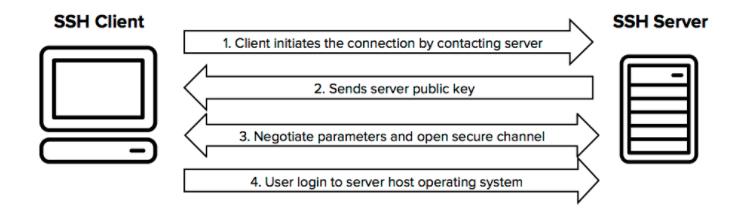
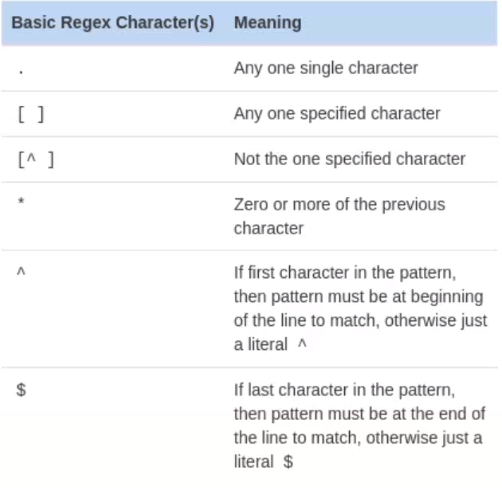
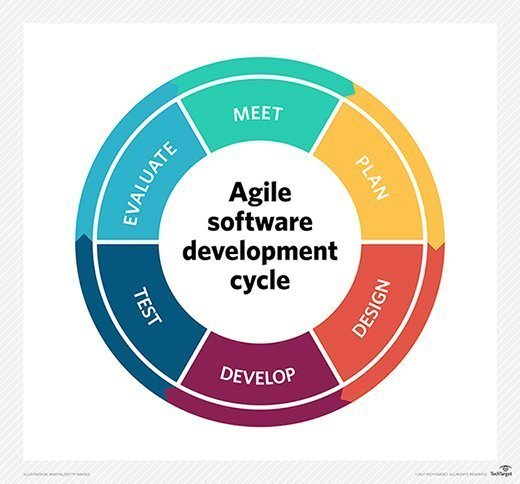
