anonymous login
`anon_upload_enable=YES`	Allow anonymous uploads?
`chown_uploads=YES`	Change ownership of anonymously uploaded files?
`chown_username=ftp`	User to whom set ownership of anonymously uploaded files
`anon_world_readable_only=NO`	Allow anonymous users to only download world-readable files?
`ssl_enable=YES`	Enable SSL?
`force_local_data_ssl=NO`	Encrypt local data?
`force_local_logins_ssl=YES`	Force encrypted authentication?
`allow_anon_ssl=YES`	Allow anonymous users to use SSL?

`ssl_tlsv1=YES`
`ssl_tlsv2=NO`
`ssl_tlsv3=NO`
	Allowed SSL/TLS versions

`rsa_cert_file=/etc/pki/tls/certs/vsftpd.pem`	Location of certificate file
`rsa_private_key_file=/etc/pki/tls/certs/vsftpd.pem`	Location of private key file

Pure-FTP is a free and easy-to-use FTP server.

`pure-ftpd`	Pure-FTP daemon
`pure-ftpwho`	Show clients connected to the Pure-FTP server
`pure-mrtginfo`	Show connections to the Pure-FTP server as a MRTG graph
`pure-statsdecode`	Show Pure-FTP log data
`pure-pw`	Manage Pure-FTP virtual accounts
`pure-pwconvert`	Convert the system user database to a Pure-FTP virtual accounts database
`pure-quotacheck`	Manage Pure-FTP quota database
`pure-uploadscript`	Run a command on the Pure-FTP server to process an uploaded file

CUPS

In Linux, printers are managed by `cupsd`, the **CUPS (Common Unix Printing System)** daemon.
Printers are administered via a web interface on the URL http://localhost:631.

/etc/cups/cupsd.conf	CUPS configuration file
/etc/cups/printers.conf	Database of available local CUPS printers
/etc/printcap	Database of printer capabilities, for old printing applications
/var/spool/cups/	Printer spooler for data awaiting to be printed
/var/log/cups/error_log	CUPS error log
/var/log/cups/page_log	Information about printed pages

/etc/init.d/cupsys start	Start the CUPS service

gnome-cups-manager	Run the CUPS Manager graphical application
cupsenable *printer0*	Enable a CUPS printer
cupsdisable *printer0*	Disable a CUPS printer
cupsaccept *printer0*	Accept a job sent on a printer queue
cupsreject -r "*Message*" *printer0*	Reject a job sent on a printer queue, with an informational message
cupstestppd *LEXC510.ppd*	Test the conformance of a PPD file to the format specification
cupsaddsmb *printer0*	Export a printer to Samba (for use with MS Windows clients)

cups-config --cflags	Show the necessary compiler options
cups-config --datadir	Show the default CUPS data directory
cups-config --ldflags	Show the necessary linker options
cups-config --libs	Show the necessary libraries to link to
cups-config --serverbin	Show the default CUPS binaries directory that stores filters and backends
cups-config --serverroot	Show the default CUPS configuration file directory

lpstat	Show CUPS status information
lpadmin	Administer CUPS printers
lpadmin -p *printer0* -P *LEXC750.ppd*	Specify a PPD (Adobe PostScript Printer Description) file to associate to a printer
lp -d *printer0 file*	Print a file on the specified printer
lpq	View the default print queue
lpq -P *printer0*	View a specific print queue
lpq *user*	View the print queue of a specific user
lprm -P *printer0 jobnumber*	Delete a specific job from a printer queue
lprm -P *printer0 user*	Delete all jobs from a specific user from a printer queue
lprm -P *printer0* -	Delete all jobs from a printer queue
lpc	Manage print queues

a2ps *file.txt*	Convert a text file to PostScript
ps2pdf *file.ps*	Convert a file from PostScript to PDF
mpage *file.ps*	Print a PostScript document on multiple pages per sheet on a PostScript printer
gv *file.ps*	View a PostScript document (the `gv` software is a derivation of GhostView)

IPv4 addressing

		Address range	Prefix	Number of addresses	Reference
Classful	Class A (Unicast)	0.0.0.0 – 127.255.255.255 first octet: 0XXX XXXX	/8	128 networks × 16,777,216 addresses	RFC 791
	Class B (Unicast)	128.0.0.0 – 191.255.255.255 first octet: 10XX XXXX	/16	16,384 networks × 65,536 addresses	RFC 791
	Class C (Unicast)	192.0.0.0 – 223.255.255.255 first octet: 110X XXXX	/24	2,097,152 networks × 256 addresses	RFC 791
	Class D (Multicast)	224.0.0.0 – 239.255.255.255 first octet: 1110 XXXX	/4	268,435,456	RFC 3171
	Class E (Experimental)	240.0.0.0 – 255.255.255.255 first octet: 1111 XXXX	/4	268,435,456	RFC 1166
Private	Private Class A	10.0.0.0 – 10.255.255.255	10.0.0.0/8	16,777,216	RFC 1918
	Private Class B	172.16.0.0 – 172.31.255.255	172.16.0.0/12	1,048,576	RFC 1918
	Private Class C	192.168.0.0 – 192.168.255.255	192.168.0.0/16	65,536	RFC 1918
Reserved	Source	0.0.0.0 – 0.255.255.255	0.0.0.0/8	16,777,216	RFC 1700
	Loopback	127.0.0.0 – 127.255.255.255	127.0.0.0/8	16,777,216	RFC 1700
	Autoconf	169.254.0.0 – 169.254.255.255	169.254.0.0/16	65,536	RFC 3330
	TEST-NET	192.0.2.0 – 192.0.2.255	192.0.2.0/24	256	RFC 3330
	6to4 relay anycast	192.88.99.0 – 192.88.99.255	192.88.99.0/24	256	RFC 3068
	Device benchmarks	198.18.0.0 – 198.19.255.255	198.18.0.0/15	131,072	RFC 2544

An IPv4 address is 32-bit long, and is represented divided in four octets (dotted-quad notation), e.g. 193.22.33.44.

There are approximately 4×10^9 total possible IPv4 addresses.

IPv4 classful addressing is obsolete and has been replaced by CIDR (Classless Inter-Domain Routing).

IPv6 addressing

Unicast	64-bit network prefix (>= 48-bit routing prefix + <= 16-bit subnet id) + 64-bit interface identifier A 48-bit MAC address is transformed into a 64-bit EUI-64 by inserting ff:fe in the middle. A EUI-64 is then transformed into an IPv6 interface identifier by inverting the 7th most significant bit.
Link-local	fe80:0000:0000:0000 + 64-bit interface identifier
Multicast	ff + 4-bit flag + 4-bit scope field + 112-bit group ID

An IPv6 address is 128-bit long, and is represented divided in eight 16-bit groups (4 hex digits).
Leading zeros in each group can be deleted. A single chunk of one or more adjacent 0000 groups can be deleted.
e.g. 2130:0000:0000:0000:0007:0040:15bc:235f which can also be written as 2130::7:40:15bc:235f.

There are approximately 3×10^{38} total possible IPv6 addresses.

The IANA (Internet Assigned Numbers Authority) manages the allocation of IPv4 and IPv6 addresses, assigning large blocks to RIRs (Regional Internet Registries) which in turn allocate addresses to ISPs (Internet Service Providers) and other local registries. These address blocks can be searched via a WHOIS query to the appropriate RIR, which is:

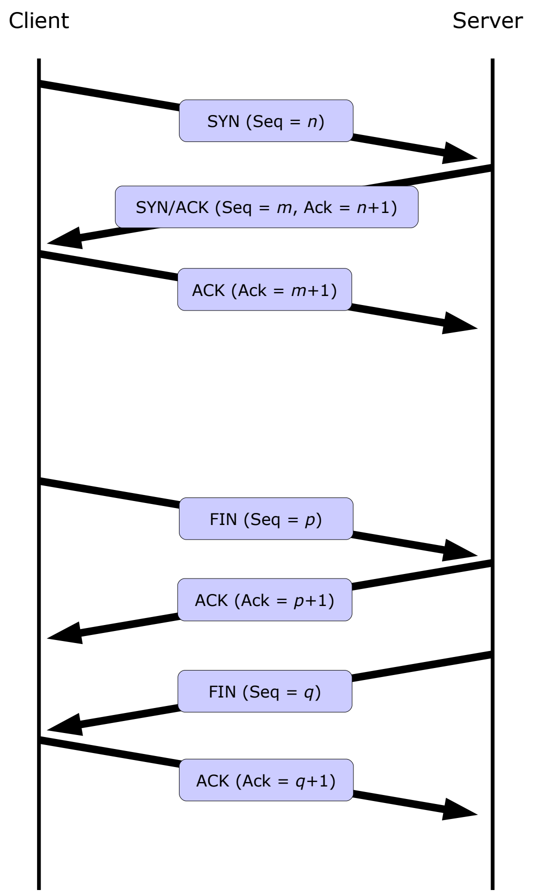
AFRINIC	for Africa
ARIN	for US, Canada, and Antarctica
APNIC	for Asia and Oceania
LACNIC	for Latin America
RIPE NCC	for Europe, Middle East, and Russia

VLSM chart - Last octet subnetting (CIDR notation)						
Prefix: /24 Netmask: .0 00000000 1 subnet 254 hosts each 254 total hosts	Prefix: /25 Netmask: .128 10000000 2 subnets 126 hosts each 252 total hosts	Prefix: /26 Netmask: .192 11000000 4 subnets 62 hosts each 248 total hosts	Prefix: /27 Netmask: .224 11100000 8 subnets 30 hosts each 240 total hosts	Prefix: /28 Netmask: .240 11110000 16 subnets 14 hosts each 224 total hosts	Prefix: /29 Netmask: .248 11111000 32 subnets 6 hosts each 192 total hosts	Prefix: /30 Netmask: .252 11111100 64 subnets 2 hosts each 128 total hosts
0	.0	.0	.0	.0	.0	.0
						.4
					.8	.8
						.12
				.16	.16	.16
						.20
					.24	.24
						.28
			.32	.32	.32	.32
						.36
					.40	.40
						.44
				.48	.48	.48
						.52
					.56	.56
						.60
		.64	.64	.64	.64	.64
						.68
					.72	.72
						.76
				.80	.80	.80
						.84
					.88	.88
						.92
			.96	.96	.96	.96
						.100
					.104	.104
						.108
				.112	.112	.112
						.116
					.120	.120
						.124
	0.13	.128	.128	.128	.128	.128
						.132
					.136	.136
						.140
				.144	.144	.144
						.148
					.152	.152
						.156
			.160	.160	.160	.160
						.164
					.168	.168
						.172
				.176	.176	.176
						.180
					.184	.184
						.188
		.192	.192	.192	.192	.192
						.196
					.200	.200
						.204
				.208	.208	.208
						.212
					.216	.216
						.220
			.224	.224	.224	.224
						.228
					.232	.232
						.236
				.240	.240	.240
						.244
					.248	.248
						.252

Each block of a column identifies a subnet, whose range of valid hosts addresses is [network address +1 — broadcast address -1] inclusive.
The network address of the subnet is the number shown inside a block.
The broadcast address of the subnet is the network address of the block underneath -1 or, for the bottom block, .255.

ISO/OSI and TCP/IP protocol stack models

Layer	ISO/OSI	TCP/IP	Standards	Data transmission unit
7	Application	Application	HTTP, SMTP, POP, etc.	Message
6	Presentation			
5	Session			
4	Transport	Transport	TCP, UDP	Segment (TCP), datagram (UDP)
3	Network	Internet	IPv4, IPv6, ICMP, etc.	Packet
2	Data Link	Network Access	Ethernet, Wi-Fi, etc.	Frame
1	Physical			Bit



TCP connection establishment
(three-way handshake)

TCP connection termination

Most common wireless standards				
IEEE standard	**Known as**	**Frequency (GHz)**	**Max bandwidth (Mbps)**	**Max range (m)**
802.11	Wi-Fi	2.4	2	100
802.11a		5	54	100
802.11b		2.4	11	150
802.11g		2.4	54	150
802.11n		2.4, 5	54, 600	250
802.15.1	Bluetooth	2.4	50	10 - 250
802.16	WiMax	2 - 11	1000	10000

Wireless transmission techniques	
Direct-Sequence Spread Spectrum (DSSS)	Spread-spectrum modulation technique that modulates the original data with a pseudorandom bit sequence (spreading sequence). It is used to reduce signal interference.
Frequency-Hopping Spread Spectrum (FHSS)	Radio transmission technique consisting in rapidly changing the carrier frequency amongst different frequencies, in sync between transmitter and receiver. It is used to reduce signal interference, avoid eavesdropping, and allow code-division multiple access (CDMA) communications.
Orthogonal Frequency-Division Multiplexing (OFDM)	Digital multi-carrier modulation technique which uses multiple orthogonal subcarrier signal frequencies to transmit data, mapping information on the changes in the carrier phase, frequency, or amplitude. It is used to cope with severe channel conditions.
Multiple-Input Multiple-Output Orthogonal Frequency-Division Multiplexing (MIMO-OFDM)	Access mode for 4G and 5G broadband wireless communications. It is used to increase spectral efficiency and reduce signal interference.

Wireless encryption algorithms	
WEP (Wired Equivalent Privacy) IEEE 802.11	WEP uses a pre-shared key with a length of 40, 104, or 232 bits, with a random 24-bit **IV (Initialization Vector)** added to the key. A CRC-32 checksum is computed on the data and added to it as **ICV (Integrity Check Value)**. WEP key and IV are fed to the RC4 stream cipher to generate a key stream, which is XORed with the data and ICV to obtain the encrypted data. WEP is insecure because of the short length of the IV, which leads to IV reuse; furthermore, the WEP standard does not even require a different IV for each packet. Key reuse in a stream cipher is bad practice and leads to weak encryption. A weak IV may even allow to deduce the WEP pre-shared key. In case of an IV collision, it is possible to reconstruct the RC4 key stream from the IV and the packet's decrypted payload. Therefore, WEP does not provide cryptographic integrity protection of a packet, and is now obsolete. Some wireless APs use **LEAP (Lightweight Extensible Authentication Protocol)**, a Cisco proprietary version of the EAP authentication method for WLANs. LEAP can use either dynamic WEP keys (keys that change very often to minimize cracking exposure) or TKIP. LEAP uses either the MS-CHAP or the EAP-FAST authentication protocol. However, WEP with LEAP is still considered vulnerable.
WPA (Wi-Fi Protected Access) draft IEEE 802.11i	In WPA, the **TKIP (Temporal Key Integrity Protocol)** feeds a 128-bit temporal key and a 64-bit **MIC (Message Integrity Check)** to the RC4 stream cipher to obtain the encrypted data. It uses the CRC-32 checksum algorithm strengthened by the use of Michael MIC codes. IV size is 48 bits. TKIP adds a rekeying mechanism to provide fresh encryption and integrity keys, changing temporal keys every 10000 packets in sync between Access Point and client.
WPA2 (Wi-Fi Protected Access II) IEEE 802.11i	WPA2 is encrypted using **CCMP (Counter Mode CBC-MAC Protocol)**, which utilizes AES encryption. IV size is 48 bits. **WPA2-Personal** uses a **PSK (Pre-Shared Key)**. The Access Point encrypts the data using a 128-bit key derived from a passphrase with length from 8 to 63 characters. Encryption keys are unique for each client, and change frequently. **WPA2-Enterprise** uses centralized client authentication via **802.1X**, either **EAP (Extensible Authentication Protocol)** or **RADIUS (Remote Authentication Dial-In User Service)**. A TLS-encapsulated secured version of EAP, called **PEAP (Protected Extensible Authentication Protocol)**, is also available. After PSK or 802.1X authentication, a shared secret key called **PMK (Pairwise Master Key)** is generated, and is validated through a four-way handshake between wireless client and Access Point: 1. AP sends a nonce to the client, which uses it to build the **PTK (Pairwise Transient Key)** 2. The client sends a nonce and a MIC to the AP 3. The AP builds and sends the **GTK (Group Temporal Key)** with another MIC to the client 4. The client acknowledges reception to the AP
Wireless encryption can be used in conjunction with other security measures such as SSID cloaking (security by obscurity) and MAC address filtering (whitelisting), which however are not very effective.	

Most common well-known ports	
Port number	**Service**
13 TCP	Daytime Protocol
20 TCP	FTP (data)
21 TCP	FTP (control)
22 TCP	SSH
23 TCP	Telnet
25 TCP	SMTP
53 TCP/UDP	DNS
67 UDP	BOOTP/DHCP (server)
68 UDP	BOOTP/DHCP (client)
69 TCP	TFTP
80 TCP	HTTP
88 TCP	Kerberos
110 TCP	POP3
119 TCP	NNTP
123 UDP	NTP
135 TCP/UDP	Microsoft RPC
137 TCP/UDP	Microsoft NetBIOS Name Service / WINS
138 TCP/UDP	Microsoft NetBIOS Datagram Service
139 TCP/UDP	Microsoft NetBIOS Session Service
143 TCP	IMAP
161 UDP	SNMP
162 TCP/UDP	SNMP Trap
389 TCP/UDP	LDAP
443 TCP	HTTPS (HTTP over SSL/TLS)
445 TCP/UDP	Microsoft SMB
465 TCP	SMTP over SSL
500 UDP	IPSec ISAKMP / IKE
514 UDP	Syslog
515 TCP/UDP	Line Printer Daemon
901 TCP	Samba SWAT
993 TCP	IMAPS (IMAP over SSL)
995 TCP	POP3S (POP3 over SSL)
4500 UDP	IPSec NAT Traversal

1-1023: privileged ports, used server-side
1024-65535: unprivileged ports, used client-side

The file `/etc/services` lists all well-known ports.

Network configuration - commands

`ip a`
`ip addr`
`ip addr show`
`ifconfig -a`
	Display configuration of all network interfaces

`ip link show eth0`
`ifconfig eth0`
	Display configuration of `eth0`

`ip addr add dev eth0 10.1.1.3/24`
`ifconfig eth0 10.1.1.3 netmask 255.255.255.0 broadcast 10.1.1.255`
	Configure IP address of `eth0`

`ifconfig eth0 hw ether 45:67:89:ab:cd:ef`	Configure MAC address of `eth0`

`ip link set eth0 up`
`ifconfig eth0 up`
`ifup eth0`
	Activate `eth0`

`ip link set eth0 down`
`ifconfig eth0 down`
`ifdown eth0`
	Shut down `eth0`

`dhclient eth0`
`pump -i eth0`
`dhcpcd eth0` (SUSE)
	Request an IP address via DHCP

`ip neigh`
`arp -a`
	Show the ARP cache table (containing mappings of MAC to IP addresses)

`ip neigh show 10.1.1.4`
`arp 10.1.1.4`
	Show the ARP cache entry for a host

`ip neigh add 10.1.1.5 lladdr 01:23:45:67:89:ab dev eth0`
`arp -s 10.1.1.5 01:23:45:67:89:ab`
	Add a new ARP entry for a host

`ip neigh del 10.1.1.5 dev eth0`
`arp -d 10.1.1.5`
	Delete an ARP entry

`ip neigh flush all`	Delete the ARP table for all interfaces

`hostname`	Get the hostname
`hostname -f`	Get the FQDN (Fully Qualified Domain Name)
`hostname mybox` `hostnamectl set-hostname --static "mybox"` (Red Hat)	Set the hostname
`hostnamectl` (Red Hat)	Get the hostname, OS, and other information

`/etc/init.d/networking restart` (Debian) `/etc/init.d/network restart` (Red Hat)	Restart network services

`ethtool` *`option device`*	Query or control network driver and hardware settings
`ethtool eth0`	View hardware settings of `eth0`

`rdisc`	Network router discovery daemon. Client for IRDP (ICMP Router Discover Protocol). Runs at bootup to populate the network routing tables with default routes

`xinetd`	Extended Internet services daemon. Many network services are run by it rather than standalone; `xinetd` operates as a super server, listening on all service ports defined in its configuration, and upon receiving a connection request it starts the appropriate service

Network configuration - files

`/etc/hosts`

Mappings between IP addresses and hostnames, for name resolution

```
127.0.0.1  localhost.localdomain  localhost
10.2.3.4   myhost.domain.org      myhost
```

`/etc/nsswitch.conf`

Sources that must be used by various system library lookup functions

```
passwd:  files nisplus nis
shadow:  files nisplus nis
group:   files nisplus nis
hosts:   files dns nisplus nis
```

`/etc/host.conf`

Sources for name resolution, for systems before `glibc2`.
Obsolete, superseded by `/etc/nsswitch.conf`

```
order hosts,bind
multi on
```

`/etc/resolv.conf`

Domain names that must be appended to bare hostnames, and DNS servers that will be used for name resolution

```
search domain1.org domain2.org
nameserver  192.168.3.3
nameserver  192.168.4.4
```

`/etc/networks`

Mappings between network addresses and names

```
loopback  127.0.0.0
mylan     10.2.3.0
```

`/etc/services`

List of service TCP/UDP port numbers

`/etc/protocols`

List of available protocols

`/sys/class/net`

List of all network interfaces in the system

Network configuration - distro-specific files

Red Hat

`/etc/sysconfig/network`

Network configuration file

```
ADDRESS=10.2.3.4
NETMASK=255.255.255.0
GATEWAY=10.2.3.254
HOSTNAME=mylinuxbox.example.org
NETWORKING=yes
```

`/etc/sysconfig/network-scripts/ifcfg-eth0`

Configuration file for `eth0`.
This file is read by the `ifup` and `ifdown` scripts

```
DEVICE=eth0
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FF
BOOTPROTO=none
ONBOOT=yes
NM_CONTROLLED=no
IPADDR=10.2.3.4
NETMASK=255.255.255.0
GATEWAY=10.2.3.254
DNS1=8.8.8.8
DNS2=4.4.4.4
USERCTL=no
```

`/etc/sysconfig/network-scripts/ifcfg-eth0:0`
`/etc/sysconfig/network-scripts/ifcfg-eth0:1`
`/etc/sysconfig/network-scripts/ifcfg-eth0:2`

Multiple configuration files for a single `eth0` interface, which allows binding multiple IP addresses to a single NIC

`/etc/sysconfig/network-scripts/route-eth0`

Static route configuration for `eth0`

```
default 10.2.3.4 dev eth0
10.7.8.0/24 via 10.2.3.254 dev eth0
10.7.9.0/24 via 10.2.3.254 dev eth0
```

`/etc/ethertypes`

Ethernet frame types.
Lists various Ethernet protocol types used on Ethernet networks

Debian

`/etc/network/interfaces`

List and configuration of all network interfaces

```
allow-hotplug eth0
iface eth0 inet static
   address 10.2.3.4
   netmask 255.255.255.0
   gateway 10.2.3.254
   dns-domain example.com
   dns-nameservers 8.8.8.8 4.4.4.4
```

`/etc/hostname`

Hostname of the local machine

`/etc/ethers`

ARP mappings

In RHEL7 and later the network configuration is managed by the NetworkManager daemon.
A **connection** is a network configuration that applies to a **device** (aka network interface). A device can be included in multiple connections, but only one of them may be active at a time.
The configuration for *connection* is stored in the file `/etc/sysconfig/network-scripts/ifcfg-`*connection*. Although it is possible to set up networking by editing these configuration files, it is much easier to use the command `nmcli`.

`nmcli device status`	Show all network devices
`nmcli device disconnect` *iface*	Disconnects the device *iface*. This command should be used instead of `nmcli connection down` *connection* because if *connection* is set to autoconnect, Network Manager will bring the connection (and the device) up again short time later
`nmcli connection show`	Show all connections. Connections with an empty device entry are inactive
`nmcli connection show --active`	Show active connections
`nmcli connection show` *connection*	Show the configuration of *connection*

`nmcli connection add con-name` *connection* `\`
`type ethernet ifname` *iface* `ipv4.method manual \`
`ipv4.addresses 10.0.0.13/24 ipv4.gateway 10.0.0.254`
	Configure a new *connection* that uses the Ethernet interface *iface* and assigns it an IPv4 address and gateway

`nmcli connection modify` *connection options*	Modify the configuration of *connection*
`nmcli connection up` *connection*	Brings up a *connection*
`nmcli connection reload`	Reload any manual change made to the files `/etc/sysconfig/network-scripts/ifcfg-*`

The manpage `man nmcli-examples` contains examples of network configuration.

Network teaming allows binding together two or more network interfaces to increase throughput or provide redundancy. RHEL7 and later implement network teaming via the `teamd` daemon.

How to set up a teaming connection

1.
   ```
   nmcli connection add type team con-name teamcon ifname teamif \
   config '{"runner":{"name":"loadbalance"}}'
   ```
   Set up a team connection *teamcon* and a team interface *teamif* with a runner (in JSON code) for automatic failover

2.
   ```
   nmcli connection modify teamcon ipv4.method manual \
   ipv4.addresses 10.0.0.14/24 ipv4.gateway 10.0.0.254
   ```
   Assign manually an IP address and gateway

3.
   ```
   nmcli connection add type team-slave ifname iface \
   master teamcon
   ```
   Add an existing device *iface* as a slave of team *teamcon*.
   The slave connection will be automatically named `team-slave-iface`
4. Repeat the previous step for each slave interface.

`teamdctl teamif state`	Show the state of the team interface *teamif*
`teamnl teamif command`	Debug a team interface *teamif*

A **network bridge** emulates a hardware bridge, i.e. a Layer 2 device able to forward traffic between networks based on MAC addresses.

How to set up a bridge connection

1.
   ```
   nmcli connection add type bridge con-name brcon ifname brif
   ```
   Set up a bridge connection *brcon* and a bridge interface *brif*

2.
   ```
   nmcli connection modify brcon ipv4.method manual \
   ipv4.addresses 10.0.0.15/24 ipv4.gateway 10.0.0.254
   ```
   Assign manually an IP address and gateway

3.
   ```
   nmcli connection add type bridge-slave ifname iface \
   master brcon
   ```
   Add an existing device *iface* as a slave of bridge *brcon*.
   The slave connection will be automatically named `bridge-slave-iface`
4. Repeat the previous step for each slave interface.

`brctl show brif`	Display information about the bridge interface *brif*

The manpage `man teamd.conf` contains examples of team configurations and runners.
The manpage `man nmcli-examples` contains examples of teaming and bridging configuration.

dig *example.org*	Perform a DNS lookup for the specified domain or hostname. Returns information in BIND zone file syntax; uses an internal resolver and hence does not honor /etc/resolv.conf
host *example.org* nslookup *example.org* (deprecated)	Perform a DNS lookup for the specified domain or hostname. Does honor /etc/resolv.conf
dig @*nameserver* -t MX *example.org* host -t MX *example.org nameserver*	Perform a DNS lookup for the MX record of the specified domain, querying *nameserver*
dig *example.org* any host -a *example.org*	Get all DNS records for a domain
dig -x *a.b.c.d* host *a.b.c.d*	Perform a reverse DNS lookup for the IP address *a.b.c.d*
host -la *example.org nameserver*	Perform a DNS Zone Transfer for zone *example.org*, querying the DNS server *nameserver* with a DNS ANY query. This lists all DNS records
nslookup -norecurse *example.org*	Check if the specified domain is present in the DNS cache
whois *example.org*	Query the WHOIS service for an Internet resource (usually a domain name)
ping *host*	Test if a remote host can be reached and measure the round-trip time to it. This is done by sending an ICMP Echo Request datagram and awaiting an ICMP Echo Response
ping -M do -s *size host*	Ping a remote host using an ICMP packet of size *size* (default is 56 bytes) and setting the DF (Don't Fragment) bit. This command can be used to find what is the maximum frame size allowed on the network, by trying increasingly higher values for *size* until it exceeds the MTU and the datagram is unable to reach the destination host
fping -a *host1 host2 host3*	Ping multiple hosts in parallel and report which ones are alive
bing *host1 host2*	Calculate point-to-point throughput between two hosts
traceroute *host*	Print the route, hop by hop, packets trace to a remote host. This is done by sending a sequence of ICMP Echo Request datagrams with increasing TTL values, starting with TTL=1, and expecting ICMP Time Exceeded datagrams
tracepath *host*	Simpler traceroute
tcptraceroute *host*	Implementation of traceroute that uses TCP packets
mtr *host*	traceroute and ping combined
telnet *host*	Establish a telnet connection to a remote host
telnet *host port*	Establish a telnet connection to a remote *host* on the specified *port*. Useful for a quick-and-dirty test of network services
uucp *srchost*!*path desthost*!*path*	Unix-to-Unix copy. Copies files between hosts, identified by a bang path. Obsolete

`echo >/dev/tcp/ipaddress/port \`
`>/dev/null 2>&1 && echo "port is open"`
	Check if *port* at *ipaddress* is open

`redir --laddr=ip1 --lport=port1 \`
`--caddr=ip2 --cport=port2`
	Redirect all connections, coming to local IP address *ip1* and port *port1*, to remote IP address *ip2* and port *port2*

`stunnel`	TLS encryption wrapper. Can be used to secure any client-server protocol
`socat`	Establish two bidirectional data stream and transfer data between them
`socat TCP-LISTEN:80,fork  TCP:host:80`	Forward local HTTP port to remote *host*'s HTTP port
`socat TCP:timeserver:13 -`	Query a *timeserver* using the Daytime Protocol
`wget`	Download a file via HTTP, HTTPS, or FTP

`wget --no-clobber --html-extension \`
`--page-requisites --convert-links \`
`--recursive --domains example.org \`
`--no-parent www.example.org/path`
	Download a whole website *www.example.org/path*

`curl`	Transfer data to or from a remote host via HTTP, HTTPS, FTP, FTPS, SCP, SFTP, TFTP, DICT, TELNET, LDAP, or FILE
`curl www.example.org/file -o myfile`	Download a file via HTTP and save it locally under another name
`curl -u user:psw 'ftp://server/file'`	Download a file via FTP, after logging in to the server
`curl -XPUT webserver -d'data'`	Send an HTTP PUT command with *data* to *webserver*
`tcpd`	Monitor and intercept incoming requests for services mapped one-to-one to executable files (e.g. telnet, finger, ftp, rsh, rlogin, tftp). `inetd` redirects these incoming service requests to `tcpd`, which logs the request and performs some checks before running the specific server program
`inetsim`	Simulate Internet services. This is useful when setting up a confined lab for malware analysis
`pktgen`	Network packet generator. Uses the DPDK (Data Plane Development Kit) packet processing framework
`trafgen`	Network packet generator
`packETH`	Ethernet packet generator (GUI)
`packETHcli`	Ethernet packet generator (command line)

Wireless network tools

`iwlist wlan0 scan`	List all wireless devices in range, with their quality of signal and other information
`iwlist wlan0 freq`	Display transmission frequency settings
`iwlist wlan0 rate`	Display transmission speed settings
`iwlist wlan0 txpower`	Display transmission power settings
`iwlist wlan0 key`	Display encryption settings
`iwgetid wlan0` *`option`*	Print NWID, ESSID, AP/Cell address or other information about the wireless network that is currently in use
`iwconfig wlan0`	Display configuration of wireless interface `wlan0`
`iwconfig wlan0` *`option`*	Configure wireless interface `wlan0`
`iw dev wlan0 station dump`	On a wireless card configured in AP Mode, display information (e.g. MAC address, tx/rx, bitrate, signal strength) about the clients
`rfkill list`	List installed wireless devices
`rfkill unblock` *`n`*	Enable wireless device number *n*
`hostapd`	Daemon that allows a wireless card to function in Host AP Mode, i.e. perform all functions of an Access Point
`hcidump -i` *`device`*	Display raw HCI (Host Controller Interface) data exchanged with a Bluetooth *device*

`netstat`	Display current network connections. Options: `-t` Display active TCP connections `-l` Display only listening sockets `-a` Display all listening and non-listening sockets `-p` Display PID and name of program to which each socket belongs `-i` Display network interfaces `-s` Display protocol statistics `-r` Display kernel routing tables (equivalent to `route -e`) `-n` Do not resolve hostnames or portnames `-c` Continuously display connections
`ss`	Display socket statistics (similarly to `netstat`)
`ss -t -a`	Display all TCP sockets
`arp-scan`	Scan all hosts on the current LAN. Uses ARP (Layer 2) packets; therefore it is able to also find hosts configured to drop all IP or ICMP traffic, and it cannot scan hosts outside the LAN
`snoop` (Solaris)	Packet sniffer
`tcptrace`	Tool for the analysis of TCP dump files such as those generated by `tcpdump`, `snoop`, etc.
`ipgrab`	Packet sniffer that includes full header fields
`dhcpdump`	DHCP packet sniffer
`ngrep`	Filter data payload of network packets matching a specified regex
`nload`	Display a graph of the current network usage
`ntop`	Network usage analyzer
`ntopng`	Network usage analyzer
`nethogs`	Display bandwidth network usage by process
`iptraf`	Interactive IP LAN monitor (ncurses UI)
`iptraf-ng`	Interactive IP LAN monitor (ncurses UI)
`netserver`	Run a network performance benchmark server
`netperf`	Execute network performance benchmarks, by connecting to a `netserver` server
`iperf -s`	Run a network throughput benchmark server
`iperf -c` *`server`*	Execute network throughput tests in client mode, by connecting to an `iperf` server

Nmap is a network analyzer, auditing tool, and penetration testing tool. The GUI equivalent is **Zenmap**.

`nmap` *`options host`*	Scan a host, or all hosts in a subnet

Port state	
open	An application is listening for connections on the port
closed	No application is listening for connections on the port
filtered	Port is not responding to probe due to a firewall blocking the port, so port may be open or closed
unfiltered	Port is responding to probe, but it is impossible to tell whether port is open or closed

Scan technique options	
`-sT`	**TCP connect scan** aka **full-open scan**. Completes the three-way handshake; response will be SYN/ACK if port is open, RST if port is closed. Slow and likely to trigger IDS
`-sS`	**SYN scan** aka **stealth scan** or **half-open scan**. Sends a TCP packet with SYN flag set; same response as the TCP connect scan. Fast and stealth
`-sN`	**Null scan**. Sends a TCP packet with no flag set; response will be none if port is open, RST if port is closed. Stealth. Works only if the target host's OS TCP/IP implementation is based on RFC 793
`-sF`	**FIN scan**. Sends a TCP packet with FIN flag set. Same response as null scan. Scanning using a TCP packet with FIN and/or PSH and/or URG flags set is known as **inverse TCP flag scan**.
`-sX`	**Xmas tree scan**. Sends a TCP packet with FIN, PSH, and URG flags set. Same response as null scan
`-sA`	**ACK scan**. Sends a TCP packet with ACK flag set; response will be RST if port is open or closed (unfiltered), no response or ICMP error if port is filtered. Further analysis can be done on the TCP/IP RST response packet: if the TTL field is less than the max value, or the Window Size field is nonzero, port is open. Used to discover firewall rules, and to determine firewall type: if unfiltered ports are reported the firewall is stateless, otherwise the firewall is stateful
`-sI` *`zombiehost`*	**IP ID idle scan**. Uses a zombie host to perform the scan and predicts the port state of the target host by analyzing the IP fragmentation ID sequence numbers from the zombie host; if the sequence number has increased by 2 port is open, if it has increased by 1 port is closed. Fully stealth as no packets are sent from the scanning machine to the target host
`-sU`	**UDP scan**. Sends a UDP packet; response will be none if port is open or filtered, ICMP Port Unreachable (Type 3 Destination Unreachable) error if port is closed
`-sO`	**IP protocol scan**. Cycles through IP protocol numbers (instead of TCP or UDP ports) to try to determine which IP protocols (TCP, ICMP, IGMP, etc.) the target host supports; response will be any protocol response or none if port is open, ICMP Protocol Unreachable (Type 3 Destination Unreachable) error if port is closed, other ICMP Type 3 Destination Unreachable errors or none if port is filtered
`-sR`	**RPC scan**. Floods all TCP/UDP ports found open with SunRPC program NULL commands to try to determine whether they are RPC ports, and if yes, the service program and version number. It is recommended to use `-sV` instead as it gives more information
`-sC`	Script scan. Uses the default script set. The Nmap Scripting Engine permits writing scripts (in the Lua programming language) to perform automatically various types of network scans
`--script=`*`name`*	Script scan. Runs the *name* scan script. Examples: `--script=sniffer-detect` *`host`* Check if a host has its NIC in promiscuous mode (sniffer) `--script=firewalk` *`host`* Attempt to detect firewall or gateway rules `--script=http-trace -d` *`host`* Send an HTTP TRACE request to find if TRACE method is enabled `--script=http-enum` *`host`* Enumerate dirs used by common web applications and webservers

nmap - options part 1

Host discovery options	
-sL	**List scan**. Simple network host discovery, with reverse DNS resolution. No packet is sent to target hosts
-sn	**No port scan** aka **ping sweep**. Used to detect how many hosts are up
-Pn	No ping; skip host discovery. The subsequent scan operation will be performed against all hosts, instead of only those discovered to be up during this phase
-PS*port*	Send a TCP SYN packet to the specified port
-PA*port*	Send a TCP ACK packet to the specified port
-PU*port*	Send a UDP packet to the specified port
-PY*port*	Send a SCTP packet containing a minimal INIT chunk to the specified port
-PE	Send a ICMP Echo Request (type 8)
-PP	Send a ICMP Timestamp Request (type 13)
-PM	Send a ICMP Address Mask Request (type 17)
-PO*protocol*	Send IP packets with the specified protocol number set in their header
-PR	**ARP scan**. Default discovery type when scanning the current LAN
--traceroute	After the scan, trace path to host to determine port and protocol most likely to reach the target host

IDS evasion, firewall evasion, and spoofing options	
-f	Use tiny fragmented packets (8 bytes or less) for IDS evasion. Might crash the target host
--mtu *offset*	Use fragmented packets of size *offset* (must be a multiple of 8) for IDS evasion
-D *ip*	Spoof the scanning machine IP address as *ip*
-D RND:*n*	Spoof the scanning machine IP address using *n* randomly generated addresses. The real IP address is included among the decoys
--ip-options "L *ip1 ip2*"	Use loose source routing for IDS evasion, requiring that the packet is loose source routed through the waypoints with IP address *ip1* and *ip2*
--ip-options "S *ip1 ip2*"	Use strict source routing for IDS evasion, requiring that the packet is strictly source routed through the waypoints with IP address *ip1* and *ip2*. All waypoints must be specified

Timing options	
-T0	Paranoid. Extremely slow serialized scan for IDS evasion. Will take a long time to complete
-T1	Sneaky. Very slow serialized scan for IDS evasion. Will take a long time to complete
-T2	Polite. Slow serialized scan to consume less bandwidth and resources of the target
-T3	Normal. Parallel scan. Default
-T4	Aggressive. Fast parallel scan, to be used on networks with a high bandwidth. Recommended
-T5	Insane. Very fast parallel scan, to be used on networks with a very high bandwidth. Might be less accurate

Other options	
`-A`	Aggressive scan. Equivalent to `-O -sV -sC --traceroute`
`-O`	OS fingerprinting, to find out which operating system is running on target host
`-sV`	Version detection, to determine protocol, application name, version number, device type, etc.
`-6`	Enable IPv6 scanning
`-p` *`port`*	Scan only the specified port or port range, instead of the most common 1000 ports for each protocol
`-p-`	Scan all ports (from 1 to 65535)
`--top-ports` *`n`*	Scan only the *n* most popular ports
`-F`	Fast mode; scan fewer ports than the default, hence enumerating all hosts faster
`-r`	Scan ports in numerical order, instead of random order
`-n`	Do not do DNS resolution
`-R`	Always do DNS resolution
`-oN` *`file.nmap`*	Save output to *file* in standard format (slightly different from interactive mode output)
`-oX` *`file.xml`*	Save output to *file* in XML format

Tcpdump is a packet sniffer (aka packet analyzer) which uses the `libpcap` library for packet capture. The GUI equivalent of tcpdump is **Wireshark**, formerly called Ethereal.

