Week 1

1. Operating System

Software that manages hardware and helps other program to communicate with the hardware

2. Open Source Software

Source code is publicly available
Modification by any individual allowed on global scale
It is free for use

3. User Interfaces

Command Line Interface (CLI)
Graphical User Interface (GUI)

4. Kernel

Core of OS
Allocates time and memory to programs
Allows communication between different processes

5. Shell

Interface between the user and kernel
A shell is command-line interpreter

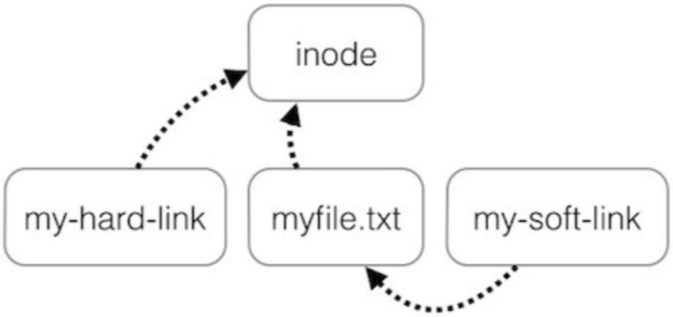
Basic Shell Commands:

```
!!     - replace with previous command
![str] - refer to previous command with str
```

6. Everything is a file

Everything is either a process or a file
Process - a executing program identified by PID
File - collection of data

7. Links


inode
my-hard-link
myfile.txt
my-soft-link

Week 2

1. POSIX built-in shell variables

    # - Number of arguments given to current process
    ? - Exit status of previous command
    IFS - Internal Field Separator. a variable which defines the character or characters used to separate a pattern into tokens for some operations
    ${1} - first argument
    $@ - an array with all arguments passed

2. Exit status

    0 : The command exited successfully
    1-125 : The command exited unsuccessfully
    127 : command not found

3. Quotes

    ‘’ - literal meaning, do not expand
    “” - Expand only backticks and $
    `` - Expand as shell commands

4. Redirection

    program < file_in : redirects file_in to stdin
    program > file_out : redirects file_out to stdout
    Program 2> file_err : redirects stderr to file_err
    Program >> file_out : appends stdout to file_out
    Program 1 | program 2 : redirects stdout from program 1 to stdin of program 2

5. BRE vs ERE

    Basic Regular Expression : ‘?’, ‘+’, ‘{’, ‘}’, ‘(’, ‘)’ lose their special meanings
    Extended Regular Expression

Week 6

1. Processor Modes

Supervisor (Kernel) mode : processor executes every instruction in it's hardware Repertoire

When CPU is in kernel mode, the code being executed can access any memory address and any hardware resource.

User mode : can use subsets of instructions

The program don't have direct access to memory and hardware resources.

E.g. I/O instructions are protected. If an application needs to do I/O, it needs to get the OS to do it on its behalf

Mode Bit : used to distinguish between execution on behalf of OS & behalf of user
Define areas of memory to be used when the processor is in supervisor mode vs user mode.

2. System Call

System call interface is a safe way to expose privileged functionality and services of the processor. The kernel executes privileged operations on behalf of untrusted user Processes.

3. Library Functions

Many library functions invoke system calls indirectly. Usually equivalent library functions make fewer system calls. Non-frequent switches from user mode to kernel mode (i.e. Less Overhead)

4. Unbuffered vs Buffered I/O

Buffered I/O output improves I/O performance and can reduce system calls.

Unbuffered output when you want to ensure that the output has been written before continuing.

Week 7

1. Thread

- smallest unit that can be scheduled to run on the CPU.
- Normally, each program has only one thread: it can only be executed on one CPU at any time.
- By using certain libraries, we can create more than one threads for the program.
- This allows the program to occupy more than one CPUs and thus to boost its performance.

- All the threads share global variables, static variables in the function, dynamically allocated memory address. (share heap and static data)
- Each thread has its own copy of local variables (do not share the stack)

2. Multithreading vs Multitasking

Multithreading

- An error in one thread can bring down all threads in process
- Light-weight creation/destruction
- Easy inter-thread communication

Multitasking

- An error in one process cannot bring down another process
- Expensive inter-process communication (IPC)
- Expensive creation/destruction

3. Thread Synchronization

Race condition - the output depends on the order of execution

Critical section - a section of code that can only be executed by one thread at a time, if the program is to function correctly.

Week 8

1. Static Library

- Statically Linked
- Every program has its own copy
- More space in memory
- Tied to a specific version of the lib. New version of the lib requires recompile of source
- code

2. Shared Library

Dynamic Linking - The OS loads the library when needed. A dynamic linker does the linking for the symbol used.

Dynamic load - The program actively loads the library it needs. More control to the program at runtime.