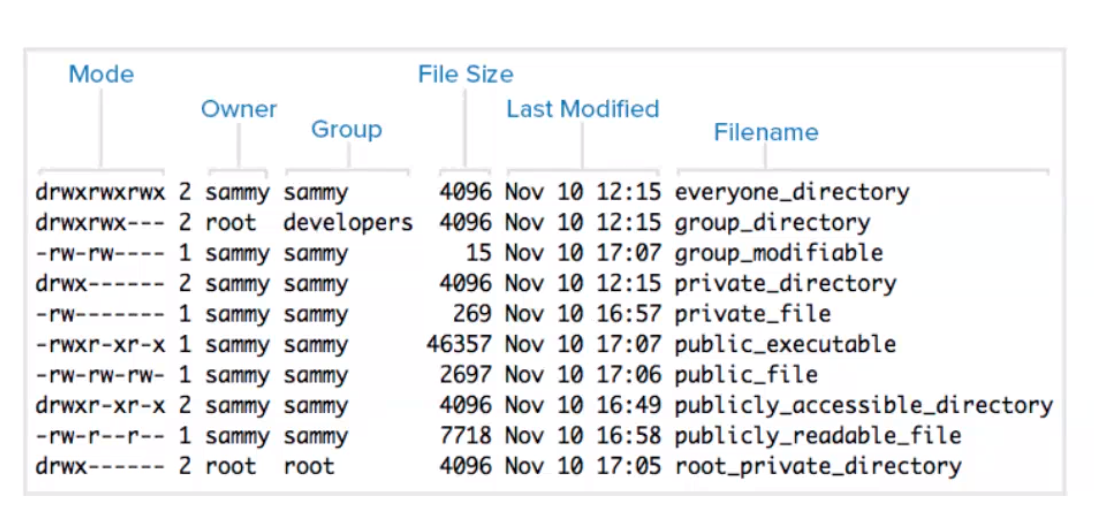
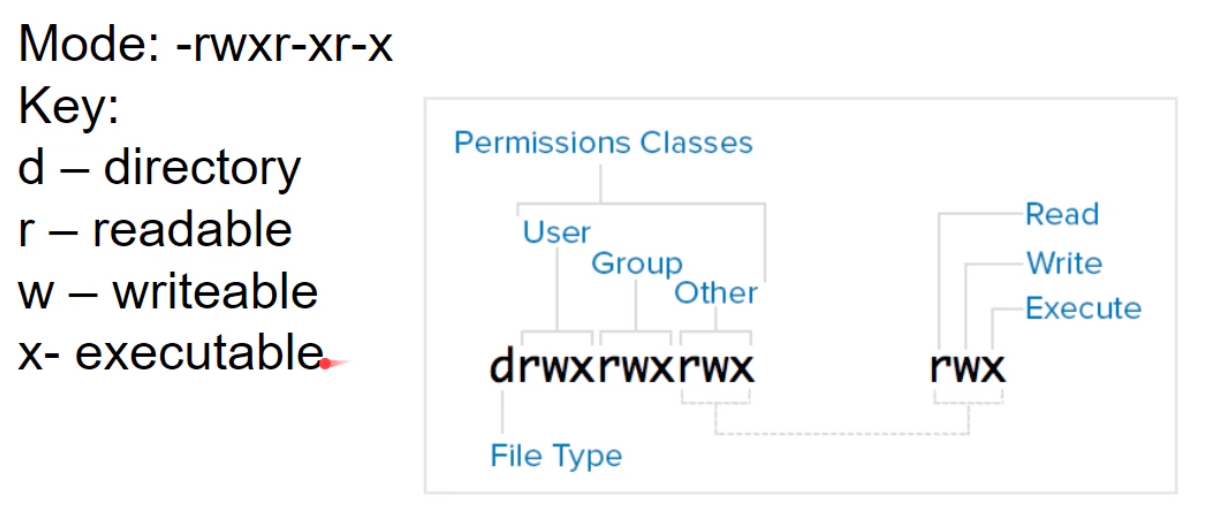
Other stuff

