*nix Study Guide

An overview of things to know to be comfortable in a *nix computing environment (MacOS or Linux).
If you feel comfortable with the things here, you're ready to face anything in the Real World.
If anything is new or surprising, follow the links and learn more!

Commands

Networking	Unix	Modern
dig	vi	brew
mtr	grep	git
curl	sed	node
nc	awk	python3
ssh	cut	docker
	tr	aws
System	sort	jq
which	wc	
launchctl		
systemctl		

Shell

Commands
Flags & Arguments
Environment Variables
Pipes

Variables

```
PATH
PORT
AWS_PROFILE
```

File System

Core		
pwd	/home	Per-user files, configurations, etc.
cd	/etc	System-wide configuration settings.
ls	/var	System-wide generated files (logs, database files, etc).
cat	/opt	System-specific programs, eg databases, web servers.
cp	/usr/bin	General utility programs provided by the distribution.
mv	/bin	Core system programs.
ln		
rm		
echo		
env		
sh		

Core File System Commands

These core file system commands are fundamental to working in a Unix system. Unix systems are designed around the notion of a file system, and the shell is the main tool to work interactively with the file system. These commands are the core operations on the fs.

- `pwd` Print the current working directory (where am I?).
- `cd` Change the current working directory (move around the file system).
- `ls` List the files in the current directory. `-l` in long form. `-a` include hidden files, which start with a `.` (period). `-h` (with `-l`) print file sizes in human readable values. `-s` expand symlinks.
- `cat` Write the contents of the file to standard out. With no pipe, prints the file to the command line. With a pipe, makes the file the `stdin` for the next command.
- `cp` Copy one file to another. `-r` recursive to copy a directory.
- `mv` Move (rename) a file, possibly changing directories.
- `ln` Create a symlink to a file.
- `rm` Remove (delete) a file. `-r` recursive to remove a directory. `-f` force to not ask permission.(DANGEROUS)
- `echo` Print a string, expanding any variables.
- `env` Print the current environment variables.
- `sh` Start executing a new shell.

What's the Difference Between Bash, Zsh, and Other Linux She...
Most Linux distributions include the bash shell by default, but you could ...
hostingseek.com

https://linuxize.com/post/linux-cat-command/

vi survival guide
1 Introduction 2 Basic crash course 3 Other Modes 3.1 A guide to...
ucsd.edu

Using Grep & Regular Expressions to Search for Text...
Grep is a tool used to search for text/text patterns within text input a...
digitalocean.com

The Basics of Using the Sed Stream Editor to Manipulate...
Sed is a powerful text editor that can be integrated into other tools and ...
digitalocean.com

How to Use the awk Command on Linux
On Linux, awk is a command-line tool for data manipulation (formerly, as well as a pro...
hostingseek.com

Unix System Commands

The Linux system commands are common commands that slice and dice the text within files. The Unix philosophy revolves around storing data in files, moving data in pipes, and composing small tools for bigger results.

- `vi` A common, command-line text editor. It uses an unusual editing paradigm, so be comfortable doing the most basic tasks with it.
- `cut` Process line-oriented files as columns.
- `tr` Translate or Transform characters. For instance, change
- `grep` Search for lines in a file that match a pattern. Patterns are specified using a simplified regex format. `-ABC` n for n lines After, Before or around Context. `-n` print matching line number.
- `sort` Sort the lines in a file. `-f` ignore case. `-n` sort numerically, instead of alphabetically. `-h` sort numerically, but using K/M/etc number suffixes for scale.
- `uniq` Remove repeated lines in a file. Works best when placed after sort. `-c` print a count of each line. `-d` print only lines with count greater than one. `-u` print only lines with count equal to 1.
- `wc` Word (line, character) count. Prints the number of characters, words (space separated), and lines in the file.
- `sed` Stream Editor to perform complex editing patterns on text.
- `awk` The most useful tiny programming language that no one ever uses. Can make very useful programs for parsing logs and other line-structured text.

Homebrew
The missing package manager for macOS (or Linux)
Homebrew
The Missing Package Manager for macOS (or Linux)...
brew.sh

Common Variables

These variables are used often, or contribute to common issues.

- `PATH` The PATH variable controls where executable commands are found.
- `PORT` PORT is used by many CI/CD and production deployment systems to control how a server listens for new requests.
- `AWS_PROFILE` When using the AWS CLI or any AWS program tools, AWS_PROFILE allows selecting between one of several configured profiles.