Sniffers operate at the Data Link layer (Layer 2).
On a wired medium, for a sniffing machine to be able to capture all network traffic, and not only the traffic from/to the machine itself, the machine's NIC must be set to promiscuous mode. Furthermore, only traffic within a network segment connected via a hub (i.e. the collision domain) can be sniffed; in the case of a switched network, the sniffing machine needs to be connected to the switch's SPAN port (which performs port mirroring) in order to be able to capture all traffic.
In the case of a wireless NIC, the chipset also determines capabilities for modes of operation.
Active sniffing refers to sniffing through a switch. Passive sniffing refers to sniffing through a hub.

`tcpdump` *`options expression`*	Print the content of sniffed packets that match *expression*. Options: `-v -vv` Increasing levels of verbosity `-n` Do not perform DNS resolution on host addresses `-nn` Do not convert protocol and port numbers to names
`tcpdump -i eth0`	Sniff all network traffic on interface `eth0`
`tcpdump ip host 10.0.0.2 tcp port 25`	Sniff network packets on TCP port 25 from and to 10.0.0.2
`tcpdump ether host '45:67:89:ab:cd:ef'`	Sniff traffic from and to the network interface having MAC address 45:67:89:ab:cd:ef

`tcpdump 'src host 10.0.0.2 and \`
`(tcp port 80 or tcp port 443)'`
	Sniff HTTP and HTTPS traffic having as source host 10.0.0.2

`tcpdump -i eth0 not port 22`	Sniff all traffic on `eth0` except that belonging to a SSH connection
`tcpdump -i eth0 arp`	Sniff ARP traffic on `eth0`
`tcpdump ip host 10.0.0.2 and not 10.0.0.9`	Sniff IP traffic between 10.0.0.2 and any other host except 10.0.0.9

	PCAP filter syntax
`tcp.port==25 or icmp`	Show SMTP and ICMP traffic

`ip.addr==10.0.0.2`
`ip.src==10.0.0.2 or ip.dst==10.0.0.2`
	Show traffic from and to 10.0.0.2

`ip.src==10.0.0.3 and frame.pkt_len > 400`	Show packets coming from 10.0.0.3 with frame length higher than 400
`http.request`	Show HTTP requests
`udp contains 76:54`	Show UDP packets containing the 2-byte hex sequence 0x76, 0x54 in the header or the payload, at any offset

`==`
`eq`
	Equal to	`>`
		`gt`
			Greater than	`>=`
				`ge`
					Greater than or equal to	`&&`
						`and`
							Logical AND

`!=`
`ne`
	Not equal to	`<`
		`lt`
			Less than	`<=`
				`le`
					Less than or equal to	`||`
						`or`
							Logical OR

Netcat is "the Swiss Army knife of networking", a very flexible generic TCP/IP client/server.
Depending on the distribution, the binary is called `nc`, `ncat` (Red Hat), or `netcat` (SUSE).

```
nc -z 10.0.0.7 22
ncat 10.0.0.7 22
```

Scan for a listening SSH daemon on remote host 10.0.0.7

```
nc -l -p 25
```

Listen for connections on port 25 (i.e. mimic an SMTP server). Send any input received on stdin to the connected client and dump on stdout any data received from the client

```
nc 10.0.0.7 389 < file
```

Push the content of *file* to port 389 on remote host 10.0.0.7

```
echo "GET / HTTP/1.0\r\n\r\n" | nc 10.0.0.7 80
```

Connect to web server 10.0.0.7 and issue an HTTP GET

```
while true; \
do nc -l -p 80 -q 1 < page.html; done

while true; \
do echo "<html><body>Hello</body></html>" \
| ncat -l -p 80; done
```

Start a minimal web server, serving the specified HTML page to clients

```
nc -v -n -z -w1 -r 10.0.0.7 1-1023
```

Run a TCP port scan against remote host 10.0.0.7.
Probes randomly all privileged ports with a 1-second timeout, without resolving service names, and with verbose output

```
echo "" | nc -v -n -w1 10.0.0.7 1-1023
```

Retrieve the greeting banner of any network service that might be running on remote host 10.0.0.7

Hping3 is a packet crafting tool, able to send any custom TCP/IP packet to a remote host and display the reply.
It is an extension of hping2, and is command-line compatible with it while having extended capabilities for packet generation.

`hping3 options host`	Send a crafted packet to *host*. By default, it sends TCP headers to port 0 of remote host with no TCP flag set and a window size of 64

`hping3` **options**	
`-c n`	Send *n* packets
`-p n`	Use port *n*

`-a src`
`--spoof src`
	Set *src* as a fake IP source address for sent packets

`-1`
`--ICMP`
	Use the ICMP protocol. By default, hping3 uses TCP

`-2`
`--UDP`
	Use the UDP protocol

`-8 n1-n2`
`--scan n1-n2`
	Operate in scan mode, scanning the port range from *n1* to *n2*

`-9 signature`
`--listen signature`
	Operate in listening mode, trying to intercept *signature*

`-A`	Set the ACK flag in probe packets. An ACK scan can be used to check if the remote host is alive, when it does not respond to ping packets
`-S`	Set the SYN flag in probe packets
`-F`	Set the FIN flag in probe packets
`-P`	Set the PSH flag in probe packets
`-U`	Set the URG flag in probe packets
`-Q`	Collect all TCP sequence numbers generated by the remote host
`--tcp-timestamp`	Attempt to guess the timestamp update frequency and uptime of the remote host

`hping3 -S -p 25 -c 5 host`	Send 5 TCP packets, with the SYN flag set, to port 25 of remote host
`hping3 --scan 1-1024 -S host`	Perform a SYN scan on ports 1 to 1024 against the remote host
`hping3 --udp --rand-source --data 512 host`	Send UDP packets with random source address and a data body size of 512 bytes
`hping3 -S -p 80 --flood host`	Perform a TCP SYN flood DoS attack against a webserver
`hping3 -A -p 25 host`	Verify if a mailserver is alive (if it is, it will reply with an RST)

The **TCP Wrapper** feature provides basic traffic filtering of incoming network connections. To use this feature, the service binary must have been compiled with the `libwrap.a` library.

`ldd` *`service_binary`* `| grep libwrap`	Find if a network service is TCP Wrapped

`/etc/hosts.allow`
`/etc/hosts.deny`

Host access control files used by the TCP Wrapper system.

Each file contains zero or more *`daemon`*`:`*`client`* lines. The first matching line is considered.

Access is granted when a *`daemon`*`:`*`client`* pair matches an entry in `/etc/hosts.allow`.
Otherwise, access is denied when a *`daemon`*`:`*`client`* pair matches an entry in `/etc/hosts.deny`.
Otherwise, access is granted.

`/etc/hosts.allow` and `/etc/hosts.deny` lines syntax	
`ALL: ALL`	All services to all hosts
`ALL: .example.edu`	All services to all hosts of the example.edu domain
`ALL: .example.edu EXCEPT host1.example.edu`	All services to all hosts of example.edu, except host1
`in.fingerd: .example.com`	Finger service to all hosts of example.com
`in.tftpd: LOCAL`	TFTP to hosts of the local domain only
`sshd: 10.0.0.3 10.0.0.4 10.1.1.0/24`	SSH to the hosts and network specified

`sshd: 10.0.1.0/24`
`sshd: 10.0.1.`
`sshd: 10.0.1.0/255.255.255.0`
	SSH to 10.0.1.0/24

`in.tftpd: ALL: spawn (/safe_dir/safe_finger -l @%h \`
`| /bin/mail -s %d-%h root) &`
	Send a finger probe to hosts attempting TFTP and notify the root user via email

`portmap: ALL: (echo Illegal RPC request from %h \`
`| /bin/mail root) &`
	When a client attempts an RPC request via portmapper (NFS access), echo a message to the terminal and notify the root user via email

Output of command `route -en`

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.3.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.3.1     0.0.0.0         UG    0      0        0 eth0
```

Destination	*`network or host`*	destination network or host
	`0.0.0.0`	default route
Gateway	*`host`*	gateway

	`0.0.0.0`
	`*`
		no gateway needed, network is directly connected

	`-`	rejected route
Genmask	*`network mask`*	network mask to apply for the destination network
	`255.255.255.255`	destination host
	`0.0.0.0`	default route
Flags	`U`	route is up
	`G`	use gateway
	`H`	target is host
	`!`	rejected route
	`D`	dynamically installed by daemon
	`M`	modified from routing daemon
	`R`	reinstate route for dynamic routing

`ip route`
`route -en`
`route -F`
`netstat -rn`
	Display IP routing table

`ip route show cache`
`route -C`
	Display kernel routing cache

`ip route add default via 10.1.1.254`
`route add default gw 10.1.1.254`
	Add a default gateway 10.1.1.254

`ip route add 10.2.0.1 dev eth0`
`ip route add 10.2.0.1 via 10.2.0.254`
`route add -host 10.2.0.1 gw 10.2.0.254`
	Add a route for a host 10.2.0.1

`ip route add 10.2.0.0/16 via 10.2.0.254`
`route add -net 10.2.0.0 netmask 255.255.0.0 gw 10.2.0.254`
	Add a route for a network 10.2.0.0/16

`ip route delete 10.2.0.1 dev eth0`
`route del -host 10.2.0.1 gw 10.2.0.254`
	Delete a route for a host 10.2.0.1

`ip route flush all`	Delete the routing table for all interfaces

The Netfilter framework provides firewalling capabilities in Linux. It is implemented by the user-space application programs `iptables` for IPv4 (which replaced `ipchains`, which itself replaced `ipfwadm`) and `ip6tables` for IPv6.
`iptables` is implemented in the kernel and therefore does not have a daemon process or a service.
The ability to track connection state is provided by the `ip_conntrack` kernel module.

In RHEL 6, the service `iptables` provides all firewall functionalities; the GUI frontend is `system-config-firewall`.
In RHEL 7, the firewall is managed by the `firewalld` daemon, which uses `iptables` as backend. It is possible, but discouraged, to use `iptables` directly by disabling `firewalld` and installing the package `iptables-services`, which provides systemd units for `iptables`.
In RHEL 8, the firewall is managed by `firewalld`, with `nftables` (a replacement for `iptables`) as backend.
In Ubuntu, firewall capabilities are provided by the `ufw` (Uncomplicated Firewall) service, with `iptables` as backend.

`/etc/sysconfig/iptables`	Default file containing the firewall rules

`iptables-restore <` *`file`*	Load into iptables the firewall rules specified in the *file*
`iptables-save >` *`file`*	Save into iptables the firewall rules specified in the *file*

iptables rules file	

`*filter`
`:INPUT ACCEPT [0:0]`
`:FORWARD ACCEPT [0:0]`
`:OUTPUT ACCEPT [0:0]`
`COMMIT`
	Delete all rules and open the firewall to all connections

Iptables uses **tables** containing sets of **chains**, which contain sets of **rules**. Each rule has a **target** (e.g. ACCEPT).
The "filter" table contains chains INPUT, FORWARD, OUTPUT (built-in chains); this is the default table to which all iptables commands are applied, unless another table is specified via the `-t` option.
The "nat" table contains chains PREROUTING, OUTPUT, POSTROUTING.
The "mangle" table contains chains PREROUTING, OUTPUT.
When a packet enters the system, it is handed to the INPUT chain. If the destination is local, it is processed; if the destination is not local and IP forwarding is enabled, the packet is handed to the FORWARD chain, otherwise it is dropped.
An outgoing packet generated by the system will go through the OUTPUT chain.
If NAT is in use, an incoming packet will pass at first through the PREROUTING chain, and an outgoing packet will pass last through the POSTROUTING chain.

`iptables -A INPUT -s 10.0.0.6 -j ACCEPT`	Add a rule to accept all packets from 10.0.0.6
`iptables -A INPUT -s 10.0.0.7 -j REJECT`	Add a rule to reject all packets from 10.0.0.7 and send back a ICMP response to the sender
`iptables -A INPUT -s 10.0.0.8 -j DROP`	Add a rule to silently drop all packets from 10.0.0.8
`iptables -A INPUT -s 10.0.0.9 -j LOG`	Add a rule to log (via syslog) all packets from 10.0.0.9

`iptables -D INPUT -s 10.0.0.9 -j LOG`	Delete a specific rule
`iptables -D INPUT 42`	Delete rule 42 of the INPUT chain
`iptables -F INPUT`	Flush all rules of the INPUT chain
`iptables -F`	Flush all rules, hence disabling the firewall
`iptables -t mangle -F`	Flush all rules of the "mangle" table
`iptables -t mangle -X`	Delete all user-defined (not built-in) rules in the "mangle" table

`iptables -L INPUT`	List the rules of the INPUT chain
`iptables -L -n`	List all rules, without translating numeric values (IP addresses to FQDNs and port numbers to services)

`iptables -N mychain`	Define a new chain
`iptables -P INPUT DROP`	Define the chain policy target, which takes effect when no rule matches and the end of the rules list is reached

`iptables -A OUTPUT -d 10.7.7.0/24 -j DROP`	Add a rule to drop all packets with destination 10.7.7.0/24
`iptables -A FORWARD -i eth0 -o eth1 -j LOG`	Add a rule to log all packets entering the system via eth0 and exiting via eth1

`iptables -A INPUT -p 17 -j DROP`
`iptables -A INPUT -p udp -j DROP`
	Add a rule to drop all incoming UDP traffic (protocol numbers are defined in `/etc/protocols`)

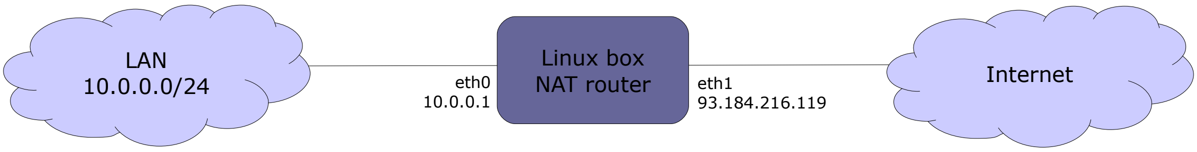
`iptables -A INPUT --sport 1024:65535 --dport 53 \`
`-j ACCEPT`
	Add a rule to accept all packets coming from any unprivileged port and with destination port 53

`iptables -A INPUT -p icmp --icmp-type echo-request \`
`-m limit --limit 1/s -i eth0 -j ACCEPT`
	Add a rule to accept incoming pings through eth0 at a maximum rate of 1 ping/second

`iptables -A INPUT -m state --state ESTABLISHED \`
`-j ACCEPT`
	Load the module for stateful packet filtering, and add a rule to accept all packets that are part of a communication already tracked by the state module

`iptables -A INPUT -m state --state NEW -j ACCEPT`	Add a rule to accept all packets that are not part of a communication already tracked by the state module
`iptables -A INPUT -m state --state RELATED -j ACCEPT`	Add a rule to accept all packets that are related (e.g. ICMP responses to TCP or UDP traffic) to a communication already tracked by the state module
`iptables -A INPUT -m state --state INVALID -j ACCEPT`	Add a rule to accept all packets that do not match any of the states above

iptables - NAT routing



SNAT (Source Network Address Translation)

```
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o eth1 \
-j SNAT --to-source 93.184.216.119
```

Map all traffic leaving the LAN to the external IP address 93.184.216.119

```
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o eth1 \
-j SNAT --to-source 93.184.216.119:93.184.216.127
```

Map all traffic leaving the LAN to a pool of external IP addresses 93.184.216.119-127

```
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

Map all traffic leaving the LAN to the address dynamically assigned to eth1 via DHCP

DNAT (Destination Network Address Translation)

```
iptables -t nat -A PREROUTING -i eth1 -d 93.184.216.119 \
-j DNAT --to-destination 10.0.0.13
```

Allow the internal host 10.0.0.13 to be publicly reachable via the external address 93.184.216.119

PAT (Port Address Translation)

```
iptables -t nat -A PREROUTING -i eth1 -d 93.184.216.119 \
-p tcp --dport 80 -j DNAT --to-destination 10.0.0.13:8080
```

Make publicly accessible a webserver that is located in the LAN, by mapping port 8080 of the internal host 10.0.0.13 to port 80 of the external address 93.184.216.119

```
iptables -t nat -A PREROUTING -i eth0 -d ! 10.0.0.0/24 \
-p tcp --dport 80 -j REDIRECT --to-ports 3128
```

Redirect all outbound HTTP traffic originating from the LAN to a proxy running on port 3128 on the Linux box

```
sysctl -w net.ipv4.ip_forward=1
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Enable IP forwarding; necessary to set up a Linux machine as a router. (This command causes other network options to be changed as well.)

In firewalld, a network interface (aka **interface**) or a subnet address (aka **source**) can be assigned to a specific **zone**.
To determine to which zone a packet belongs, first the zone of the source is analyzed, then the zone of the interface; if no source or interface matches, the packet is associated to the default zone (which is "public", unless set otherwise).
If the zone is not specified (via `--zone=`*`zone`*), the command is applied to the default zone.
By default, commands are temporary; adding the `--permanent` option to a command sets it as permanent, or shows permanent settings only.
Temporary commands are effective immediately but are canceled at reboot, firewall reload, or firewall restart.
Permanent commands are effective only after reboot, firewall reload, or firewall restart.

Firewalld zones	
block	Rejects incoming connections with an ICMP HOST_PROHIBITED; allows only established connections
dmz	Used to expose services to the public; allows only specific incoming connections
drop	Drops all incoming packets; allows only outgoing connections
external	Used for routing and masquerading; allows only specific connections
home	Allows only specific incoming connections
internal	Used to define internal networks and allow only private network traffic
public	Allows only specific incoming connections. Default zone
trusted	Accepts all traffic
work	Used to define internal networks and allow only private network traffic

The list of firewall zones can be obtained via the command `firewall-cmd --get-zones`.

`systemctl status firewalld`
`firewall-cmd --state`
	Check the status of the firewall

`firewall-config`	Firewall management GUI

`firewall-cmd --reload`	Reload firewall configuration; this applies all permanent changes and cancels all temporary changes. Current connections are not terminated
`firewall-cmd --complete-reload`	Reload firewall configuration, stopping all current connections
`firewall-cmd --runtime-to-permanent`	Transform all temporary changes to permanent

`firewall-cmd --list-all-zones`	List all zones and their full settings
`firewall-cmd --get-default-zone`	Show the default zone
`firewall-cmd --set-default-zone=home`	Set "home" as the default zone
`firewall-cmd --get-active-zones`	Show the active zones i.e. zones bound to either an interface or a source
`firewall-cmd --get-zones`	Show all available zones
`firewall-cmd --get-zone-of-interface=eth0`	Show the zone assigned to `eth0`
`firewall-cmd --new-zone=test`	Create a new zone called "test"
`firewall-cmd --zone=home --change-interface=eth0`	Assign eth0 to the "home" zone
`firewall-cmd --zone=home --list-all`	List temporary settings of the "home" zone
`firewall-cmd --zone=home --list-all --permanent`	List permanent settings of the "home" zone
`firewall-cmd --zone=home --add-source=10.1.1.0/24`	Assign 10.1.1.0/24 to the "home" zone i.e. route all traffic from that subnet to that zone
`firewall-cmd --zone=home --list-sources`	List sources bound to the "home" zone

`firewall-cmd --zone=trusted --add-service=ssh`
`firewall-cmd --zone=trusted --add-port=22/tcp`
	Add the SSH service to the "trusted" zone

`firewall-cmd --zone=trusted --add-service={ssh,http,https}`	Add the SSH, HTTP, and HTTPS services to the "trusted" zone
`firewall-cmd --zone=trusted --list-services`	Show temporary and permanent services bound to the "trusted" zone
`firewall-cmd --zone=trusted --list-ports`	Show temporary and permanent ports open on the "trusted" zone

`firewall-cmd --get-services`	List all predefined services

Predefined services are configured in `/usr/lib/firewalld/services/`*`service`*`.xml`.
User-defined services are configured in `/etc/firewalld/services/`*`service`*`.xml`.

`firewall-cmd --get-icmptypes`	Show all known types of ICMP messages
`firewall-cmd --add-icmp-block=echo-reply`	Block a specific ICMP message type
`firewall-cmd --query-icmp-block=echo-reply`	Tell if a specific ICMP message type is blocked
`firewall-cmd --list-icmp-block`	Show the list of blocked ICMP message types

`firewall-cmd --add-rich-rule='`*`richrule`*`'`	Set up a **rich rule** (for more complex and detailed firewall configurations)

`firewall-cmd --add-rich-rule='rule \`
`family=ipv4 source address=10.2.2.0/24 service name=tftp`
`log prefix=tftp level=info limit value=3/m accept'`
	Set up a rich rule to allow tftp connections from subnet 10.2.2.0/24 and log them via syslog at a rate of 3 per minute

`firewall-cmd --list-rich-rules`	List all rich rules

The manpage `man firewalld.richlanguage` contains several examples of rich rules.

`firewall-cmd --direct --add-rule `*`directrule`*	Set up a **direct rule** (in iptables format)

`firewall-cmd --direct --add-rule \`
`ipv4 filter INPUT 0 -p tcp --dport 22 -j ACCEPT`
	Set up a direct rule to allow SSH connections

`firewall-offline-cmd `*`directrule`*	Set up a direct rule when firewalld is not running
`firewall-cmd --direct --get-all-rules`	Show all direct rules

User-defined direct rules are stored in `/etc/firewalld/direct.xml`.
The manpage `man firewalld.direct` documents the syntax of direct rules.

`firewall-cmd --zone=`*`zone`*` --add-masquerade`	Set up masquerading for hosts of *zone*; packets originating from *zone* will get the firewall's IP address on the "external" zone as source address

`firewall-cmd --zone=`*`zone`*` --add-rich-rule='rule \`
`family=ipv4 source address=10.2.2.0/24 masquerade'`
	Set up masquerading only for those hosts of *zone* located in subnet 10.2.2.0/24

`firewall-cmd --zone=`*`zone`*` --add-forward-port=\`
`port=22:proto=tcp:toport=2222:toaddr=10.7.7.7`
	Set up port forwarding for hosts of *zone*; incoming connections to port 22 for hosts of *zone* will be forwarded to port 2222 on host 10.7.7.7

Secure Shell (SSH) is a protocol (not a shell) for encrypted secure communications. It is mostly used as a replacement to Telnet to securely login to a remote server's terminal, but can be applied to any network protocol: some of the most common applications of SSH are Secure Copy (SCP) and SSH File Transfer Protocol (SFTP).

`ssh user@host`	Connect to a remote *host* via SSH and login as *user*. Options: `-v -vv -vvv` Increasing levels of verbosity `-p n` Use port *n* instead of standard port 22
`ssh user@host command`	Execute a command on a remote host
`autossh user@host`	Connect to a remote host, monitoring the connection and restarting it automatically if it dies
`sshpass -p password ssh user@host`	Connect to a remote host using the specified password
`pssh -i -H "host1 host2 host3" command`	Execute a command in parallel on a group of remote hosts
`ssh-keygen -t rsa -b 2048`	Generate interactively a 2048-bit RSA key pair; will prompt for a passphrase
`ssh-keygen -t dsa`	Generate a DSA key pair
`ssh-keygen -p -t rsa`	Change passphrase of the private key
`ssh-keygen -q -t rsa -f keyfile -N '' -C ''`	Generate an RSA key with no passphrase (for non-interactive use) and no comment
`ssh-keygen -lf keyfile`	View key length and fingerprint of a public or private key

`< keyfile.pub awk '{print $2}' \`
`| base64 -d | openssl hashfunction`
	View fingerprint of a key, calculated using *hashfunction*.
	RSA keys fingerprint use `sha1` (deprecated) or `md5`

`ssh-keyscan host >> ~/.ssh/known_hosts`	Get the public key of *host* and add it to the user's known hosts file
`ssh-agent`	Echo to the terminal the environment variables that must be set in order to use the SSH Agent
``eval `ssh-agent` ``	Start the SSH Agent daemon that caches decrypted private keys in memory; also shows the PID of ssh-agent and sets the appropriate environment variables. Once ssh-agent is started, the keys to cache must be added via the `ssh-add` command; cached keys will then be automatically used by any SSH tool e.g. `ssh`, `sftp`, `scp`
`ssh-agent bash -c 'ssh-add keyfile'`	Start ssh-agent and cache the specified key
`ssh-add`	Add the default private keys to the ssh-agent cache
`ssh-add keyfile`	Add a specific private key to the ssh-agent cache
`ssh-copy-id user@host`	Use locally available keys to authorize, via public key authentication, login of *user* on a remote *host*. This is done by copying the user's local public key `~/.ssh/id_rsa.pub` to `~/.ssh/authorized_keys` on the remote host

```
scp /path1/file user@host:/path2/
scp user@host:/path1/file /path2/
scp user1@host1:/path1/file user2@host2:/path2/
```

Non-interactive secure file copy via SSH.
Can transfer files from local to remote, from remote to local, or between two remote hosts

```
sftp user@host
```

SSH FTP-like tool for secure file transfer

```
scponly
```

SSH wrapper pseudo-shell providing access to remote users for secure file transfer, but without execution privileges

```
sshfs user@host:/dir/ mountpoint/
```

SSH tool that allows mounting a remote directory as an SSH filesystem on a mountpoint on the local machine. Uses the FUSE kernel module.
The filesystem can be unmounted via the command `fusermount -u mountpoint/`

SSH port forwarding (aka SSH tunneling)

```
ssh -L 2525:mail.foo.com:25 user@mail.foo.com
```

Establish a SSH encrypted tunnel from localhost to remote host mail.foo.com, redirecting traffic from local port 2525 to port 25 of remote host mail.foo.com.
Useful if the local firewall blocks outgoing port 25. In this case, port 2525 is used to go out; the application must be configured to connect to localhost on port 2525 (instead of mail.foo.com on port 25)

```
ssh -L 2525:mail.foo.com:25 user@login.foo.com
```

Establish a SSH encrypted tunnel from localhost to remote host login.foo.com.
Remote host login.foo.com will then forward, unencrypted, all data received over the tunnel on port 2525 to remote host mail.foo.com on port 25

SSH reverse forwarding (aka SSH reverse tunneling)

```
ssh -R 2222:localhost:22 user@login.foo.com
```

Establish a SSH encrypted reverse tunnel from remote host login.foo.com back to localhost, redirecting traffic sent to port 2222 of remote host login.foo.com back towards local port 22.
Useful if the local firewall blocks incoming connections so remote hosts cannot connect back to local machine. In this case, port 2222 of login.foo.com is opened for listening and connecting back to localhost on port 22; remote host login.foo.com is then able to connect to the local machine on port 2222 (redirected to local port 22)

SSH as a SOCKS proxy

```
ssh -D 33333 user@login.foo.com
```

The application supporting SOCKS must be configured to connect to localhost on port 33333. Data is tunneled from localhost to login.foo.com, then unencrypted to destination

X11 Forwarding