* Telnet: like SSH, but doesn’t have many security features, because back then not many people use the internet, so people didn’t think much about cybersecurity. Still usable today using telnet clients such as PuTTY. Text-based communication
  + Towel.blinkenlights.nl port 23 has ASCII Star Wars
  + Towel.blinkenlights.nl port 666 has a random excuse generator
* SSH: Secure SHell. A secure way to connect to a remote computer



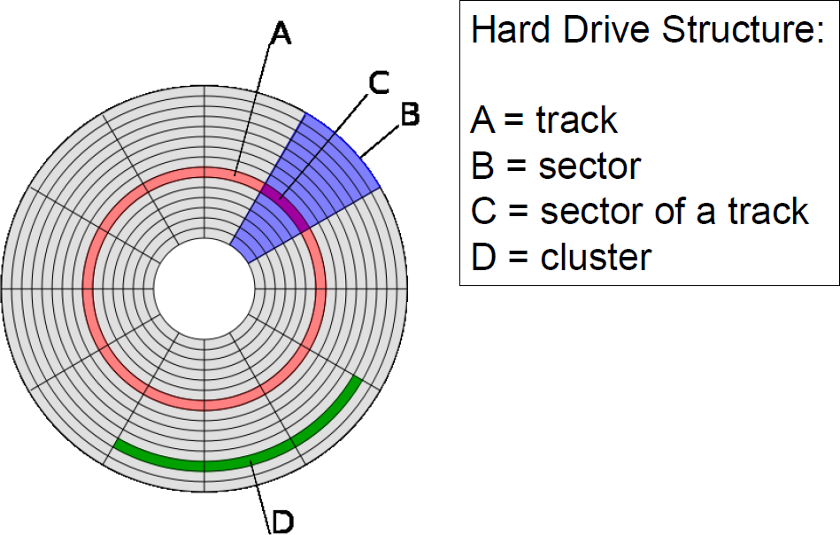
* Command-line interface
  + Interacting with a computer using text
  + Most computers don’t have a GUI, so CLI is the only way to interact
* Regular expression
  + A string alone is looking for that sequence of characters
  + Multiple regex operators can be combined
  + [<start character>-<end character>] is a range
    - Ex: [a-d][c-e] searches: ac, bc, dc, ad, bd, cd, ae, be, ce
    - Range ordering based on ASCII order
    - Chained: [<start>-<end><resume>-<finish]
  + A list of characters in [ ] means to contain any of those characters
    - Ex: [123] searches for 1, 2 and 3
  + [^ ]
    - Evaluates the same as the previous [ ] stuff, but the ^ inside it means not matching
  + Stuff inside [ and ] evaluates to a single character at a time
  + ^
    - Everything that starts with…
  + .
    - a single any character
  + {start,end}
    - Repeat the immediate previous thing greater than or equal to start number of times, but less than end number of times
  + “ “
    - Combines to a single unit
  + $
    - End with the thing immediately preceding it
  + \*
    - Containing 0 or more of the previous character
  + 
* Agile software development
  + 