Git
git-scm.com

Modern Tools

The Internet changed much of how computing is achieved and how collaboration happens. These tools are "modern" developments, that have achieved widespread use and adoption for their respective areas. Each tool is an entire topic in itself.

- `brew` A package manager for a number of environments.
- `git` Source control with git allows tracking changes to files over time and enables easily sharing versions of files in group and distributed settings.
- `node` Node.JS is a robust platform for developing web servers and I/O heavy tasks using the JavaScript programming language.
- `python3` Python is a widely distributed programming environment, known for ease of use & setup. It has seen widespread usage in data analytics and machine learning.
- `docker` Docker containerization makes it possible to distribute a self-contained computing environment. This environment has only the files and configurations necessary for an application.
- `aws` The AWS CLI is the main tool for interacting with AWS cloud resources in a DevOps environment, and serves as the basic entry point for troubleshooting AWS tasks.
- `jq` Most web APIs return data in json format. jq is a command line tool for parsing, filtering, and inspecting JSON files.

Welcome to Python.org
The official home of the Python Programming language
python.org

About | Node.js
Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.
nodejs.org

System Control Commands

These commands are used to control system processes or "daemons", such as databases and long running webservers.

- `which` Determine what program will be executed for a single-word command.
- `cron` Used to schedule background tasks at consistent times.
- `launchctl` Start and stop services on MacOS.
- `systemctl` Start and stop services on Linux using systemd.

Tutorial
manpages.debian.org

Command Line Interface - AWS CLI - AWS
aws.amazon.com

Docker Documentation: How to Build, Share, and Run
Docker Documentation is the official Docker library of resources, tutorials, ...
docs.docker.com

How to Use Linux dig Command (DNS Lookup) (With Examples)
Dig command-line utility on the Linux operating system is a powerful ...
phoenixnap.com

curl - Tutorial
curlie.io

Network Commands

These commands are the workhorses of troubleshooting network connections. Network connections are whenever any programs talk to each other not using files and pipes. Usually, this means communicating over a local intranet or the Internet.

- `ping` Verify connectivity to another computer, either by IP address or hostname.
- `dig` Perform a DNS lookup. Used to verify that a hostname or prefix is correctly associated with an IP Address or CNAME.
- `traceroute` Troubleshoot where, exactly, the network connectivity is failing.
- `curl` Perform a wide number of Internet network request. `-vv` for verbose information, including SSL. `-H` set an HTTP header. `-X` send an HTTP post message body.
- `nc` Send and receive arbitrary data across a network connection. Useful when for sending malformed or invalid data, especially for testing.
- `ssh` Open a shell on a remote computer.
- `scp` Copy files between the local and a remote computer.
- `ssh-keygen` Manage secure keys & tokens for SSH connections. `-t ed25519` current (2023) best in class encryption algorithm.

How to Use The Netcat (nc) Command: An In-Depth Tutorial...
Netcat (nc) is a powerful networking tool for system administ...
hostinger.com

GitHub

Every Linux networking tool I know
explainshell.com

How to Create & Manage SSH Keys on Linux Machines - Jump...
SSH keys are a secure method for gaining an SSH connection between two p...
jumpcloud.com

Connecting to GitHub with SSH - GitHub Docs
You can connect to GitHub using the Secure Shell Protocol (SSH), which pr...
docs.github.com

File Systems

Data is stored in *nix operating computers in a structure called the file system. A file system stores individual units of data in a file, and organizes files using directories, which themselves can contain either files or directories.

Plain text vs Binary

Files can store any type of data - program source code, employee records, rich text documents, and more. Files whose data can be read normally by a person are called "plain text", while files whose content requires specific computer code to decode is called "binary". Program source code, configuration data, and plain text data like CSV and XML are typically stored in plain text. Compiled executable programs, database data files, or rich text documents are typically stored in binary.

What is the difference between plaintext and binary data?
Many languages have functions which only process "plaintext" not binary. D...
stackoverflow.com

Hierarchy

Directories allow for a hierarchical organization of files. Directories have a name, a parent (except the root directory, /), and zero or more children. Children within a directory are either files or directories. The top level directories (those immediately under root) are specified, though are slightly different between Linux and MacOS.

The most commonly user- or admin-accessed folders are

- `/home` Per-user files, configurations, etc. on Linux
- `/Users` Per-user files, configurations, etc. on MacOS
- `/etc` System-wide configuration settings.
- `/var` System-wide generated files (logs, database files, etc).
- `/opt` System-specific programs, eg databases, web servers.
- `/usr/bin` General utility programs provided by the distribution.
- `/bin` Core system programs.