```
ssh -X user@login.foo.com
```

Enable the local display to execute locally an X application stored on a remote host login.foo.com

How to enable public key authentication

1. On remote host, set `PubkeyAuthentication yes` in `/etc/ssh/sshd_config`
2. On local machine, do `ssh-copy-id you@remotehost` (or copy your public key to the remote host by hand)

How to enable host-based authentication amongst a group of trusted hosts

1. On all hosts, set `HostbasedAuthentication yes` in `/etc/ssh/sshd_config`
2. On all hosts, create `/etc/ssh/shosts.equiv` and enter in this file all trusted hostnames
3. Connect via SSH manually from your machine on each host so that all hosts' public keys go into `~/.ssh/known_hosts`
4. Copy `~/.ssh/known_hosts` from your machine to `/etc/ssh/ssh_known_hosts` on all hosts

How to enable X11 Forwarding

1. On remote host 10.2.2.2, set `X11Forwarding yes` in `/etc/ssh/sshd_config`, and make sure that `xauth` is installed
2. On local host 10.1.1.1, type `ssh -X 10.2.2.2`, then run on remote host the graphical application e.g. `xclock &`

It is also possible to enable X11 Forwarding via telnet (unencrypted, therefore insecure and not recommended):
1. On remote host 10.2.2.2, type `export DISPLAY=10.1.1.1:0.0`
2. On local host 10.1.1.1, type `xhost +`
3. On local host 10.1.1.1, type `telnet 10.2.2.2`, then run on remote host the graphical application e.g. `xclock &`

SSH - configuration

/etc/ssh/sshd_config	SSH server daemon configuration file
/etc/ssh/ssh_config	SSH client global configuration file
/etc/ssh/ssh_host_key	Host's private key (should be mode 0600)
/etc/ssh/ssh_host_key.pub	Host's public key
/etc/ssh/shosts.equiv	Names of trusted hosts for host-based authentication
/etc/ssh/ssh_known_hosts	Database of host public keys that were previously accepted as legitimate
~/.ssh/	User's SSH directory (must be mode 0700)
~/.ssh/config	SSH client user configuration file
~/.ssh/id_rsa ~/.ssh/id_dsa	User's RSA or DSA private key, as generated by ssh-keygen
~/.ssh/id_rsa.pub ~/.ssh/id_dsa.pub	User's RSA or DSA public key, as generated by ssh-keygen
~/.ssh/known_hosts	Host public keys that were previously accepted as legitimate by the user
~/.ssh/authorized_keys ~/.ssh/authorized_keys2 (obsolete)	Trusted public keys; the corresponding private keys allow the user to authenticate on this host

/etc/ssh/sshd_config	**SSH server configuration file**
PermitRootLogin yes	Control superuser login via SSH. Possible values are: yes Superuser can login no Superuser cannot login without-password Superuser cannot login with password forced-commands-only Superuser can only run commands in SSH command line
AllowUsers jdoe ksmith DenyUsers jhacker	List of users that can/cannot login via SSH, or * for everybody
AllowGroups geeks DenyGroups *	List of groups whose members can/cannot login via SSH, or * for all groups
PasswordAuthentication yes	Permit authentication via login and password
PubKeyAuthentication yes	Permit authentication via public key
HostbasedAuthentication yes	Permit authentication based on trusted hosts
Protocol 1,2	Specify protocols supported by SSH. Value can be 1 or 2 or both
X11Forwarding yes	Allow X11 Forwarding

/etc/ssh/ssh_config and ~/.ssh/config	**SSH client configuration file**
Host *	List of hosts to which the following directives will apply, or * for all hosts
StrictHostKeyChecking yes	Ask before adding new host keys to the ~/.ssh/known_hosts file, and refuse to connect if the key for a known host has changed. This prevents MITM attacks
GSSAPIAuthentication yes	Support authentication using GSSAPI
ForwardX11Trusted yes	Allow remote X11 clients to fully access the original X11 display
IdentityFile ~/.ssh/id_rsa	User identity file for authentication. Default values are: ~/.ssh/identity for protocol version 1 ~/.ssh/id_rsa and ~/.ssh/id_dsa for protocol version 2

The **X.509** standard defines the format of public key certificates and other related files; it includes cryptographic standards and protocols such as SSL/TLS, PKCS7, PKCS12, and OCSP. The **Public Key Infrastructure X.509 (PKIX)** is described in RFC 5280.

X.509 file formats	
DER	Binary-encoded certificate
PEM	ASCII-armored Base64-encoded certificate, included between these two lines: -----BEGIN *FILE_TYPE*----- -----END *FILE_TYPE*----- where *FILE_TYPE* is one of the X.509 file types (see below)
DER and PEM are also used as file extensions for different types of files (see below).	

X.509 file type extensions	
CRT **CER**	Certificate or certificate chain
CSR	Certificate Signing Request
KEY	Private key
CRL	Certificate Revocation List
DER	Certificate; DER-encoded
PEM	Certificate (including or not the private key), certificate chain, or Certificate Signing Request; PEM-encoded

Other file type extensions	
P12 **PFX**	Certificate (including or not the private key), certificate chain, or Certificate Signing Request; bundled in a PKCS#12 archive file format

OpenSSL is an open source cryptographic library containing an implementation of the SSL (Secure Sockets Layer) and TLS (Transport Layer Security) protocols, plus various cryptographic utilities.

`openssl`	OpenSSL command line tool
`genkey`	Interactive utility for the generation of SSL certificates and Certificate Signing Requests
`certwatch`	Program that issues email warnings when an SSL certificate is close to its expiration date
`keyrand`	Utility that collects random bits from `/dev/random` and appends them to a file

`CA.pl`	User-friendly command for common certificate operations
`CA.pl -newca`	Create a Certification Authority hierarchy
`CA.pl -newreq`	Generate a Certificate Signing Request
`CA.pl -newreq-nodes`	Generate a Certificate Signing Request, creating also a key pair (unencrypted, for non-interactive use)
`CA.pl -signreq`	Sign a Certificate Signing Request
`CA.pl -pkcs12 "`*`Cert name`*`"`	Generate a PKCS#12 certificate from a Certificate Signing Request
`CA.pl -newcert`	Generate a self-signed certificate
`CA.pl -verify`	Verify a certificate against the Certification Authority certificate for "demoCA"

openssl x509 -text -in *cert.crt* -noout	Read a certificate
openssl req -text -in *cert.csr* -noout	Read a Certificate Signing Request
openssl req -new -key *cert.key* -out *cert.csr*	Generate a Certificate Signing Request, given a private key
openssl req -new -keyout *cert.key* -out *cert.csr* \\ -newkey rsa:2048 -nodes	Generate a Certificate Signing Request, creating also a 2048-bit RSA key pair (unencrypted, for non-interactive use)
openssl x509 -req -in *cert.csr* -CAcreateserial \\ -CA *ca.crt* -CAkey *ca.key* -out *cert.crt* -days *validity*	Sign a certificate as a CA, given a Certificate Signing Request
openssl req -x509 -keyout *cert.key* -out *cert.crt* \\ -newkey rsa:2048 -nodes -days *validity*	Generate a self-signed root certificate, and create a new CA private key
openssl ca -config *ca.conf* -in *cert.csr* \\ -out *cert.crt* -days *validity* -verbose	Sign a certificate
openssl ca -config *ca.conf* -gencrl -revoke *cert.crt* \\ -crl_reason *why*	Revoke a certificate
openssl ca -config *ca.conf* -gencrl -out *list.crl*	Generate a Certificate Revocation List containing all revoked certificates so far
openssl x509 -in *cert.pem* -outform DER -out *cert.der*	Convert a certificate from PEM to DER
openssl pkcs12 -export -in *cert.pem* \\ -inkey *cert.key* -out *cert.pfx* -name *friendlyname*	Convert a certificate from PEM to PKCS#12 including the private key
openssl pkcs12 -in *cert.p12* -out *cert.crt* -clcerts \\ -nokeys	Convert a certificate from PKCS#12 to PEM
openssl pkcs12 -in *cert.p12* -out *cert.key* -nocerts \\ -nodes	Extract the private key from a PKCS#12 certificate
openssl pkcs12 -in *cert.p12* -out *ca.crt* -cacerts	Extract the CA certificate from a PKCS#12 certificate
cat *cert.crt* *cert.key* > *cert.pem*	Create a PEM certificate from CRT and private key
openssl dgst -*hashfunction* -out *file.hash* *file*	Generate the digest (hash) of a file
openssl dgst -*hashfunction* *file* | cmp -b *file.hash*	Check the hash of a file; no output means OK
openssl dgst -*hashfunction* -sign *private.key* \\ -out *file.sig* *file*	Sign a file
openssl dgst -*hashfunction* -verify *public.key* \\ -signature *file.sig* *file*	Verify the signature of a file
openssl enc -e -*cipher* -in *file* -out *file.enc* -salt	Encrypt a file
openssl enc -d -*cipher* -in *file.enc* -out *file*	Decrypt a file
openssl genpkey -algorithm RSA -cipher 3des \\ -pkeyopt rsa_keygen_bits:2048 -out *keypair.pem*	Generate a 2048-bit RSA key pair protected by a TripleDES-encrypted passphrase
openssl pkey -text -in *private.key* -noout	Examine a private key
openssl pkey -in *old.key* -out *new.key* -*cipher*	Change the passphrase of a private key
openssl pkey -in *old.key* -out *new.key*	Remove the passphrase from a private key
1. openssl s_client -connect *www.site.com*:443 > *tmpfile* 2. CTRL C 3. openssl x509 -in *tmpfile* -text	Inspect an SSL certificate from a website
openssl list-message-digest-commands	List all available hash functions
openssl list-cipher-commands	List all available ciphers

In **symmetric cryptography**, a symmetric cipher (i.e. cryptographic algorithm) is used with a **shared secret key** to encrypt a message. The message can then be decrypted using the same key.
In **asymmetric cryptography** aka **Public Key Cryptography**, ciphers do not operate with a single key but with a **key pair**, composed of a **public key** and a **private key**. Public and private key are created together at the same time using a special algorithm and are strictly related to each other; however, deriving a private key from its public key is computationally infeasible. A message is encrypted with a public key and can only be decrypted with the companion private key. Similarly, a message is digitally signed with a private key and can only be verified with the companion public key.

Encryption guarantees confidentiality (only sender and receiver are able to know the contents of the message).
Digital signature guarantees authentication (the receiver can verify that the message originates from the intended sender), integrity (the receiver can verify that the message was not modified since it was signed), and non-repudiation (the sender cannot deny having signed the message).

In a **block cipher**, the input plaintext is split into blocks of fixed size, fed to the algorithm, and transformed according to the key. If the input plaintext is shorter than the block length, padding is used.
In a **stream cipher**, the input plaintext is combined with a pseudorandom digit stream. The key is applied to each bit, one at a time.

A **hash function** maps a bitstring of arbitrary length to another bitstring of fixed length, hence outputting a condensed representative image of the bitstring fed in input. Changing just one bit of the input string results in a very different hash value in output (avalanche effect).
A hash function must have the following properties:
- be one-way, i.e. given an output value it is computationally infeasible to find the matching input (preimage resistance)
- given a specific input, it is computationally infeasible to find another input that results in the same output (second preimage resistance)
- it is computationally infeasible to find two different inputs which would result in the same output (collision resistance)

A **Public Key Infrastructure (PKI)** handles creation, management, distribution, use, and revocation of Digital Certificates.
It is composed of the following entities:

Certification Authority (CA)	Issues and verifies Digital Certificates
Registration Authority (RA)	Verifies the CA, verifies the subject, and ensures valid and correct registration
Validation Authority (VA)	Verifies the validity of a Digital Certificate

`bcrypt` File encryption tool. Uses the Blowfish algorithm

`ccrypt` File and stream encryption tool. Uses the Rijndael block cipher

`ccr` Codecrypt, an encryption and signing tool that uses only algorithms resistant to quantum-computer cryptanalysis

`age` File encryption tool

`stegsnow` Steganography tool for text files. The secret message is concealed in additional tab and whitespace characters at the end of lines

`steghide` Steganography tool for image and audio files

`shasum`
`sha1sum`
`sha224sum`
`sha256sum`
`sha384sum`
`sha512sum` Print or check the digest of a file generated by the SHA hashing algorithm

`md5sum` Print or check the digest of a file generated by the MD5 hashing algorithm

`md5pass` Create an MD5 password hash. If no salt is specified, a random salt will be generated

Symmetric ciphers	
DES (Data Encryption Standard)	Block cipher with a 64-bit block size. Uses DEA (Data Encryption Algorithm) with a 56-bit key. Obsolete and insecure.
3DES (Triple DES)	Cipher which uses a key bundle of three DES keys: K_1, K_2, and K_3. The algorithm consists in applying DES three times: encrypt with K_1, decrypt with K_2, then encrypt with K_3. Key options are: - K_1, K_2, and K_3 are all different (most secure) - $K_1 = K_3$ - $K_1 = K_2 = K_3$ (fallback to DES, insecure)
AES (Advanced Encryption Standard)	Iterated block cipher with a 128-bit block size. NIST standard. Can use a 128-bit, 192-bit, or 256-bit key.
IDEA (International Data Encryption Algorithm)	Block cipher with a 64-bit block size and a 128-bit key. Intended as a replacement for DES, is now patent-free.
RC4 (Rivest Cipher 4) aka ARC4 or ARCFOUR	Stream cipher with byte-oriented operations, based on the use of a random permutation. Key size is variable. Patented by RSA Security.
RC5 (Rivest Cipher 5)	Fast, parameterized block cipher with a 32-bit, 64-bit, or 128-bit block size. Number of rounds can be up to 255, and key size can be up to 2040 bits. Uses two 2-bit registers. Patented by RSA Security.
RC6 (Rivest Cipher 6)	Block cipher derived from RC5, with the improvement of integer multiplication and four 4-bit registers. Patented by RSA Security.
SEAL (Software-optimized Encryption Algorithm)	Stream cipher optimized for 32-bit machines. Patented by IBM.
Blowfish	16-round Feistel block cipher with a 64-bit block size. Key size can be from 32 to 448 bits, and is expanded into a set of subkeys 4168 bytes long. Designed by Bruce Schneier as a replacement for DES or IDEA.
Twofish	Feistel block cipher with a 128-bit block size. Highly configurable. Designed by Bruce Schneier et al.
Asymmetric ciphers	
DSA (Digital Signature Algorithm)	NIST standard used in the Digital Signature Standard (DSS). Generates a 320-bit digital signature. Based on the algebraic properties of modular exponentiation and the discrete logarithm problem.
RSA (Rivest-Shamir-Adleman)	Cryptosystem for encryption and authentication. Based on modular arithmetic and large prime numbers.
DH (Diffie-Hellman)	Cryptographic protocol for securely establishing a shared secret key over an insecure channel. Based on modular arithmetic, exponentiation, and large prime numbers. Strength and algorithm of key exchange are defined by groups: - group 1 (768 bit) - group 2 (1024 bit) - group 5 (1536 bit) - group 14 (2048 bit) - group 19 (256 bit, elliptic curve) - group 20 (384 bit, elliptic curve)

Hash functions	
MD2 **(Message Digest v2)**	Takes in input a message which is a multiple of 512 bits (if not, padding is used) and has a maximum length of 2^{64}-1 bits, and generates a 128-bit hash. Supports 8-bit machines (i.e. word size is 8 bits). Broken, but still used in digital certificates together with RSA.
MD4 **(Message Digest v4)**	Same properties as MD2. Supports 32-bit machines. Proven severely insecure, thus obsolete.
MD5 **(Message Digest v5)**	Same properties as MD2. Supports 32-bit machines. This is the hash function of the MD family currently in use.
SHA-0 **(Secure Hash Algorithm v0)**	Original version of the SHA hash function. Vulnerable, thus not in use anymore.
SHA-1 **(Secure Hash Algorithm v1)**	Takes in input a message with a maximum length of 2^{64}-1 bits, and generates a 160-bit hash. Vulnerable and therefore no longer approved for cryptographic use.
SHA-2 **(Secure Hash Algorithm v2)**	Family of hash functions: **SHA-256** (for 32-bit machines, generates a 256-bit hash) **SHA-224** (for 32-bit machines, generates a 224-bit hash, truncated version) **SHA-512** (for 64-bit machines, generates a 512-bit hash) **SHA-384** (for 64-bit machines, generates a 384-bit hash, truncated version) Max input message length is 2^{128}-1 bits.
SHA-3 **(Secure Hash Algorithm v3)**	Family of hash functions: **SHA3-256** (for 32-bit machines, generates a 256-bit hash) **SHA3-224** (for 32-bit machines, generates a 224-bit hash, truncated version) **SHA3-512** (for 64-bit machines, generates a 512-bit hash) **SHA3-384** (for 64-bit machines, generates a 384-bit hash, truncated version) **SHAKE128** (generates a hash of arbitrary length) **SHAKE256** (generates a hash of arbitrary length) Max input message length is unlimited.
RIPEMD **(RACE Integrity Primitives Evaluation Message Digest)**	Family of hash functions: **RIPEMD** (generates a 128-bit hash; insecure) **RIPEMD-128** (generates a 128-bit hash; insecure) **RIPEMD-160** (generates a 160-bit hash; most commonly used) **RIPEMD-256** (generates a 256-bit hash) **RIPEMD-320** (generates a 320-bit hash)

Authentication systems	
HMAC **(Hash-based Message Authentication Code)**	Message Authentication Code used to verify data integrity and sender authentication. Uses a hash function in conjunction with a secret key.
PAP **(Password Authentication Protocol)**	Password-based authentication protocol used by Point-to-Point Protocol (PPP) to authenticate remote clients. Uses a weak authentication scheme, vulnerable to attacks; passwords are transmitted in plaintext over the network. For this reason, is not recommended and it is advised to use CHAP or EAP instead.
CHAP **(Challenge-Handshake Authentication Protocol)**	Authentication protocol used by Point-to-Point Protocol (PPP) to authenticate remote clients. Client identity is verified via a three-way handshake. It uses an incrementally changing identifier and a variable challenge value in order to thwart replay attacks.
EAP **(Extensible Authentication Protocol)**	Authentication framework able to use different authentication systems (passwords, smart tokens, one-time passwords, Secure ID cards, digital certificates, public key cryptography protocols, etc.) via a challenge-response mechanism.
LEAP **(Lightweight Extensible Authentication Protocol)**	Cisco proprietary version of EAP, used for WEP. Uses either the MS-CHAP or the EAP-FAST authentication protocol. Vulnerable and not recommended.
PEAP **(Protected Extensible Authentication Protocol)**	TLS-encapsulated secured version of EAP, used in WPA2.

GPG (GNU Privacy Guard) aka **GnuPG** is a well-known implementation of the OpenPGP standard described in RFC 4880.

The OpenPGP standard derives from **PGP (Pretty Good Privacy)**, the first tool for strong encryption available to the public. It specifies a suite of algorithms: ElGamal, DSA, Triple DES, SHA-1, RSA, AES-128, CAST-128, IDEA, Camellia, ECC (Elliptic Curve Cryptography) i.e. ECDSA and ECDH.

gpg --gen-key	Generate a key pair
gpg --import *alice.asc*	Import Alice's public key *alice.asc* into your keyring
gpg --list-keys	List the keys contained into your keyring
gpg --list-secret-keys	List your private keys contained into your keyring
gpg --list-public-keys	List the public keys contained into your keyring
gpg --export -o *keyring.gpg*	Export your whole keyring to a file *keyring.gpg*
gpg --export-secret-key -a "You" -o *private.key*	Export your private key to a file *private.key*
gpg --export-public-key -a "Alice" -o *alice.pub*	Export Alice's public key to a file *alice.pub*
gpg --edit-key "Alice"	Sign Alice's public key
gpg -e -u "You" -r "Alice" *file*	Sign *file* (with your private key) and encrypt it to Alice (with Alice's public key)
gpg -d *file.gpg* -o *file*	Decrypt *file.gpg* (with your own private key) and save the decrypted file to *file*

LUKS (Linux Unified Key Setup) is a platform-independent specification for the encryption of a block device. It uses **dm-crypt**, a transparent disk encryption subsystem which is part of the device mapper, as a backend.
The `lsblk` command can be used to list devices and partitions and identify LUKS-encrypted ones.

`cryptsetup`	Frontend command for dm-crypt. Will prompt for a passphrase for most operations on a LUKS-encrypted device
`cryptsetup luksFormat` *device*	Initialize a LUKS partition, prompting for an encryption passphrase
`cryptsetup luksChangeKey` *device*	Change the passphrase of a LUKS partition
`cryptsetup luksAddKey` *device*	Add a new passphrase to a LUKS partition
`cryptsetup luksAddKey` *device keyfile*	Add a new keyfile to a LUKS partition
`cryptsetup luksRemoveKey` *device*	Remove a passphrase from a LUKS partition
`cryptsetup luksRemoveKey` *device keyfile*	Remove a keyfile from a LUKS partition
`cryptsetup luksKillSlot` *device keyslot*	Remove a key from a LUKS partition
`cryptsetup isLuks` *device*	Return true if the device is a LUKS partition
`cryptsetup luksDump` *device*	Dump the header information of a LUKS partition
`cryptsetup luksUUID` *device*	Print the UUID a LUKS partition
`cryptsetup luksOpen` *device name*	Open a LUKS device and set up a mapping name
`cryptsetup luksClose` *name*	Close a LUKS device and remove the mapping name
`cryptsetup luksSuspend` *name*	Suspend a LUKS device and wipe the encryption key from memory
`cryptsetup luksResume` *name*	Resume a suspended LUKS device
`cryptsetup luksHeaderBackup` *device* `--header-backup-file` *file*	Backup header and keyslot areas of a LUKS device to a file
`cryptsetup luksHeaderRestore` *device* `--header-backup-file` *file*	Restore header and keyslot areas of a LUKS device from a file

OpenVPN is an open source software that implements a Virtual Private Network (VPN) between two endpoints. The encrypted VPN tunnel uses UDP port 1194.

`openvpn --genkey --secret` *`keyfile`*	Generate a shared secret keyfile for OpenVPN authentication. The keyfile must be copied on both server and client
`openvpn server.conf`	Start the VPN on the server side
`openvpn client.conf`	Start the VPN on the client side

`/etc/openvpn/server.conf`

Server-side configuration file:

```
dev tun
ifconfig server_IP client_IP
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
secret keyfile
```

`/etc/openvpn/client.conf`

Client-side configuration file:

```
remote server_public_IP
dev tun
ifconfig client_IP server_IP
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
secret keyfile
```

`auditd` is the Linux Auditing System daemon, developed and maintained by Red Hat. It is used by SELinux to log events.

`auditctl`	Control and query the kernel audit system
`auditctl -a exit,always -S open -F path=`*`file`*	Audit processes trying to open *file*
`auditctl -a exit,always -S open -F auid=`*`uid`*	Audit files opened by the user with UID *uid*

`auditctl -w `*`file`*` -p waauditctl \`
`-a exit,always -F path=`*`file`*` -F perm=wa`
	Audit *file* for changes

`ausearch -f `*`file`*	Search the audited events matching *file*
`aureport`	Produce a summary report about audited events

`last`	Print the list of users that logged in and out. Searches through the file `/var/log/wtmp`
`lastb`	Print the list of bad login attempts. Searches through the file `/var/log/btmp`
`fail2ban`	Temporarily ban IP addresses (via firewall rules) that have too many failed password logins. This information is taken from authentication logs
`pam_tally2`	Deny access to users that have too many failed logins

`acct on`
`acct off`
	Turn process accounting on or off

`ac`	Print statistics about connect time of users
`lastcomm`	Print information about previously executed commands
`sa`	Print summarized information about previously executed commands

Key	Alternate key	Function
CTRL F	→	Move cursor forward one character
CTRL B	←	Move cursor backward one character
CTRL A	HOME	Move cursor to beginning of line
CTRL E	END	Move cursor to end of line
CTRL H	BACKSPACE	Delete character to the left of cursor
CTRL W		Delete word to the left of cursor
CTRL U		Delete all characters to the left of cursor
CTRL K		Delete all characters to the right of cursor
CTRL T		Swap current character with previous one
ESC T		Swap current word with previous one
SHIFT PAGE UP		Scroll up the screen buffer
SHIFT PAGE DOWN		Scroll down the screen buffer
CTRL L		Clear screen (same as `clear`)
CTRL P	↑	Previous command in history
CTRL N	↓	Next command in history
CTRL R		Reverse history search
ALT .		Get argument from previous command in history
CTRL I	TAB	Autocomplete commands, filenames, and directory names
ALT /		Autocomplete filenames and directory names only
CTRL ALT E		Expand the Bash alias currently entered on the command line
CTRL J	RETURN	Line feed
CTRL M		Carriage return
CTRL S		Pause transfer to terminal Forward history search (if XON/XOFF flow control is disabled)
CTRL Q		Resume transfer to terminal
CTRL Z		Send a SIGTSTP to put the current job in background
CTRL C		Send a SIGINT to stop the current process
CTRL D		Send an EOF to the current process (same as `logout` if process is a shell)
CTRL ALT DEL		Send a SIGINT to reboot the machine (same as `shutdown -r now`), as specified in `/etc/inittab` and `/etc/init/control-alt-delete`
CTRL ALT F1 ... F6		Switch between text consoles (same as `chvt` *n*)

Key	Alternate key	Function
CTRL ALT F7 ... F11		Switch between X Window consoles
CTRL ALT +		Increase X Window screen resolution
CTRL ALT -		Decrease X Window screen resolution
CTRL TAB		Switch between X Window tasks
CTRL ALT →	CTRL ALT ↓	Switch to next workspace
CTRL ALT ←	CTRL ALT ↑	Switch to previous workspace
CTRL ALT BACKSPACE		Reboot the X Window server
	GNOME	
ALT TAB		Switch between windows in the current workspace
SUPER		Show activities overview
SUPER L		Lock screen
SUPER M		Show tray messages
SUPER ↑		Maximize current window
SUPER ↓		Restore normal size of current window
SUPER ←		Maximize current window to left half screen
SUPER →		Maximize current window to right half screen
ALT F2		Run command
CTRL +		Increase terminal font size
CTRL -		Decrease terminal font size

The Hardware Abstraction Layer (HAL) manages device files and provides plug-and-play facilities. The HAL daemon hald maintains a persistent database of devices.
udev is the device manager for the Linux kernel. It dynamically generates the device nodes in /dev/ for devices present on the system; it also provides persistent naming for storage devices in /dev/disk.
When a device is added, removed, or changes state, the kernel sends an uevent received by the udevd daemon which will pass the uevent through a set of rules stored in /etc/udev/rules.d/*.rules and /lib/udev/rules.d/*.rules.

udevadm monitor udevmonitor	Show all kernel uevents and udev messages
udevadm info --attribute-walk --name=/dev/sda	Print all attributes of device /dev/sda in udev rules key format
cat /sys/block/sda/size	Print the size attribute of disk sda in 512-byte blocks. This information is retrieved from sysfs
udevadm test /dev/sdb	Simulate an udev event run for the device and print debug output
gnome-device-manager	Browser for the HAL device manager
lshal	Show items in the HAL device database

/etc/udev/rules.d/*.rules and /lib/udev/rules.d/*.rules	udev rules
KERNEL=="hda", NAME="mydisk"	Match a device which was named by the kernel as hda; name the device node as "mydisk". The device node will be therefore /dev/mydisk
KERNEL=="hdb", DRIVER=="ide-disk", SYMLINK+="mydisk myhd"	Match a device with kernel name and driver as specified; name the device node with the default name and create two symbolic links /dev/mydisk and /dev/myhd pointing to /dev/hdb
KERNEL=="fd*[0-9]**", NAME="floppy/%n", SYMLINK+="%k"	Match all floppy disk drives (i.e. fd*n*); place device node in /dev/floppy/*n* and create a symlink /dev/fd*n* to it
SUBSYSTEM=="block", ATTR{size}=="41943040", SYMLINK+="mydisk"	Match a block device with a size attribute of 41943040; create a symlink /dev/mydisk
KERNEL=="fd*[0-9]**", OWNER="jdoe"	Match all floppy disk drives; give ownership of the device file to user "jdoe"
KERNEL=="sda", PROGRAM="/bin/mydevicenamer %k", SYMLINK+="%c"	Match a device named by the kernel as sda; to name the device, use the defined program which takes on stdin the kernel name and output on stdout e.g. *name1 name2*. Create symlinks /dev/*name1* and /dev/*name2* pointing to /dev/sda
KERNEL=="sda", ACTION=="add", RUN+="/bin/myprogram"	Match a device named by the kernel as sda; run the defined program when the device is connected
KERNEL=="sda", ACTION=="remove", RUN+="/bin/myprogram"	Match a device named by the kernel as sda; run the defined program when the device is disconnected

%n = kernel number (e.g. = 3 for fd3)
%k = kernel name (e.g. = fd3 for fd3)
%c = device name as output from program

A kernel version number has the form *major.minor.patchlevel*.
Kernel images are usually gzip-compressed and can be of two types: zImage (max 520 Kb) and bzImage (no size limit).
Kernel modules can be loaded dynamically into the kernel to provide additional functionalities on demand, instead of being included when the kernel is compiled; this reduces memory footprint.
`kerneld` (daemon) and `kmod` (kernel thread) facilitate the dynamic loading of kernel modules.

`/lib/modules/`*`X.Y.Z`*`/*.ko`	Kernel modules for kernel version *X.Y.Z*
`/lib/modules/`*`X.Y.Z`*`/modules.dep`	Modules dependencies. This file needs to be recreated (via the command `depmod -a`) after a reboot or a change in module dependencies
`/etc/modules.conf` `/etc/conf.modules` (deprecated)	Modules configuration file
`/usr/src/linux/`	Directory containing the kernel source code to be compiled
`/usr/src/linux/.config`	Kernel configuration file
`/etc/initramfs-tools/initramfs.conf` (Debian)	Configuration file for the `initrd` image file
`freeramdisk`	Free the memory used for the `initrd` image. This command must be run directly after unmounting `/initrd`
`mkinitrd` *`initrd_image kernel_version`* (Red Hat) `mkinitramfs` (Debian)	Create an `initrd` image file
`dracut`	Create initial ramdisk images for preloading modules
`lsinitramfs`	Show the contents of an `initramfs` image
`dbus-monitor`	Monitor messages going through a D-Bus message bus
`dbus-monitor --session`	Monitor session messages (default)
`dbus-monitor --system`	Monitor system messages

`kexec -l` *`kernel_image`* `--append=`*`options`* `\`
`--initrd=`*`initrd_image`* `&& kexec -e`
	Load a kernel image file into memory and boot it. This allows running a different kernel without rebooting the machine

The runtime loader `ld.so` loads the required shared libraries of the program into RAM, searching in this order:

1.	`LD_LIBRARY_PATH`	Environment variable specifying the list of dirs where libraries should be searched for first
2.	`/etc/ld.so.cache`	Cache file
3.	`/lib` and `/usr/lib`	Default locations for shared libraries

Shared library locations (other than the default ones `/lib` and `/usr/lib`) can be specified in the file `/etc/ld.so.conf`.

`ldconfig`	Create a cache file `/etc/ld.so.cache` of all available dynamically linked libraries. This command should be run when the system complains about missing libraries
`ldd` *`program_or_lib`*	Print library dependencies

`lspci`	List PCI devices
`lspci -d 8086:`	List all Intel hardware present. PCI IDs are stored in: `/usr/share/hwdata/pci.ids` (Red Hat) `/usr/share/misc/pci.ids` (Debian)
`lsusb`	List USB devices
`lsusb -d 8086:`	List all Intel USB devices present. USB IDs are stored in: `/usr/share/hwdata/usb.ids` (Red Hat) `/var/lib/usbutils/usb.ids` (Debian)
`lspcmcia`	List PCMCIA devices
`lsdev`	List information about the system hardware
`lshw`	List system hardware
`lscpu`	List information about the CPU architecture
`uname`	Print system information. Options to show the relevant piece of information are: `-s` Kernel name `-n` Network node hostname `-r` Kernel release number *X.Y.Z* `-v` Kernel version number `-m` Machine hardware name `-p` Processor type `-i` Hardware platform `-o` Operating system `-a` All the above information, in that order
`evtest`	Monitor and query input device events in /dev/input/event*n*
`dmesg`	Print the messages of the kernel ring buffer. Each entry is prepended by a timestamp showing the number of seconds since the machine booted up. Options are: `-T` Print human-readable timestamps `-n 1` Set the logging level to 1 (i.e. only panic messages)

`journalctl`	Display the Systemd journal, which contains the kernel logs
`journalctl -n` *n*	Display the most recent *n* log lines (default is 10)
`journalctl --since "1 hour ago"`	Display events happened in the last hour
`journalctl -x`	Display events, adding explanations from the message catalog
`journalctl -f`	Display the journal in real-time

`journalctl -u crond.service`
`journalctl _SYSTEMD_UNIT=crond.service`
	Display the log entries created by the cron service

`mkdir -p /var/log/journal/ && \`
`systemctl restart systemd-journald`
	Enable persistent storage of logs in /var/log/journal/
	(by default, journalctl stores the logfiles in RAM only)

`journalctl --rotate && \`
`journalctl --vacuum-time=1s`
	Remove all current journal entries

Kernel compile and patching

Kernel compile		
Download	Download the kernel source code `linux-X.Y.Z.tar.bz2` from `http://www.kernel.org` to the base of the kernel source tree `/usr/src/linux`	
Clean	`make clean`	Delete most generated files
	`make mrproper`	Delete all generated files and kernel configuration
	`make distclean`	Delete temporary files, patch leftovers, and similar files
Configure	`make config`	Create configuration (terminal-based; options must be set in sequence)
	`make menuconfig`	Create configuration (ncurses UI)

	`make xconfig`
	`make gconfig`
		Create configuration (GUI)

	`make oldconfig`	Create a new configuration file, based on the options in the old configuration file and in the source code
	Components (e.g. device drivers) can be either: - not compiled - compiled into the kernel binary, for support of devices always used on the system or necessary for the system to boot - compiled as a kernel module, for optional devices The configuration command creates a configuration file `/usr/src/linux/.config` containing instructions for the kernel compilation	
Build	`make bzImage`	Compile the kernel
	`make modules`	Compile the kernel modules
	`make all`	Compile kernel and kernel modules
	`make -j2 all` will speed up compilation by allocating 2 simultaneous compile jobs	
Modules install	`make modules_install`	Install the previously built modules present in `/lib/modules/X.Y.Z`
Kernel install	`make install`	Install the kernel automatically
	To install the kernel by hand: 1. Copy the new compiled kernel and other files into the boot partition: `cp /usr/src/linux/arch/boot/bzImage /boot/vmlinuz-X.Y.Z` (kernel) `cp /usr/src/linux/arch/boot/System.map-X.Y.Z /boot` `cp /usr/src/linux/arch/boot/config-X.Y.Z /boot` (config options used for this compile) 2. Create an entry in GRUB to boot on the new kernel	
Package	Optionally, the kernel can be packaged for install on other machines	
	`make rpm-pkg`	Build source and binary RPM packages
	`make binrpm-pkg`	Build binary RPM package
	`make deb-pkg`	Builds binary DEB package

Kernel patching		
Download	Download and decompress the patch to `/usr/src`	
Patch	`patch -p1 < file.patch`	Apply the patch
	`patch -Rp1 < file.patch`	Remove (reverse) a patch. Alternatively, applying the patch again reverses it
Build	Build the patched kernel as explained above	
Install	Install the patched kernel as explained above	

Kernel modules allow the kernel to access functions (symbols) for kernel services e.g. hardware drivers, network stack, or filesystem abstraction.

`lsmod`	List the modules that are currently loaded into the kernel
`insmod` *`module`*	Insert a module into the kernel. If the module requires another module or if it does not detect compatible hardware, insertion will fail. It is better to use `modprobe` instead
`rmmod` *`module`*	Remove a module from the kernel. If the module is in use by another module, it is necessary to remove the latter module first. It is better to use `modprobe -r` instead
`modinfo` *`module`*	Display the list of parameters accepted by the module
`depmod -a`	Probe all modules in the kernel modules directory and generate the file that lists their dependencies
`modprobe` *`module option=value`*	Insert a module into the running kernel, with the specified parameters. Prerequisite modules will be inserted automatically. It is recommended to use `modprobe` instead of `insmod` and `rmmod`, because it automatically handles prerequisites when inserting modules, is more specific about errors, and accepts just the module name alone instead of requiring the full path
`modprobe -a`	Insert all modules
`modprobe -t` *`directory`*	Attempt to load all modules contained in the directory until a module succeeds. This action probes the hardware by successive module-insertion attempts for a single type of hardware, e.g. a network adapter
`modprobe -r` *`module`*	Remove a module
`modprobe -c` *`module`*	Display module configuration
`modprobe -l`	List loaded modules

Configuration of device drivers		
Device drivers support the kernel with instructions on how to use that device.		
Device driver compiled into the kernel	Configure the device driver by passing a kernel parameter in the GRUB menu: `kernel /vmlinuz ro root=/dev/vg0/root vga=0x33c`	
Device driver provided as a kernel module	Edit module configuration in `/etc/modprobe.conf` or `/etc/modprobe.d/` (Red Hat):	
	`alias eth0 3c59x`	Specify that eth0 uses the `3c59x.ko` driver module
	`options 3c509 irq=10,11`	Assign IRQ 10 and 11 to 3c509 devices

/proc

/proc is a pseudo filesystem that gives access to process data held in the kernel.

File	Information stored (can be viewed via cat)	Equivalent command
/proc/bus	Buses (e.g. PCI, USB, PC Card)	
/proc/cpuinfo	CPUs information	
/proc/devices	Drivers currently loaded	
/proc/dma	DMA channels in use	
/proc/filesystems	Filesystems supported by the system	
/proc/interrupts	Current IRQs (Interrupt Requests)	procinfo
/proc/ioports	I/O addresses in use	
/proc/kcore	Memory allocatable by the kernel	
/proc/loadavg	System load averages	uptime
/proc/mdstat	Information about RAID arrays and devices	
/proc/meminfo	Total and free memory	free
/proc/modules	Kernel modules currently loaded	lsmod
/proc/mounts	Mounted partitions	mount
/proc/net/dev	Network interface statistics	
/proc/partitions	Drive partition information	fdisk -l
/proc/swaps	Size of total and used swap areas	swapon -s
/proc/sys/	sysfs: exposes tunable kernel parameters	
/proc/sys/kernel/	Kernel information and parameters	
/proc/sys/net/	Network information and parameters	
/proc/uptime	Time elapsed since boot	uptime
/proc/version	Linux version	uname -a
/proc/*n*/	Information about process with PID *n*	ps *n*
/proc/*n*/cmdline	Command by which the process was launched	
/proc/*n*/cwd	Symlink to process' working directory	
/proc/*n*/environ	Values of environment variables of process	
/proc/*n*/exe	Symlink to process' executable	
/proc/*n*/fd	Files currently opened by the process	lsof -p *n*
/proc/*n*/root	Symlink to process' filesystem root	
/proc/*n*/status	Status of process	

/proc/sys is the only writable branch of /proc and can be used to tune kernel parameters on the fly.
All changes are lost after system shutdown, unless applied via sysctl -p.

`sysctl fs.file-max`
`cat /proc/sys/fs/file-max`
	Get the maximum allowed number of open files

`sysctl -w "fs.file-max=100000"`
`echo "100000" > /proc/sys/fs/file-max`
	Set the maximum allowed number of open files to 100000

`sysctl -a`	List all available kernel tuning options
`sysctl -p`	Apply all tuning settings listed in /etc/sysctl.conf. This command is usually run at boot by the system initialization script, to make permanent changes to kernel parameters

/dev - storage devices

/dev contains the device files to access all devices in the system.

File	Device
/dev/sda	SCSI, PATA, or SATA hard drive
/dev/hda	IDE hard drive
/dev/pda	Parallel port IDE hard drive
/dev/vda	Virtual disk for KVM-based virtual machines
/dev/sda, /dev/sdb, /dev/sdc ...	First, second, third ... hard drive
/dev/sda1, /dev/sda2, /dev/sda3 ...	First, second, third ... partition of the first hard drive
/dev/md0	Metadisk group, for use with RAID
/dev/sr0	SCSI CD-ROM
/dev/pcd0	Parallel port CD-ROM
/dev/cdrom	CD-ROM. Usually symlinked to /dev/sr0
/dev/fd0	Floppy disk drive
/dev/ht0	IDE tape drive
/dev/pt0	Parallel port tape drive
/dev/sg0	Generic SCSI device
/dev/loop0	Loopback pseudo device. Makes a file accessible as a block device, hence allowing a file containing an entire filesystem to be mounted as if it were a disk device
/dev/autofs	AutoFS device
/dev/fuse	FUSE device
/dev/shm	Shared memory device (tmpfs). Can be used like /tmp to store temporary files, but is bound by the amount of RAM in the system

File	Device
/dev/dsp	Digital Signal Processor device. Interfaces with the soundcard
/dev/fb0	Framebuffer device. Interfaces with the graphics hardware
/dev/lp0	Parallel port printer device
/dev/parport0	Raw parallel port device
/dev/mem	Physical memory
/dev/kmem	Kernel virtual memory
/dev/core	Obsolete. Symlink to /proc/kcore
/dev/stdin	Standard Input
/dev/stdout	Standard Output
/dev/stderr	Standard Error
/dev/null	Null device, aka blackhole or bit bucket. Discards any received data
/dev/zero	Zero device. Outputs an infinite stream of zero bytes (NUL) on reads
/dev/full	"Always full" device. Similar to /dev/zero, and also returns an error "No space left on device" (ENOSPC) on writes
/dev/random	Non-deterministic random number generator. Gathers entropy from the system to generate randomness; once the entropy pool is depleted, the device blocks all reads until it can collect more entropy
/dev/urandom	Unlimited pseudo random number generator. Faster but unsafe for cryptographic purposes
/dev/console	System console
/dev/tty	Terminal for current process
/dev/tty0	Current virtual console
/dev/ttyS0	Serial port, usually used for modem connections
/dev/ptyp0	Pseudo-TTY master
/dev/ttyp0	Pseudo-TTY slave

If the kernel has booted in emergency mode and `init` has not run, some initial configuration is necessary e.g.

```
mount /proc
mount -o remount,rw /
mount -a
```

If mounting the filesystems fails:

```
mknod /dev/sda
mknod /dev/sda1
fdisk -l /dev/sda
fsck -y /dev/sda1
mount -t ext3 /dev/sda1 /mnt/sysimage
chroot /mnt/sysimage
```

To install a package using an alternative root directory (useful if the system has been booted from a removable media):

```
rpm -U --root /mnt/sysimage package.rpm
```

To install GRUB on the specified directory (which must contain `/boot/grub/`):

```
grub-install --root-directory=/mnt/sysimage /dev/sda
```

Alternative method:

```
chroot /mnt/sysimage && grub-install /dev/sda
```

Run `sync` and unmount all filesystems before exiting the shell, to ensure that all changes have been written on disk.

How to reset the root password (RHEL 7 and 8)

1. Power up the system and, once on the GRUB 2 boot screen, press E to edit the current entry
2. On the kernel line that mentions `linux16`, remove the `rhgb` and `quiet` parameters and add `rd.break` at the end
3. Press CTRL X; the system will boot on the initramfs `switch_root` prompt
4. Remount the filesystem as writable `mount -o remount,rw /sysroot`
5. Change the filesystem root `chroot /sysroot`
6. Modify the root password `passwd root`
7. Force SELinux to relabel context on next boot `touch /.autorelabel`
8. Remount the filesystem as read-only (not strictly necessary) `mount -o remount,ro /sysroot`
9. Exit the chroot environment `exit`
10. Resume system boot `exit`

System recovery - chmod

If the executable permission has been removed from the `chmod` command binary by mistake, any of the following procedures allows to restore it.

Copy attributes and permissions from another command binary (preserving ownership and timestamps):

```
cp --attributes-only -p /usr/bin/true /usr/bin/chmod
```

Copy the contents of `chmod` to another command binary via `cat`:

```
cp /usr/bin/true /usr/bin/true.bak
cat /usr/bin/chmod > /usr/bin/true
mv /usr/bin/true /usr/bin/chmod
mv /usr/bin/true.bak /usr/bin/true
```

Add temporarily an ACL via `setfacl`, set the executable permission, then remove the ACL:

```
setfacl -m u::rx /usr/bin/chmod
chmod +x /usr/bin/chmod
setfacl -b /usr/bin/chmod
```

Copy the binary and set permissions via `rsync`:

```
rsync /usr/bin/chmod /usr/bin/chmod2 --chmod=ugo+x
mv /usr/bin/chmod2 /usr/bin/chmod
```

Run `chmod` via the `ld` linker:

```
/usr/lib64/ld-linux-x86-64.so.2 /usr/bin/chmod +x /usr/bin/chmod   (on 64-bit systems)
/usr/lib/ld-linux.so /usr/bin/chmod +x /usr/bin/chmod              (on 32-bit systems)
```

Run the `busybox` version of `chmod`:

```
busybox chmod +x /usr/bin/chmod
```

Use the command interpreter of a programming language:

```
perl -e 'chmod 0755, "/usr/bin/chmod"'                  (via Perl)
python -c "import os; os.chmod('/usr/bin/chmod',0755)"  (via Python)
```

Domain Name System (DNS) is a decentralized hierarchical naming system, mostly used to resolve domain names to IP addresses. It uses TCP and UDP port 53.

DNS implementations	
BIND	Berkeley Internet Name Domain system, is the standard DNS server for UNIX
Unbound	Standard DNS server in RHEL 7
dnsmasq	Lightweight DNS, DHCP and TFTP server for a small network
djbdns	Security-hardened DNS server that also includes DNS debugging tools
PowerDNS	Alternative open-source DNS server

`named`	BIND Name Daemon
`ndc`	Name Daemon Controller for BIND 8
`rndc`	Remote Name Daemon Controller for BIND 9, uses a shared key to communicate securely with `named`

`named -u named -g named`	Run BIND as user/group "named" (must be created if needed) instead of root
`named -t /var/cache/bind`	Run BIND in a chroot jail `/var/cache/bind` (it is actually the `chroot` command that starts the `named` server)
`dnswalk` *`example.org.`*	DNS debugger
`rndc reconfig`	Reload BIND configuration and new zones
`rndc reload` *`example.org`*	Reload the zone *example.org*
`rndc freeze` *`example.org`*	Suspend updates for the zone *example.org*
`rndc thaw` *`example.org`*	Resume updates for the zone *example.org*
`rndc tsig-list`	List all currently active TSIG keys

DNSSEC was designed to secure the DNS tree and hence prevent cache poisoning.
The TSIG (Transaction SIGnature) standard, which authenticates communications between two trusted systems, is used to sign zone transfers and DDNS (Dynamic DNS) updates.