Linux

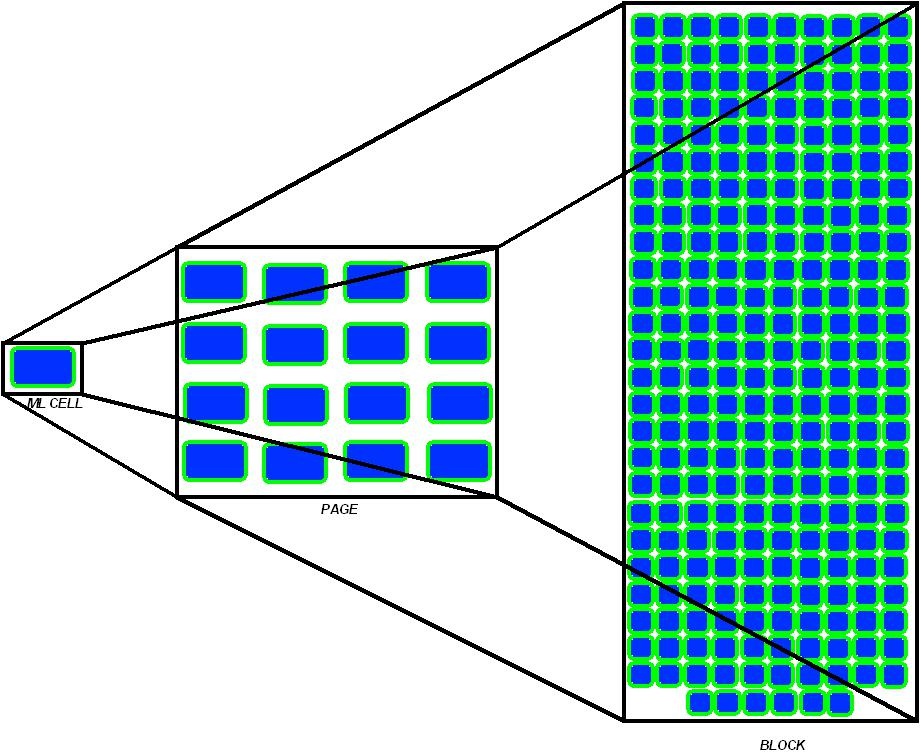
* Shell commands
  + ls: output the directories and files located in the current directory
    - -l option is for the long format, which shows more details (such as permissions, owners, dates)
  + cd LOCATION: change the current directory to LOCATION
    - .. means up a level
    - . means current directory
  + less FILE: output the contents of FILE, scrollable
    - Press q to close
    - K to scroll up
    - J to scroll down
  + more FILE: output the contents of FILE, can’t scroll back up
    - Scroll by line using enter
    - Scroll by page using space
  + cat: dumps all the content at once (no need to scroll)
  + pwd: prints the current directory
  + man COMMAND: show the manual for COMMAND
    - Shows how to use it, syntax, and options
  + help COMMAND: show help for COMMAND
  + mv FILE LOCATION: moves FILE to LOCATION
  + rm FILE: deletes FILE
  + mkdir NAME: creates a directory named NAME at the current directory
  + sudo COMMAND: runs COMMAND with elevated permissions
    - sudo -i: switch user to root
  + grep EXPRESSION FILE: outputs lines from FILE containing EXPRESSION
    - Use the -E switch flag to use complex regex
    - -c counts occurrences
  + touch FILE: creates FILE at the current directory
  + cp OLD\_FILE NEW\_FILE: creates a copy of OLD\_FILE named NEW\_FILE
  + file FILE\_NAME: prints the encoding of the file FILE\_NAME
  + echo TEXT: prints TEXT
  + clear: clears the screen
  + sort: output lines sorted by alphabetical order
  + uniq: outputs the lines that are unique (comparison is done using adjacent lines)
  + wc: line, word, character counter
  + chmod PERM\_NUMBER ITEM: change the permission of ITEM to PERM\_NUMBER
  + ps: shows running processes
  + kill PID: send kill signal to the process with PID of PID
    - Add a -9 flag to forcefully execute the PID
  + trap <command> <signal>
    - Sets a trap so that if <signal> is received, <command> will execute instead of what that <signal> usually does
  + jobs
    - Print a list of background jobs and their numbers
  + fg %<number>: bring background job <number> to foreground
  + sha512sum <file>
    - Prints the SHA-512 checksum of <file>
    - Changing just one bit of <file> will result in a completely different checksum
  + disown %JOBNUMBER: gives up JOBNUMBER to init, making it immune to SIGHUP
    - Add -h to make the transition happen when you logout of the session
  + nohup COMMAND: runs COMMAND and make that process immune to SIGHUP by giving the process to init when you logout of the session
  + nice -n NICENESS\_NUMBER COMMAND: run COMMAND in a process with the niceness number of NICENESS\_NUMBER
  + renice -n NICENESS -p PID: sets process with PID to have niceness number NICENESS
  + pgrep DESCRIPTOR: output the pid associated with DESCRIPTOR
    - Use -u to specify user
  + pkill DESCRIPTOR: sends SIGTERM to the pids associated with DESCRIPTOR
    - Use --signal to specify another signal
  + pidof PROCESS\_NAME: return the pids with the exact same PROCESS\_NAME
  + which COMMAND: outputs the location of COMMAND
  + find <start> <expression>: starts searching from <start> to search for file names using <expression>
  + gzip <source>: compresses <source> using gzip, and output compressed version as <source>.gz
  + gzip -d <source>: un-gzip <source>
  + tar -cvzf <name> <directory>: create a tar of <directory>, then gzip it, with <name>
  + tar -xvzf <name>: extracts tar <name>
  + adduser <username>: creates a new user with username <username>
  + passwd: change your password. You could add a username as argument to change the password of another user
  + usermod: change the permissions of a user
  + uname -a: print the OS version
  + systemctl: system control
    - systemctl start <name>: starts process <name>
    - systemctl restart <name>: restarts process <name>
    - systemctl stop <name>: stops process <name>
    - systemctl status <name>: get status of process <name>
  + To pass arguments, -optionName value at the end
  + If an argument has a space, then put quotes around it, or escape the space using \