Classification of Unix File System
Ordinary Files
Directories
Special Files
Pipes
Sockets
Symbolic Links
Unix File System - GeeksforGeeks
A Computer Science portal for geeks. It contains well written, well thought ...
geeksforgeeks.org

Names and Extensions

File and directory names are unique within the parent directory. By convention, file names end with an extension. An extension is a followed by (usually) two or three characters that describe what sort of file it is. For instance, data.csv would contain comma separated data values, while image.png would be an image using the PNG image format. However, this is convention only, and cannot be relied upon.

Filename extension - Wikipedia
en.wikipedia.org

What do some common Unix file extensions mean?
Extension .pbx file extensions include: ...
ask.ubuntu.com

Do file extensions have any purpose in Linux?
Linux determines a file's type via code in the file's header. This process does...
askubuntu.com

Size

Files have a size, which is the number of bytes of content in the file. One English character is encoded in a single byte, so for plain text documents that only have alphabetical English and punctuation, the number of bytes is the same as the number of characters in the file. For files containing other languages, or binary files, the size is just how big it is.

Permissions

Files and directories have permissions, which control who is allowed to do what with the file or directory. Control is typically broken into three access levels and three permissions. The permissions are read, write, and execute. The access levels are user, group, and world. When a program access a file, the system checks the type of access & the user running the program against the permissions of the file. For instance, if the file is owned by the accountants group with group write permissions, and an accountant user tried to write to the file, the operation would be allowed. If an engineer user tried to write to the file, the operation would be denied.

Unix / Linux - File Permission / Access Modes
Unix Linux File Permission Access Modes - In this chapter, we will ...
tutorialspoint.com

How UNIX File Permissions Work
mason.gmu.edu

Symlinks

Symbolic links, or shortcuts, are special entries in a directory which "point" to another location. This other location can be a file or a directory. If it's a directory, it's "as if" the other directory were a child of the directory that the symlink is in.

Symlink Tutorial in Linux - How to Create, Remove, Use ...
Symlinks (soft links) are a special type of file in Linux that points to ...
linuxhint.com

Networking

Computers are most useful when they work together. Computers work together by communicating, and that communication is handled by the operating systems' networking stack. The networking stack, like the command line and file system, are a concept to streamline the communication process.

The networking is built on four conceptual layers, called as a whole the TCP/IP model. Having a high-level understanding of each layer in the stack is necessary to troubleshoot the entire range of issues when networking is involved. From bottom to top, the layers are Data Link, Network, Transport, and Application. The Data Link layer handles actually moving the data from one computer to the next. This is typically done with Ethernet for wired networks, or WiFi for wireless. The Network layer handles routing data through the network of computers. This uses IP (Internet Protocol) to route the data from the source to its destination. The Transport layer handles getting the data from the right program on one end to the desired program on the other. The application layer is how the program itself handles the data.

TCP/IP Model - GeeksforGeeks
A Computer Science portal for geeks. It contains well written, well thought ...
geeksforgeeks.org

Application Layer

The Application layer describes how two programs, once they've connected, will agree to communicate. These protocols describe the data that the two programs will send between each other.

Most networking happens using the HTTP protocol, often with TLS encryption. HTTP, or Hyper Text Transfer Protocol, is a plain text protocol that treats all communication as requests for a specific file. The computer that initiates the request, the client, sends a request for a URL to another computer, the server. The computers optionally agree to encrypt their communication when using HTTPS. After that optional encryption, the server sends the contents of one file back to the client, who decides what to do with it.

While most data transfer happens over HTTP, the DNS (Domain Name Service) protocol is perhaps more fundamentally important to the operation of the internet. DNS is responsible for converting queries for a hostname, like www.google.com, into an IP address, like `142.250.189.196`. See the Network layer for more on the importance of IP addresses.

Other Application level protocols include SMTP, for email; SSH, for a secure remote shell; and RDP, for running a remote desktop connection.

An overview of HTTP - HTTP | MDN
HTTP is a protocol for fetching resources such as HTML documents...
developer.mozilla.org

https://www.cloudflare.com/learning/ssl/what-is-ssl/

Transport Layer

The Transport connects programs running on different computers. A single computer runs many programs. To distinguish between those programs for the networking stack, each server claims a port number. While any program may request any port number, certain ports are "well known" and typically used for one specific type of application. Handling this distribution is the responsibility of TCP (Transmission Control Protocol).