```
dnssec-keygen -a dsa -b 1024 \
-n HOST dns1.example.org
```

Generate a TSIG key with DNSSEC algorithm *nnn* and key fingerprint *fffff*.
This will create two key files
`Kdns1.example.org.+nnn+fffff.key`
`Kdns1.example.org.+nnn+fffff.private`
which contain a key number that must be inserted both in `/etc/named.conf` and `/etc/rndc.conf`

```
rndc-confgen -a
```

Generate a `/etc/rndc.key` key file:

```
key "rndc-key" {
   algorithm hmac-md5;
   secret "vyZqL3tPHsqnA57e4LT0Ek==";
};
options {
   default-key "rndc-key";
   default-server 127.0.0.1;
   default-port 953;
};
```

This file is automatically read both by `named` and `rndc`

```
dnssec-signzone example.org
```

Sign the zone *example.org*

/etc/named.conf DNS server configuration file

```
controls {
   inet 127.0.0.1 allow {localhost;} keys {rndckey;};
};
key "rndc-key" {                                 // TSIG key
   algorithm dsa;
   secret "HYZur46fftdUQ43BJKI093t4t78lkp";
};

acl "mynetwork" {10.7.0.0/24;};                 // Alias definition
                                                // Built-in ACLs: any, none, localhost, localnets

options {
   directory "/var/named";                      // Working directory
   version "0.0";                               // Hide version number by replacing it with 0.0
   listen-on port 53 {10.7.0.1; 127.0.0.1;};   // Port and own IP addresses to listen on
   blackhole {172.17.17.0/24;};                 // IPs whose packets are to be ignored
   allow-query {mynetwork;};                    // IPs allowed to make iterative queries
   allow-query-on {any;};                       // Local IPs that can accept iterative queries
   allow-query-cache {any;};                    // IPs that can get an answer from cache
   allow-recursion {mynetwork;};       // IPs to accept recursive queries from (typically the
                                       // own network's IPs).  The DNS server does the full
                                       // resolution process on behalf of these client IPs,
                                       // and returns a referral for the other IPs
   allow-recursion-on {mynetwork;};    // Local IPs that can accept recursive queries
   allow-transfer {10.7.0.254;};       // Zone transfer is restricted to these IPs (slaves);
                                       // on slave servers, this option should be disabled
   allow-update {any;};                // IPs to accept DDNS updates from
   recursive-clients 1000;             // Max number of simultaneous recursive lookups
   dnssec-enable yes;                  // Enable DNSSEC
   dialup no;                          // Not a dialup connection: external zone maintenance
                                       // (e.g. sending heartbeat packets, external zone transfers)
                                       // is then permitted
   forward first;                               // Site-wide cache: bypass the normal resolution
   forwarders {10.7.0.252; 10.7.0.253;};        // method by querying first these central DNS
                                                // servers if they are available
};

// Define the root name servers
zone "." {
   type hint;
   file "root.cache";
}

// Configure system to act as a master server for the example.org domain
zone "example.org" IN {
   type master;
   file "master/example.org.zone";      // Zone file for the example.org domain
};
zone "240.123.224.in-addr.arpa" IN {    // Configure reverse lookup zone (for 224.123.240.0/24)
   type master;
   file "slave/example.org.revzone";
};

// Configure system to act as a slave server for the example2.org domain
zone "example2.org" IN {
   type slave;
   file "slave/example2.org.zone";      // Slave: do not edit this zone file!
   masters {10.7.0.254;};
};
zone "0.7.10.in-addr.arpa" IN {         // Configure reverse lookup zone (for 10.7.0.0/24)
   type slave;
   file "slave/10.7.0.revzone";
   masters {10.7.0.254;};
};
```

/var/named/master/example.org.zone **DNS zone file for the example.org zone**

```
$TTL 86400      ; TTL (1 day)
$ORIGIN example.org.
example.org IN SOA dns1.example.org. help.example.org. (   ; Master DNS server is dns1.example.org
   2014052300   ; serial                                   ; Contact help@example.org if problems
   28800        ; refresh (8 hours)
   7200         ; retry (2 hours)
   604800       ; expire (1 week)
   600 )        ; negative TTL (10 mins)

        IN NS     dns1.example.org.
        IN NS     dns2.example.org.
        IN MX     10 mail1.example.org.
        IN MX     20 mail2.example.org.

dns1    IN A      224.123.240.3
dns2    IN A      224.123.240.4
mail1   IN A      224.123.240.73
mail2   IN A      224.123.240.77
foo     IN A      224.123.240.12
bar     IN A      224.123.240.13
www     IN A      224.123.240.19
baz     IN CNAME  bar

_sip._tcp.example.org.   IN SRV   10 50 5060 224.123.240.166
_sip._tcp.example.org.   IN SRV   10 30 5060 224.123.240.167
_sip._tcp.example.org.   IN SRV   20  0 5060 224.123.240.169

subdomain   IN NS   ns1.subdomain.example.org.   ; Glue records
            IN NS   ns2.subdomain.example.org.
ns1.subdomain.example.org.   IN A   224.123.240.201
ns2.subdomain.example.org.   IN A   224.123.240.202
```

/var/named/master/example.org.revzone **DNS reverse zone file for the example.org zone**

```
$TTL 86400      ; TTL (1 day)
example.org IN SOA dns1.example.org. help.example.org. (
   2014052300   ; serial
   28800        ; refresh (8 hours)
   7200         ; retry (2 hours)
   604800       ; expire (1 week)
   600 )        ; negative TTL (10 mins)

12.240.123.224.in-addr.arpa   IN PTR   foo
13.240.123.224.in-addr.arpa   IN PTR   bar
19.240.123.224.in-addr.arpa   IN PTR   www
```

DNS Resource Records		
	$TTL	How long to cache a positive response
	$ORIGIN	Suffix appended to all names not ending with a dot. Useful when defining multiple subdomains inside the same zone
SOA	Start Of Authority for the example.org zone	
	serial	Serial number. Must be increased after each edit of the zone file
	refresh	How frequently a slave server refreshes its copy of zone data from the master
	retry	How frequently a slave server retries connecting to the master
	expire	How long a slave server relies on its copy of zone data. After this time period expires, the slave server is not authoritative anymore for the zone unless it can contact a master
	negative TTL	How long to cache a non-existent answer
A	Address: maps names to IPv4 addresses. Used for DNS lookups.	
AAAA	IPv6 address: maps names to IPv6 addresses. Used for DNS lookups.	
PTR	Pointer: maps IP addresses to names. Used for reverse DNS lookups. Each A record must have a matching PTR record	
CNAME	Canonical Name: specifies an alias for a host with an A record (even in a different zone). Discouraged as it causes multiple lookups; it is better to use multiple A records instead	
NS	Name Service: specifies the authoritative name servers for the zone	
MX	Mailserver: specifies address and priority of the servers able to handle mail for the zone	
SRV	Service: specifies address and port number of the host providing a specific service. It is indicated as *`_service._protocol.domain.`*, where *protocol* is TCP or UDP	
Glue Records are not really part of the zone; they delegate authority for other zones, usually subdomains		

Most common HTTP response codes		
1XX Informational	**100 Continue**	The server received the request headers, so the client should continue by sending the remainder of the request
	101 Switching Protocols	The server agreed to switch protocol upon client's demand
	102 Processing	The server received the request and is processing it, but response is not yet available. Used for WebDAV requests which may contain many subrequests requiring a long time to complete; this prevents client timeout
2XX Success	**200 OK**	The request was successful
	201 Created	The request was successful, and resulted in a resource being created
	204 No Content	The request was successful, and the server does not need to return any content
	206 Partial Content	The request was successful, and the server is returning only partial content because the client sent a Range header field
3XX Redirection	**301 Moved Permanently**	The requested resource was permanently moved to a new URI
	302 Found	The requested resource was temporarily moved to a new URI
	303 See Other	The requested resource can be found on another URI, and should be retrieved from there via a GET
	304 Not Modified	The client sent a conditional GET request, and the resource has not been modified since last time it was requested
	307 Temporary Redirect	The requested resource was temporarily moved to a new URI, but future requests should use the original URI
4XX Client Error	**400 Bad Request**	The server was unable to understand the request due to bad syntax
	401 Unauthorized	The request requires user authentication
	403 Forbidden	The client did not have the necessary permissions to access the requested resource
	404 Not Found	The requested resource was not found on the server
	408 Request Timeout	The server timed out while waiting for the request
	409 Conflict	The request could not be processed because of a conflict in the resource state
	410 Gone	The requested resource is no longer available on the server and will not be available again
	451 Unavailable for Legal Reasons	The requested resource is not available due to government censorship
5XX Server Error	**500 Internal Server Error**	The server encountered a generic error while trying to fulfill the request
	501 Not Implemented	The server was unable to recognize the request method
	502 Bad Gateway	The server is acting as a gateway or proxy, and received an invalid response from the upstream server
	503 Service Unavailable	The server is temporarily unavailable due to overload or maintenance
	504 Gateway Timeout	The server is acting as a gateway or proxy, and a request to the upstream server timed out
	505 HTTP Version Not Supported	The server does not support the HTTP protocol version used in the request

Apache is an open source and widespread HTTP server, originally based on the NCSA HTTPd server.

`/etc/httpd/conf/httpd.conf` `/etc/httpd/conf.d/*.conf`	(Red Hat)	Apache configuration files
`/etc/apache2/httpd.conf`	(Debian and SUSE)	

`/var/www/html`	Default document root directory
`$HOME/public_html`	Default document root directory for users' websites

Web content must be readable by the user/group the Apache process runs as. For security reasons, it should be owned and writable by the superuser or the webmaster user/group (usually `www-data`), not the Apache user/group.

`apachectl`	(Red Hat)	Manage the Apache webserver
`httpd`	(Red Hat)	
`apache2ctl`	(Debian)	

`apachectl start`	Start the Apache webserver daemon
`apachectl status`	Display a brief status report
`apachectl fullstatus`	Display a detailed status report
`apachectl graceful`	Gracefully restart Apache; currently open connections are not aborted
`apachectl graceful-stop`	Gracefully stop Apache; currently open connections are not aborted

`apachectl configtest`
`apachectl -t`
	Test the configuration file, reporting any syntax error

`apachectl -M`	List all loaded and shared modules

The Apache webserver contains a number of MPMs (Multi-Processing Modules) which can operate following two methods:

prefork MPM	A number of child processes is spawned in advance, with each child serving one connection. Highly reliable due to Linux memory protection that isolates each child process.
worker MPM	Multiple child processes spawn multiple threads, with each thread serving one connection. More scalable but prone to deadlocks if third-party non-threadsafe modules are loaded.

HTTPS

HTTPS (i.e. HTTP over SSL/TLS) allows securing communications between the webserver and the client by encrypting all communications end-to-end between the two. A webserver using HTTPS hands over its public key to the client when the client connects to the server via port 443. The server's public key is signed by a CA (Certification Authority), whose validity is ensured by the root certificates stored into the client's browser.

The commands of the OpenSSL cryptographic library (`openssl`, `CA.pl`, and `genkey`) can be used to accomplish all public key cryptography operations e.g. generate key pairs, Certificate Signing Requests, and self-signed certificates.

Virtual hosting with HTTPS requires assigning a unique IP address for each virtual host; this because the SSL handshake (during which the server sends its certificate to the client's browser) takes place before the client sends the `Host:` header (which tells to which virtual host the client wants to talk).
A workaround for this is SNI (Server Name Indication) which makes the browser send the hostname in the first message of the SSL handshake. Another workaround is to have all multiple name-based virtual hosts use the same SSL certificate with a wildcard domain e.g. *.example.org.

Apache configuration file	
Server configuration directives	
`ServerName www.mysite.org:80`	Name and port (if omitted, uses default HTTP port 80) of server
`ServerRoot /etc/httpd`	Root directory for configuration and log files
`ServerAdmin webmaster@mysite.org`	Contact address that the server includes in any HTTP error messages to the client. Can be an email address or a URL
`StartServers 5`	Number of servers to start initially

`MinSpareServers 5`
`MaxSpareServers 10`
	Minimum and maximum number of idle child server processes

`MaxClients 256` (before v2.3.13) `MaxRequestWorkers 256` (v2.3.13 and later)	Max number of simultaneous requests that will be served; clients above this limit will get an HTTP error 503 - Service Unavailable. Prefork MPM: max number of child processes launched to serve requests. Worker MPM: max total number of threads available to serve requests
`ServerLimit 256`	Prefork MPM: max configured value for `MaxRequestWorkers`. Worker MPM: in conjunction with `ThreadLimit`, max configured value for `MaxRequestWorkers`
`ThreadsPerChild 25`	Worker MPM: number of threads created by each child process
`ThreadLimit 64`	Worker MPM: max configured value for `ThreadsPerChild`
`MaxRequestsPerChild 16` (v2.2) `MaxConnectionsPerChild 16` (v2.4)	Max number of connections allowed per child
`LoadModule mime_module modules/mod_mime.so`	Load the module `mime_module` by linking in the object file or library `modules/mod_mime.so`

`Listen 10.17.1.1:80`
`Listen 10.17.1.5:8080`
	Make the server accept connections on the specified IP addresses (optional) and ports

`User nobody`
`Group nobody`
	User and group the Apache process runs as. For security reasons, this should not be root

Apache configuration file	
Main configuration directives	
`DocumentRoot /var/www/html`	Directory in filesystem that maps to the root of the website
`Alias /image /mydir/pub/image`	Map the URL `http://www.mysite.org/image/` to the directory `/mydir/pub/image` in the filesystem. This allows Apache to serve content placed outside of the document root
`TypesConfig conf/mime.types`	Media types file. The path is relative to `ServerRoot`
`AddType image/jpeg jpeg jpg jpe`	Map the specified filename extensions onto the specified content type. These entries add to or override the entries from the media types file `conf/mime.types`
`Redirect permanent /foo /bar`	Redirect to a URL on the same host. Status can be: `permanent` Return an HTTP status "301 - Moved Permanently" `temp` Return an HTTP status "302 - Found" (default) `seeother` Return an HTTP status "303 - See Other" `gone` Return an HTTP status "410 - Gone"
`Redirect /foo http://www.example.com/foo`	Redirect to a URL on a different host
`AccessFileName .htaccess`	Name of the distributed configuration file, which contains directives that apply to the document directory it is in and to all its subtrees

`<Directory "/var/www/html/foobar">`
`   AllowOverride AuthConfig Limit`
`</Directory>`
	Specify which global directives an `.htaccess` file can override:
	`AuthConfig` Authorization directives for directory protection
	`FileInfo` Document type and metadata
	`Indexes` Directory indexing
	`Limit` Host access control
	`Options` Specific directory features
	`All` All directives
	`None` No directive

Limited scope directives	

`<Directory "/var/www/html/foobar">`
    [list of directives]
`</Directory>`
	Limit the scope of the specified directives to the directory `/var/www/html/foobar` and its subdirectories

`<Location /foobar>`
    [list of directives]
`</Location>`
	Limit the scope of the specified directive to the URL `http://www.mysite.org/foobar/` and its subdirectories

Logging directives	
`LogFormat "%h %l %u %t \"%r\" %>s %b"`	Specify the format of a log
`LogFormat "%h %l %u %t \"%r\" %>s %b" common`	Specify a nickname for a log format. In this case, specifies "common" for the CLF (Common Log Format) which is defined as such: `%h` IP address of the client host `%l` Identity of client as determined by `identd` `%u` User ID of client making the request `%t` Timestamp the server completed the request `%r` Request as done by the user `%s` Status code sent by the server to the client `%b` Size of the object returned, in bytes
`CustomLog /var/log/httpd/access_log common`	Set up a log filename, with the format or (as in this case) the nickname specified
`TransferLog /var/log/httpd/access_log`	Set up a log filename, with format determined by the most recent `LogFormat` directive which did not define a nickname
`TransferLog "|rotatelogs access_log 86400"`	Set log rotation every 24 hours
`HostnameLookups Off`	Disable DNS hostname lookup to save network traffic. Hostnames can be resolved later by processing the log file: `logresolve <access_log >accessdns_log`

Apache configuration file	
Virtual hosts directives	
`NameVirtualHost *` (v2.2)	Specify which IP address will serve virtual hosting. The argument can be an IP address, an *address:port* pair, or `*` for all IP addresses of the server. The same argument need to be inserted in the relevant `<VirtualHost>` directive

`<VirtualHost *:80>`
`   ServerName www.mysite.org`
`   ServerAlias mysite.org *.mysite.org`
`   DocumentRoot /var/www/vhosts/mysite`
`</VirtualHost>`
	The first listed virtual host is also the default virtual host.
	It inherits those main settings that does not override.
	This virtual host answers to `http://www.mysite.org`, and also redirects there all HTTP requests on the domain mysite.org

`<VirtualHost *:80>`
`   ServerAdmin webmaster@www.mysite2.org`
`   ServerName www.mysite2.org`
`   DocumentRoot /var/www/vhosts/mysite2`
`   ErrorLog /var/www/logs/mysite2`
`</VirtualHost>`
	Name-based virtual host `http://www.mysite2.org`.
	Multiple name-based virtual hosts can share the same IP address; DNS must be configured accordingly to map each name to the correct IP address. Cannot be used with HTTPS

`<VirtualHost *:8080>`
`   ServerName www.mysite3.org`
`   DocumentRoot /var/www/vhosts/mysite3`
`</VirtualHost>`
	Port-based virtual host answering to connections on port 8080.
	A `Listen 8080` directive must also be present

`<VirtualHost 10.17.1.5:80>`
`   ServerName www.mysite4.org`
`   DocumentRoot /var/www/vhosts/mysite4`
`</VirtualHost>`
	IP-based virtual host answering to `http://10.17.1.5`

Apache configuration file	
Authorization directives	
`AuthName "Protected zone"`	Name of the realm. The client will be shown the realm name and prompted to enter a user and password
`AuthType Basic`	Type of user authentication: `Basic`, `Digest`, `Form`, or `None`
`AuthUserFile "/var/www/.htpasswd"`	User database file. Each line has the format *`user`*`:`*`encryptedpassword`*. To add a user to the database file, use the command: `htpasswd /var/www/.htpasswd` *`user`* (will prompt for a password)
`AuthGroupFile "/var/www/.htgroup"`	Group database file. Each line specifies a group followed by the usernames of all its members: *`group`*`:` *`user1 user2 user3`*
`Require valid-user`	Control who can access the protected resource. `valid-user` Any user in the user database file `user` *`user`* Only the specified user `group` *`group`* Only the members of the specified group
`Satisfy Any`	Set the access policy concerning user and host control. `All` Both `Require` and `Allow` criteria must be satisfied `Any` Any of `Require` or `Allow` criteria must be satisfied
`Allow from 10.13.13.0/24` `Deny from 10.13.14.0/24` (deprecated)	Control which host can or cannot access the protected resource
`Order Allow,Deny` (deprecated)	Control the evaluation order of `Allow` and `Deny` directives. `Allow,Deny` First, all `Allow` directives are evaluated; at least one must match, or the request is rejected. Next, all `Deny` directives are evaluated; if any matches, the request is rejected. Last, any requests which do not match an `Allow` or a `Deny` directive are denied `Deny,Allow` First, all `Deny` directives are evaluated; if any match, the request is denied unless it also matches an `Allow` directive. Any requests which do not match any `Allow` or `Deny` directives are permitted

Apache configuration file	
SSL/TLS directives (`mod_ssl` module)	

`SSLCertificateFile \`
`/etc/httpd/conf/ssl.crt/server.crt`
	SSL server certificate

`SSLCertificateKeyFile \`
`/etc/httpd/conf/ssl.key/server.key`
	SSL server private key (for security reasons, this file must be mode 600 and owned by root)

`SSLCACertificatePath \`
`/usr/local/apache2/conf/ssl.crt/`
	Directory containing the certificates of CAs. Files in this directory are PEM-encoded and accessed via symlinks to hash filenames

`SSLCACertificateFile \`
`/usr/local/apache2/conf/ssl.crt/ca-bundle.crt`
	Certificates of CAs. Certificates are PEM-encoded and concatenated in a single bundle file in order of preference

`SSLCertificateChainFile \`
`/usr/local/apache2/conf/ssl.crt/ca.crt`
	Certificate chain of the CAs. Certificates are PEM-encoded and concatenated from the issuing CA certificate of the server certificate to the root CA certificate. Optional

`SSLEngine on`	Enable the SSL/TLS Protocol Engine

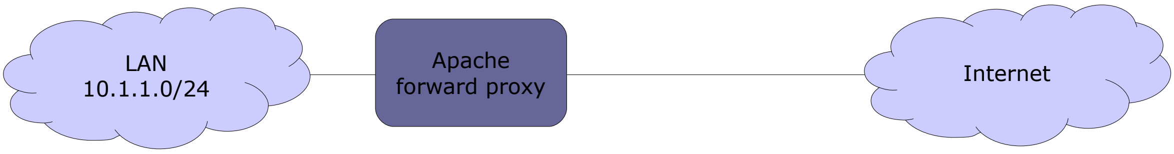
`SSLProtocol +SSLv3 +TLSv1.2`	SSL protocol flavors that the client can use to connect to server. Possible values are:
	`SSLv2` (deprecated)
	`SSLv3`
	`TLSv1`
	`TLSv1.1`
	`TLSv1.2`
	`All` (all the above protocols)

`SSLCipherSuite \`
`ALL:!aDH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP`
	Cipher suite available for the SSL handshake (key exchange algorithms, authentication algorithms, cipher/encryption algorithms, MAC digest algorithms)

`ServerTokens Full`	Server response header field to send back to client. Possible values are:
	`Prod` sends `Server: Apache`
	`Major` sends `Server: Apache/2`
	`Minor` sends `Server: Apache/2.4`
	`Minimal` sends `Server: Apache/2.4.2`
	`OS` sends `Server: Apache/2.4.2 (Unix)`
	`Full` sends `Server: Apache/2.4.2 (Unix) \`
	`PHP/4.2.2 MyMod/1.2` (default)

`ServerSignature Off`	Trailing footer line on server-generated documents. Possible values are:
	`Off` no footer line (default)
	`On` server version number and `ServerName`
	`EMail` as above, plus a mailto link to `ServerAdmin`

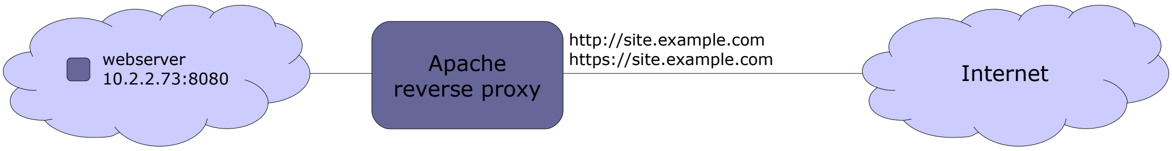
`SSLVerifyClient none`	Certificate verification level for client authentication. Possible values are: `none` no client certificate is required `require` the client needs to present a valid certificate `optional` the client may present a valid certificate (this option is unused as it doesn't work on all browsers) `optional_no_ca` the client may present a valid certificate but it doesn't need to be successfully verifiable (this option is practically used only for SSL testing)
`TraceEnable on`	Enable TRACE requests



A **forward proxy** provides proxy services, typically web content caching and/or filtering, for clients located in a LAN.
All outgoing requests from the clients, and the responses from the Internet, pass through the proxy.
The clients must be manually configured (e.g. in the browser's connection settings) to use the proxy.

Apache configuration file	
Forward proxy	
`ProxyRequests On`	Enable forward proxy requests
`ProxyVia On`	Add a `Via:` HTTP header line to every request and reply

`<Proxy "*">`
`   Require ip 10.1.1`
`</Proxy>`
	Serve only proxy requests coming from 10.1.1.0/24



A **reverse proxy** aka **gateway** allows to expose a single entry point for one or more webservers in a LAN. This improves security and simplifies management, as features (e.g. load balancing, firewalling, automatic redirection from HTTP to HTTPS, redirection on default ports) can be configured centrally.
It is necessary to create a DNS A record that maps site.example.com to the public IP address of the proxy.

Apache configuration file	
Reverse proxy	
`<VirtualHost *:80>`	Virtual host for HTTP
`   ServerName site.example.com`	Define website name

`   RewriteEngine On`
`   RewriteCond %{HTTPS} off`
`   RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}`
Alternatively:
`   Redirect "/" "https://10.2.2.73:443/"`
	Redirect all HTTP requests to HTTPS

`</VirtualHost>`	
`<VirtualHost *:443>`	Virtual host for HTTPS
`   ServerName site.example.com`	Define website name
`   ServerSignature On`	Set a footer line under server-generated pages

`   <Proxy *>`
`      Require all granted`
`   </Proxy>`
	Serve all proxy requests

`   SSLEngine on`
`   SSLProtocol ALL -SSLv2 -SSLv3`
`   SSLHonorCipherOrder on`
`   SSLCipherSuite DEFAULT`
`   SSLCertificateFile    /etc/httpd/ssl/site.crt`
`   SSLCertificateKeyFile /etc/httpd/ssl/site.key`
`   SSLCACertificateFile  /etc/httpd/ssl/site.ca.crt`
	Enable and configure SSL

`   ProxyPass        "/" "http://10.2.2.73:8080/"`
`   ProxyPassReverse "/" "http://10.2.2.73:8080/"`
	Enable reverse proxying for server 10.2.2.73

`</VirtualHost>`	

Apache Tomcat is an open source Java Servlet Container implementing several Java EE specifications, originally part of the Jakarta Project. It is composed of:
- Catalina, the core component and servlet container implementation;
- Coyote, an HTTP connector component, providing a pure Java webserver environment to run Java code;
- Jasper, a JSP (Java Server Pages) engine, which parses JSP files and compiles them into Java servlets.

Tomcat has been removed from RHEL 8; instead, it is suggested to use the JBoss Enterprise Application Platform, which includes Apache and Tomcat.

`$JAVA_HOME`	Root of the Java installation e.g. `/usr/lib/jvm/java-1.8.0-openjdk.x86_64/`
`$CATALINA_HOME`	Root of the Tomcat installation e.g. `/usr/share/tomcat7/`
`$CATALINA_BASE`	Tomcat may also be configured for multiple instances by defining the variable `$CATALINA_BASE` for each instance. If a single instance of Tomcat is running, `$CATALINA_BASE` is the same as `$CATALINA_HOME`

Tomcat global files	
`$CATALINA_BASE/conf/server.xml`	Tomcat main configuration file
`$CATALINA_BASE/conf/web.xml`	Options and values applied to all web applications running on a specific Tomcat instance. These can be overridden by the application-specific servlet configuration defined in `$CATALINA_BASE/webapps/`*`appname`*`/WEB-INF/web.xml`
`$CATALINA_BASE/conf/context.xml`	Context applied to all web applications running on a specific Tomcat instance
`$CATALINA_BASE/conf/tomcat-users.xml`	Users, passwords, and roles applied to a specific Tomcat instance
`$CATALINA_BASE/conf/catalina.policy`	Tomcat's core security policy for the Catalina class
`$CATALINA_BASE/conf/catalina.properties`	Java properties file for the Catalina class
`$CATALINA_BASE/conf/logging.properties`	Java properties file for Catalina's built-in logging functions
`$CATALINA_BASE/lib/`	JAR files accessible by both web applications and internal Tomcat code
`$JAVA_HOME/jre/lib/security/`*`keystore`*`.jks`	Java keystore
Tomcat application-specific files	
`$CATALINA_BASE/webapps/`*`appname`*`/WEB-INF/`	HTML, JSP, and other files to serve to the client browser
`$CATALINA_BASE/webapps/`*`appname`*`/WEB-INF/web.xml`	Description of servlets and other components of the application, and initialization parameters
`$CATALINA_BASE/webapps/`*`appname`*`/WEB-INF/classes/`	Java class files that aren't in JAR format. The directory hierarchy from here reflects the class hierarchy
`$CATALINA_BASE/webapps/`*`appname`*`/WEB-INF/lib/`	Other JAR files (e.g. third-party libraries, JDBC drivers) required by the application
Tomcat log files	
`$CATALINA_BASE/logs/catalina.out`	Tomcat log
`$CATALINA_BASE/logs/localhost.log`	Host log
`$CATALINA_BASE/logs/localhost_access.log`	Host HTTP access log
`$CATALINA_BASE/logs/manager.log`	Application log
`$CATALINA_BASE/logs/host-manager.log`	Application log

`java -X`	Display all available `-X` options (nonstandard HotSpot JVM options)
`java -XshowSettings:properties -version`	Print Java runtime settings

Samba is a free-software, cross-platform implementation of SMB/CIFS. **SMB (Server Message Block)** is a Microsoft proprietary protocol for file and printer sharing, while **CIFS (Common Internet File System)** is the public version of SMB.

Commonly used ports in Samba		
TCP/UDP 137	netbios-ns	NetBIOS Name Service requests and responses
TCP/UDP 138	netbios-dgm	NetBIOS Datagram Service e.g. server announcements
TCP/UDP 139	netbios-ssn	NetBIOS Session Service e.g. file and printer sharing
TCP 445	microsoft-ds	Active Directory; registration and translation of NetBIOS names, network browsing
TCP 389		LDAP
TCP 901		SWAT service

The full list of used ports can be found via the command `grep -i netbios /etc/services`.

`smbd` Server Message Block daemon. Provides SMB file and printer sharing, browser services, user authentication, and resource lock. An extra copy of this daemon runs for each client connected to the server

`nmbd` NetBIOS Name Service daemon. Handles NetBIOS name lookups, WINS requests, list browsing and elections. An extra copy of this daemon runs if Samba functions as a WINS server; another extra copy of this daemon runs if DNS is used to translate NetBIOS names.
WINS (Windows Internet Name Service) is a name service used to translate NetBIOS names to IP addresses

`/etc/smb/` `/etc/samba/` (RHEL 7)	Samba directory
`/etc/samba/lmhosts`	Samba NetBIOS hosts file
`/etc/samba/netlogon`	User logon directory

`smbd -V`
`smbclient -V`
	Show the version of the Samba server

`testparm`	Check the Samba configuration file and report any error
`smbpasswd` *`user`*	Change the Samba password of *user*
`smbpasswd -a` *`user`*	Create a new Samba *user* and set their password
`nmblookup` *`smbserver`*	Look up the NetBIOS name of a server and map it to an IP address
`nmblookup -U` *`winsserver`* `-R` *`WORKGROUP`*`#1B`	Query recursively a WINS server for the Domain Master Browser for the specified workgroup
`nmblookup -U` *`winsserver`* `-R` *`WORKGROUP`*`#1D`	Query recursively a WINS server for the Domain Controller for the specified workgroup
`net`	Tool for administration of Samba and remote CIFS servers
`net rpc shutdown -r -S` *`smbserver`* `-U root%`*`password`*	Reboot a CIFS server
`net rpc service list -S` *`smbserver`*	List available services on a CIFS server
`net status sessions`	Show active Samba sessions
`net status shares`	Show Samba shares
`net rpc info`	Show information about the domain
`net groupmap list`	Show group mappings between Samba and MS Windows

mount.cifs smbmount	Mount a Samba share on a Linux filesystem, using the CIFS filesystem interface
mount //*smbserver*/share1 /mnt/share1 \ -t cifs -o username=*user*	Mount a Samba share as *user*
smbstatus	Display current information about shares, clients connections, and locked files
smbclient //*smbserver*/share1	Access a Samba share on a server (with an FTP-like interface)
smbclient -L //*smbserver* -W *WORKGROUP* -U *user*	List the Samba resources available on a server, belonging to the specified workgroup and accessible to the specified user
cat msg.txt | smbclient -M *client* -U *user*	Show a message popup on the client machine, using the WinPopup protocol

Samba mount options	
username=*user*	Mount the share as *user*
password=*password*	Specify the mount user's *password*

credentials=*credfile*	Mount the share as the user defined in the credentials file *credfile* which must have this format:
	username=*user*
	password=*password*

multiuser	Mount the share in multiuser mode
sec=ntlmssp	Set the security level to NTLMSSP. This is required in RHEL 7 to enable multiuser mode

`/etc/samba/smb.conf`	**Samba configuration**
`[global]`	Global server settings: defines parameters applicable for the whole Samba server and sets the defaults that will be used for the parameters not mentioned in other sections
`workgroup = MYWORKGROUP`	Make Samba join the specified workgroup
`server string = Linux Samba Server %L`	Describe server to the clients
`hosts allow = 10.9.9.0/255.255.255.0`	Allow only the specified machines to connect to the server
`security = user`	Set up user-level authentication
`encrypt passwords = yes`	Use encrypted passwords

`smb passwd file = /etc/samba/smbpasswd`	Refer to the specified password file for user authentication.
	A new user's password will need to be set both in Linux and Samba by running these commands from a shell prompt:
	`passwd` *`newuser`*
	`smbpasswd` *`newuser`*

`unix password sync = yes`	When the password of a client user (e.g. under MS Windows) is changed, change the Linux and Samba passwords accordingly

`username map = /etc/samba/smbusers`	Map each Samba server user name to client user name(s).
	The file `/etc/samba/smbusers` has the following format:
	`root = Administrator Admin`
	`jdoe = "John Doe"`
	`kgreen = "Kim Green"`

`netbios name = Mysambabox`
`netbios aliases = Mysambabox1`
	Set NetBIOS name and alias

`wins support = yes`	Make Samba play the role of a WINS server. Note: There should be only one WINS server on a network
`logon server = yes`	Enable logon support. Logon script parameters will be defined in a `[netlogon]` section
`log file = /var/log/samba/log.%m`	Use a separate logfile for each machine that connects
`max log size = 1000`	Maximum size of each logfile, in Kb
`syslog only = no`	Do not use only syslog to log
`syslog = 0`	Log everything to the logfiles `/var/log/smb/log.smbd` and `/var/log/smb/log.nmbd`, and log a minimum amount of information to syslog. This parameter can be set to a higher value to have syslog log more information

`panic action = \`
`/usr/share/samba/panic-action %d`
	Mail a backtrace to the sysadmin in case Samba crashes

`[netlogon]`
`comment = Netlogon for Windows clients`
	Section defining a logon script

`path = /home/netlogon`
`logon script = %U.bat`
	Specifies a per-user script e.g. `/home/netlogon/jdoe.bat` will be called when user jdoe logs in.
	It is also possible to specify a per-clientname script `%m.bat`, which will be called when a specific machine logs in.

`browseable = no`
`writeable = no`	

`guest ok = no`	Guest access to the service (i.e. access without entering a password) is disabled

`[Canon LaserJet 3]`
`printer name = lp`
`comment = Canon LaserJet 3 main printer`
`path = /var/spool/lpd/samba`
`printable = yes`
`writeable = no`
	Section defining a printer accessible via the network

/etc/samba/smb.conf	Samba configuration
[public]	Section defining a public share accessible on read/write by anyone
comment = Public Storage on %L	Describe the public share to users
path = /home/samba	Path of the public share on the server
browsable = yes	Show the public share when browsing
writeable = yes	Allow all users to write in this directory
[homes]	Section enabling users that have an account and a home directory on the Samba server to access it and modify its contents from a Samba client. The path variable is not set, by default is path=/home/%S
comment = %U's home directory on %L from %m	Describe the share to the user
browseable = no	Do not show the homes share when browsing
writeable = yes	Allow the user to write in their home directory
[foobar]	Section defining a specific share
path = /foobar	Path of the share on the server
comment = Share Foobar on %L from %m	Describe the share to users
browsable = yes	Show the share when browsing
writeable = yes	Allow the users to write in this share
valid users = jdoe, kgreen, +geeks	Allow access only to users "jdoe" and "kgreen", and to local group "geeks"
invalid users = csmith	Deny access to user "csmith"
read list = bcameron	Allow read-only access to user "bcameron"
write list = fcastle	Allow read-write access to user "fcastle"

Samba - access configuration and macros

`/etc/samba/smb.conf` **Samba configuration**	
User-level authentication	
`[global]`	
`security = user`	Set up user-level authentication
`guest account = nobody`	Map the guest account to the system user nobody (default)
`map to guest = Never`	Specify how incoming requests are mapped to the guest account: `Bad User` redirect from an invalid user to guest account on server `Bad Password` redirect from an invalid password to guest account on server `Never` reject unauthenticated users
Server-level authentication	
`[global]`	
`security = server`	Set up server-level authentication
`password server =` *`srv1 srv2`*	Authenticate to server *srv1*, or to server *srv2* if the first one is unavailable
Domain-level authentication	
`[global]`	
`security = ADS`	Set up domain-level authentication as an Active Directory member server
`realm = KRB_REALM`	Join the specified realm. Kerberos must be installed and an administrator account must be created: `net ads join -U Administrator%`*`password`*
Share-level authentication	
`[global]`	
`security = share`	Set up share-level authentication

`[foobar]`
`path = /foobar`
`username =` *`user`*
`only user = yes`
	Define a "foobar" share accessible to any user which can supply *user*'s password.
	The *user* must be created on the system:
	`useradd -c "Foobar account" -d /tmp -m -s /sbin/nologin` *`user`*
	and added to the Samba password file:
	`smbpasswd -a` *`user`*

Samba macros			
`%S`	Username	These macros are applied only to configuration options used once a connection has been established:	
`%U`	Session username i.e. the username that the client requested, not necessarily the same as the one the client obtained		
`%G`	Primary group of session username	`%S`	Name of the current service, if any
`%h`	Samba server hostname	`%P`	Root directory of the current service, if any
`%M`	Client hostname	`%u`	Username of the current service, if any
`%L`	NetBIOS name of the server	`%g`	Primary group name of username
`%m`	NetBIOS name of the client	`%H`	Home directory of username
`%d`	Process ID of the current server process	`%N`	Name of the NIS home directory server as obtained from the NIS `auto.map` entry. Same as `%L` if Samba was not compiled with the `--with-automount` option
`%a`	Architecture of remote machine		
`%I`	IP address of client machine		
`%i`	Local IP address to which a client connected	`%p`	Path of service's home directory as obtained from the NIS `auto.map` entry. The NIS `auto.map` entry is split up as `%N:%p`
`%T`	Current date and time		
`%D`	Domain or workgroup of the current user		
`%w`	Winbind separator		
`%$(`*`var`*`)`	Value of the environment variable *var*		

Samba setup

This procedure allows sharing on read-write the local directory `/smbshare` on server 10.1.1.1 to client 10.2.2.2.

Server setup:

1.	Create the group for write access to the share	`groupadd -r geeks`
2.	Create the user and assign it to the group	`useradd -G geeks jdoe`
3.	Add the user to Samba. You will be prompted to enter a password	`smbpasswd -a jdoe`
4.	Assign correct ownership to the share	`chgrp geeks /smbshare`
5.	Set the SGID bit to the share	`chmod 2775 /smbshare`

6.	Set the correct SELinux label to the share	`semanage fcontext -a -t samba_share_t '/smbshare'`
		`restorecon -FR /smbshare`

7.	Enable the SELinux boolean for write access to the share	`setsebool -P samba_export_all_rw=on`

8. Add a section for the share on `/etc/samba/smb.conf`:

```
[smbshare]
   path = /smbshare
   hosts allow = 10.2.2.2
   write list = @geeks
```

9. Ensure that the `smb` and `nmb` services are running

Client setup:

1. Add an entry to `/etc/fstab` to mount the Samba share device automatically:

```
//10.1.1.1/smbshare  /mountpoint  cifs  username=jdoe,password=s3cr3t  0 0
```

Client multiuser setup:

1. Add an entry to `/etc/fstab` to mount the Samba share device automatically in multiuser mode:

```
//10.1.1.1/smbshare  /mountpoint  cifs  username=jdoe,password=s3cr3t,multiuser,sec=ntlmssp  0 0
```

2.	Login as another user (there must be a matching Samba user on the Samba server 10.1.1.1)	`su - ksmith`
3.	Store the Samba username and password in the kernel keyring for the current session	`cifscreds add 10.1.1.1`

A **Network File System (NFS)** server makes filesystems available to remote clients for mounting.

NFS requires the portmapper to map incoming TCP/IP connections to the appropriate NFS RPC calls. Some Linux distributions use rpcbind instead of the portmapper.
For security reasons, the TCP Wrapper should be configured to limit access to the portmapper to NFS clients only:
file /etc/hosts.deny should contain portmap: ALL
file /etc/hosts.allow should contain portmap: *IP_addresses_of_clients*

NFS handles user permissions across systems by considering users with same UID and username as the same user.
Group permission is evaluated similarly, by GID and groupname.

rpc.nfsd rpc.mountd rpc.lockd rpc.statd	NFS daemons
/etc/exports	List of the filesystems to be exported (via the command exportfs)
/var/lib/nfs/xtab	List of exported filesystems, maintained by exportfs
/proc/fs/nfs/exports	Kernel export table (can be examined via the command cat)
exportfs -ra	Export or reexport all directories. When exporting, fills the kernel export table /proc/fs/nfs/exports. When reexporting, removes the entries in /var/lib/nfs/xtab that are deleted from /etc/exports (therefore synchronizing the two files), and removes the entries from /proc/fs/nfs/exports that are no longer valid
exportfs -ua	Unexport all directories. Removes from /proc/fs/nfs/exports the entries that are listed in /var/lib/nfs/xtab, and clears the latter file
mount -t nfs *nfsserver*:/share /usr	Command to be run on a client to mount locally a remote NFS share. NFS shares accessed frequently should be added to /etc/fstab e.g. *nfsserver*:/share /usr nfs intr 0 0
showmount	Show the remote client hosts currently having active mounts
showmount --directories	Show the directories currently mounted by a remote client host
showmount --exports	Show the filesystems currently exported i.e. the active export list
showmount --all	Show both remote client hosts and directories
showmount -e *nfsserver*	Show the shares a NFS server has available for mounting
rpcinfo -p *nfsserver*	Probe the portmapper on a NFS server and display the list of all registered RPC services there
rpcinfo -t *nfsserver* nfs	Test a NFS connection by sending a null pseudo request (using TCP)
rpcinfo -u *nfsserver* nfs	Test a NFS connection by sending a null pseudo request (using UDP)
nfsstat	Display NFS/RPC client/server statistics.

Options:

	NFS	RPC	both
server	-sn	-sr	-s
client	-cn	-cr	-c
both	-n	-r	-nr

NFS - export table

/etc/exports

```
/export/            10.3.3.3(rw)
/export2/           10.4.4.0/24
/export3/           *(ro,sync)
/home/ftp/pub       myhost(rw)   *.example.org(ro)
/home/crew          @FOOWORKGROUP(rw)   (ro)
```

filesystem	Filesystem on the NFS server to be exported to clients	
client identity	Client systems permitted to access the exported directory. Can be specified by hostname, IP address, wildcard, subnet, or @NIS workgroup. Multiple client systems can be listed, and each one can have different options	
client options	`ro`	Read-only access (default)
	`rw`	Read and write access. The client might choose to mount read-only anyway
	`sync`	Reply to requests only after the changes made by these requests have been committed to stable storage
	`async`	Reply to requests without waiting that changes are committed to stable storage. Improves performances but might cause loss or corruption of data if server crashes
	`root_squash`	Requests by user `root` on client will be done as user `nobody` on server (default)
	`no_root_squash`	Requests by user `root` on client will be done as same user `root` on server
	`all_squash`	Requests by a non-root user on client will be done as user `nobody` on server
	`no_all_squash`	Requests by a non-root user on client will be attempted as same user on server (default)

NFS mount options	
rsize=*nnn*	Size for read transfers (from server to client)
wsize=*nnn*	Size for write transfers (from client to server)
nfsvers=*n*	Use NFS version *n* for transport
retry=*n*	Keep retrying a mount attempt for *n* minutes before giving up
timeo=*n*	A mount attempt times out after *n* tenths of a second
intr	User can interrupt a mount attempt
nointr	User cannot interrupt a mount attempt (default)
hard	The system will try a mount indefinitely (default)
soft	The system will try a mount until an RPC timeout occurs
bg	Try a mount in the foreground; all retries occur in the background
fg	All mount attempts occur in the foreground (default)
tcp	Connect using TCP
udp	Connect using UDP
sec=krb5p	Use Kerberos to encrypt all requests between client and server
v4.2	Enable NFS v4.2, which allows the server to export the SELinux context

NFS setup

This procedure allows sharing on read-write the local directory `/nfsshare` on server 10.1.1.1 to client 10.2.2.2.

Server setup:

1. Ensure that the `nfs-server` service is running
2. Change ownership of the share `chown nfsnobody /nfsshare`
3. Add an entry for the share on `/etc/exports`:

```
/nfsshare  10.2.2.2(rw)
```

4. Reload the exports file `exportfs -r`

Client setup:

1. Add an entry to `/etc/fstab` to mount the NFS share device automatically:

```
10.1.1.1:/nfsshare  /mountpoint  nfs  defaults  0 0
```

Secure NFS setup

This procedure allows sharing on read-write the local directory `/nfsshare` on server 10.1.1.1 to client 10.2.2.2, securely with Kerberos enabled.

Server setup:

1. Install the appropriate server keytab on `/etc/krb5.keytab`
2. Ensure that the `nfs-secure-server` service is running
3. Change ownership of the share `chown nfsnobody /nfsshare`
4. Add an entry for the share on `/etc/exports`:

```
/nfsshare  10.2.2.2(sec=krb5p,rw)
```

5. Reload the exports file `exportfs -r`

Client setup:

1. Install the appropriate client keytab on `/etc/krb5.keytab`
2. Ensure that the `nfs-secure` service is running
3. Add an entry to `/etc/fstab` to mount the NFS share device automatically:

```
10.1.1.1:/nfsshare  /mountpoint  nfs  defaults,sec=krb5p  0 0
```

iSCSI (Internet Small Computer System Interface) is a network protocol that allows emulating an SCSI local storage device over a TCP/IP network. By default it uses TCP port 3260.
An iSCSI server can use a local block device (physical or virtual disk, disk partition, or Logical Volume), a file, a physical SCSI device, or a ramdisk as the underlying storage resource (**backstore**) and make it available by assigning it a **LUN** (Logical Unit Number). An iSCSI server provides one or more **targets**, each of which presents one or more LUNs and is able to accept connections from an iSCSI client (**initiator**).
Targets and initiators are called **nodes** and are identified by a unique **IQN** (iSCSI Qualified Name) e.g. `iqn.2017-11.org.example.subdomain:foo:bar`. The IP address and port of a node is called a **portal**.
A target accepts connections from an initiator via a **TPG** (Target Portal Group) i.e. its IP address and port. A TPG may have an ACL in place so to accept connections only from a specific initiator's IQN.

`targetcli`	Target configurator (server side). Can be used as a command line tool or as an interactive shell. Configuration is saved to `/etc/target/saveconfig.json`
`iscsiadm`	Administration tool for iSCSI devices (client side)

iSCSI setup

This procedure makes available the local disk `/dev/sbd` on server 10.1.1.1 to the client having IQN `iqn.2017-11.org.example:client`.

Server (target) setup:

1. Ensure that the `targetcli` service is running
2. Enter the targetcli shell

```
targetcli
```

3. Create a backstore

```
cd /backstores/block
create mydisk /dev/sdb
```

4. Create a IQN for the target.
   This automatically creates a TPG for the IQN

```
cd /iscsi
create iqn.2017-11.org.example:target
```

5. On the TPG, create an ACL to allow connections from the initiator with a specific IQN

```
cd /iscsi/iqn.2017-11.org.example:target/tpg1/acls
create iqn.2017-11.org.example:client
```

6. On the TPG, create a LUN for the backstore

```
cd /iscsi/iqn.2017-11.org.example:target/tpg1/luns
create /backstores/block/mydisk
```

7. On the TPG, create a portal listening from the server's IP address

```
cd /iscsi/iqn.2017-11.org.example:target/tpg1/portals
delete 0.0.0.0 ip_port=3260
create 10.1.1.1
```

8. Verify the configuration

```
ls /
```

```
o- / ....................................................................................... [...]
  o- backstores ............................................................................ [...]
  | o- block ................................................................ [Storage Objects: 1]
  | | o- mydisk ....................................... [/dev/sdb (100.0MiB) write-thru activated]
  | |   o- alua ................................................................. [ALUA Groups: 1]
  | |     o- default_tg_pt_gp ..................................... [ALUA state: Active/optimized]
  | o- fileio ............................................................... [Storage Objects: 0]
  | o- pscsi ................................................................ [Storage Objects: 0]
  | o- ramdisk .............................................................. [Storage Objects: 0]
  o- iscsi .......................................................................... [Targets: 1]
  | o- iqn.2017-11.org.example:target .................................................. [TPGs: 1]
  |   o- tpg1 ............................................................. [no-gen-acls, no-auth]
  |     o- acls ........................................................................ [ACLs: 1]
  |     | o- iqn.2017-11.org.example:client ..................................... [Mapped LUNs: 1]
  |     |   o- mapped_lun0 ............................................... [lun0 block/mydisk (rw)]
  |     o- luns ........................................................................ [LUNs: 1]
  |     | o- lun0 .................................. [block/mydisk (/dev/sdb) (default_tg_pt_gp)]
  |     o- portals ................................................................... [Portals: 1]
  |       o- 10.1.1.1:3260 ...................................................................... [OK]
  o- loopback ....................................................................... [Targets: 0]
```

9. Exit the targetcli shell.
   Configuration is automatically saved

```
exit
```

Client (initiator) setup:

1. Set the correct initiator IQN in the file `/etc/iscsi/initiatorname.iscsi`:

```
InitiatorName=iqn.2017-11.org.example:client
```

2. Ensure that the `iscsi` service is running
3. Discover the iSCSI target(s) provided by the portal. This echoes the target(s) IQN found

```
iscsiadm -m discovery -t sendtargets -p 10.1.1.1
```

4. Login to the target IQN found

```
iscsiadm -m node -T iqn.2017-11.org.example:target -p 10.1.1.1 -l
```

   The iSCSI device is now locally available and can be formatted and mounted. Node records remain after logout or reboot; the system will login again to the target IQN automatically

5. Add an entry to `/etc/fstab` to mount the iSCSI device automatically:

```
UUID=nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn  /mountpoint  fstype  _netdev  0 0
```

DHCP (Dynamic Host Configuration Protocol) is a protocol for network management that automatically assigns to a requesting host an IP address and other network configuration parameters. It is based on **BOOTP (Bootstrap Protocol)**.
A DHCP server listens for requests on UDP port 67 and answers to UDP port 68. The assignment of an IP address to a host is done through a sequence of DHCP messages initiated by the client host, which is (for DHCPv4): DHCP Discover, DHCP Offer, DHCP Request, and then DHCP ACK.
Because DHCP Discover messages are broadcast and therefore not routed outside a LAN, a DHCP relay agent is necessary for those clients situated outside the DHCP server's LAN. The DHCP relay agent listens to DHCP Discover messages and relays them in unicast to the DHCP server.

`/etc/dhcpd.conf`	Configuration file for the DHCP server
`/etc/sysconfig/dhcrelay` (SUSE)	Configuration file for the DHCP relay agent
`/var/lib/dhcpd/dhcpd.leases`	DHCP current leases

`/etc/dhcpd.conf`	**DHCP server configuration**

`option domain-name-servers 10.2.2.2;`
`option smtp-servers 10.3.3.3;`
`option pop-servers 10.4.4.4;`
`option time-servers 10.5.5.5;`
`option nntp-servers 10.6.6.6;`
	Global parameters for DNS, mail, NTP, and news servers specification

`shared-network geek-net {`	Definition of a network
`   default-lease-time 86400;`	Time, in seconds, that will be assigned to a lease if a client does not ask for a specific expiration time
`   max-lease-time 172800;`	Maximum time, in seconds, that can be assigned to a lease if a client asks for a specific expiration time

`   option routers 10.0.3.252;`
`   option broadcast-address 10.0.3.255;`	

`   subnet 10.0.3.0 netmask 255.255.255.128 {`
`      range 10.0.3.1 10.0.3.101;`
`   }`
`   subnet 10.0.3.128 netmask 255.255.255.128 {`
`      range 10.0.3.129 10.0.3.229;`
`   }`
 	
`}`
	Definition of different subnets in the network, with specification of different ranges of IP addresses that will be leased to clients depending on the client's subnet

`group {`	Definition of a group

`   option routers 10.0.17.252;`
`   option broadcast-address 10.0.17.255;`
`   netmask 255.255.255.0;`	

`   host linuxbox1 {`
`      hardware ethernet AA:BB:CC:DD:EE:FF;`
`      fixed-address 10.0.17.42;`
`      option host-name "linuxbox1";`
`   }`
`   host linuxbox2 {`
`      hardware ethernet 33:44:55:66:77:88;`
`      fixed-address 10.0.17.66;`
`      option host-name "linuxbox2";`
`   }`
 	
`}`
	Definition of different hosts to whom static IP addresses will be assigned to, depending on their MAC address

DHCPv4 message	DHCPv6 message	
DHCP Discover	**Solicit**	Sent from client. Tries to find any available DHCP server
DHCP Offer	**Advertise**	Sent from server to client in response to DHCP Discover or Solicit. Advertises that server is available for DHCP services
DHCP Request	**Request**	Sent from client to server. Requests IP address and other configuration parameters
	Renew	Sent from client to the server that originally provided the IP address. Asks to extend IP address lease
	Rebind	Sent from client to any available server if a past Renew message produced no answer. Asks to extend IP address lease
	Confirm	Sent from client to any available server. Asks to determine whether the allocated IP address is still valid on the link
DHCP ACK	**Reply**	Sent from server to client in response to multiple types of message. Content varies
DHCP Release	**Release**	Sent from client to the server that originally provided the IP address. Asks to cancel the IP address lease
DHCP Decline	**Decline**	Sent from client to server. Client has detected that the IP address assigned by the server is already in use
	Reconfigure	Sent from server to client. Server has new configuration parameters and the client is asked to initiate a Renew or Information-Request
DHCP Inform	**Information-Request**	Sent from client to server. Requests configuration parameters without any IP address
	Relay-Forward	Sent from relay agent to server or another relay agent. Content is another encapsulated message
	Relay-Reply	Sent from server to relay agent. Content is another encapsulated message
DHCP NAK		Sent from server to client. Client has incorrect parameters for the link or its lease has expired

PAM (Pluggable Authentication Modules) is an abstraction layer that allows applications to use authentication methods while being implementation-agnostic.

`/etc/pam.d/`*`service`*	PAM configuration for *service*
`/etc/pam.conf` (obsolete)	PAM configuration for all services
`ldd /usr/sbin/`*`service`*` | grep libpam`	Check if *service* is enabled to use PAM

/etc/pam.d/*service*

```
auth       requisite   pam_securetty.so
auth       required    pam_nologin.so
auth       required    pam_env.so
auth       required    pam_unix.so nullok
account    required    pam_unix.so
session    required    pam_unix.so
session    optional    pam_lastlog.so
password   required    pam_unix.so nullok obscure min=4 max=8
```

type	`auth`	Authentication module to verify user identity and group membership
	`account`	Authorization module to determine user's right to access a resource (other than their identity)
	`password`	Module to update a user's authentication credentials
	`session`	Module (run at end and beginning of a user session) to set up the user environment
control	`optional`	Module is not critical to the success or failure of *service*
	`sufficient`	If this module successes, and no previous module has failed, module stack processing ends successfully. If this module fails, it is non-fatal and processing of the stack continues
	`required`	If this module fails, processing of the stack continues until the end, and *service* fails
	`requisite`	If this module fails, *service* fails and control returns to the application that invoked *service*
	`include`	Include modules from another PAM service file
module	PAM module and its options, e.g.:	
	`pam_unix.so`	Standard UNIX authentication module via `/etc/passwd` and `/etc/shadow`
	`pam_nis.so`	Module for authentication via NIS
	`pam_ldap.so`	Module for authentication via LDAP
	`pam_fshadow.so`	Module for authentication against an alternative shadow passwords file
	`pam_cracklib.so`	Module for password strength policies (e.g. length, case, max number of retries)
	`pam_limits.so`	Module for system policies and system resource usage limits
	`pam_listfile.so`	Module to deny or allow the service based on an arbitrary text file

LDAP (Lightweight Directory Access Protocol) is a simplified version of the X.500 standard and uses TCP port 389. LDAP allows organizing hierarchically a database of entries, each one of which is identified by a unique **DN (Distinguished Name)**. Each DN has a set of **attributes**, and each attribute has a **value**; an attribute may appear multiple times. Special attributes called **objectClass** define which attributes are allowed and which are required, and determine the **schema** of the LDAP.

<table>
<tr><td colspan="2">dn: cn=John Doe,ou=IT Dept,dc=example,dc=org</td><td>Distinguished Name</td></tr>
<tr><th colspan="3">Examples of LDAP attributes</th></tr>
<tr><th>Attribute</th><th>Attribute with value</th><th>Meaning</th></tr>
<tr><td>cn</td><td>cn: John Doe</td><td>Common Name</td></tr>
<tr><td>dc</td><td>dc=example,dc=org</td><td>Domain Component</td></tr>
<tr><td>givenName</td><td>givenName: John</td><td>First name</td></tr>
<tr><td>sn</td><td>sn: Doe</td><td>Surname</td></tr>
<tr><td>mail</td><td>mail: jdoe@example.org</td><td>Email address</td></tr>
<tr><td>telephoneNumber</td><td>telephoneNumber: +1 555 1234 567</td><td>Telephone number</td></tr>
<tr><td>uid</td><td>uid: jdoe</td><td>User ID</td></tr>
<tr><td>c</td><td>c: US</td><td>Country code</td></tr>
<tr><td>l</td><td>l: San Francisco</td><td>Locality</td></tr>
<tr><td>st</td><td>st: California</td><td>State or province</td></tr>
<tr><td>street</td><td>street: 42, Penguin Road</td><td>Street</td></tr>
<tr><td>o</td><td>o: The Example Foundation</td><td>Organization</td></tr>
<tr><td>ou</td><td>ou: IT Dept</td><td>Organizational Unit</td></tr>
<tr><td>manager</td><td>manager: cn=Kim Green,ou=RD,dc=example,dc=org</td><td>Manager</td></tr>
</table>

LDIF (LDAP Data Interchange Format) is a plaintext data format for representing LDAP content and changes. The following LDIF file will change the email address of user "jdoe", add a picture, and delete the description attribute for the entry:

```
dn: cn=John Doe,dc=example,dc=org
changetype: modify
replace: mail
mail: johndoe@example.org
-
add: jpegPhoto
jpegPhoto:< file://tmp/jdoe.jpg
-
delete: description
-
```

ldapsearch	Query an LDAP server and return the output in LDIF	
	-b *base*	Start searching from *base*
	-z *n*	Retrieve at maximum *n* entries as result
	-LLL	Terse output. Outputs the result in LDIFv1, does not print comments, and omits the LDIF version number
	filter	Search filter. If not specified, uses the default filter (objectClass=*)
	attributes	Attributes to return. If not specified, returns all attributes
ldapmodify	Modify an LDAP entry	
ldapadd ldapmodify -a	Add an LDAP entry	
ldapdelete	Delete an LDAP entry	
	-f *file.ldif*	Modify, add, or delete an entry according to the LDIF file specified
ldappasswd	Change the password of an LDAP entry	
	-s *password*	Set the new password as *password*
	-S	Prompt for the new password

In addition to the command-specific arguments, all LDAP commands above accept the following generic arguments:

-H ldap://*srv*	Connect to the specified LDAP server
-H ldapi://	Connect to the localhost LDAP server using IPC instead of a network socket
-D *binddn*	Bind (authenticate) to the LDAP server as the specified DN
-w *password*	Authenticate with the specified *password*
-W	Prompt for authentication
-x	Use simple authentication instead of SASL
-v	Use verbose mode for output

```
ldapsearch -H ldap://ldap.example.org \
-s base -b "ou=people,dc=example,dc=com" "(sn=Doe)" \
cn sn telephoneNumber
```

Query a LDAP server for entries in the OU "people" whose surname is "Doe"; print common name, surname, and telephone number of the entries found

```
ldapmodify -b -r -f file.ldif
```

Modify an entry according to the LDIF file specified

```
ldapadd -h ldap.example.org \
-D "cn=Admin,dc=example,dc=org" -W -f file.ldif
```

Authenticating as "Admin", add an entry by adding the content of the specified LDIF file to the directory

```
ldapdelete -h ldap.example.org \
-D "cn=Admin,dc=example,dc=org" -W \
"uid=jdoe,dc=example,dc=org"
```

Authenticating as "Admin", delete the user "jdoe"

```
ldappasswd -h ldap.example.org \
-D "cn=Admin,dc=example,dc=org" -W -x \
-S "uid=jdoe,ou=IT Dept,dc=example,dc=org"
```

Authenticating as "Admin" on example.org, change the password of user "jdoe" in the OU "IT Dept"

OpenLDAP is an open source implementation of LDAP, and was initially developed together with the LDAP protocol. Its related service is `slapd`, the Standalone LDAP daemon.
SSSD can be configured to provide access to OpenLDAP (or any other LDAP server) as an authentication and identity provider.

`/var/lib/ldap/`	Files constituting the OpenLDAP database
`/etc/openldap/slapd.conf` `/usr/local/etc/openldap/slapd.conf`	OpenLDAP configuration file (deprecated)
`/usr/local/etc/openldap/slapd.d/` (v2.3 and later)	Directory containing the LDIF database that stores the OpenLDAP configuration. These LDIF files must not be edited by hand

`slapcat -b cn=config`
`ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config`
	Show the OpenLDAP configuration

`slaptest -u`	Verify that the OpenLDAP configuration is correct
`slapcat -l file.ldif`	Dump the contents of an OpenLDAP database to an LDIF file
`slapadd -l file.ldif`	Import an OpenLDAP database from an LDIF file
`slapindex`	Regenerate OpenLDAP's database indexes
`yum install openldap openldap-clients authconfig \` `sssd nss-pam-ldapd authconfig-gtk` (RHEL 7)	Install the OpenLDAP client

`authconfig --enableldap --enableldapauth \`
`--ldapserver=ldap://`*`ldapserver`*` \`
`--ldapbasedn="dc=example,dc=org" \`
`--enablesssd --update` (RHEL 7)
	Set up the LDAP client to connect to a *ldapserver*.
	This will update the configuration files
	`/etc/sssd/sssd.conf` and `/etc/openldap/ldap.conf`

`authselect select sssd --force` (RHEL 8)	Set up LDAP client authentication via `sssd`
`authconfig-gtk` `system-config-authentication`	OpenLDAP configuration GUI

389 Directory Server is an enterprise-class open source LDAP server. It is derived from OpenLDAP and is part of the Fedora Project. A commercial version is also available with the name **Red Hat Directory Server**.

It features TLSv1 encryption, SASL, synchronization with MS Windows Active Directory, and a web console (on port 9090). It also includes Lib389, a Python base library that can be used to manage, test, and perform all operations on a 389 DS instance.

`/etc/dirsrv/slapd-`*`instancename`*`/dse.ldif`	Instance configuration (`cn=config` entry)
`/var/lib/dirsrv/slapd-`*`instancename`*`/`	Directory containing the database and other data relative to an instance

`dsctl`	Start, stop, display status, backup, and generally manage a local instance
`dsconf`	Configure a local or remote instance
`dsidm`	Manage backend data (users, groups, permissions)

Security-Enhanced Linux (SELinux) is a Linux kernel security module that provides a mechanism for supporting access control security policies.

SELinux implements a Mandatory Access Control framework that allows the definition of fine-grained permissions for how **subjects** (i.e. processes) access **objects** (i.e. other processes, files, devices, ports, sockets); this improves security with respect to the traditional Discretionary Access Control, which defines accesses based on users and groups.
Processes, files, and users have a **security context** structured as *user:role:type:level* e.g. `unconfined_u:object_r:user_home_t:s0`. The third field defines a *type* for files or a *domain* for processes.
The security context of a file is stored in its extended attributes.
The decisions SELinux takes about allowing or disallowing access are stored in the **AVC (Access Vector Cache)**.

SELinux creates a pseudo filesystem (SELinuxfs) containing commands used by the kernel for its operations; this filesystem is usually mounted on `/selinux/` or `/sys/fs/selinux/`.

`setenforce 0`
`echo 0 > /selinux/enforce`
	Enter permissive mode (SELinux must be enabled)

`setenforce 1`
`echo 1 > /selinux/enforce`
	Enter enforcing mode (SELinux must be enabled)

`getenforce`
`cat /selinux/enforce`
	Display current mode

`sestatus -v`	Show SELinux mode, SELinuxfs mount point, etc.

SELinux state can be configured permanently in `/etc/selinux/config` (symlinked in `/etc/sysconfig/selinux`):

mode	`SELINUX=`	`enforcing`	SELinux fully enforces security policies
		`permissive`	SELinux does not enforce security policies, but logs all violations
		`disabled`	SELinux security policies are disabled
policy	`SELINUXTYPE=`	`targeted`	SELinux protects targeted daemons
		`strict`	(up to RHEL 6) SELinux fully protects the system
		`minimum`	(RHEL 7 and later) SELinux only protects selected processes
		`mls`	(RHEL 7 and later) Multi Level Security protection

`ls -Z`	List files and their security context
`ps -eZ`	List processes and their security context
`cp --preserve=context` *`file file2`*	Copy a file and its context. By default, the `cp` command will create a new SELinux file context

`tar --selinux` *`otherargs`*
`star -xattr -H=exustar` *`otherargs`*
	Create or extract archives that retain the security context of the original files

SELinux - semanage

`semanage`	Manage SELinux policies
`semanage fcontext -l`	List files and their assigned SELinux labels
`semanage fcontext -a -t` *`label file`*	Assign the SELinux *label* to *file*. Afterwards, it is necessary to apply the label via `restorecon -f` *`file`*

`semanage fcontext -a -t httpd_sys_content_t \`
`"/var/www2/html2(/.*)?"`
	Allow a local webserver to serve content stored in the directory `/var/www2/html2`

`semanage login -l`	List mappings between users and SELinux users
`semanage port -l`	List port numbers and their assigned SELinux type definitions
`semanage port -a -t` *`portlabel`* `-p tcp` *`n`*	Assign the SELinux *portlabel* to TCP port *n*
`semanage port -a -t http_port_t -p tcp 8888`	Allow a local webserver to serve content on port 8888
`semanage port -d -t http_port_t -p tcp 8888`	Remove the binding of `http_port_t` port label to TCP 8888
`semanage port -m -t http_cache_port_t -p tcp 8888`	Modify the port label bound to TCP 8888
`semanage permissive -a auditd_t`	Add `auditd_t` to the list of permissive types/domains. In this case, SELinux allows the `auditd` daemon all access while logging its AVC violations
`semanage permissive -d auditd_t`	Delete `auditd_t` from the list of permissive types/domains
`semanage permissive -l`	List all permissive types/domains

/var/log/audit/audit.log	Logfile containing AVC denials, if auditd is running (default). AVC denials can also be seen via dmesg
/var/log/messages	Logfile containing AVC denials, if rsyslogd is running
sepolicy	Inspect a SELinux policy
sepolicy manpage -a -p /usr/local/man/man8 && mandb	Generate all SELinux policy manpages
seinfo	Query the components of a SELinux policy
chcon *context file*	Change the security context of *file* to the specified *context*. Changes made via chcon are not persistent across filesystem relabels or the execution of restorecon; for persistent changes, use semanage fcontext followed by restorecon
chcon --reference=*file0 file*	Change the security context of *file* to be the same as *file0*
restorecon -f *file*	Restore the security context of *file* to the system default
getsebool *boolean*	Get the value of a SELinux boolean
setsebool *boolean*=*value*	Set the value of a SELinux boolean
sealert -a *logfile*	Analyze a SELinux logfile and display verbosely SELinux policy violations. SELinux violation events are logged as type=AVC msg=audit(*timest.amp:id*): avc: denied (...)
grep *timest.amp:id logfile* | audit2why	Diagnostic a specific AVC denial event entry (identified by a *timestamp* and an *id*) from a SELinux *logfile*
audit2why -d	Read AVC violations from the output of dmesg
ausearch -a *id*	Query the SELinux log for event *id*
audit2allow -i *inputfile* -M *module*	Generate a loadable *module* containing the appropriate SELinux policy from a denied operation stored in *inputfile*
ausearch -c '(*exe*)' --raw | audit2allow -M *module*	Generate a loadable module to allow access on an executable which caused an AVC violation. This module will then need to be installed via semodule -i *module.pp*
semodule -l	List installed SELinux policy modules
semodule -X *n* -i *module*.pp	Install a SELinux policy module at priority *n*. Installed modules persist after reboot. Module files have usually the suffix .pp (policy package)
semodule -X *n* -r *module*	Remove a SELinux policy module at priority *n*. Modules must be removed at the same priority at which they were installed

Kickstart is a method to perform automatic installation and configuration of RHEL machines.
This can be done by specifying `inst.ks=hd:/dev/sda:/root/path/ksfile` either as a boot option, or an option to the kernel command in GRUB 2.

`/root/anaconda-ks.cfg`	Kickstart file describing the current system. This file is automatically generated during the installation
`system-config-kickstart`	GUI tool to create a Kickstart file
`ksvalidator` *`ksfile`*	Check the validity of a Kickstart file
`ksverdiff -f RHEL6 -t RHEL7`	Show the differences in the Kickstart syntax between RHEL 6 and RHEL 7

Red Hat Satellite 6

Red Hat Satellite is a system management software platform that allows provisioning and configuration of RHEL machines. Repository content is provided via Red Hat Subscription Management (RHSM).
Satellite 5 was based on Spacewalk, an open source system management software for Linux machines. Satellite 6 is a complete overhaul of it and is composed of:
- **Foreman**, an open source lifecycle management tool able to provision servers via Kickstart and Puppet;
- **Katello**, a tool that handles Red Hat repository management (via the **Pulp** service) and subscription management (via the **Candlepin** service).

All these components above need a PostgreSQL database, except Pulp which needs a MongoDB database.
As a separate component, **Capsule** servers act as proxies for many of the main Satellite functions e.g. repository storage.
A Capsule is also integrated in each Satellite server.

`subscription-manager register`	Register a system to the RHSM portal
`subscription-manager attach`	Attach a RHSM subscription to a registered system
`foreman-maintain service list`	List all Satellite services

`foreman-maintain service status`
`foreman-maintain service start`
`foreman-maintain service stop`
`foreman-maintain service restart`
	Display status or start, stop, restart all Satellite services.
	Performed via `systemctl`

`foreman-maintain backup`	Make a backup of Satellite
`foreman-rake` *command:option*	Perform various administrative tasks
`hammer`	CLI tool for Foreman
`pulp-admin-client`	Tool to administer the Pulp server
`virt-who`	Agent for scanning and reporting virtual guest IDs and hypervisors to a Satellite server
`foreman-debug`	Collect Satellite configuration, log, and backend data for debug purposes
`sosreport`	Collect diagnostic and configuration data for technical support
`citellus.py` *sosreportfile*	Perform some automated checks for troubleshooting a system

Virtualization software technologies (KVM, VMware, Xen, User-mode Linux, etc.) allow running a virtual instance of a system (i.e. a Virtual Machine) in a layer abstracted from the actual hardware. A **hypervisor** (either bare-metal aka type-1 or software/hosted aka type-2) installed on the host machine allows running multiple guest OSes with different kernels and their applications; these OSes coexist separately as they were on dedicated machines.
In **full virtualization** the hardware is fully simulated. In **paravirtualization**, hardware is not simulated; guest applications are executed in their own isolated domains as if they were running on separate systems, but need to be specifically modified to run in that environment.

KVM (Kernel-based Virtual Machine) is a virtualization infrastructure for the Linux kernel that allows it to function as a hypervisor. It was introduced in version 2.6.20 of the Linux kernel.
Red Hat Virtualization, formerly known as Red Hat Enterprise Virtualization (RHEV), is based on KVM.

`/etc/libvirt/qemu/`	Directory containing the XML files that define VMs properties. `libvirtd` must be restarted after modifying an XML file
`/var/lib/libvirt/`	Directory containing files related to the VMs
`virt-manager`	KVM GUI
`virt-install --prompt`	Interactive command-line program to create a VM

`virt-install -n` *vmname* `-r 2048 \`
`--disk path=/var/lib/libvirt/images/vmname.img \`
`-l /root/vmstuff/inst/ \`
`-x "ks=/root/vmstuff/kickstart.cfg"`
	Create a VM with 2 Gb of RAM, specifying path of virtual disk, location of installation files, and (as extra argument) the Kickstart configuration to use

`virt-clone --prompt`	Interactive command-line program to clone a VM. A VM must be shut off or paused before it can be cloned
`virt-clone -o` *vmname* `-n` *vmclonename*	Clone a VM
`virsh`	Interface for VM management
`virsh list --all`	List all VMs present on the system
`virsh start` *vmname*	Start a VM
`virsh destroy` *vmname*	Brutally shut down a VM
`virsh shutdown` *vmname*	Gracefully shut down a VM
`virsh autostart` *vmname*	Set a VM to be automatically started when the system boots. Done by symlinking the VM to `/etc/libvirt/qemu/autostart/`
`virsh autostart --disable` *vmname*	Disable the autostart of a VM at system boot
`virsh edit` *vmname*	Edit the XML file defining a VM's properties
`virt-what`	Detect whether the current machine is a VM

In **containerization** aka **OS-level virtualization** (Docker, rkt, OpenVZ, Virtuozzo, etc.) the kernel allows the existence of multiple isolated user space instances, called **containers**. A container is a sandboxed software application packaged with all its dependencies and its configuration so that it is able to run in any environment. There is no hypervisor; all containers are run by the container runtime engine, which is placed on top of the OS.
Containerization is more lightweight and efficient that virtualization, because programs in OS-level virtual partitions can use the OS's normal system call interface, with no need for emulation. However, it is also less flexible as it can only run guest OSes that share the same kernel version.

Docker is a PaaS platform that implements containerization.
Docker uses a **Dockerfile** as a blueprint to build an **image**, which is a read-only file containing the source code, libraries, and dependencies of an application. A **container registry** or **Docker registry**, identified by *hostname*/*repository*:*tag*, is used to push (store) or pull (retrieve) images. Then, the image is run, resulting in a **container** (in execution).
Images are built layer after layer, and can also share common layers; this optimizes disk space and network bandwidth when pushing or pulling large images.
Docker registries are usually cloud-based; the default registry is Docker Hub, the primary and largest library and community for Docker images. A **Docker repository** is a collection of Docker images with the same name and different tag; it can be considered as the combination of a registry and an image.
A container can create, modify, and delete files; however, these changes are isolated to that container and lost when the container is removed. A **volume** allows connecting specific filesystem paths of the container to the filesystem on the host machine, so to ensure persistence of files created while the container is running. A named volume has its host mountpoint decided automatically by Docker, while bind mounts allow choosing the host mountpoint.
Creating a **network** allows to have multi-container applications, as only containers in the same network can communicate.
Docker Compose is a tool to define and share multi-container applications. It uses a YAML file to define all services so that they can be operated with a single command.

docker	Docker CLI
docker build -t *image*:v1 */path*	Create an image from the Dockerfile, using *path* (where the Dockerfile is located) as the build context
docker run *image*:v1	Run an image as a container
docker push *image*:v1	Push an image to the previously specified registry
docker pull *image3*	Pull an image from the previously specified registry
docker images	List the created images
docker scan *image*	Scan an image for security vulnerabilities (via Snyk)
docker image history *image*	Show the layers composing an image
docker tag *image*:v1 *image2*:v1	Create a new tag for the same image. Both images will have the same image ID
docker ps	List all running containers
docker ps -a	List all containers that ran and exited successfully
docker start *contID*	Start a container
docker stop *contID*	Stop a container
docker stop $(docker ps -q)	Stop all running containers
docker logs *contID*	Show the logs for a container
docker rm *contID*	Remove a container. A container must be in a stopped state to be removed
docker volume create *volname*	Create a named volume
docker volume inspect *volname*	Show info about a named volume, including its mountpoint on the host machine
docker network create *app*	Create a network

Dockerfile example	
FROM ubuntu:18.04	Create a layer from the "ubuntu:18.04" Docker image
COPY . /app	Add files from the current directory on the Docker client
RUN make /app	Run the "make" command to build the application
CMD python /app/app.py	Run a command inside the container

Container orchestration helps in the provisioning and deployment of containers, provides scaling and load balancing, ensures redundancy and availability, and allows performing scheduling and health checks.

Kubernetes, an open source software, is the most used container orchestration platform.

Kubernetes cluster Composed of:	**control plane**	Manages the cluster and consists of:	**Kubernetes API server**	Used for all communications, both intra-cluster and with external components
			etcd	Key-value store containing the cluster state and configuration
			Kubernetes scheduler	Determines on which nodes should the workload run by assigning Pods to them
			Kubernetes controller manager	Monitors the cluster state and ensures it matches the desired state
			cloud controller manager	Allows the cluster to interact with cloud providers
	nodes	Are physical machines or VMs that serve as workers; run the applications, are created by the cloud provider, and consist of:	**kubelet**	Agent that runs and monitors the Pods, and communicates with the control plane
			Kubernetes proxy	Allows communications to Pods

Kubernetes **objects** are persistent entities with two properties: **spec** (provided by the user, describing the desired state) and **status** (provided by Kubernetes, describing the current state). The principal Kubernetes objects are:

Pod	Represents a process running in the cluster i.e. a single instance of an application deployed in the cluster. It usually wraps a single container
ReplicaSet	Ensures that a given number of redundant Pods are running at any time
Deployment	Provides declarative updates to an application. It can deploy a Pod or a ReplicaSet, perform updates and rollbacks, and provide scaling
Namespace	Provides a scope for names and is a way to create a virtual cluster
Volume	Is a directory on a disk or on a container
ConfigMap	Allows providing configuration data on-the-fly to Pods and Deployments, avoiding hardcoding it in the application
Secret	Similar to a ConfigMap, but is used to hold confidential data; e.g. this data is not printed when using the `kubectl describe` command

`kubectl`	Kubernetes CLI
`kubeadm`	Tool for quickly bootstrapping a cluster. It does not include options for machine provisioning
`minikube`	User-friendly tool to easily set up a small-scale local Kubernetes cluster, for learning or testing purposes

kubectl run nginx --image nginx	Create a Pod running a Nginx container
kubectl create -f nginx.yaml	Create an object based on a configuration template
kubectl apply -f nginx/	Apply all files in a directory
kubectl exec *podname* -- *command*	Run *command* on a Pod
kubectl exec -ti *podname* -- bash	Run a Bash shell session on a Pod
kubectl get *resources*	List the specified resources
kubectl get nodes	List the available nodes in the cluster
kubectl get pods	List Pods
kubectl describe *resources*	Display detailed information about the specified resources
kubectl cluster-info	Display information about the cluster
kubectl logs *podname*	Display logs for a specific Pod

Cloud computing is the on-demand delivery of computing services through the Internet. The cloud provider supplies these services according to different models:

IaaS **(Infrastructure as a Service)**	Provides virtual machines, storage, load balancing, networking, etc.
PaaS **(Platform as a Service)**	Provides OS, databases, development environments, web servers, etc.
SaaS **(Software as a Service)**	Provides access to ready-to-use application software
Serverless computing	Provides computing execution power, by provisioning computing resources (virtual machines, storage, and network) as needed, transparently to the user

Red Hat OpenShift Container Platform is a hybrid cloud PaaS solution built on Kubernetes and RHEL.
OpenShift provides developer services, application services, and platform services e.g. service mesh, pipelines for CI/CD (Continuous Integration / Continuous Delivery or Deployment), and full stack logging. It ships packaged with OKD (Origin Kubernetes Distribution).

`oc`	OpenShift CLI. It includes `kubectl`

OpenStack is an open standard cloud computing platform, usually deployed as a IaaS solution for public and private clouds.
It has a modular architecture, composed of many elements:
- **Aodh**, a rule-based alarm service;
- **Barbican**, a REST API for management of keys and secrets;
- **Ceilometer**, for telemetry;
- **Cinder**, a block storage service;
- **Designate**, a REST API for DNS management;
- **Glance**, a service to upload and distribute images;
- **Heat**, an orchestration service;
- **Horizon**, a web-based dashboard;
- **Ironic**, for provisioning bare metal servers;
- **Keystone**, for identity and authentication;
- **Magnum**, for container orchestration;
- **Manila**, a shared file system;
- **Mistral**, to manage workflows;
- **Neutron**, to manage networking;
- **Nova**, for provisioning of computing instances (either VMs or bare metal servers);
- **Sahara**, to provision Hadoop clusters;
- **Searchlight**, an ElasticSearch-based search tool for OpenStack cloud services;
- **Swift**, a distributed object store;
- **Trove**, a relational and non-relational database engine;
- **Vitrage**, the OpenStack Root Cause Analysis service for organizing and handling alarms;
- **Zaqar**, a cloud messaging service with REST API.

In cloud-native development, a **service mesh** is a dedicated layer to make communications between microservices secure and reliable.
CI/CD (Continuous Integration / Continuous Delivery or **Deployment)** is a method to frequently deploy applications in production. It is based on the automation of new code build, test, merge, release to repository, and deployment.

Kerberos is an authentication protocol that allows hosts to authenticate each other over an insecure network.

The central authentication entity is the **Key Distribution Center (KDC)**, composed of three parts: the database, the Authentication Server, and the Ticket Granting Server.
The **database** stores entries associated with users, hosts, and services. Each entry is called a **principal** and is in the form *username*/*instance*@*REALM* (for users) or *service*/*hostname*@*REALM* (for services). A **realm** is an authentication administrative domain. A trust relationship between different realms allows users from a realm to authenticate and access the services of another realm, via **cross-authentication**.
The **Authentication Server (AS)** replies to the initial authentication request from the client by issuing a **Ticket Granting Ticket (TGT)**.
The **Ticket Granting Server (TGS)** issues service tickets to clients that own a valid TGT. A **ticket** is encrypted with the secret key of the service it is intended for, has a limited validity (10 hours by default), and contains a **session key** (which is a secret shared between the client and the service). The client will then submit the ticket to an application server in order to prove its identity. Along with the ticket, the client submits an **authenticator** packet containing the user principal and the timestamp, encrypted with the session key.

A client authenticates via Kerberos to an application server through the following steps:
1. The client contacts the AS, making an initial user authentication request
2. The AS replies to the client, sending a TGT (encrypted with the TGS's secret key) and a session key (encrypted with the user's secret key)
3. The client contacts the TGS, sending the TGT and an authenticator (encrypted with the session key)
4. The TGS replies to the client, sending the requested service ticket (encrypted with the service's secret key) and a service session key (encrypted with the session key)
5. The client contacts the application server, sending the service ticket and an authenticator (encrypted with the service session key)

A **keytab** (key table) stores keys for principals. A keytab is usually a file, named as `FILE:/`*`path`*`/krb5.keytab`.
Each entry in a keytab consists of: timestamp, principal name, key version number, encryption type, and encryption key.
The keytab file is present in any host that uses Kerberos.

`/etc/krb5/kadm5.keytab`	Keytab file on the KDC
`/etc/krb5/krb5.keytab`	Keytab file on application servers providing kerberized services

User commands:

`kinit`	Request a TGT for a principal and store it in the credential cache
`klist`	List principal and tickets contained in the credentials cache, or the keys contained in a keytab file
`kswitch`	Switch to another credential cache
`kdestroy`	Destroy the credential cache, deleting all tickets
`kvno`	Acquire a service ticket for a principal and print out its key version number
`kpasswd`	Change a principal's password
`ksu`	Kerberos version of `su`
`krb5-config`	Print information useful for compiling and linking programs against the installed Kerberos libraries

Administration commands:

`kadmin`	Administer a Kerberos system (via `kadmind`)
`kadmin.local`	Administer a Kerberos system (via the local KDC database)
`kadmind`	Start the Kerberos administration server
`krb5kdc`	Manage the AS and the KDC
`kdb5_util`	Manage the Kerberos database
`kdb5_ldap_util`	Manage realms, Kerberos services, and ticket policies
`ktutil`	Edit a keytab
`k5srvutil`	Edit keys stored in a keytab
`kprop`	Propagate the Kerberos database from the primary KDC server to a replica KDC server
`kpropd`	Listen for and apply updates from `kprop`. Runs on the replica KDC server
`kproplog`	Display the log of the Kerberos database updates

Other commands:

`sclient` `sserver`	Simple client and server, useful for testing or demo of Kerberos authentication

The **Name Service Switch (NSS)** is a scheme that allows the local machine to connect and use different name resolution mechanisms e.g. local files, LDAP, DNS, NIS (Network Information Service), NIS+.

`/etc/nsswitch.conf` NSS configuration file.
Each line specifies a database name, followed by the list of possible sources, which will be tried in order to perform name resolution.