  + COMMAND > FILE\_NAME: sends the stdout of running COMMAND into the file FILE\_NAME. Overwrites if file exists already
  + >> would do the same thing, but append, not overwrite
  + <command> 2> <file>: appends stderr from running <command> to <file>
  + Piping:
  + Command1 | command2
  + The output of command1 is passed to command2 as argument
* Shells
  + Bash: Bourne again shell
  + Ksh: Korn shell
  + Csh: C shell
* You work in the /home/<username>/ directory
* ~ means home
  + ~USERNAME can go to the home of that USER
* Press up or down keys to navigate command history
* Ctrl+k = delete everything after cursor
* Tab to autocomplete file/directory name
* Esc+d deletes one word at the cursor
* Ctrl+a moves the cursor to home
* getent passwd shows all users, human and program
* Text editors
  + Vi
    - To open a file: vim <filename>
    - Press esc then type :w to save
    - Add a file name after :w to save as another name
    - Press esc then type /<word> to search for <word>
      * Press n for the next instance
    - Search and replace the next instance: :s/<search>/<replace>
    - Use %s to search and replace all instances
    - J = up arrow
    - K = down arrow
    - H = left arrow
    - L = right arrow
  + Emacs
    - To open a file, ctrl + x then ctrl + f, then type the location of the file to open
    - To save a file, ctrl + x then ctrl + s
    - To save the file as another file, ctrl + x then ctrl + w
    - To search: ctrl + s
    - To search and replace: alt + shift + 5. Follow prompts. In the end, press y to replace the next instance or ! to replace all instances
    - Exit the editor: Ctrl + z then ctrl + c
    - Scroll to top: alt + <
    - Scroll to the bottom: alt + >
    - Undo: ctrl + x then u
  + Nano
    - To open a file, press ctrl + r
    - To save: ctrl + o
    - To search, press ctrl + w
      * Use alt + w to go the next instance
    - To search and replace, use ctrl + \
      * Press y to replace the next instance
      * Press a to replace all instances
    - To exit editor, use ctrl + x
    - To get help, use ctrl + g
* File properties, in ls -l
  + 
  + The number after mode shows the number of hard links
    - For files, it usually shows how many metadata files are associated
    - For directories, it usually shows how many directories are under it in ls -a
  + If last modified has no year, then it’s the current year
  + File sizes are in bytes
  + Mode
    - 
    - The user is the owner. The owner is the creator unless ownership transferred
    - Group is a selected group of users the file is shared with
    - As a number:
      * 3 digits: first digit is sum of permissions of user, second is permissions of group, third is permissions of other
      * r = 4, w = 2, x = 1
      * (octal)
      * 755: 111101101: -rwxr-xr-x
* Processes
  + A program can have multiple instances, multiple processes
  + Processes are started by the user or the OS
  + Each process is identified by its PID
  + Processes can become a parent by spawning a child process
  + Ctrl+c kills (SIGINT) the current foreground process
  + Ctrl+z pauses (SIGTSTP) the current foreground process
    - Find all paused (background) processes using jobs
    - Bring back the process using fg %<job number>
    - Pause means frozen; not running. It’s memory contents are placed into a swap file in storage
  + Once you log out, all the processes forked from your session receive a SIGHUP
  + Daemons are background jobs, immune to SIGHUP
  + Zombie process: A process that has completed its execution, but its process still exists, so its parent process can request for its status code.
  + Orphan process: a child process that’s still running despite its parent process has finished running
  + Priorities
    - “Niceness number”
    - A number between -19 to 20, with 20 being the lowest priority (nicest)
    - Default value of 0, require sudo to change
    - ls -l shows priorities
* Kernel
  + Small piece of software
  + Closest piece of software to the hardware
  + Manages communication between processes
  + Stored in /boot/vmlinuz