SSH: 22 Secure remote Shell
DNS: 53
HTTP: 80 Unencrypted HTTP communication
HTTPS: 443 Encrypted HTTP communication
RDP: 3389 Windows Remote Desktop
psql: 5432 PostgresQL database connections

Additionally, development servers will often choose a port. 8080 is common for testing HTTP servers, while 3000 and 5000 are common choices for testing JavaScript and Python web applications.

https://www.cyberciti.biz/faq/linux-unix-open-port/

Network Layer

The network layer connects the computers themselves. While the two computers may each be running several programs, the communication between those programs is opaque to the network layer. Instead, the network handles routing between several intermediary computers through the Internet Service Providers (ISPs). Most home computers do not connect directly to computers at the web host's data center. Instead, the home computer connects to the home router. The home router connects to the user's ISP. The user's ISP itself connects to the host's ISP, which connects to the data center computer.

Each point on that path needs to hop to the next point on the path. The IP (Internet Protocol) achieves this by allowing each computer to know what direction to send the data. The computer knows it wants to send data to a certain IP address (which it learns using DNS). The computer has a map of IP Subnets, which are ranges of IP addresses. The subnet `142.250.189.0/24` describes all IP addresses that start with `142.250.189`. If the computer has a map that includes that subnet, it will forward the data to the computer responsible for the subnet. Every computer also has an "upwards" host (its internet gateway) which will know about a wider range of addresses.

Using this hierarchical scheme, a computer will always know where to send data to get it closer to its intended IP address.

Subnetting, Subnet Masks, and Routing
What is subnetting? Why use subnetting? How do you calculate a s...
youtube.com

IP Routing and Subnets
IP routing is explained with an example of a small internet with subnetting.
examples.com

Data Link

The Data Link layer handles the physical connection between two single computers. Each of these computers is one part of the network hop, and the protocols in the data link layer ensure the data physically gets from one machine to the next. Wired connections use a protocol called Ethernet, and wireless connections can use WiFi for short-range home networks or LTE/5G for longer range cellular communication.

Using *nix as a personal computing (laptop or desktop) probably won't connect to anything other than WiFi or Ethernet, but Internet of Things (IoT) devices make use of a truly astonishing number of Data Link connections. Knowing them all isn't necessary, but knowing the ones in use by the network devices will be.

There is a 7-layer Networking model often used called the OSI model. In the OSI model, the Network Access Layer is divided into the Physical and Data Link layers. The Application layer is broken into the Session, Presentation, and Application layers. For common *nix usage, the additional complexity is not necessary.

Troubleshooting Ethernet network connectivity issues
Ethernet network controller troubleshooting information
ibm.com

Wi-Fi Not Working in Linux? Here's How to Fix It - Make...
There are lots of issues getting non-Intel Wi-Fi cards to work well.
makeuseof.com

Physical and Data Link Layer Protocols - Ethernet, Bluetooth, Wi-Fi, ...
Ethernet is the most common LAN architecture for IEEE was contained an...
engineersgarage.com

Executing a Command

When executing a command in the shell (or terminal), four steps happen.

1. Find the correct executable program in the current PATH.
2. Prepare the Environment Variables and Arguments for the program.
3. Execute the program with the Arguments as the `argv` and with any Environment Variables made available.
   a. Connect the program's standard input, output, and error.
4. The program completes with an exit code.

PATH Lookup

PATH is an environment variable with a list of folders, separated by `:`. When a command is run, the system looks for a file with the program's name in the folders in the path, in order, taking the first that is found.

PATH (variable) - Wikipedia
en.wikipedia.org

Unix / Linux - Environment
Unix Linux Environment - In this chapter, we will discuss in detail...
tutorialspoint.com

https://linuxize.com/post/how-to-set-and-list-environ...

Environment Variables

Environment Variables are a map of string keys to string values. They can be set in the shell by running `export NAME=VALUE` on its own line, or including `NAME=VALUE` before the program name in a command line.

Arguments

Arguments are the values after the program name on the command line. The meaning of these values are left entirely to the program; however, there are some common conventions.

- Flags are arguments that have a `-` in front of them.
  - "Short" flags are typically one letter mnemonics for some options for a program, sometimes with an additional value.
  - "Long" flags start with `--`, are full words, and may have arguments.
  - Short flags are usually preferred when running commands interactively, and long flags are preferred in script files.
- Positional Arguments don't start with a `-`, and don't have a flag that requires an argument before them.
  - Most commands take a file path as a positional argument to take which file to work on.