```
passwd:     files ldap
shadow:     files
group:      files ldap

hosts:      dns nis nisplus files

ethers:     files nis
netmasks:   files nis
networks:   files nis
protocols:  files nis
rpc:        files nis
services:   files nis

automount:  files
aliases:    files
```

`getent`	Get entries from NSS libraries
`getent passwd` *`user`*	Get *user*'s password entry
`getent group` *`groupname`*	Get entries matching the group *groupname*

SSSD (System Security Services Daemon) is a set of daemons providing local or remote identity authentication.
It is derived from the FreeIPA project.
SSSD features its own NSS (Name Service Switch) and PAM (Pluggable Authentication Module) client interfaces, and has its own cache for offline support. Furthermore, it is capable to interface and query different types of directories, databases, and frameworks such as NIS, LDAP, Kerberos, etc.

`/etc/sssd/sssd.conf` `/etc/sssd/conf.d/*`	SSSD configuration files

Identity Management (IdM) is a framework of policies and technologies to ensure that the proper people have access to the proper resources. Similar frameworks are **Identity and Access Management (IAM)** and **IPA (Identity, Policy, and Audit)**.
Single Sign-On (SSO) is an authentication scheme that allows a user to log in to multiple independent services using one set of credentials. It is a subset of **Federated Identity Management (FIdM)**, which handles identity federation i.e. the linking of multiple identities of a user across multiple IdM systems.

Authentication procedures, and especially SSO, make large use of **SAML (Security Assertion Markup Language)**, an open standard, XML-based markup language. SAML exchanges authentication and authorization data between a subject aka principal (i.e. a user), an identity provider, and a service provider. SAML is build upon XML, HTTP, and SOAP.

The **OAuth** open standard is designed specifically to operate with HTTP. OAuth provides secure delegated access i.e. a way for resource owners to authorize third-party applications (consumers) to access their resources from a service provider without disclosing secret credentials; this is done by the means of access tokens. Its latest version is OAuth 2.0.
OIDC (OpenID Connect) is an authentication layer built on top of OAuth 2.0.

Keycloak is an open source IAM and SSO solution, and the upstream project for **Red Hat SSO**.
It supports several standard protocols for authentication and authorization, such as SAML, OAuth 2.0, and OIDC.

FreeIPA is an open source IdM system, and the upstream project for **Red Hat Identity Management**.
Its main components are: 389 Directory Server (LDAP server), Dogtag Certificate System (CA), Kerberos, SSSD, NTP, and bind-dyndb-ldap (for integration with DNS).
It features a web interface (Web UI, built as a JavaScript Single Page Application) as well as a CLI (`ipa`).

`ipa`	FreeIPA CLI

The **Dogtag Certificate System** is an open source Certification Authority, written in Java and running on Tomcat.

It is composed of the following six subsystems:

Certificate Authority (CA)	Issues, renews, revokes, and publishes certificates. It also creates and publishes CRLs
Registration Authority (RA)	Authenticates enrollment requests and forwards them to the CA to generate a certificate
Key Recovery Authority (KRA) aka **Data Recovery Manager (DRM)**	Stores private keys. It can also provide server-side key pair generation
OCSP Manager	Provides OCSP (Online Certificate Status Protocol) functionalities i.e. determine the state of a certificate, and particularly its revocation status, without the need to check a CRL
Token Key Service (TKS)	Manages the master keys used to establish secure channels to the token management system, allowing e.g. smart card tokens to communicate securely with the TPS
Token Processing System (TPS)	Provides RA functionality in the token management system, and establishes secure channels between the client (e.g. smart card management infrastructure) and the backend subsystems (CA, KRA, and TKS)

Git is an open source version control system with a small footprint and very high performances. A Git directory is a complete repository with full history and version tracking abilities, independent of any remote repository.
Git commits are identified by a 40-hex-digit hash number, usually shortened to 7 digits, or even less if unambiguous.

`git init`	Initialize the current directory as a repository

`git clone` *repo*	Clone a remote repository.
	repo can be a URL (SSH, HTTP, HTTPS, FTP, FTPS, Git) or a local path e.g.
	`ssh://user@example.com:8888/path/to/repo.git`
	`git://example.com:9999/path/to/repo.git`
	`/path/to/repo.git`

`git checkout` *branch*	Start working into an already existing *branch*
`git checkout -B` *branch*	Create *branch* and start working into it
`git checkout --` *file*	Discard local changes done to *file*
`git checkout` *branch file*	Copy *file* from *branch* to the current branch, and add it to the staging area
`git pull`	Pull the changes from the remote repository branch to the local branch
`git add` *file*	Add *file* to the staging area (i.e. content staged for the next commit), hence starting to track it
`git restore --staged` *file*	Remove *file* from the staging area, undoing the command `git add` *file*
`git add .`	Add all modified files to the staging area
`git rm` *file*	Remove *file* from the content staged for the next commit
`git status`	See the status (e.g. files changed but not yet staged) of the current branch
`git commit -m "`*Message*`"`	Commit all staged files in the current branch
`git commit -am "`*Message*`"`	Add all changed files to the staging area in the current branch, and commit them
`git merge` *branch*	Merge changes made on *branch* to the master branch
`git push`	Push the local commits from the current branch to the remote repository
`git push origin` *branch*	Push the local commits from *branch* to the remote repository
`git revert` *commit*	Revert a specific commit
`git branch`	Show local branches
`git branch -r`	Show remote branches
`git branch -a`	Show remote and local branches
`git branch -a --contains` *commit*	Show on which branch was done a specific commit number
`git branch -d` *branch*	Delete a local branch (which must have been merged in its upstream branch)
`git branch -D` *branch*	Delete a local branch (irrespective of its merged status)

Git - search and configuration

`git diff`	Show the differences between local and remote branch
`git diff` *commit1 commit2*	Show the differences between two commits
`git diff` *branch1 branch2*	Show the differences between two branches
`git diff` *branch1 branch2 file*	Show the differences between two branches for a specific file
`git log --all --` *file*	Show the commits which involved *file*, across all branches

`git log -p --all -S '`*string*`'`
`git log -p --all -G '`*regex*`'`
	Show the commits whose added or deleted lines contain a specific word

`git grep` *string* `` `git show-ref --heads` ``	Search for *string* across all branches' heads (i.e. in the latest content only, and not in all the previous commits)
`git config --list`	Get all currently set options and their values in the Git configuration
`git config` *option*	Get the value of *option*
`git config user.name` *name*	Set your username
`git config user.email` *email*	Set your email address

Vagrant is an open source software that allows building and maintaining lightweight and portable virtual environments for software development. It relies on an underlying virtualization or containerization solution e.g. VirtualBox, KVM, VMware, or Docker.

`vagrant -h`	Print the list of commands recognized by Vagrant
`vagrant` *`command`* `-h`	Print help about the Vagrant *command*
`vagrant init hashicorp/precise64`	Initialize the current directory as a specific Vagrant environment (in this case, Ubuntu 12.04 64-bit) by creating a Vagrantfile on it
`vagrant up` *`vmname`*	Start a guest virtual machine and do a first provisioning according to the Vagrantfile
`vagrant provision` *`vmname`*	Provision a virtual machine
`vagrant ssh` *`vmname`*	Connect via SSH to a virtual machine
`vagrant halt` *`vmname`*	Shut down the virtual machine
`vagrant destroy` *`vmname`*	Delete the virtual machine and free any resource allocated to it
`vagrant status`	Print the status of the virtual machines currently managed by Vagrant
`vagrant global-status`	Print the status of all Vagrant environments on the system, by reading cached data. Completes quickly but results may be outdated
`vagrant global-status --prune`	Print the status of all Vagrant environments on the system, after rebuilding the environment information cache. Results are always correct but completion takes longer

The directory containing the Vagrantfile on the host machine can be accessed on the guest machine at `/vagrant`.

Ceph is an open source solution for a storage cluster, providing redundancy and high availability, with a single system for access to object storage, block device storage, and file storage.
Ceph is based on RADOS (Reliable Autonomic Distributed Object Store). Storage and retrieval of data are determined by the CRUSH (Controlled Replication Under Scalable Hashing) algorithm, which builds a hierarchical map of the cluster and assigns data to pseudorandomly-chosen OSDs; this improves scalability, while avoiding performance bottlenecks and Single Points of Failure.

The components of a **Ceph Storage Cluster** are:

Ceph OSDs **(Object Storage Daemons)**	Store data and handle data replication, recovery, and rebalancing. At least 3 OSDs are usually required
Ceph Monitors	Maintain maps of the cluster state and handle daemon-client authentication. Uses the Paxos parliament protocol. At least 3 Monitors are usually required
Ceph Managers	Track runtime cluster metrics and allow access to cluster information. At least 2 Managers are usually required
Ceph MDS **(Metadata Server)**	Stores Ceph File System metadata in order to relieve burden from the cluster. Required when running Ceph File System clients

Ceph File System aka **CephFS** is a POSIX-compliant file system built on top of RADOS.
Ceph Object Storage supports interfaces to the Amazon S3 RESTful API and the OpenStack Swift API. Via the Ceph Object Gateway, it provides a RESTful gateway to a Ceph Storage Cluster.
Ceph Block Device is implemented through RBD (RADOS Block Device) images, which are thin-provisioned and store data striped over multiple OSDs.

`ceph-osd`	Object Storage daemon for CephFS
`ceph-mon`	Cluster Monitor daemon for CephFS
`ceph-mgr`	Manager daemon. It is recommended to place Manager and Monitor daemons in the same nodes
`ceph-mds`	Metadata Server daemon

`ceph`	Ceph administration tool for deploying and managing a cluster. CLI tool for Cephadm (since v15.2.0)
`ceph-authtool`	Management tool for Ceph keyring files (used for authentication)
`ceph-volume`	Deploy logical volumes as OSDs
`ceph-clsinfo`	Display information about a specific class object
`ceph-run`	Restart a daemon
`ceph-conf`	Display information about Ceph configuration
`osdmaptool`	Manipulate OSD cluster maps, and export or import CRUSH maps
`monmaptool`	Manipulate Monitor cluster maps
`crushtool`	Create, compile, decompile, and test CRUSH map files
`rados`	RADOS utility

Puppet is a software configuration management tool, based on a client-server architecture. It works as follows:
1. A **Puppet agent** (client, running as `root` on each managed node) periodically gathers information (**facts**) about the local node state via the **Facter** system inventory tool.
2. The Puppet agent then communicates this information to the **Puppet master** (server, running as the `puppet` user and listening on TCP port 8140).
3. The Puppet master sends back to the Puppet agent a **catalog** containing the desired configuration for that node.
4. The Puppet agent applies the needed changes (which are idempotent) so that the node's configuration converges with the desired configuration, and sends back a report to the Puppet master.

Puppet configurations are based on **resources** (e.g. "package", "service", "file", "user"). For each resource, a list of **attributes** is specified, with the desired value for each attribute.
Each resource type is implemented through **providers** (e.g. `yum`, `rpm`, `apt`, `opkg` for the resource "package").
Resources managed together as a single unit can be grouped into **classes**; classes are contained in **manifests** which are files with the `.pp` extension.
Modules are directories containing self-contained pieces of configuration and classes for a specific complex setting, e.g. an Apache webserver or a MySQL server.

`/etc/puppet/puppet.conf`	Configuration file (Open Source Puppet)
`/etc/puppetlabs/puppet/puppet.conf`	Configuration file (Puppet Enterprise)
`facter`	Gather the facts about the managed node, returning a list of key-value pairs
`puppet agent`	Main Puppet client. Retrieves the node's desired configuration from the Puppet master and applies it
`puppet agent --enable`	Enable the Puppet agent on the node
`puppet agent --disable "`*`Reason for disabling`*`"`	Disable the Puppet agent on the node
`cat $(puppet config print vardir)/state/agent_disabled.lock`	Print the reason why the Puppet agent is currently disabled. If the Puppet agent is enabled instead, this lockfile does not exist
`puppet agent --noop`	Perform a dry run, displaying the changes that Puppet would have applied without actually applying them

`puppet --version`
`puppet agent --version`
`puppet master --version`
	Show version of different Puppet components

`puppet module list`	List all modules installed in Puppet
`puppet resource user `*`username`*	Inspect the state of the resource "user" with respect to *username*
`puppet resource service httpd enable=false`	Modify the state of the resource "service" (in this case, disable the HTTP server)
`puppet describe user`	Show information about the resource "user"
`puppet describe --list`	List all resource types
`puppet describe user --providers`	Return the list of providers for the resource "user"
`puppet apply `*`modulename`*`/init.pp`	Apply a manifest one time only
`puppet cert `*`operation`*	Manage the SSL certificates used for communications between master and agents
`puppet master --configprint basemodulepath`	Display the specified configuration value

Manifest and other related files

basemodulepath/*modulepath*/ssh/manifests/init.pp

Default class for the SSH service. Ensures that SSH is installed and running

```
class ssh {
  package { 'ssh':
    ensure => present,
    name => $::operatingsystem ? {
      'Redhat' => 'openssh',
      'Ubuntu' => 'ssh',
      default  => 'openssh',
    },
  }
  file { '/etc/ssh/sshd_config':
    ensure  => file,
    owner   => 'root',
    group   => 'root',
    mode    => '0644',
    require => Package['openssh'],
    source  => 'puppet:///modules/ssh/sshd_config',
  }
  service { 'sshd':
    ensure  => running,
    enable  => true,
    require => File['/etc/ssh/sshd_config'],
  }
  notify { 'Test message for SSH class': }
}
```

basemodulepath/*environment*/hiera.yaml

Hiera configuration file

```
---
ntp::servers:
  - time.example.org
  - 0.pool.ntp.org
```

basemodulepath/*modulepath*/templates/host.erb

ERB (Embedded Ruby) template. Allows embedding dynamic variables that can be resolved in the calling manifest

```
The IP address is <%= @ipaddress %>
```

Node definition. Assigns one or more classes to a node

```
node 'n1.example.org' {
  include ssh
  include apache
}
```

Resource relationship	

`package { 'openssh-server':`
`  ensure => present,`
`  before => File['/etc/ssh/sshd_config'],`
`}`
or
`file { '/etc/ssh/sshd_config':`
`  ensure  => file,`
`  mode    => '0600',`
`  source  => 'puppet:///modules/sshd/sshd_config',`
`  require => Package['openssh-server'],`
`}`
	Ensures that the SSH server package is installed before the SSH daemon configuration file

`file { '/etc/ssh/sshd_config':`
`  ensure => file,`
`  mode   => '0600',`
`  source => 'puppet:///modules/sshd/sshd_config',`
`  notify => Service['sshd'],`
`}`
or
`service { 'sshd':`
`  ensure    => running,`
`  enable    => true,`
`  subscribe => File['/etc/ssh/sshd_config'],`
`}`
	Notifies the SSH service (restarting it) any time the SSH daemon configuration file is changed

Ansible is an open source tool (made by Red Hat) for configuration management, software provisioning, and application deployment. It is agentless and connects to the managed machines via SSH pubkey authentication, requiring only OpenSSH and Python to be installed on the managed nodes.
The configuration for a managed node is specified in a **playbook**, written in YAML. A playbook contains one or more **plays** to be run in sequence, each of which lists a number of small specific **tasks** to be run in sequence. Each task calls a **module**, which describes the desired state of the system and executes the operation to do so (e.g. start the Apache server, copy a file, verify that a package is installed, rename a database table). A module should be idempotent i.e. it should set the system always in the same state regardless of how many times it is run.
Handlers are tasks that are run only when notified (via the command **`notify`**), when a change is made on a machine.
The **inventory** lists and categorizes all hosts on which tasks have to be executed. It is also possible to define **roles** to categorize hosts and automatically load variables, tasks, handlers, and other artifacts pertaining to that role.
When a playbook is run, first it collects system and environment information (**facts**) which is then stored in multiple variables named ansible_*varname*.

Ansible uses the Jinja template engine to enable access to tests, filters, variables, and dynamic execution.
Also available is a set of plugins that expand Ansible's core functionalities.

Ansible Tower is a web console for centralized configuration and management of Ansible elements. It provides web services, a REST API, RBAC, job scheduling, Tower clustering, and graphical inventory management.

Red Hat Ansible Automation Platform allows implementing enterprise-wide automation. Its upstream open source project is **AWX**.

/etc/ansible/hosts	Inventory file, containing the list of hosts managed by Ansible. Can be in INI or YAML format
ansible *hosts* -m *module* -a *options*	Run the *module* with the specific *options* on the *hosts*
ansible *hosts* -m ansible.builtin.copy \\ -a "src=*/path/to/file* dest=/tmp/"	Copy a file to the *hosts*
ansible *hosts* -m ansible.builtin.yum \\ -a "name=httpd state=latest"	Ensure that *hosts* have the `httpd` package installed and updated to the latest version
ansible *hosts* -a "/sbin/shutdown"	Shutdown the *hosts*
ansible all -m ping	Ping all hosts in the inventory (by executing the "ping" module)
ansible all -m ansible.builtin.setup	Show all facts
ansible-playbook *playbook*.yml	Apply the specified playbook
ansible-lint *playbook*.yml	Check the syntax of the specified playbook
ansible-pull	Pull a remote copy of Ansible on each managed node and download the playbooks from a source repository. This inverts the default push architecture of Ansible into a pull architecture

`ansible` command line options	
-m *module*	Run the specified *module*. If not specified, Ansible uses the default module "ansible.builtin.command"
-a *options*	Apply the specified *options* when running the module
-f *n*	Fork *n* processes when running the module. If not specified, default is 5
-u *user*	Run the module as *user*. If not specified, default is current user
--become	Run the module as root
--become --ask-become-pass	Run the module as root, asking for the password

Ansible - playbook example part 1

Inventory and variables files

`hosts`
Inventory file, defining two groups

```
[webservers]
10.0.1.17
10.0.1.18

[dbservers]
10.0.2.42
```

`group_vars/all`
Variables applied to all host groups

```
---
httpd_port: 80
ntpserver: 192.168.0.13
repository: https://foobar.org/repo/website.git
```

`group_vars/dbservers`
Variables applied to hosts in the "dbservers" group

```
---
mysqlservice: mysqld
mysql_port: 3306
dbuser: jdoe
dbname: mydb
dbpassword: mys3cr3t
```

Main playbook file

`lamp.yml`
Playbook that deploys the whole LAMP stack on the site

```
---
- name: Apply common configuration to all nodes
  hosts: all
  remote_user: root

  roles:
    - common

- name: Deploy and configure webservers
  hosts: webservers
  remote_user: root

  roles:
    - web

- name: Deploy MySQL and configure databases
  hosts: dbservers
  remote_user: root

  roles:
    - db
```

Files for role "common"

`roles/common/handlers/main.yml`
Handler for general notifications, called from other plays

```
---
- name: Restart NTP
  service:
    name: ntpd
    state: restarted
```

`roles/common/tasks/main.yml`
General play run on all nodes

```
---
- name: Install NTP
  yum:
    name: ntp
    state: present
  tags: ntp

- name: Configure NTP
  template:
    src: ntp.conf.j2
    dest: /etc/ntp.conf
  tags: ntp
  notify: Restart NTP

- name: Start NTP
  service:
    name: ntpd
    state: started
    enabled: yes
  tags: ntp
```

`roles/common/templates/ntp.conf.j2`
Jinja template for NTP configuration file

```
driftfile /var/lib/ntp/drift
restrict 127.0.0.1
server {{ ntpserver }}
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys
```

Ansible - playbook example part 2

Files for role "db"

`roles/db/handlers/main.yml`
Handler for DB tier notifications

```
---
- name: Restart MySQL
  service:
    name: mysqld
    state: restarted

- name: Restart iptables
  service:
    name: iptables
    state: restarted
```

`roles/db/tasks/main.yml`
Install MySQL, then create database and database user

```
---
- name: Install the MySQL package
  yum:
    name: "{{ item }}"
    state: installed
  with_items:
   - mysql-server
   - MySQL-python

- name: Configure MySQL
  template:
    src: my.cnf.j2
    dest: /etc/my.cnf
  notify: Restart MySQL

- name: Start MySQL
  service:
    name: mysqld
    state: started
    enabled: yes

- name: Insert iptables rule for MySQL
  lineinfile:
    dest: /etc/sysconfig/iptables
    state: present
    regexp: "{{ mysql_port }}"
    insertafter: "^:OUTPUT "
    line: "-A INPUT -p tcp  --dport {{ mysql_port }} -j  ACCEPT"
  notify: Restart iptables

- name: Create database
  mysql_db:
    name: "{{ dbname }}"
    state: present

- name: Create database user and set password
  mysql_user:
    name: "{{ dbuser }}"
    password: "{{ dbpassword }}"
    priv: "*.*:ALL"
    host: '%'
    state: present
```

`roles/db/templates/my.cnf.j2`
Jinja template for MySQL configuration file

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
symbolic-links=0
port={{ mysql_port }}

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Ansible - playbook example part 3

Files for role "web"

`roles/web/handlers/main.yml`
Handler for web tier notifications

```
---
- name: Restart iptables
  service:
    name: iptables
    state: restarted
```

`roles/web/tasks/main.yml` Main task, calls the other two playbooks

```
---
- include: install_httpd.yml
- include: copy_website.yml
```

`roles/web/tasks/copy_website.yml` Copy the code from the git repository

```
---
- name: Copy website from repo
  git:
    repo: "{{ repository }}"
    dest: /var/www/html/

- name: Create the index.php file
  template:
    src: index.php.j2
    dest: /var/www/html/index.php
```

`roles/web/tasks/install_httpd.yml` Install HTTP, PHP, and Git modules

```
---
- name: Install httpd packages
  yum:
    name: "{{ item }}"
    state: present
  with_items:
   - httpd
   - php
   - php-mysql
   - git

- name: Insert iptables rule for httpd
  lineinfile:
    dest: /etc/sysconfig/iptables
    create: yes
    state: present
    regexp: "{{ httpd_port }}"
    insertafter: "^:OUTPUT "
    line: "-A INPUT -p tcp --dport {{ httpd_port }} -j  ACCEPT"
  notify: Restart iptables

- name: Check that httpd is running
  service:
    name: httpd
    state: started
    enabled: yes
```

`roles/web/templates/index.php.j2`
Jinja template for the website root file `index.php`