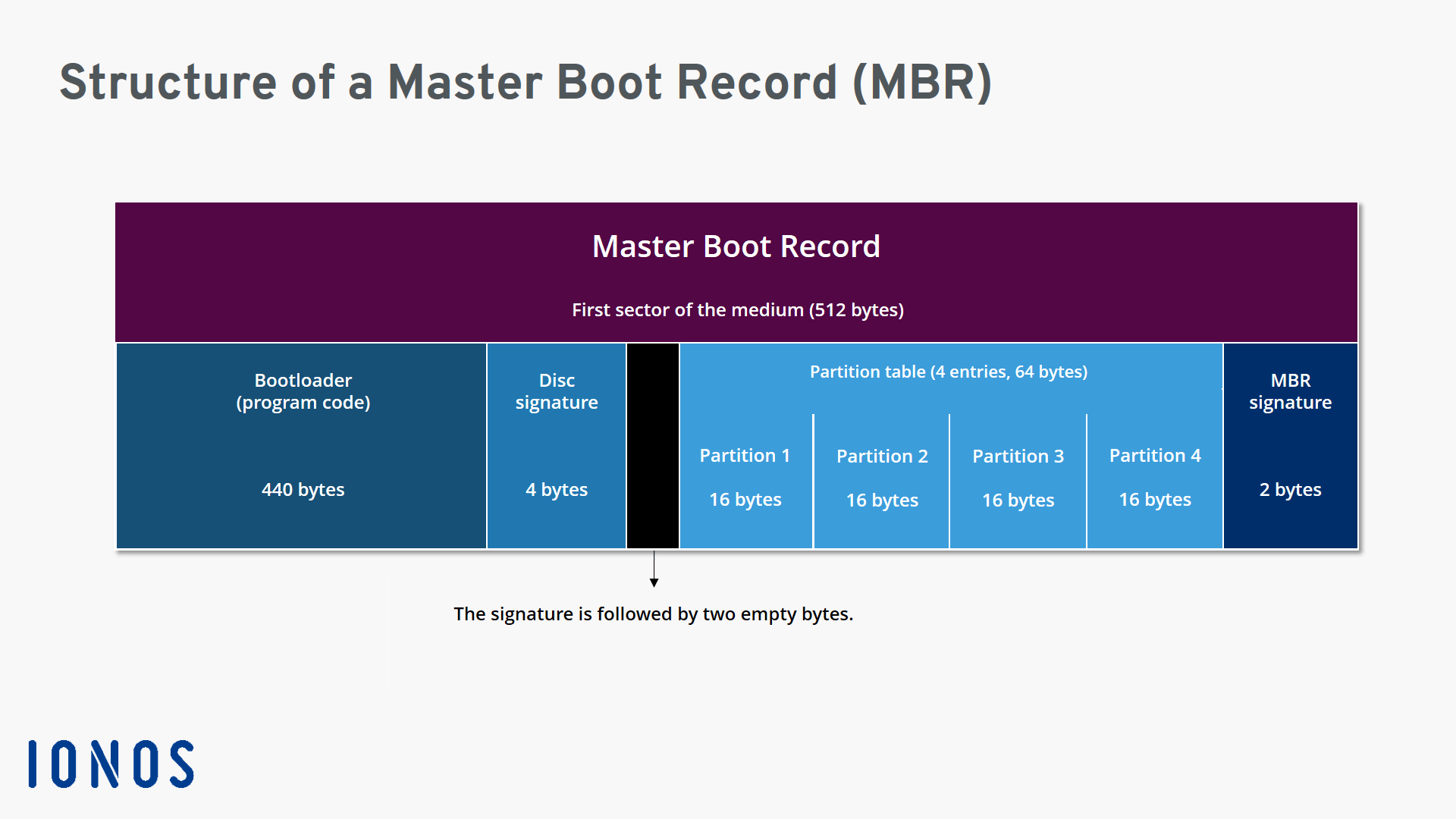
File Systems

* A structure to organizing files, with rules on how to access and store things
* Root
  + Where the tree starts from
  + In Windows, each drive has a root
  + In Linux, the root is /
    - Only one root exists, regardless of the number of drives
* Absolute path: shows all the directions needed to pass through to get to a file, starting from the root
* Relative path: shows all directories needed to pass through to get to a file, starting at the current directory
* Tree diagram: Show the hierarchical relationships between directories, and the files within the directories
* Files and directories starting with . are hidden by default. They’re usually used to hide config files for organization
* Storage devices
  + HDD
    - A stack of spinning disks (platter)
    - Each disk is divided into rings (tracks), which are then divided into sectors
    - Multiple sectors of a track form a cluster