Linux - Commands-and-arguments - w...
Linux - Commands-and-arguments - In this lesson, we learn about the pha...
w3resource.com

File Descriptors

Executing a program means starting its binary executable, providing the Arguments and Environment, and connecting three special files: `stdin`, `stdout`, and `stderr`.

- `stdin` is the input for the program. It will operate on the lines of this file.
- `stdout` is where common, correct output is logged. These are the values computed during normal operation of the program.
- `stderr` is where the program provides any exceptional or error case output. This can describe any errors in the execution and running of the program.

What Are stdin, stdout, and stderr on Linux?
In this tutorial, we'll take a look at how data is produced and consumed when we launch a c...
baeldung.com

File Descriptor Redirects

With no additional configuration, these are connected to the terminal shell itself. Output, both stdout and stderr, are printed directly to the terminal. Input comes from what the user types. These can be redirected using several special operators at the end of the command.

- `>` redirect stdout to a file.
- `2>` redirect stdout to a file.
- `<` redirect stdin from a file.
- `|` connect stdout from one program to stdin of the next.

There are several special files of interest

- `/dev/null` is a special file that discards any information written to it.
- `/dev/random` is a special file that always has random bytes to read.

Confused about stdin, stdout and stderr?
I am rather confused with the purpose of these three files. Can it my understand...
stackoverflow.com

Exit Code

When a program finishes, it exits with a final integer error code. Exit code 0 means the program completed successfully. Any other number means there was an error; the meaning of the error is program dependant.

https://shapeshed.com/unix-exit-codes

Sample Command Breakdown

```
PORT=8080 node --cpu-prof test.js -n 12 > ./prof-run-out
```

- `node` The command to run, the NodeJS javascript runtime
- `PORT=8080` An environment variable, specifying the PORT variable is 8080
- `--cpu-prof` A long flag that will be sent to NodeJS
- `test.js` A positional argument, that NodeJS will use as the JavaScript to execute.
- `-n` A short flag, that node will pass to the test.js program.
- `12` The value for the `-n` short flag.
- `> ./prof-run-out` Redirecting the standard output (stdout) of the NodeJS runtime.

Scripting

Scripting involves writing a plain text program file (called a script) that has several shell commands to execute in order. Scripts can use repetition and conditional logic to respond to a variety of situations in the computer.

What is a simple explanation for how pipes work in Bash?
I often use pipes in Bash, e.g. cmdsrcp | sed -e'cmd1' | tee -a log cmd2...
stackoverflow.com

Bash Pipe Tutorial
Comprehensive tutorial on the different common uses of the pipe (|).
linuxhint.com

Pipelines

Pipelines join the output of one command to the input of the next using shell stdin/stdout redirection. The two commands run concurrently, sharing data through the pipe. The success of the pipeline is the success of the last command in the pipe.

Commands can be listed on one line using `;` to end one command before starting the next.

How to Use Bash && Operator in Linux
Bash && command can be used in multiple ways as it is a logical...
linuxhint.com

Control Operators

Commands can be conditionally joined using the `&&` and `||` operators.

- `command1 && command 2` Executes `command2` if `command1` succeeds.
- `command1 || command 2` Executes `command2` if `command1` fails.

Bash OR Operator
Practical tutorial on the OR logical operator which can be used in the ...
linuxhint.com

OPEN SOURCE
An Introduction to parameter expansion in Bash
Get started with this quick tutorial to guide on expansion modifiers that ha...
opensource.com

Parameter Expansion

Bash parameter expansion reads a variable and replaces the argument with its value.

Parameter expansion can additionally do string manipulation on the value in the variable.

Shell Parameter Expansion (Bash Reference Manual)
Shell Parameter Expansion Bash Reference Manual.
gnu.org

Loops and Conditionals

Shell scripts can use the `if ... ; then ... fi` structure to conditionally execute a block of code. The condition is usually expressed using the `test` command, which is shortened to `[`

Shell scripts can use the `for ... ; do ... done` to repeat blocks of code. Lists of strings are separated at word boundaries and assigned to a variable inside the body. Subshells are often used to generate the line-by-line lists for the loop.

IF Statements - Bash Scripting Tutorial
Basic if statements. Learn if statements, elif and else cases.
linuxhint.com

Introduction to IF
tldp.org

https://www.cyberciti.biz/faq/bash-for-loop/

Bash nested subshell argument expansion
Why is the Blue being printed here as a literal, even though the outer...
stackoverflow.com