```
<html>
   <head>
      <title>LAMP stack and website deployed via Ansible</title>
   </head>
   <body>
      <a href=http://{{ ansible_default_ipv4.address }}/index.html>Homepage</a></br>
      <?php
echo "Hostname: " . exec('hostname') . "</br>";
echo "Database list: </br>";
{% for host in groups['dbservers'] %}
$link = mysqli_connect(
'{{ hostvars[host].ansible_default_ipv4.address }}',
'{{ hostvars[host].dbuser }}',
'{{ hostvars[host].dbpassword }}'
) or die(mysqli_connect_error($link));
{% endfor %}
while ($r = mysqli_fetch_assoc(mysqli_query($link, "SHOW DATABASES;"))) {echo $r['Database'] . "\n";}
      ?>
   </body>
</html>
```

Tag		Attributes	
`<h1>` … `<h6>` **Heading**		`align=left|center|right|justify`	Heading alignment †
`<br>` **Line break**	Line break and carriage return		
`<hr>` **Horizontal line**		`align=left|center|right`	Line alignment †
		`noshade`	Solid rendering instead of 3D †
		`size=`*`npixels`*	Line height
		`width=`*`npixels`*`|`*`percent`*`%`	Line width
`<p>` **Paragraph** `<div>` **Section**		`align=left|center|right|justify`	Paragraph or section alignment †
`<span>` **Group**	Group of elements		
`<a>` **Anchor**	Hyperlink	`charset=`*`encoding`*	Character encoding of target URL
		`coords=`*`left,top,right,bottom`*`|` *`cx,cy,radius`*`|`*`x1,y1,...,xn,yn`*	Coordinates of region; depends on `shape`
		`href=`*`url`*	Target URL for the link
		`hreflang=`*`language`*	Language of document at the target URL
		`name=`*`section`*	Name of anchor for document bookmarking
		`rel|rev=alternate|stylesheet|start|next|prev|contents|index|glossary|copyright|chapter|section|subsection|appendix|help|bookmark`	Relationship between this document and the target URL (`rel`) or vice versa (`rev`)
		`shape=rectangle|circle|polygon`	Shape of region
		`target=_blank|_parent|_self|_top`	Destination of target URL
		`type=`*`mimetype`*	MIME type of target URL
`<dl>` **Definition list**			
`<dt>` **Definition term**			
`<dd>` **Definition description**	Description of a definition term		
`<ol>` **Ordered list**		`compact=compact`	List must be more compact †
		`start=`*`firstnumber`*	Number to start the list on †
		`type=A|a|I|i|1`	List numbers type †
`<ul>` **Unordered list**		`compact=compact`	List must be more compact †
		`type=disc|square|circle`	List type †
`<li>` **List item**		`type=disc|square|circle|A|a|I|i|1`	List item type †
		`value=`*`itemno`*	List item value †

† = deprecated

Tag			Attributes	
`<i>`	**Italic**			
`<b>`	**Bold**			

`<s>`
`<strike>`
	Strike-through	Strike-through text †		

`<u>`	**Underlined**	Underlined text †		
`<big>`	**Bigger**			
`<small>`	**Smaller**			
`<sub>`	**Subscript**			
`<sup>`	**Superscript**			
`<tt>`	**Teletype**	Monospaced text		
`<em>`	**Emphasized**			
`<strong>`	**Strong**			

`<del>`
`<ins>`
	Deleted
	Inserted
		Deleted/inserted text	`cite=`*`url`*	URL to document explaining deletion/insertion

			`datetime=`*`yyyy-mm-dd`*	When the text was deleted/inserted
`<pre>`	**Preformatted**		`width=`*`ncharacters`*	Max number of characters per line †
`<code>`	**Code**	Source code text		
`<samp>`	**Sample**	Sample code text		
`<kbd>`	**Keyboard**	Keyboard key		
`<var>`	**Variable**	Variable name		
`<cite>`	**Citation**	Citation block		

`<blockquote>`
`<q>`
	Quotation
	Short quotation
			`cite=`*`url`*	URL to document containing the quote

`<address>`	**Address**	Address block		
`<abbr>`	**Abbreviation**			
`<acronym>`	**Acronym**			
`<dfn>`	**Definition**	Definition term		
`<font>`	**Font**	Font †	`color=rgb(`*`r`*`,`*`g`*`,`*`b`*`)|#`*`rrggbb`*`|`*`color`*	Text color
			`face=`*`fontname`*	Text font
			`size=`*`[1 ... 7]|[-6 ... +6]`*	Text size
`<bdo>`	**Bidirectional override**		`dir=ltr|rtl`	Direction of text: left-to-right or right-to-left
`<xmp>`	**XMP**	Non-formatted text † (ignores other HTML tags)		
other tags		Attributes common to almost all other tags	`class=`*`class|style`*	Class of the element
			`id=`*`id`*	Unique ID of the element
			`style=`*`styledef`*	Inline style definition
			`title=`*`tooltip`*	Text of the tooltip to display
			`dir=ltr|rtl`	Direction of text: left-to-right or right-to-left
			`lang=`*`language`*	Language of the content
			`accesskey=`*`character`*	Keyboard shortcut for the element
			`tabindex=`*`ntab`*	N of tab for the element

† = deprecated

Tag	Attributes	
<img> **Image**	`align=top|bottom|left|middle|right`	Image alignment with respect to surrounding text †
	`alt=`*alternatetext*	Description of the image for text-only browsers
	`border=`*npixels*	Border width around the image †
	`height=`*npixels*|*percent*`%`	Image height
	`hspace=`*npixels*	Blank space on the left and right side of image †
	`ismap=`*url*	URL for server-side image map
	`longdesc=`*url*	URL containing a long description of the image
	`src=`*url*	URL of the image
	`usemap=`*url*	URL for client-side image map
	`vspace=`*npixels*	Blank space on top and bottom of image †
	`width=`*npixels*|*percent*`%`	Image width
<map> **Image map**	`id=`*id*	Unique ID for the map tag
	`name=`*name*	Unique name for the map tag
<area> **Area of image map**	`alt=`*alternatetext*	Description of area for text-only browsers
	`coords=`*left*,*top*,*right*,*bottom*| *cx*,*cy*,*radius*|*x1*,*y1*,...,*xn*,*yn*	Coordinates of clickable area; depends on `shape`
	`href=`*url*	Target URL of area
	`nohref=true|false`	Excludes or includes the area from image map
	`shape=rectangle|circle|polygon`	Shape of area
	`target=_blank|_parent|_self|_top`	Destination of target URL

† = deprecated

Tag	Attributes	
<table> **Table**	align=left|center|right	Table alignment †
	bgcolor=rgb(*r*,*g*,*b*)|#*rrggbb*|*color*	Table background color †
	border=*npixels*	Border width
	cellpadding=*npixels*|*percent*%	Space around the content of each cell
	cellspacing=*npixels*|*percent*%	Space between cells
	frame=void|above|below| lhs|rhs|hsides|vsides|box|border	Visibility of sides of the table border
	rules=none|groups|rows|cols|all	Horizontal or vertical divider lines
	summary=*summary*	Summary of the table for text-only browsers
	width=*npixels*|*percent*%	Table width
<tr> **Table row**	align=left|center|right|justify|char	Horizontal text alignment
	bgcolor=rgb(*r*,*g*,*b*)|#*rrggbb*|*color*	Row background color †
	char=*character*	Character to align text on, if align=char
	charoff=*npixels*|*percent*%	Alignment offset to first character, if align=char
	valign=top|middle|bottom|baseline	Vertical text alignment
<td> **Table cell** <th> **Table header**	abbr=*content*	Abbreviated content in a cell
	align=left|center|right|justify|char	Horizontal text alignment
	axis=*category*	Cell name
	bgcolor=rgb(*r*,*g*,*b*)|#*rrggbb*|*color*	Cell background color †
	char=*character*	Character to align text on, if align=char
	charoff=*npixels*|*percent*%	Alignment offset to first character, if align=char
	colspan=*ncolumns*	Number of columns this cell spans on
	headers=*headerid*	Cell header information for text-only browsers
	height=*npixels*	Cell height †
	nowrap	Text in cell stays on a single line †
	rowspan=*nrows*	Number of rows this cell spans on
	scope=col|colgroup|row|rowgroup	Target for cell header information
	valign=top|middle|bottom|baseline	Vertical text alignment
	width=*npixels*|*percent*%	Cell width †

† = deprecated

7-bit ASCII table

Dec	Hex	Char		Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	
0	0	NUL	Null	32	20	space	64	40	@	96	60	`	
1	1	SOH	Start of heading	33	21	!	65	41	A	97	61	a	
2	2	STX	Start of text	34	22	"	66	42	B	98	62	b	
3	3	ETX	End of text	35	23	#	67	43	C	99	63	c	
4	4	EOT	End of transmission	36	24	$	68	44	D	100	64	d	
5	5	ENQ	Enquiry	37	25	%	69	45	E	101	65	e	
6	6	ACK	Acknowledge	38	26	&	70	46	F	102	66	f	
7	7	BEL	Bell	39	27	'	71	47	G	103	67	g	
8	8	BS	Backspace	40	28	(	72	48	H	104	68	h	
9	9	TAB	Horizontal tab	41	29	)	73	49	I	105	69	i	
10	A	LF	Line feed	42	2A	*	74	4A	J	106	6A	j	
11	B	VT	Vertical tab	43	2B	+	75	4B	K	107	6B	k	
12	C	FF	Form feed	44	2C	,	76	4C	L	108	6C	l	
13	D	CR	Carriage return	45	2D	-	77	4D	M	109	6D	m	
14	E	SO	Shift out	46	2E	.	78	4E	N	110	6E	n	
15	F	SI	Shift in	47	2F	/	79	4F	O	111	6F	o	
16	10	DLE	Data link escape	48	30	0	80	50	P	112	70	p	
17	11	DC1	Device control 1	49	31	1	81	51	Q	113	71	q	
18	12	DC2	Device control 2	50	32	2	82	52	R	114	72	r	
19	13	DC3	Device control 3	51	33	3	83	53	S	115	73	s	
20	14	DC4	Device control 4	52	34	4	84	54	T	116	74	t	
21	15	NAK	Negative ACK	53	35	5	85	55	U	117	75	u	
22	16	SYN	Synchronous idle	54	36	6	86	56	V	118	76	v	
23	17	ETB	End of Tx block	55	37	7	87	57	W	119	77	w	
24	18	CAN	Cancel	56	38	8	88	58	X	120	78	x	
25	19	EM	End of medium	57	39	9	89	59	Y	121	79	y	
26	1A	SUB	Substitute	58	3A	:	90	5A	Z	122	7A	z	
27	1B	ESC	Escape	59	3B	;	91	5B	[	123	7B	{	
28	1C	FS	File separator	60	3C	<	92	5C	\	124	7C	|	
29	1D	GS	Group separator	61	3D	=	93	5D	]	125	7D	}	
30	1E	RS	Record separator	62	3E	>	94	5E	^	126	7E	~	
31	1F	US	Unit separator	63	3F	?	95	5F	_	127	7F	DEL	Delete

Characters 0-31 and 127 are non-printable.

`ascii`
`man ascii`
	Display an ASCII table

`showkey -a`	Prompt for pressing a key and display its ASCII value in decimal, octal, and hex

Confidentiality, **Integrity**, and **Availability** (aka the **CIA triad**) are the basic policies of Information Security. Confidentiality ensures that access to information is limited to people and groups with the correct rights, integrity ensures that information has not been improperly modified, and availability ensures that a system is operable and functioning.

Access control types	
Discretionary Access Control (DAC) aka **need-to-know**	Allows the user that has access to the resource to decide with whom to share it. File access is regulated by user and group permissions. In Linux, this is the standard access model.
Mandatory Access Control (MAC)	A particular user can access a resource only if they have been given explicit access right to it. The end user is not allowed to choose who can access the resource, or to pass privileges. In Linux, this is implemented via SELinux.
Role-Based Access Control (RBAC)	Access permissions are based on the access policies determined by the system. Users are assigned access to resources on a one-to-one basis.
Rule-Based Access Control (RuBAC)	Endpoint devices (e.g. firewalls) verify the requests to access network resources against a set of rules based on IP addresses, port numbers, etc.

A **threat** is an entity, circumstance, or event with the potential to adversely impact a computer system through unauthorized access, destruction, disclosure, modification of data, or Denial of Service.
A **vulnerability** is a weakness in a computer system that could be exploited by a threat source.
A **risk** is the probability that a particular security threat will exploit a system vulnerability, according to the risk equation: risk = threat × vulnerability.
An **attack** is an attempt to gain unauthorized access to a computer system's services, resources, or information; can also be considered an attempt to compromise a computer system's confidentiality, integrity, or availability.

Vulnerability management life cycle	
Creation of a baseline	Identify and prioritize critical assets
↓	
Vulnerability assessment	Identify and prioritize vulnerabilities. Examine the abilities of a system, applications, security procedures, and controls, to withstand an attack
↓	
Risk assessment	Analyze and evaluate risks in order to determine possible incidents, their likelihood, their consequences, and the tolerance of each critical asset for such events. The components of risk assessment are **technical safeguards**, **organizational safeguards**, **physical safeguards**, and **administrative safeguards**
↓	
Remediation	Reduce the severity of vulnerabilities (via action plans, patches, hotfixes, etc.)
↓	
Verification	Verify all the previous phases (via scanners, reports, etc.)
↓	
Monitor	Monitor regularly the system to maintain the required level of security

The **Common Vulnerability Scoring System (CVSS)** assessment consists of three metrics for measuring vulnerabilities: **base metrics**, **temporal metrics**, and **environmental metrics**. Each metric sets a severity score from 1 to 10.

Common Vulnerabilities and Exposures (CVE) is a public list of identifiers for software vulnerabilities.

The **Metasploit Framework** is a well-known tool and the de facto standard for developing and exploiting security vulnerabilities in systems and applications. The GUI equivalent is **Armitage**.

Metasploit has a modular architecture. **Exploit** modules are the basic modules used to encapsulate an exploit.
Payload modules can be **singles** (self-contained), **stagers** (modules that establish a network connection to the attacked system), and **stages** (downloaded by stagers).
Auxiliary modules are used to perform one-time actions such as port scanning, DoS, or fuzzing.
NOP modules generate no-op instructions (0x90 for x86 microchips) used to keep the payload size consistent by padding out buffers.
The steps for carrying out an attack are, in order: configuring an exploit, setting up the exploit options, selecting a target, selecting a payload, and launching the exploit.

`msfconsole`	Launch the Metasploit Framework CLI
`msfupdate`	Update the Metasploit Framework
`msfvenom`	Generate and encode a payload for an exploit. Replaces the old `msfpayload` and `msfencode` tools

Base commands	
`help`	Show the list of Metasploit commands
`help` *`command`*	Show detailed help about *command*
`db_status`	Check database connection status
`creds`	Display all credentials in the database
`use` *`module`*	Load and use *module*
`setg` *`variable value`*	Assign *value* to a global *variable*
`getg` *`variable`*	Get the value of a global *variable*
`unsetg` *`variable`*	Unset the value of a global *variable*
`connect` *`host port`*	Connect to a remote *host* on *port*
`sessions`	Display information about active sessions
`threads`	Display information about background threads and manipulate them
`banner`	Display a Metasploit banner
`history`	Show Metasploit command history

Module commands	
`info`	See information about the currently loaded module
`show payloads`	Show the list of compatible payloads for the currently loaded module
`show options`	Show all options (variables) available for the module, along with their descriptions and set values
`set` *`variable value`*	Assign *value* to a context-specific *variable*
`get` *`variable`*	Get the value of a context-specific *variable*
`unset` *`variable`*	Unset the value of a context-specific *variable*
`check` *`host`*	Check if *host* is vulnerable
`reload`	Reload the module
`rexploit` `rerun`	Reload and execute the module
`exploit` `run`	Execute the module

Meterpreter is an advanced and dynamically extensible payload for a Metasploit attack that provides the attacker with an interactive shell (Meterpreter session) to the target remote machine. This session is obtained by running from Metasploit an exploit with the appropriate payload e.g. `php/meterpreter/reverse_tcp`.
The payload is deployed using in-memory DLL injection.

Meterpreter commands	
`help`	Show a list of all Meterpreter commands
`background`	Send the Meterpreter session to background and return to the MSF CLI
`cat` *`file`*	Display the contents of *file*
`edit` *`file`*	Edit a text *file* (via Vim)
`ls`	List files on the target machine
`ps`	Display processes on the target machine
`download` *`file`*	Download *file* from the target machine
`upload` *`file path`*	Upload *file* from the local machine to the *path* on the target machine
`execute -f` *`command`*	Execute a *command* on the target machine
`resource` *`file`*	Execute on the target machine the Meterpreter commands listed in the local *file*
`shell`	Run an OS shell on the target machine
`sysinfo`	Get information about the target machine: OS, machine name, etc.
`ipconfig`	Display network configuration of the target machine
`localtime`	Display date and time of the target machine
`clearev`	On a MS Windows target machine, clear all Application, Security, and System logs
`webcam_list`	List all operative webcams on the target machine
`webcam_snap`	Take a snapshot from the webcam on the target machine
`quit` `exit`	End the Meterpreter session

Aircrack-ng

Aircrack-ng is a suite of tools for WiFi security. It includes utilities for wireless network sniffing, attack, key cracking, and testing.

`aircrack-ng` *`options file`*

Crack WEP or WPA/WPA2-PSK keys from the capture *file* (in `.cap` or `.ivs` format).
Possible *options*:

`-a` *`n`*	Attack mode (*n*=1 for WEP, *n*=2 for WPA/WPA2-PSK)
`-e` *`essid`*	Specify the Access Point to use
`-K`	Use the Korek WEP cracking technique
`-z`	Use the PTW WEP cracking technique
`-k` *`n`*	Disable the Korek WEP attack number *n* (where *n* is between 1 and 17)
`-n` *`len`*	Specify WEP key length
`-s`	Show WEP key in ASCII while cracking
`-w` *`file`*	Wordlist file to use for WEP or WPA/WPA2 key cracking

`aireplay-ng` *`attack options`*

Replay packets to perform an attack, where *attack* is one of:

`-0`	Deauthentication attack
`-1`	Fake authentication attack
`-2`	Interactive packet replay attack
`-3`	ARP Request replay attack
`-4`	Chopchop attack
`-5`	Fragmentation attack (against WEP)
`-6`	Caffe Latte attack
`-7`	Hirte attack
`-8`	WPA Migration Mode attack
`-9`	Injection test

`airodump` *`options interface`*

Capture packets by listening to the network *interface*.
Possible *options*:

`--ivs`	Save only captured IVs
`-w` *`file`*	Write sniffed packets in a capture *file*
`-o` *`format`*	Use *format* for the capture file: may be `pcap`, `ivs`, `csv`, `gps`, `kismet`, `netxml`, or `logcsv`

How to crack WEP

1.	`airmon-ng start wlan0`	Start the wireless NIC into monitor mode on the same channel as the AP, and test injection capabilities to the AP
2.	`airodump-ng --ivs -w capture wlan0`	Discover the list of active wireless machines. Note ESSID and BSSID of the target AP (let us assume they are respectively *ap_essid* and *ap_mac*). Keep this command running to capture the generated IVs

3.	`aireplay-ng -1 0 -e ap_essid \`
	`-a ap_mac -h mac wlan0`
		Do a fake authentication with the target AP using your NIC (with MAC address *mac*)

4.	`aireplay-ng -3 -b ap_mac -h mac wlan0`	To capture a large number of IVs in a short time, run `aireplay-ng` in ARP Request replay mode
5.	`aircrack-ng -s capture.ivs`	Once `airodump-ng` has captured at least 50000 IVs, crack the WEP key

How to crack WPA-PSK or WPA2-PSK

1.	`airmon-ng start wlan0`	Put the wireless NIC into monitor mode
2.	`airodump-ng -w capture wlan0`	Discover the list of active wireless machines. Note BSSID of the target client (let us assume it's *cl_mac*). Keep this command running
3.	`aireplay-ng --deauth 11 -a cl_mac`	Deauthenticate the client. The client will try to authenticate again, and `airodump-ng` will capture the authentication packet sent during the WPA/WPA2 four-way handshake
4.	`aircrack-ng -a 2 -w capture.cap`	Analyze the capture dump to crack the WPA/WPA2 key

The only way to crack WPA/WPA2 is to sniff the Pairwise Master Key associated with the four-way handshake authentication process. Therefore it is important to choose a complex WPA/WPA2 random password at least 20 characters long.

How to decloak a hidden SSID

1.	`airmon-ng start wlan0`	Put the wireless NIC into monitor mode
2.	`airodump-ng --ivs -w capture wlan0`	Discover the list of active wireless clients. Find the one where the ESSID is hidden (it shows only its string length) and note its BSSID (let us assume it's *cl_mac*). Keep this command running
3.	`aireplay-ng --deauth 11 -a cl_mac`	Deauthenticate the client
4.	The output of `airodump-ng` will now show the hidden ESSID	

How to perform a MitM attack

1.	`airmon-ng start wlan0`	Put the wireless NIC into monitor mode
2.	`airodump-ng --ivs -w capture wlan0`	Discover the list of active wireless clients. Note BSSID of the target client (let us assume it's *cl_mac*). Note ESSID and BSSID of the AP you are currently accessing (let us assume they are respectively *ap_essid* and *ap_mac*). Keep this command running
3.	`aireplay-ng --deauth 11 -a cl_mac`	Deauthenticate the client

4.	`aireplay-ng -1 0 -e ap_essid \`
	`-a ap_mac -h cl_mac wlan0`
		Perform a fake authentication attack, associating your NIC with the AP you are currently accessing

The **Firmware Mod Kit** is a toolkit to extract, deconstruct, modify, rebuild, and flash firmware images for Linux-based routers, IoT devices, embedded devices, and most other devices that use common firmware formats and filesystems such as TRX/uImage and SquashFS/CramFS.
It can be used by an attacker after compromising a device, to maintain access.

`extract-firmware.sh`	Extract the firmware
`build-firmware.sh`	Rebuild the firmware
`ddwrt-gui-extract.sh`	Extract Web GUI files from extracted DD-WRT firmware
`ddwrt-gui-rebuild.sh`	Restore modified Web GUI files to extracted DD-WRT firmware

`arpspoof`	ARP spoofing tool
`arpoison`	ARP cache update utility. Can be used to craft custom ARP packets
`arpstraw`	ARP spoofing detection tool
`arpon`	ARP handler inspector. Useful to detect ARP spoofing
`arpwatch`	Tracker of MAC to IP address pairings. Useful to detect ARP spoofing
`ettercap`	Network security tool for ARP poisoning and man-in-the-middle attacks over the LAN
`macchanger`	Tool to perform MAC spoofing. Changes the NIC to a random MAC address
`macof`	DoS tool for MAC flooding
`yersinia`	DoS tool for DHCP starvation attack
`dhcpstarv`	DoS tool for DHCP starvation attack
`dhcpig`	DoS tool for DHCP starvation attack
`nbtscan`	Network scanner for NetBIOS name information
`p0f`	Passive traffic fingerprinting tool. Identifies hosts performing any incidental TCP/IP communication
`bannergrab`	Network service banner grabbing tool. Sends a trigger to the service and collects basic information
`nscan`	Fast network scanner optimized for Internet-wide scanning
`zmap`	Fast single packet network scanner. An improved version of `nmap` designed for Internet-wide scanning
`masscan`	Fast Internet port scanner
`fragrouter`	IDS evasion toolkit. Reroutes network traffic
`dnsspoof`	DNS spoofing tool. Forges replies to DNS queries on the LAN
`responder`	LLMNR, NBT-NS, and MDNS poisoner
`scapy`	Packet manipulation tool. Features packet forging, decoding, injection, and other network operations
`mitmf`	Framework for MitM attacks
`loki`	Firewall evasion tool that encapsulates commands into the payload of ICMP packets
`hts`	HTTPTunnel server. Used in conjunction with the HTTPTunnel client `htc` to tunnel network connections through pure HTTP traffic (GET and POST requests), hence bypassing restrictive firewalls or proxies
`htc`	HTTPTunnel client
`iodined`	Firewall evasion tool. Tunnels IPv4 traffic through a DNS server. Replaces the obsolete `tcp-over-dns`
`iodine`	Client for `iodined`
`loic`	Low Orbit Ion Cannon, a GUI tool for network stress testing and DoS/DDoS attacks
`hoic`	High Orbit Ion Cannon, a GUI tool for network stress testing and DoS/DDoS attacks
`zz`	Zombie Zapper, a countermeasure tool capable of stopping DDoS packet flooding attacks carried out by Trin00, TFN, Stacheldraht, etc.

`aircrack-ng`	WEP and WPA/WPA2-PSK key cracker. Part of the Aircrack-ng suite for Wi-Fi network cracking
`airoscript-ng`	User-friendly interface for `aircrack-ng`
`airodump-ng`	Packet sniffer
`airgraph-ng`	Tool to generate graphs of relationships between wireless devices, using data from `airodump-ng`
`aireplay-ng`	Packet injector
`packetforge-ng`	Generator of encrypted packets for injection
`airbase-ng`	Tool for attacks against wireless clients (and not Access Points)
`airserv-ng`	Tool to access the wireless NIC from other machines
`tkiptun-ng`	WPA-TKIP attack tool
`wesside-ng`	WEP key recovery tool
`airdecloak-ng`	Tool to remove WEP cloaking from PCAP dump files
`airolib-ng`	Manager for ESSID and password lists. These are used for WPA and WPA2 cracking
`airdecap-ng`	Decryption tool for WEP and WPA
`airmon-ng`	Tool to set up wireless NICs in monitor mode
`airtun-ng`	Virtual tunnel interface creator
`airdriver-ng`	Tool that displays information about wireless drivers on the system
`airdrop-ng`	Tool to force deauthentication of users
`ivstools`	Tool to extract or merge IVs from a PCAP dump file
`kstats`	Program that displays statistical FMS algorithm votes for an IVs dump with a specific WEP key
`makeivs`	Tool to generate a dummy IVs dump file with a specific WEP key
`versuck-ng`	Tool to calculate the default WEP key for Verizon Actiontec wireless routers
`easside-ng`	Tool to communicate with an Access Point without knowing its WEP key
`buddy-ng`	Helper server for `easside-ng`
`fern-wifi-cracker`	Wi-Fi auditing and attack tool, with GUI
`airsnort`	WEP key cracker using the Fluhrer, Mantin, and Shamir attack (FMS)
`wepattack`	WEP key cracker. Takes a dumpfile as input
`WEPCrack`	WEP key cracker
`airfart`	GUI tool that detects Wi-Fi devices and displays their signal strength
`cowpatty`	WPA-PSK key cracker via dictionary attack. Part of the coWPAtty package
`genpmk`	WPA-PSK key cracker via precomputation attack

`kismet` Wireless sniffer and IDS

`hackrf_info` Tool that probes the HackRF One device and shows its configuration.
HackRF One is an open source SDR (Software Defined Radio) peripheral for testing RF systems. It is able to transmit and receive radio signals in the 1 MHz - 6 GHz range and can be used to carry out attacks such as replay, jamming, fuzzing, BlueBorne, etc.
Usually used with the Ubuntu, Gentoo, and Pentoo distros

`rfcat` ISM band radio multipurpose tool

`rfcat-rolljam.py` Exploit for the rolling code attack. Jams, captures, and replays radio signals

`scapy-radio` Modified version of the `scapy` packet manipulation tool to include RF capabilities

`RFCrack.py` Radio Frequency communications scanner and test workbench. Can be used to perform the rolling code attack

`btscanner` Bluetooth scanner with ncurses UI

`bluescan` Bluetooth scanner

`bluediving` Bluetooth pentesting suite

`bluesniff` Bluetooth device discovery (wardriving) utility. Proof of concept

`sqlmap`	Automatic SQL injection exploitation/pentesting tool. The GUI version is Tyrant SQL
`mole`	Automatic SQL injection exploitation/pentesting tool
`jsql`	jSQL Injection, a GUI tool to retrieve database information from a remote server
`sqlsus`	Fast and efficient SQL injection and takeover tool for MySQL
`ISR-sqlget`	Blind SQL injection tool
`fatrat`	TheFatRat, a Remote Access Trojan and exploiting tool
`horsepill`	Ramdisk-based rootkit
`amap`	Application Mapper, a scanning and fingerprinting tool for pentesting
`amapcrap`	Fuzzer that sends random data to a TCP/UDP port and gathers a response for use by `amap`
`svmap`	Scanner for discovering VoIP SIP devices. Part of the SIPVicious package
`svwar`	Scanner for identifying active extensions on a PBX
`svcrack`	Online password cracker for SIP PBX
`svcrash`	Countermeasure for unauthorized `svwar` and `svcrack` scans
`svreport`	SIP audit session manager and reports exporter
`ike-scan`	Tool for discovering IKE hosts (IPsec VPN servers) and determine which IKE implementation they use
`ddosim`	Layer 7 DDoS simulator. Simulates zombie hosts connecting to a specific application on the target host
`fat`	Firmware Analysis Toolkit, a toolkit to find and analyze vulnerabilities in the firmware of IoT devices and embedded devices
`balbuzard`	Tool to extract patterns of interest (e.g. URLs, IP addresses, typical malware strings) from malicious files
`bbcrack`	Tool to bruteforce typical malware obfuscation transforms (XOR, ROL, ADD, etc.) and discover the algorithms and keys used. Based on patterns of interest
`bbharvest`	Tool to extract all patterns of interest found when applying typical malware obfuscation transforms (XOR, ROL, ADD, etc.) trying all possible keys
`bbtrans`	Tool to apply any of the malware obfuscation transforms from `bbcrack` to a file

w3af	Web Application Attack and Audit Framework, GUI. Finds and exploits web application vulnerabilities
w3af-console	Web Application Attack and Audit Framework, CLI version
nikto	Webserver vulnerability scanner
skipfish	Webserver vulnerability scanner
uniscan	Simple webserver vulnerability scanner
burp	Burp Suite, a comprehensive testing platform for web applications security. Allows intercepting and modifying HTTP/HTTPS requests and replies to perform HTTP session hijacking
zap	OWASP Zed Attack Proxy, a pentesting GUI tool for web applications
webscarab	Tool for testing the security of web applications. Runs as a proxy
arachni_web	Modular framework and penetration testing platform to test the security of web applications
nuclei	Pentesting tool, based on templates, for configurable targeted scanning
xsser	Automatic framework to detect, exploit, and report XSS vulnerabilities in web applications
vega	GUI tool to test the security of web applications
paros	GUI tool with HTTP/HTTPS proxy functionality for assessing web application vulnerability
wapiti	Vulnerability scanner for web applications
httpbee	Web application testing tool
wsfuzzer	Fuzzer for black box testing of web services (HTTP and SOAP)
ffuf	Fast web fuzzer
httprint	Web server fingerprinting tool
wafw00f	Tool that detects and fingerprints a WAF (Web App Firewall) i.e. a firewall that protects a webserver
wafninja	Tool that circumvents a WAF by automating the steps necessary for bypassing input validation
whatweb	Web scanner. Detects and identifies web technologies, plugins, JavaScript libraries, etc.
sublist3r	Enumerator of website subdomains. Uses common web search engines
scrapy	Application framework for web scraping, web crawling, data mining, and other web content extraction
dirbuster	Webspider with GUI that attempts to find via brute-force all non-linked (hidden) directories and files
sf	Spiderfoot, an OSINT website reconnaissance tool
ferret	Cookie sniffer
hamster	Session hijacker. It runs as a proxy, reusing cookies stolen by ferret from other clients' sessions
fiddler	Web debugging proxy tool, with GUI. Can be used to test the security of web applications
sslstrip	Tool for the HTTPS Stripping attack
sqlninja	Tool to exploit SQL injection vulnerabilities against web applications that use Microsoft SQL Server as database backend
SQLiX	Web crawler that attempts to find SQL injection vulnerabilities on explored websites
slowloris	Tool for the DoS Slowloris attack against webservers
slowhttptest	Tool for testing slow HTTP DoS attacks such as Slowloris, Slow Read, R U Dead Yet, etc.
wpscan	WordPress vulnerability scanner

Security tools - passwords

`john`	John The Ripper, a password cracker
`hashcat`	Password cracker and password recovery tool
`dsniff`	Password sniffer, able to operate over many cleartext network protocols
`hydra`	Login cracker tool supporting many network protocols and password cracking techniques
`medusa`	Brute-force login cracker
`ncrack`	High-speed network authentication cracker
`ophcrack-cli`	Password cracker for MS Windows passwords. Uses rainbow tables
`patator`	Multi-purpose brute-forcer (for FTP, SSH, telnet, SMTP, DNS, MySQL, PostgreSQL, etc.)
`rcrack`	Hash cracker that uses rainbow tables. Part of the RainbowCrack package
`rtgen`	Rainbow table generator for `rcrack`

`snort`	Network IDS/IPS with real-time traffic analysis and packet sniffing. It is configured via a customized ruleset that uses `libpcap`
`ossec-hids`	OSSEC, a HIDS with log monitoring and SIEM capabilities
`tripwire`	HIDS with integrity-based detection of unauthorized filesystem changes
`aide`	Advanced Intrusion Detection Environment, a HIDS with integrity-based detection. It makes a snapshot of the filesystem state and records it in a database, to check integrity of files at a later time
`nessusd`	Nessus, a well-known and comprehensive vulnerability scanner
`saint`	Security Administrator's Integrated Network Tool, a vulnerability scanner. Originally based on SATAN (Security Administrator Tool for Analyzing Networks)
`pentbox`	Security suite including password crackers, honeypots, DoS tools, etc.
`websploit`	Exploit framework containing reconnaissance and attack tools for various technologies
`psad`	Port Scan Attack Detector. Uses `iptables` log messages to detect and block port scans and other malicious network traffic
`honeyd`	Honeypot daemon. It creates virtual hosts, and simulates their networking stack and any desired network service
`labrea`	Honeypot for incoming IP connections. Replies to unanswered ARP requests, creating a virtual host with the related unused IP address, which then performs Layer 4 tarpitting
`sshhipot`	SSH honeypot
`artillery`	Honeypot with monitoring and alerting system
`honeytrap`	Extensible toolkit for running and monitoring honeypots
`kojoney`	Low-interaction honeypot that emulates an SSH server
`honeypy`	Medium-interaction honeypot
`cowrie`	High-interaction SSH and Telnet honeypot
`nexphisher`	Automated phishing toolkit featuring many social media websites
`stegdetect`	Detector of stenographic content in graphic image files
`inspy`	LinkedIn enumerator. Attempts to find technologies and people at a specified target company
`recon-ng`	Web reconnaissance framework
`dog`	Recon Dog, an OSINT reconnaissance tool
`maltego`	OSINT tool with GUI that visualizes discovered data in a graph format for link analysis
`JustMetadata`	OSINT tool that gathers information about a large number of IP addresses and attempts to extrapolate relationships between them

Advanced persistent threat (APT)	Stealthy attack where the attacker gains unauthorized access to a system and remains undetected for a long period.
Zero-day attack (0day)	Attack exploiting a software vulnerability that is still unknown or for which no fix exists yet.
Man-in-the-middle (MitM)	Network-based threat where the attacker inserts itself undetected in the communication channel between two legitimate parties (network-level hijacking) and then proceeds to sniff, relay, and possibly modify the traffic. Countermeasure: mutual authentication of parties.
Replay attack **Playback attack**	Attack where the attacker eavesdrops on a communication, then maliciously sends again parts of a valid data transmission. Countermeasure: data tagging e.g. nonces, rolling code.
Side channel attack	Attack based on information obtained from the implementation of a system (e.g. analysis of power consumption, timing, electromagnetic leaks, sound) and not from weaknesses in the algorithm itself (e.g. cryptanalysis, software bugs).
Rolling code attack **Hopping code attack**	Attack against the rolling code (used itself as a defense against replay attacks) used in keyless systems. The attacker jams the signal and sniffs a first code sent by the target. As the first code did not have any effect, the target sends a second code which is sniffed too by the attacker; at the same time the attacker forwards the first code which is received by the system, but the target believes it is the effect of the second code. Later on, the attacker uses the second code to gain unauthorized access to the system.
Supply chain attack	Attack against the less secure elements in an organization's supply chain, usually done by tampering with the manufacturing process of the end-user software or hardware appliance (e.g. installing a backdoor in the firmware of a router). Countermeasure: use a SBOM (Software Bill Of Materials) to analyze vulnerabilities.
Banner grabbing	Reconnaissance technique consisting in initiating a connection to the desired service and noting the software type and version mentioned in the service banner. Countermeasure: configure banners to show minimal information.
Username enumeration	Reconnaissance technique in which the attacker tries to determine whether a specific username exists or not in the target system, or attempts to obtain a list of valid users. Countermeasure: configure the system to show minimal information about a failed login.
Google hacking **Google dorking**	Reconnaissance technique consisting in using advanced operators with specific strings (i.e. dorks) in the Google search engine to find specific versions of vulnerable web applications, misconfigurations, administration panels, sensitive files not supposed to be publicly accessible, etc.
Man-in-the-mobile	Infection of a mobile device with malware to bypass 2FA, as the malware relays the information to the attacker.
Privilege escalation	Host-based threat consisting in illegally gaining elevated access to resources that are normally protected from a program or user.
Confused deputy attack	Type of privilege escalation consisting in tricking a legitimate, more privileged program into misusing its authority on the system.
Sybil attack **Pseudospoofing**	Act of subverting a system by creating multiple fake identities. This may allow the attacker to e.g. acquire a disproportionate level of control over a reputation system by affecting voting results, or disrupt statistics about vehicular traffic.

Social engineering	Wide range of non-technical attacks consisting in deception and psychological manipulation of the target individual into divulging confidential information or performing unwarranted actions.
Pretexting	Social engineering attack where the attacker invents an elaborate scenario (i.e. a pretext) to engage the target individual under a fake identity and convince them to divulge confidential information or perform unwarranted actions.
Phishing	Social engineering attack aimed at obtaining sensitive information from people via a fake but legitimately-looking website controlled by the attacker. Usually perpetrated via an email message containing an obfuscated link to the malicious website.
Spear phishing	Personalized phishing targeted at a specific individual.
Whaling	Phishing targeted at a high-value individual (CEO, CISO, etc.).
Vishing	Phishing via VoIP.
Smishing	Phishing via SMS.
Tailgating **Piggybacking**	Social engineering attack in which an attacker lacking proper authorization follows an authenticated individual into the targeted restricted area.
Shoulder surfing	Act of getting access to sensitive information by spying an individual entering the data.
USB drop attack **Baiting**	Social engineering attack consisting in leaving a bulk of malware-infected USB flash drives in public places for people to find and use.
Dumpster diving	Act of searching through discarded paper documents, left behind by the target organization, in order to find and exploit information.
Rubber hose cryptanalysis	Euphemism for extracting cryptographic secrets from the target by means of coercion or violence.
Black bag cryptanalysis	Euphemism for obtaining cryptographic secrets from the target by breaking and entering the premises, burglary, theft, etc. This term is also used for other non-cryptanalysis methods such as keystroke logging, infection via virus or trojan horse, etc.

Denial of Service (DoS)	Cyberattack towards hosts or networks, aimed at preventing or reducing availability of services to legitimate users. Countermeasures: blackhole filtering aka null routes, to drop all traffic coming from the attacker. Detection via activity profiling, sequential change-point detection (Cumulative Sum algorithm), wavelet-based signal analysis of traffic's spectral components.
Distributed Denial of Service (DDoS)	DoS launched simultaneously from several attacking hosts (usually a group of compromised machines i.e. a botnet).
Distributed Reflected Denial of Service (DRDoS)	DDoS carried out by forging requests to a large number of remote hosts using the target host's spoofed source IP address.
Permanent Denial of Service (PDoS) **Phlashing**	Hardware-targeted DoS which replaces the target device's firmware with a faulty one, bricking the device permanently.
Multi-vector attack	DoS combining volumetric, protocol, and application-layer attacks.

Ping of death	A malformed or oversized ping packet which, when reassembled by the target host, causes a buffer overflow, crashing or infecting with malicious code the target host. Obsolete; modern TCP/IP stacks, firewalls, and IDSs easily identify and discard pings of death.
Ping flood **ICMP flood**	DoS in which the attacker sends a large number of ICMP Echo Request packets to the target host.
Smurf attack	DRDoS in which the attacker sends a large number of ICMP Echo Request packets to a network broadcast address using the target host's spoofed source IP address. Obsolete; by default, ICMP requests to broadcast addresses are not forwarded anymore by routers.
Fraggle attack	DRDoS in which the attacker sends a large amount of UDP traffic to ports 7 (Echo Protocol) and 19 (CHARGEN) of multiple remote hosts, using the target host's spoofed source IP address.
SYN flood	DoS in which the attacker sends a a large number of TCP SYN packets to the target host but never responds to its SYN/ACK, hence never completing the TCP three-way handshake. This creates a large number of half-open connections which, until they time out, fill up the target host's connection queue, preventing legitimate clients to connect to it. Countermeasures: TCP Intercept (aka SYN Proxy or SYN cookies).
Teardrop	DoS in which the attacker sends mangled IP fragments with overlapping and oversized payloads to the target host, causing it to crash. Obsolete; fixed in the Linux kernel v2.1.63.
Fragmentation attack	DoS performed by sending a large number of fragmented TCP or UDP packets to the target host, which will consume resources in reassembling and inspecting them.
Shrew attack	Low-rate DoS that exploits the retransmission timeout (RTO) mechanism of TCP. Performed by sending out a burst of traffic to a bottleneck router at the same time the client sends a request to the server. Hence, the router suspends the data transmission, packets are dropped during the RTO, and after the RTO the client needs to resend the lost packets, slowing down the transmission.
MAC spoofing	Act of setting a spoofed MAC address in the NIC in order to divert communications to the host controlled by the attacker. Countermeasure: IP Source Guard (on switches) using the DHCP snooping binding table.
MAC flooding	Dispatch of multiple Ethernet frames with different source MAC addresses to a switch. This fills up a switch's CAM table and forces the switch to failover to hub mode (i.e. broadcasting to all switch ports), allowing the attacker to sniff all network traffic in the LAN. Countermeasure: port security with max one MAC address per interface (on switches).
ARP spoofing **ARP cache poisoning** **ARP poisoning** **ARP poison routing**	Dispatch of forged ARP messages into the LAN to associate the attacker's MAC address with another host's (often the gateway) IP address, diverting communications to the attacker. Often the first choice of attack for the purpose of sniffing, or in preparation to MitM attacks or session hijacking attacks. Countermeasures: Dynamic ARP Inspection (on switches) using the DHCP snooping binding table, disabling of gratuitous ARP, static IP addresses and ARP tables. On wireless networks, Client Isolation (on the wireless router) which prevents wireless clients from communicating between them.
Port stealing	ARP spoofing aimed to associate the attacker's MAC address with another host's IP address on a switch's CAM table, which will then forward packets through the wrong switch port.
IP spoofing	Act of setting a spoofed IP address in the NIC so the attacker's host can appear to be some other host. Countermeasure: direct TTL probes (however, this works only if the attacker's host is in a different subnet).

DNS spoofing	Tampering with the name resolution mechanism of the target host so that a domain name resolves to an incorrect IP address controlled by the attacker. Can be carried out either by DNS hijacking, by deceiving the target host to use a rogue DNS server, or by tampering with the hosts file of the target host. Can be done in preparation to a MitM attack, although for a LAN an ARP spoofing would serve the same purpose and is easier to do. Countermeasures: DNSSEC, restriction of DNS service, master-slave DNS setup with no Internet access for the master, DNS anti-spoofing.
DNS hijacking	Attack which consists in compromising a DNS server and changing the mapping settings to redirect towards a rogue DNS server. This can also be done by stealing the domain name upon the sponsoring domain name registrar accredited by the ICANN (which manages the DNS root zone). Countermeasure: at the registrar level, REGISTRAR-LOCK status code to prevent unauthorized changes to the domain name.
Cybersquatting	Registration of a domain name which is similar to a well-known domain, product, or entity, in order to deceive users. Can be done in preparation to phishing attacks or scams.
Typosquatting	Cybersquatting where the attacker relies on typos and other mistakes made by users when they manually type a URL into a web browser.
Domain sniping **Domain snapping**	Registration of a domain name that has just expired, with the purpose of reselling it to the original owner at a higher price.
DNS cache poisoning	Injection of forged DNS records in the DNS resolver's cache, causing the name server to return an incorrect IP address for a domain name, hence redirecting traffic to the attacker.
Blind response forgery	DNS cache poisoning attack carried out by guessing the transaction ID (birthday paradox). Countermeasure: randomization of UDP source port.
DNS water torture	DDoS done by performing a large number of DNS queries for nonexistent subdomains of a target domain. Subdomains strings are randomly-generated by the attacker, hence the queries bypass the DNS cache and hit the DNS Authoritative Servers of the target domain.
DNS amplification attack	DRDoS in which the attacker sends a large amount of DNS queries to the target host's DNS server, using the target host's spoofed source IP address. The recursive resolution of queries ends up overwhelming the target host's DNS server.
DNS sinkhole attack **Blackhole DNS attack**	Act of providing incorrect DNS information to systems so to redirect their communications to a single destination. This can also be done for beneficial purposes, e.g. to block ads or stop botnets from contacting their C&C (Command and Control) host.
DHCP spoofing	Attack consisting in setting up a rogue DHCP server and use it to send forged DHCP responses to hosts. Often done to replace the IP addresses of the default gateway and DNS server, redirecting traffic to attacker-controlled nodes. Countermeasure: DHCP snooping and Dynamic ARP Inspection (DAI) on routers.
DHCP starvation	DoS in which the attacker floods a DHCP server with DHCP requests from spoofed MAC addresses, depleting the server's IP address pool and making it unable to allocate them for legitimate clients. Also done in preparation to the deployment of a rogue DHCP server.
IRDP spoofing	Injection of forged IRDP Router Advertisements to add default route entries to a target host, redirecting traffic to the attacker-controlled node.
Sinkhole attack	Attempt to attract network traffic by advertising fake routing updates. Once traffic passes through the malicious node, the attacker may alter the payload, launch a blackhole or wormhole attack, or perform other disruptive activities.
Blackhole attack **Packet drop attack**	DoS attack where an attacker-controlled node discards packets instead of relaying them. This can be done partially and/or selectively (e.g. depending on the time of the day, the source, the destination) in order to avoid detection.
Wormhole attack	Attack (usually carried out on wireless networks) where the attacker records packets in one location and then tunnels them to another location, selectively or as a whole.
Man-in-the-Cloud (MitC)	MitM-like attack against cloud file synchronization services, carried out by stealing and reusing a synchronization token from the target cloud user to obtain access to their files. Countermeasure: hardened policies for token expiration.
Wardialing	Reconnaissance technique consisting in automatically dialing every telephone number from a list (usually in a local area code) searching for modems, BBS, or fax machines. Obsolete, as dial-up Internet connections have mostly disappeared.