* + SSD
    - Flash-based
    - Much faster than HDD



* Clusters
  + A set unit amount of storage space
  + If a file partially occupies a cluster, that whole cluster can’t be used for anything else
  + Each cluster on a drive must be of the same size
  + Dividing a drive into fewer clusters but each cluster is bigger:
    - A larger cluster size would lead to more space wasted
    - Lower cluster count would boost performance when searching for files
  + Dividing a drive into more clusters but each cluster is smaller:
    - A smaller cluster size would help prevent wasted space
    - A higher cluster count would make searching for files take longer
* Partition: divides the disk into multiple isolated spaces. It’s like having multiple disks on one
* Master boot record
  + The first sector of a drive
  + Computers don’t know how to boot up an OS without one
  + Bootloader: instruction for the computer to pick the partition to boot up
  + Disk signature: verifies the integrity of the drive
  + Partition table: layout where each partition is located on the drive
  + MBR Signature: verifies the integrity of the master boot record
  + 
* File systems
  + NTFS
    - New Technology File System
    - Used by Microsoft
    - Virtually no file size limit
  + FAT
    - File allocation table
    - For simple structures
    - Organized using a table
    - Different variant of FAT has different table sizes, which allows for different cluster counts
  + exFAT
    - Extensible file allocation table
    - Optimized for flash memory
    - Allows for large files
    - Semi-universally-compatible
  + Ext4
    - Extended filesystem 4
    - Used in Linux
    - Journal based
    - File size limit 16TiB
    - Up to 4 billion files
  + XFS
    - Extents file system
    - 64-bit system
    - Supports large files
    - Fast
    - Used in Linux
  + HFS+
    - Hierarchical file system plus
    - Used by Apple
    - Unicode file names
    - Max volume size of 8EB
    - Max file size of 8EB
    - Has file compression

Shell Script Programming

* Sequences multiple commands in one run
* Interpreted one line at a time, until it hits a fatal error that causes execution to no longer be possible
* Bash scripts have the file extension \*.sh
* To run
  + bash <filename>.sh to run without execution permission
  + ./<filename>.sh to run with execution permission
* Start file with #!/bin/sh
  + Tells computer which interpreter to use
* #comments start with a #
* Variables
  + To define from source: variableName=”variableValue”
  + To define from user input: read <variableName>
  + To use the value of a variable: ${variableName}
  + To store output from a command: variableName=”$(command)”
    - Executed in a subshell. Subshell inherits stuff from parent, but not the other way around
* $1 gets the first command line argument, $2 gets the second, etc
* Put a string between ‘ ‘ to make it a literal (don’t parse the $ and stuff in it)
* Control flow
  + For loop
    - <condition>
    - do
    - code
    - done
    - For going through a range
    - for x in {start..end..step}
      * Step is 1 by default
      * Use $x to refer to x
    - for x in “a b c”
      * Loops through the characters
  + While
    - while [ condition ]
    - do
    - #something
    - done
  + Until
    - until [ condition ]
    - do
    - #something
    - done
  + If
    - if [ condition ]
    - then
    - #do something
    - elif [ condition ]
    - then
    - #do something
    - else
    - #do something
    - fi
    - If statements can be nested
    - if [ condition ]
    - then
    - #do something
    - if [ condition ]
    - then
    - #do something
    - fi
    - fi
    - And works like this:
    - if [ condition ] && [ condition ]
    - Or works like this:
    - if [ condition ] || [ condition ]
* Operators
  + -eq: equal (for numbers)
  + -le: less than or equal to
  + -ge: greater than or equal to
  + -lt: less than
  + -gt: greater than
  + =: equal (for strings)
* Arithmetic
  + let <expression>
    - Arithmetic that updates a variable
    - Ex: let a++
    - Ex: let a=5+99\*2
  + expr <expression>
    - Arithmetic that simply prints out a result
    - Ex: expr 8+5-6
  + To plug in the value of an arithmetic expression somewhere, put the expression inside: $(())
    - Ex: $(( 1+1 ))
  + To get the length of a variable, use ${#variableName}
* When a program finishes, it returns a status code that describes what has happened
  + A status code of 0 usually means success in execution
  + A status code of 1 usually means failure in execution
  + $? Refers to the status code of the last program that ran
* Functions
  + function functionName {
  + code
  + }
  + To call, do
  + functionName
  + Arguments work like running scripts from shell: $1 for the first, $2 for second, etc
* Arrays
  + To define:
    - arrayName=()
  + To append:
    - arrayName+=(toAdd)
  + Referring to element at index
    - ${arrayName[index]}
    - Using @ for index returns everything
    - Using @ for index and adding ! before the array’s name returns the element index numbers
    - ${arrayName[@]:s:n} returns elements from index s to index n
    - Zero-indexed
  + Elements delimited with a space
  + arrayName=( $(command) ) stores the contents of running command into arrayName