Warshipping	Attack consisting in using a physical package delivery service to deliver an attack vector (e.g. a backdoored router) to a target.

Evil twin attack	Attack consisting in setting up a legitimately-looking rogue Wi-Fi Access Point to lure clients into connecting to it and then perform eavesdropping or MitM attacks. To improve effectiveness, the rogue AP can even transmit with a stronger signal. Countermeasures: network management software (on the network management side) with wired side inputs to detect devices connected to the LAN and hence also rogue APs; WIPS.
KARMA attack	Variant of the evil twin attack. Some vulnerable devices broadcast the list of their preferred networks (i.e. the SSIDs of APs to which they have already connected and are going to connect automatically). Upon receiving this information, an attacker can set up a rogue AP with a SSID from the list.
Client misassociation	Attack similar to the evil twin attack, consisting in setting up a rogue AP that duplicates the SSID and the MAC address of a legitimate AP.
Disassociation attack **Deauthentication attack**	Availability attack carried out by sending deauthentication frames to the AP to disconnect clients. This attack can be done against a specific client (by using the target client's spoofed MAC address) or all clients.
Beacon flood attack	Availability attack carried out by sending a large number of forged Wi-Fi beacons to confuse wireless clients and make it harder for them to connect to a legitimate AP.
Clear channel assessment attack **Queensland attack**	Physical layer DoS attack that exploits the CSMA/CA Clear Channel Assessment (CCA) to make the channel appear busy.
Fluhrer, Mantin, and Shamir attack (FMS)	Attack which exploits a weakness in the RC4 key scheduling algorithm to reconstruct the key from encrypted messages. This attack can be used to recover a WEP key.
Chopchop attack	Attack carried out against a WEP-encrypted wireless communication which allows to recover the unencrypted payload. The attacker chops off the last byte of data from a WEP-encrypted packet, replaces that byte, recalculates the checksum, and sends the packet to the AP. The AP discards the packet, until by trial and error the attacker eventually replaces a valid checksum and the AP accepts it. The same attack can be carried out against WPA-TKIP. This attack does not recover the WEP key.
Key Reinstallation Attack (KRACK)	Attack against the four-way handshake in the WPA2 authentication protocol. The attacker captures and replays the message in step 3 (containing the AP's nonce) to force nonce reuse; this allows the attacker to decrypt all traffic. Countermeasures: update all wireless devices with the latest security patches, patch the AP's firmware, use HTTPS, enable 2FA.
Fragmentation attack [WEP]	Attack consisting in extracting some keying information from a WEP packet, then sending ARP and LLC packets to the AP which resends them back, then extracting more keying information from the packets. This cycle is repeated until the attacker gets 1500 bytes of the PRGA (Pseudo Random Generator Algorithm) which can then be used to forge and inject packets. This attack does not recover the WEP key.
Caffe Latte attack	Attack allowing to recover a WEP key from a client by capturing an ARP packet from the client, manipulating it, and sending it back to the client.
Hirte attack **Client-oriented fragmentation attack**	Extension to the Caffe Latte attack, performed using any ARP or IP packet.
Wardriving	Detection and reconnaissance of WLANs by listening to SSID broadcasts or by sending probe requests, usually done from a moving vehicle.
Warchalking	Technique of advertising discovered WLANs in range by drawing specific symbols with chalk in public places, usually on pavements or walls.

Bluejacking	Sending of anonymous messages (e.g. spam) to a Bluetooth device, done by inserting the message in the BT connection request. Uses the OBEX (Object Exchange) protocol.
Bluesnarfing	Theft of information from a Bluetooth device. The attacker connects to the target BT device and performs a GET operation for known or guessed filenames. Carried out by exploiting a vulnerability in the OBEX protocol.
Bluebugging	Unauthorized remote access and takeover of a Bluetooth device.
Blueprinting	Footprinting performed against a Bluetooth device.
Bluesmacking	Ping of death attack carried against a Bluetooth device.
BlueBorne	Vulnerability in the Bluetooth implementation on multiple OSes that allows an attacker to take control of the target device, even if the device is not paired or even set to discoverable mode. The attacker gets the MAC address and performs OS fingerprinting on the device, then uses a BT exploit.

Network-level hijacking	Interception of TCP or UDP packets during transmission between client and server. This term is also used for the takeover of a legitimate TCP communication between two hosts, done via IP spoofing and MitM, sometimes using source routed packets. The attacker sniffs (or tries to predict) TCP Sequence and Acknowledgment numbers from the client, sends forged TCP Sequence and Acknowledgment numbers to the server to desynchronize the client, and finally inserts itself in the TCP session.
Application-level hijacking **Session hijacking**	Takeover of an HTTP session, usually done by stealing an HTTP session token. This is not a network-level hijacking.
TCP/IP hijacking	Network-level hijacking in which the attacker sniffs the communications between two hosts to get the target host's Initial Sequence Number (ISN). The attacker then sends a packet with the target host's spoofed source IP address using the captured ISN. The other host receives the packet, increments its TCP Sequence number, and sends an ACK to the target host which ignores it. The attacker continues to send spoofed packets with forged TCP Sequence and Acknowledgment numbers, causing the target host to have desynchronized values and making its connection hang. At this point, the attacker inserts itself in the TCP session, replacing the target host.
Blind hijacking	Network-level session hijacking in which the attacker tries to predict ISN and TCP Sequence and Acknowledgment numbers, without being able to see the response. Can be used to inject malicious data into the communication, and does not require source routing. This is not considered a MitM attack.
RST hijacking **TCP reset attack**	Injection of an RST packet with spoofed source IP address within a legitimate TCP communication, to terminate the connection. May be done in preparation to TCP/IP hijacking.
UDP hijacking	Network-level hijacking where the attacker forges UDP replies from the server.
Source routing attack	Network-level session hijacking that uses the source routing field in the IP header to specify a packet route so to, with the help of a trusted host, divert packets towards the attacker's node. Used in IP spoofing attacks. Obsolete; by default, network devices nowadays discard source routed packets.
Session fixation	Application-level session hijacking in which the attacker sets a session ID on behalf on the target host. This can be done e.g. via a phishing email. This attack is effective against e.g. web applications that do not change the session cookie after a successful login and instead allow additional privileges to it.
Session prediction	Application-level session hijacking in which the attacker predicts a session ID value. The attacker needs beforehand to collect valid session ID values that identify authenticated users, and to analyze and understand the session ID generation algorithm.
Session brute-forcing	Application-level session hijacking in which the attacker tries all possible session ID values until they successfully get access to the application.
Session riding	Application-level session hijacking obtained via Cross-Site Request Forgery. Countermeasures: check the HTTP Referrer header, ignore URL parameters when processing an HTTP POST command.
Session sidejacking **Sidejacking** **Cookie stealing**	Application-level session hijacking in which the attacker sniffs a session ID (session cookie) from a legitimate session and then reuses it to impersonate the legitimate client.

HTTP response-splitting attack **HTTP header injection**	Attack which consists in adding header response data into an input field so that the webserver splits the HTTP response into two; these can either be served to the attacker, served to a legitimate client, or discarded. Countermeasure: validation of client input. In particular, CR (%0d, `\r`) and LF (%0a, `\n`) characters should never be allowed in input.
HTTP request tampering	Unauthorized access to a web application obtained either by tampering with the URL query string, or by modifying the HTTP headers sent to the webserver e.g. the `Referer:` header which in vulnerable applications is used for access control.
HTTP Parameter Pollution (HPP)	Evasion technique used to bypass WAF security filters which consists in crafting an HTTP request containing multiple instances of a parameter with the same name, hence splitting the attack vector.
HTTP Parameter Fragmentation (HPF)	Evasion technique, often used along with HPP, which allows to reconstruct the parameter string passed in the HTTP request.
Webcache poisoning	Attack where the attacker uses a specially crafted request to force the webserver's cache to flush its contents and insert a URL with infected content in the cache, which is then served to legitimate clients accessing the cache. Can be performed via an HTTP response-splitting attack.
Directory traversal	Unauthorized access to directories outside the webserver's root directory, done by using repeatedly the `../` sequence in URLs.
Unvalidated redirect	Phishing in which the URL is that of a legitimate site but contains a redirect to the malicious site.
Unvalidated forwarding	Unauthorized access to a restricted webpage obtained fraudulently via an embedded forward query on the URL.
CRIME	Compression Ratio Info-leak Made Easy. Exploit against authentication web cookies transmitted over compressed HTTPS and SPDY connections, which results in session hijacking.
BREACH	Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext. Session hijacking exploit analogous to CRIME, but performed against HTTPS when using HTTP compression.
HTTPS Stripping **SSL Stripping**	MitM attack which consists in hijacking the connection establishment from the target host to a remote secure webserver, then transparently downgrading all HTTPS traffic to HTTP. This attack works for webservers with redirection from HTTP to HTTPS, and only if the initial request to the webserver is sent as HTTP; then, if the client does not explicitly specify HTTPS for links, the attacker intercepts all HTTP 302 redirections and sends the client the requested content through HTTP. Countermeasure: HSTS (however, browsers not accepting HSTS cookies will still be vulnerable).
Forbidden attack	Attack exploiting vulnerable implementations of the TLS protocol that incorrectly reuse the same cryptographic nonce when data is encrypted. This allows the attacker to sniff the HTTPS connection and inject content.
HTTP flood	Layer-7 DDoS targeted at webservers. Carried out by sending a large number of HTTP GET and POST requests towards the target webserver.
Slow HTTP attack	Low-bandwidth DoS targeted at webservers. Carried out by keeping several connections to the target webserver open as long as possible, depleting the webserver's connection pool. This is done by exchanging HTTP traffic at an extremely slow rate (1 byte/min or less), preventing the webserver from going into timeout for idle connection.
Slow Post attack	Slow HTTP attack carried out by sending correct HTTP requests and headers at an extremely slow rate.
Slow Read attack	Slow HTTP attack carried out by reading the webserver's response at an extremely slow rate.
Slowloris	Similar to the Slow Post attack, but carried out by sending partial HTTP request and headers, while never completing the request.
R U Dead Yet (RUDY)	Slow HTTP attack carried out by opening concurrent POST HTTP connections and delaying sending the body of the POST request.

Cross-Site Scripting (XSS)	Injection of malicious client-side scripts (e.g. via input textfields) in webpages, which are then executed when the webpage is viewed by other clients. This can lead to session hijacking, data theft, exploiting user privileges, etc. Identification of entry points for user input is the first step for exploiting a website. Countermeasures: set the HttpOnly flag in session cookies, sanitize user input.
Non-persistent XSS **Reflected XSS**	XSS attack where the malicious data provided by the attacker is used immediately by server-side scripts to display results to the targeted user only.
Persistent XSS **Stored XSS**	XSS attack where the malicious data provided by the attacker is saved by the server and permanently displayed to all users visiting the website.
Server-side XSS	XSS attack where the malicious data provided by the attacker is wholly processed server-side. Historically, this was the first kind of XSS attack.
DOM-based XSS	XSS attack where the malicious data provided by the attacker does not affect the webserver but it is reflected fully client-side, where all the presentation logic (often JavaScript) occurs.
Cross-Site Flashing (XSF)	XSS-like attack where the malicious data provided by the attacker is used in some specific video playing functions and variables by Flash scripts.
Cross-Site Tracing (XST)	XSS attack, involving the use of the HTTP TRACE method, that allows stealing cookies via Javascript.
Cross-Site Request Forgery (CSRF) **One-click attack**	Attack which consists in having the target client unknowingly submit a maliciously crafted web request to a webpage. Can be performed via specially designed image tags, hidden forms, JavaScript functions, etc. Countermeasure: random tokens in the web application.
Server-Side Request Forgery (SSRF)	Attack which consists in inducing the server-side application to make HTTP requests to an arbitrary domain. This results in unauthorized actions and access to data, and may lead to arbitrary command execution.
URL parameter tampering **Web parameter tampering**	Modification of parameters in the URL to exploit vulnerable applications that use them, e.g. `http://www.bank.com/account?id=345&amount=200000`
Session poisoning	Unauthorized access to a web application by exploiting (e.g. via URL parameter tampering) weak user input validation. The same term may also indicate the takeover of a session of a legitimate user e.g. by injecting malicious content, or via cookie poisoning.
Cookie sniffing	Unauthorized access to a web application by sniffing a cookie belonging to a legitimate user and reusing it to bypass the authentication process, ending up logged in as that user. Countermeasure: SSL, setting the secure attribute on cookies (which will therefore be sent only over HTTPS).
Cookie replay	Replay attack in which the attacker captures a cookie belonging to a legitimate logged in user, then reuses it to perform malicious activities on behalf of that user. The attack persists until the user logs off.
Cookie poisoning	Unauthorized access to a web application by crafting a cookie, or by sniffing and modifying a cookie belonging to a legitimate user. Countermeasures: cookie expiration, associate cookie's credentials to an IP address.
Cookie parameter tampering	Unauthorized access to a web application by tampering with the parameters of a cookie and resubmitting it.
SSI injection	Code injection technique consisting in injecting scripts in webpages via SSI. SSIs (Server Side Includes) are directives present on web applications which allow inserting dynamic content into an HTML page before it is loaded or visualized.
CORS attack	Attack exploiting the Cross-Origin Resource Sharing (CORS) mechanism. CORS allows restricted resources on a webpage to be requested from another domain outside the domain from which the first resource was served; this is safer than allowing all cross-domain requests. It bypasses the Same-Origin Policy (SOP) which forbids certain cross-domain requests (e.g. Ajax).
Connection String Parameter Pollution (CSPP)	Injection of connection string parameters into other existing parameters, often carried out by using a `;` character. This can be done e.g. in connection strings for backend databases.

Website defacement	Unauthorized changes made to the website appearance and content, to show the attacker's propaganda and/or infect legitimate clients with malware.
Pharming	Attack intended to redirect legitimate traffic from a website to a fake one. Usually carried out via DNS spoofing. Considered an advanced form of phishing.
Watering hole attack	Attack in which the attacker identifies which websites the target users visit more often and infects those websites with malware (e.g. via XSS).
Framing attack	Insertion of a malicious webpage inside a legitimate webpage by using the `<iframe>` (inline frame) HTML tag.
Clickjacking **UI redress attack** **UI redressing (UIR)**	Deceptive technique consisting in tricking web users to click on a different element from the one they think they are clicking. Usually done via a framing attack, by having an invisible iframe with malicious content on top of a visible iframe with innocuous content.
Man-in-the-browser (MitB)	Attack related to MitM where a Trojan horse infects a web browser in the target host, and injects HTML code in the browser's requests and responses. The Trojan operates between browser and OS API, allowing it to read data before encryption when it is sent from the host, and read data after decryption when it is received by the host.

XML External Entity (XXE)	SSRF attack consisting in passing malicious XML input, referencing an external entity containing infected data, to an application with a weakly configured XML parser. This may cause confidential data disclosure, DoS, remote code execution, etc. XML DoS issues are a common SOA vulnerability.
Billion laughs attack **XML bomb**	Exponential entity expansion attack consisting in a specially crafted XML document where the root element contains a defined entity which contains e.g. 10 defined entities, each of which contains e.g. 10 defined entities, and so on. This takes up a large amount of memory and results in a DoS when the XML document is processed. Countermeasures: capping the memory allocated for an individual XML parser, treating entities symbolically and expanding them only when needed.
XML injection **XML poisoning**	Insertion of specially crafted XML fields in SOAP requests for web services. This may cause confidential data disclosure, DoS, remote code execution, etc.
XPath injection	Code injection technique in which the attacker provides malicious parameters to construct the wrong XPath query and access the wrong XML node. XPath is a query language for selecting nodes from an XML document.
Wrapping attack [XML]	Attack performed during the translation of a SOAP message in the TLS layer, by intercepting the message, adding the body to the header, injecting a malicious payload in the body, and sending the message to the server as a legitimate client. The server therefore verifies the XML signature as valid. Countermeasures: XML schema validation, authenticated encryption in the XML-Enc (XML Encryption) specification.
SQL injection (SQLI)	Code injection technique in which the attacker inserts malicious SQL statements into an input field for execution. This is done by using single quotes (`'`) and double quotes (`"`). Countermeasures: user input sanitization, avoiding constructing dynamic SQL with concatenated user input values, keeping untrusted data separate from commands and queries, using least privilege account types for connection to the database.
In-band SQL injection	SQL injection where the attacker uses the same communication channel to send the query and retrieve the result. Most common type of SQL injection. Can be: **Error-based:** the attacker causes the database to throw an error, e.g. by forcing a conversion type, and gains knowledge by analyzing the error message **Illegal query:** the attacker sends a logically incorrect query and gains knowledge by analyzing the error message **Union:** the attacker uses the UNION operator to get the field values from other tables; can be coupled to other operators e.g. ORDER BY to find the number of fields in a table **Tautology:** the attacker inserts the OR operator with a tautology so that a WHERE clause is always true, e.g. `' OR '1'='1` **End-of-line comment:** the attacker uses `--` to insert an end-of-line comment to make the database parser ignore the rest of the query **Inline comment:** the attacker uses `/* */` to insert an inline comment to bypass fields **Piggybacked query** aka **stacked queries:** the attacker uses a `;` character to insert additional malicious queries to the original query **Stored procedure:** the attacker input is fed to dynamic SQL statements **Second order:** the attacker's input is saved in a database and used later when the attacker submits a second query; requires knowledge of the application backend
Out-of-band SQL injection	SQL injection where the attacker retrieves the result via a different channel e.g. email, HTTP, or file I/O functions. More difficult to perform.
Blind SQL injection **Inferential SQL injection**	SQL injection attack used when the application does not show a useful error message and hence the attacker needs to assess whether the application might be vulnerable to an SQL injection. This attack is time-intensive. Can be: **Time-based** aka **double blind:** the attacker sends a query which may be true or false embedded with sleep or benchmark functions, and gets information from the time delay that occurs in the response **Boolean-based:** the attacker sends a query which causes the application to return a different result depending on whether the query returns true or false **Heavy query:** the attacker sends a query that takes noticeable time to execute, e.g. a `SELECT COUNT(*)` from multiple tables
ORM injection	SQL injection attack against a data access object model generated by ORM (Object Relational Mapping).

Buffer overflow **Buffer overrun**	Attack carried out by writing data to a buffer over the buffer's boundaries, overwriting the adjacent memory addresses. This allows the attacker to modify the target process' address space to control process execution, crash the process, and modify variables. Countermeasure: buffer bounds checking.
Fork bomb	DoS attack consisting in a process that recursively replicates itself ad infinitum, depleting system resources and slowing down or crashing the system due to resource starvation.
Fuzzing	Attack (also a software testing technique) consisting in sending invalid, unexpected, or random data as input to a program in order to crash it or provoke an exception e.g. a buffer overflow or a memory leak. A memory leak is a form of memory consumption where the program fails to release an allocated block of memory when it is no longer needed.
Code injection	Attack in which the attacker inserts text in a data field that gets interpreted as code.
File injection	Code injection technique in which the attacker exploits dynamic file include mechanisms in vulnerable web applications (especially in PHP). The attacker provides a URL pointing to the malicious file, which is used by the web application instead of the intended local file.
DLL injection	Attack consisting in forcing a process to load a dynamic-link library, resulting in the attacker's malicious code running within the address space of that process.
LDAP injection	Code injection technique in which the attacker inserts malicious LDAP user parameters into an input field to get access to the LDAP database. Countermeasure: user input sanitization.
Insecure deserialization attack	Injection of malicious code into a serialized object; if the application uses a vulnerable algorithm for deserialization, the code is executed when the object is deserialized. Serialization is the act of converting an object into a format (e.g. XML, JSON, YAML) which can be written to disk, transmitted over the network, or sent to a stream (e.g. stdout).
Cross-guest VM breach	Side channel attack carried out by running a VM on the same physical host as the target VM and taking advantage of shared physical resources (e.g. CPU cache) to extract cryptographic secrets.
Shrink-wrap code attack	Attack consisting in exploiting holes in unpatched or misconfigured software (e.g. software with default insecure configuration options).
Logjam	Vulnerability affecting Diffie-Hellman key exchanges with key size from 512 to 1024 bits, allowing an attacker to downgrade a TLS connection and read and modify the exchanged data.
Shellshock	Vulnerability in the Bash shell which allows an attacker to execute arbitrary commands by exploiting the function export feature of Bash.
Heartbleed	Buffer over-read vulnerability in the OpenSSL cryptography library. The attacker client sends a heartbeat request specifying a length larger than the message payload. The server replies filling up all the allocated buffer with the contents of the active memory, which may include private data.
Spectre	Vulnerability affecting the branch prediction mechanism (done to improve performances) on microprocessors. An attacker can exploit the side effects of speculative execution, and particularly branch misprediction, to access private data.
Meltdown	Vulnerability affecting some Intel x86, IBM POWER, and ARM microprocessors. An attacker can have a process, even without proper authorization, read any memory location by exploiting a race condition between memory access and privilege check during instruction processing.
Rowhammer	Attack consisting in accessing physical rows inside vulnerable memory chips millions of times per second, causing bit flips in neighboring rows. This allows the attacker to e.g. bypass security sandboxes and escalate privileges of untrusted applications.
Log4Shell **LogJam**	Remote Code Execution vulnerability in the Apache Log4j library. An attacker causes the application to write one string into the log, then exploits the message lookup substitution function to upload malicious code via JNDI into the application.

Linear cryptanalysis	Cryptanalysis based on finding affine approximations to the actions of a cipher, analyzing pairs of plaintext and the corresponding ciphertext to try to recover the encryption key.
Differential cryptanalysis	Cryptanalysis based on the analysis of how differences in the input affect the output.
Integral cryptanalysis	Cryptanalysis based on the analysis of pairs of inputs differing in only one bit.
Known plaintext attack	Linear cryptanalysis technique where the attacker has access to some plaintext as well as the corresponding ciphertext.
Chosen plaintext attack	Cryptanalysis technique where the attacker is able to obtain the ciphertext corresponding to a plaintext of their choice.
Chosen ciphertext attack	Cryptanalysis technique where the attacker is able to obtain the plaintext corresponding to a ciphertext of their choice.
Adaptive chosen plaintext attack	Cryptanalysis technique where the attacker has access to the encryption device and is able to obtain the ciphertexts corresponding to plaintexts of their choice, making adaptive changes in the plaintext where needed.
Adaptive chosen ciphertext attack	Cryptanalysis technique where the attacker has access to the encryption device and is able to obtain the plaintexts corresponding to ciphertexts of their choice, making adaptive changes in the ciphertext where needed.
Non-adaptive chosen ciphertext attack **Lunchtime attack**	Chosen ciphertext attack where the attacker has access to a limited set of plaintexts and ciphertexts, or has access to the device for a limited time.
Related key attack	Cryptanalysis technique where the attacker is able to obtain the ciphertext corresponding to a plaintext of their choice, encrypted with two different but similar keys.
Chosen key attack **Known key distinguishing attack**	Cryptanalysis technique where the attacker must have access to the communication channel, and obtain the plaintexts corresponding to ciphertexts of their choice. Using this information, the attacker tries to recover the key by breaking an n-bit key cipher into 2 n/2 number of operations.
Timing attack	Cryptanalysis side channel attack where the attacker attempts to break the ciphertext by measuring the execution times of mathematical operations in the encryption process for various inputs.
Birthday attack	Cryptanalysis technique that exploits the mathematics behind the birthday paradox in probability theory, i.e. the higher likelihood of hash collisions found between random attack attempts and a fixed degree of permutations.
Brute force attack	The technique of trying every possible password or key until the correct one is found. May be very time-consuming or even infeasible. Countermeasure: long passwords and keys.
Rainbow table attack	Cracking of password hashes by using a rainbow table, i.e. a pre-computed table that caches the output of cryptographic hash functions. Countermeasure: adding a salt value to passwords before hashing.
Dictionary attack	Password cracking done by trying every entry listed in a dictionary file. The dictionary is a text file containing all words, names, entries with numbers and symbols added to words or replacing letters, finger rolls, etc. Countermeasure: truly random passwords.
Hybrid attack [password cracking]	Password cracking combining a dictionary attack and brute force attack, done by adding numbers and symbols to the dictionary entries.
Meet-in-the-Middle attack	Type of known plaintext attack carried out against ciphers that use multiple keys for encryption. The attacker performs a brute force attack on one key to encrypt the plaintext and on another key to decrypt the ciphertext, trying to find an intermediate ciphertext that matches both.
DUHK (Don't Use Hardcoded Keys)	Cryptographic vulnerability affecting devices that use the ANSI X9.31 Random Number Generator in conjunction with a hardcoded seed key. The attacker can exploit this vulnerability to recover encryption keys from VPN connections or encrypted web sessions.

Virus	
A **virus** is a piece of self-replicating code that attaches copies of itself to other executable programs, infecting them.	
File virus	Infects an executable file, overwriting it.
Boot Sector virus **System virus**	Executes its code before the target machine boots. Moves the Master Boot Record to another location on the hard disk and copies itself to the original location of the MBR.
Multipart virus **Hybrid virus**	Acts both as a file virus and a Boot Sector virus.
FAT virus	Infects the File Allocation Table in FAT filesystems.
Cluster virus	Infects files without changing them, saving instead the virus code to the hard disk and overwriting the pointer in the directory entry; in this way, the virus code is read instead of the actual program.
Stealth virus **Tunneling virus**	Alters the service call interrupts while running, to hide from AV software.
Sparse infector virus	Infects files only occasionally, upon satisfying certain conditions (time, size of the file to infect, etc.) to minimize the risk of being detected by AV software.
Encryption virus	Consists of a decryption module and an encrypted copy of the virus. Once the target machine is infected, the decryption module decrypts and executes the virus. The virus then infects files with a copy of the virus which is encrypted with a different key for each file. This is done to thwart signature detection methods in AV software.
Polymorphic virus	Reprograms itself, changing the malicious code at each infection. Consists of the encrypted virus, a decryption routine, and a mutation engine; the virus code mutates with each infection while the virus algorithm stays the same. Undetectable by AV software.
Metamorphic virus	Reprograms itself, rewriting the code each time it infects a new file: it inserts dead code, reorders instructions, and modifies the program control structure to avoid pattern recognition by AV software. Stealthier than a polymorphic virus.
Cavity virus	When infecting, overwrites empty spaces (nulls) in the original file so not to modify its size.
Camouflage virus **Companion virus**	Creates a companion file with a different file extension for each file to infect; the companion file is executed first and contains the malicious code.
Shell virus	Wraps itself around the infected file, hosting the original program as its subroutine, so that the virus code is executed first.
TSR virus	Terminate and Stay Resident virus. Remains resident in memory after the infected program has terminated execution.
Macro virus	Written as a macro (often in VBA language), infects Microsoft Office files.
File extension virus	Tries to hide itself by adding a fake safe file extension (e.g. TXT) to its executable file.
Logic bomb	Virus that is triggered in response to a specific event.

Trojan	
A **Trojan** is a malicious program packed and concealed, with the help of a wrapper, inside an innocuous program.	
Remote Access Trojan (RAT)	Provides full access to the infected host, including files, shell, screen capture, webcam, microphone, etc.
Backdoor Trojan	Allows bypassing the standard system authentication through IDSs and firewalls.
Botnet Trojan	Infects a large number of hosts (which, once compromised, become bots aka zombies) to create a botnet that can then be controlled via a Command&Control center to carry out distributed attacks.
Rootkit Trojan	Composed of a dropper, a loader, and a rootkit; the dropper runs the loader which causes a buffer overflow, causing the rootkit to be loaded into memory.
Proxy server Trojan	Allows an attacker to use the infected system as a proxy to connect to the Internet.
Covert channel Trojan	Creates a covert channel in the data stream authorized by the network access control system, allowing the attacker to tunnel malicious traffic undetected.

Rootkit	
A **rootkit** is a malicious program used to gain full, administrator-level, persistent access to a system without detection.	
Hardware/firmware rootkit	Located in the firmware (hard disks, BIOS, etc.), creates a persistent malware image.
Bootloader-level rootkit	Replaces the bootloader. Can activate itself before the OS starts.
Kernel-level rootkit	Runs at kernel level in Ring 0 with the highest OS privileges. This is the most difficult type of rootkit to detect.
Hypervisor-level rootkit	Runs in Ring 1, hosting the OS of the target machine as a VM and intercepting all hardware calls made by the target OS.
Library-level rootkit	Patches, hooks, or replaces OS system calls with backdoored versions.
Application-level rootkit	Replaces application files and modifies process' behaviour by injecting malicious code.

Other malware	
Worm	Standalone malicious program that replicates itself and executes through network connections. Usually it does not infect files, and uses just the CPU and RAM of the infected host.
Keylogger	Program that covertly intercepts and records all keys pressed on the keyboard. May also be hardware. The best location for it to be placed is the keyboard hardware or the OS.
Ransomware	Malware that encrypts files in the infected system, blocking the legitimate user from accessing them, and asks for a ransom to be paid online.

Security countermeasures - firewall

A **firewall** monitors incoming and outgoing network traffic, allowing or discarding it based on custom security rules.

Firewall architectures:

Bastion host	Defends the LAN from outside attacks. Has one public interface connected to the Internet, and one private interface connected to the LAN.
Screened subnet	Has one public interface connected to the Internet, one private interface connected to the LAN, and one interface connected to the DMZ. Designed to host servers that offer public services.
Multi-homed firewall	Composed of multiple firewalls, or one firewall with at least three NICs, which allows the protection of multiple network segments.

Firewall technologies:

Packet filtering	Filters packets according to source and destination IP address, source and destination TCP/UDP port, TCP flags, etc.
Circuit-level gateway	Forwards data between networks, making traffic appear to have originated from the gateway.
Application-level filtering	Filters traffic depending on the protocol.
Stateful multilayer inspection	Combines packet filtering, circuit-level gateway, and application-level filtering to remember the state of previous packets and determine whether session packets are legitimate.
Application proxy	Provides protection by preventing direct connections between systems on either side of the firewall, and hence by avoiding exposure of the proxied service. A client connects to the proxy firewall (stateful) which then initiates a new network connection on behalf of the request.
Virtual Private Network (VPN)	Using traffic encryption and encapsulation, provides secure access to a private network through a WAN.
Network Address Translation (NAT)	As in routers, remaps LAN's private IP addresses to Internet-routable public IP addresses.

Techniques for firewall reconnaissance and evasion:

Port scanning	The attacker performs a port scan to identify the firewall type and model.
Firewalking	The attacker sends a TCP/UDP packet with a TTL equal to the number of hops to the firewall plus one, to determine gateway ACL filters and perform network mapping.
IP address spoofing	The attacker spoofs the IP address of a trusted host in the network.
DNS poisoning	The attacker performs DNS poisoning, directing a legitimate user inside the firewalled perimeter to a malicious server which infects the user's host.
Source routing	The attacker sends source routed packets so that they bypass the firewall.
Tiny fragments	The attacker sends tiny fragments of outgoing packets forcing some TCP packet's header information into the next fragment, to bypass user-defined firewall rules.
Proxying	The attacker bypasses URL blacklisting by using an HTTP proxy.
ICMP tunneling	The attacker sends data in the payload portion of an ICMP Echo packet, which is usually not inspected by the firewall.
ACK tunneling	The attacker sends data in a TCP ACK packet, which is usually not inspected by the firewall as it is used as response to legitimate traffic.
HTTP tunneling	The attacker tunnels network traffic through HTTP, to bypass firewalls which only allow HTTP.
SSH tunneling	The attacker tunnels network traffic through SSH, as firewalls often allow SSH.
Encoded XSS	The attacker performs a XSS attack against a WAF (Web Application Firewall), encoding the payload in ASCII or hex to avoid triggering the firewall's filters.

An **IDS (Intrusion Detection System)** monitors, detects, and alerts about possible intrusions (passive mode). An **IPS (Intrusion Prevention System)** also blocks them (active mode). Can be network-based (**NIDS**) or host-based (**HIDS**).

Methods used by an IDS to detect intrusions:

Signature recognition
Pattern-matching of packets over signatures, generated at the network and transport layers, belonging to a known intrusion model.

Anomaly detection
Detection of an event outside the tolerance threshold of normal traffic and behavior.

Protocol anomaly detection
Detection of packets not following protocol standards.

Techniques for IDS evasion:

Insertion attack
The attacker obscures the exploit by inserting extra packets which will be received by the IDS but not by the target host.

TTL attack
Insertion attack in which the attacker sets the appropriate TTL on packets so that they will be received by the IDS but will not reach the target host.

Invalid RST
Insertion attack in which the attacker sends an RST packet with incorrect IP checksum, which the IDS interprets as the end of the session (and hence stops processing the traffic stream), while the target host drops the packet due to invalid checksum.

Urgency flag
Insertion attack in which the attacker sends a packet with the URG flag set. In such a packet the Urgent Pointer field indicates how much of the data in the segment, counting from the first byte, is urgent and should be prioritized. However, unlike the target host, some IDS ignore this field and consider the whole packet payload.

Pre-connection SYN
The attacker sends an initial SYN with invalid TCP checksum before the real connection is established. This desynchronizes the IDS with respect to TCP sequence numbers.

Post-connection SYN
The attacker sends a SYN after the connection is established. This desynchronizes the IDS with respect to TCP sequence numbers, but the target host ignores this packet as it references an already established connection. Then the attacker sends an RST with the correct (for the IDS) TCP sequence number to close its connection.

Evasion
The attacker sends portions of the exploit in crafted packets that the IDS mistakenly does not consider. This can be done e.g. by inserting data in the payload of a TCP SYN.

Unicode evasion
The attacker bypasses the IDS by encoding the packets in Unicode (UTF-8 and UTF-16), which features code points for multiple representations of the same character.

DoS
The attacker overwhelms the IDS capacity (CPU, RAM, disk space, network bandwidth) by sending a large amount of bogus traffic.

Obfuscation
The attacker obscures the exploit by encoding the packet payload in a way that it will be understood by the target host but not by the IDS.

Polymorphic shellcode
The attacker encrypts the packet and adds the code to decrypt it inside the packet, so that the IDS cannot recognize the attack signature. Done via a buffer overflow exploit where the return memory address points to the entrance point of the decryption code. Countermeasure: checking for no-op opcodes other than 0x90.

ASCII shellcode
The attacker encodes the attack code in ASCII characters which translate to assembly instructions, so that the IDS cannot recognize the attack signature.

Application-layer attack
The attacker hides the attack code inside application-layer compressed data (audio, video, images, etc.), so that the IDS cannot recognize the attack signature.

False positive
The attacker sends a large amount of traffic known to trigger alerts on the IDS, hiding the real attack traffic under the noise.

Session splicing
The attacker splits the exploit in several small packets so that no single packet triggers the IDS in itself. The attacker can also add a delay between packets to cause the IDS to timeout before it can reassemble and check them.

Fragmentation attack
If the IDS has a shorter timeout than the target host for reassembling fragmented packets, the attacker sends packet fragments with a long delay; the IDS drops them while the target hosts reassembles them.
If the IDS has a longer timeout, the attacker sends mixed valid and bogus packet fragments with a short delay; the IDS gets a reassembled packet with wrong checksum and drops it, while the target host gets the reassembled exploit payload.

Overlapping fragments
The attacker sends the exploit fragmented in packets with overlapping TCP sequence numbers. Different OSes handle fragments differently, so the IDS may end up with a bogus packet while the target host may end up with the exploit payload.

A **WIDS (Wireless Intrusion Detection System)** or **WIPS (Wireless Intrusion Prevention System)** monitors the radio spectrum to detect, block, and alert about possible intrusions in a wireless network.

Threats detected and prevented by a WIDS or WIPS:

Rogue Access Points

Wireless attack tools

DoS attacks

MitM attacks

Honeypots

Evil twin attacks

MAC spoofing

Ad hoc networks

Client misassociation with an AP

Unauthorized association with an AP

Misconfigured APs

Security countermeasures - honeypot

A **honeypot** is a part of a computer and network system set up as a bait for attackers. It looks like a legitimate part of the site and appears to contain information or resources valuable to attackers, but in fact it is isolated and heavily monitored. It is used to identify attackers and their techniques, as well as to deflect attacks.

Types of honeypots:

Low-interaction honeypot
Emulates a limited number of services and applications, and provides limited interaction with the attacker. Used to collect information about attack vectors such as network probes and worm infections.

Medium-interaction honeypot
Emulates a real OS with services and applications, but can only respond to preconfigured commands so the attacker might notice that its functioning is limited. Allows gathering more detailed data about an attack.

High-interaction honeypot
Not an emulation, but a real system actually running services and applications, or a whole network architecture of systems (honeynet). As such, it may be compromised or infected by an attacker. All intrusion activities are logged, so it allows collecting a large amount of data about the attacker. A honeywall gateway allows the attacker to interact with the honeynet while limiting outbound traffic via IDS technologies, to control the attacker and preventing harm to computers outside the honeynet.

Production honeypot
Emulates a real production system of an organization. Used to lure attackers to trigger alerts and hence get early warnings about an attack.

Research honeypot
High-interaction honeypot deployed in research, military, or government organizations to get detailed information about cyberattacks in order to study exploits and vulnerabilities.

Characteristics and methods used by honeypots (which can be used to identify them):

Layer 2 tarpit
A tarpit is a service purposely used to delay incoming connections. A Layer 2 tarpit is used to block network penetration of an attacker, but can be detected by the use of the MAC address 00:00:0f:ff:ff:ff which acts as a blackhole for Layer 2 connections.

Layer 4 tarpit
In a Layer 4 tarpit, the honeypot server receives the initial SYN packet from the attacker and replies with a SYN/ACK, but then does not open a socket and forgets everything about the connection. The attacker sends an ACK, believes the TCP three-way handshake to be completed, and starts sending data which never reaches its destination. The attacker does not receive acknowledgment for the sent packets and retransmits them; the connection eventually times out.
Layer 4 tarpitting can also be done by setting a TCP Window Size of 0 to block the attacker from sending further data, while keeping the connection open.

Layer 7 tarpit
Layer 7 tarpits are implemented by having services with a very high latency, e.g. a SMTP server which sends useless replies and waits a long time (15 secs or more) between lines.

SYN Proxy
As a protection against SYN flood attacks, a host applying SYN Proxy to a TCP connection responds to the initial SYN packet from the attacker with a specially crafted SYN/ACK, then waits for the ACK in response before forwarding the connection request to the server. The lack of SYN/ACK retransmissions may indicate the presence of a honeypot, especially `honeyd`.

Bait and switch
Technique consisting in redirecting all network communications to a honeypot once an intrusion attempt is detected. Attacker-side, this can be identified by the change in TCP/IP parameters (RTT, TTL, TCP timestamps, etc.)

Fake Access Points
Honeypots may create fake Wi-Fi APs. These APs have random SSIDs and send beacon frames but no other traffic.

User-mode Linux (UML)
User-mode Linux allows multiple virtual Linux kernel-based guest OSes to run as an application within a host Linux system. It is often used for honeypots, so references to UML in `/proc` subdirectories may indicate that the attacked system is a honeypot.