Servers

* tar: tape archive / tarball: used for backing up stuff
* gz: gzip: a compression algorithm based on DEFLATE
* Transferring files
  + scp: secure copy protocol
    - Faster than s/ftp
    - Cannot perform file operations
  + ftp: file transfer protocol
    - Can perform file operations
    - Not encrypted
  + sftp: secure file transfer protocol
    - Encrypted version of ftp
    - sftp <username>@<hostname>
    - put: takes a file from local machine and send to remote machine
    - get: takes a file from remote machine and send to local machine
    - File operations work
      * Prefix with l to refer to local
  + rsync
    - Remote sync
    - Syncs changes between a remote directory and local directory

Virtual machine

* Cloud
  + Another company supplies the hardware and sets up multiple virtual machines and lets you connect to and use one
  + Cost per hour: cost of hardware / lifetime of hardware in hours
  + Scaling: buy the computing power you need when you need it
* On-premise
  + You supply the hardware and set up the virtual machine
* Each virtual machine is isolated from the others

Web Server

* Needs to
  + Run in background
  + Run on boot up
  + Handle and survive errors
  + Create logs
  + Handle multiple requests simultaneously
  + Run as a service
  + Independent of keyboard input
  + Create children to handle work
* Server communicates with clients
* Static: everyone gets the same standard html files
* Dynamic: the html file you get is customized before sent
* Stack
  + OS
  + DB
  + Server and client side programming
  + server

Cybersecurity

* Hash
  + A checksum calculated based on an input
  + Changing one bit of the input would result in a completely different hash
  + One way operation
  + Can be used to verify that your download has not been poisoned
  + Also used for storing and checking passwords
    - User sets password
    - Server stores hashed password (plain text is security hole)
    - User logs in
    - User’s input gets converted to hash
    - Hashes are compared
  + Rainbow file: maps inputs to corresponding hashes, for cracking hashes stored on a server
  + Salt: random contents added to the input so that the resulting hash when checked against a rainbow file will still result in something random
* Hex editor
  + 
  + Hex editors allow you to read the binary contents of a file in hexadecimal form
  + The first column are offsets
    - Example: line 2 says 18 (hex). That means that 79 (hex) is in position 18 (hex) of the file
  + The middle part are the hex numbers
    - Example: 61 is hex for a
  + The last part is the ascii representation of the hex numbers
  + File extensions don’t mean anything. A few hex digits of each file is used for a file signature. Computers use that signature to figure out the file type
* SQL Injection
  + Attempting any of these techniques on another website that is not wholly owned by you is considered a cyber crime and very likely to fall under federal jurisdiction.
  + Error based: the server displays database errors to the user, which an attacker can use to learn key information about the stack
  + Union based: append the data that the attacker wants to a table that is already displayed in the page
  + Blind: brute force