- Library is shared by multiple programs
- Lower memory footprint
- New version of the lib doesn’t require a recompile of source code

3. Static Linking

- Carried out only once to produce an executable file
- If static libraries are called, the linker will copy all modules referenced by the program to the executable
- Static libraries are typically denoted by the .a file extension

4. Dynamic Linking

- Allow a process to add, remove, replace or relocate object modules during its execution

- Complete the linking during the loading or running time.
- Dynamic libraries are typically denoted by the .so file extension.

Week 9

1. Symmetric Key

- Encrypt and Decrypt with same key
- How to let the other side know the key safely?

2. Asymmetric Cryptography

- Use one key to encrypt, use different key to decrypt.

3. SSH (Secure Shell)

- Run processes remotely
- Encrypted session
- Session key used for encryption during the session

4. Session Encryption

- Client and server agrees on a symmetric encryption key (session key)
- Use asymmetric cryptography to exchange this session key
    - Client informs the server for connection
    - Server generates a new asymmetric key pairs, send one key (public key) to the user
    - Client chooses a random session key K, encrypt it with server's public key.
    - Client sends the encrypted key K to server
    - Server decrypt the message with the other key (private key) and got the key K
    - Server and client can encrypt and decrypt message with K

5. User Authentication

- Password-based authentication
    - Prompt for password on remote server
    - If username specified exists and remote password for it is correct, then the system lets you in
- Key-based authentication
    - Generate a key pair on client
    - Copy the public key to the server
    - Server authenticates client if it can demonstrate that is has the private key
    - The private key can be prompted with passphrase
- Key-based authentication example
    - Alice generates her public/private key pair and copy the public key to the server

    - Every time Alice tries to log into the server, server generates a random number N, encrypts it with Alice's public key and sends back to Alice
    - Alice decrypts the message with her private key, send back the N to server
    - Server sees Alice sends back N, knows it is Alice

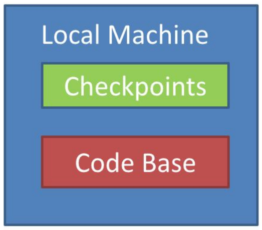
6. Digital Signature

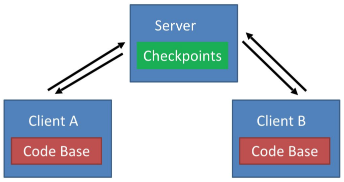
- Digital signature is extra data attached to the document (or separately) that can be used to check tampering
- Message Digest
    - Shorter version of the document
    - Generated by hash algorithm
    - Even a slight change in the original document will change the message digest with high probability
- Sender
    - Generate a message digest: Compute hash value of the document
    - Create a digital signature: Encrypt the message digest with private key
    - Send the document along with the signature of the patch
- Receiver
    - Recover the message digest: Decrypt the digital signature with the sender's public key
    - Generate the message digest: Use the same message digest algorithm used by the sender to generate a message digest of the received message
    - Compare digests

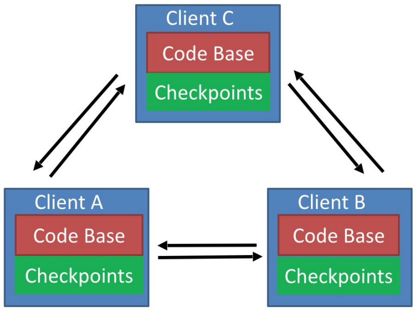
Week 10

1. Version Control Software (VCS)

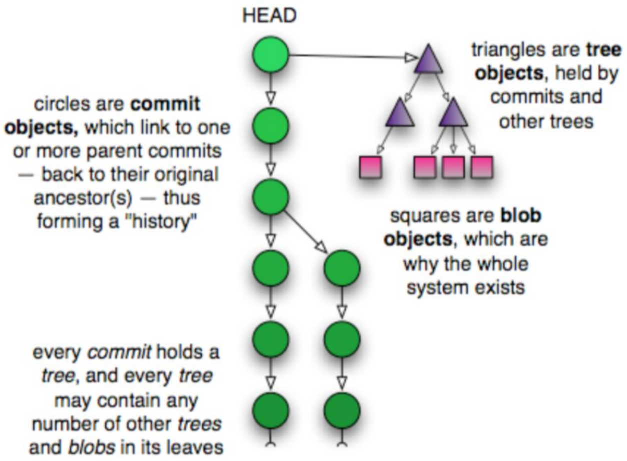
- Local Source Control System
    - Stores your checkpoint on the local disk
    - Hard to collaborate with other people
- Centralized Source Control System
    - Stores the checkpoint on remote server
    - Cannot use VCS if no access to remote server
    - Remote server under heavy pressure if number of users is large
- Distributed Source Control System
    - Users have version control all the time



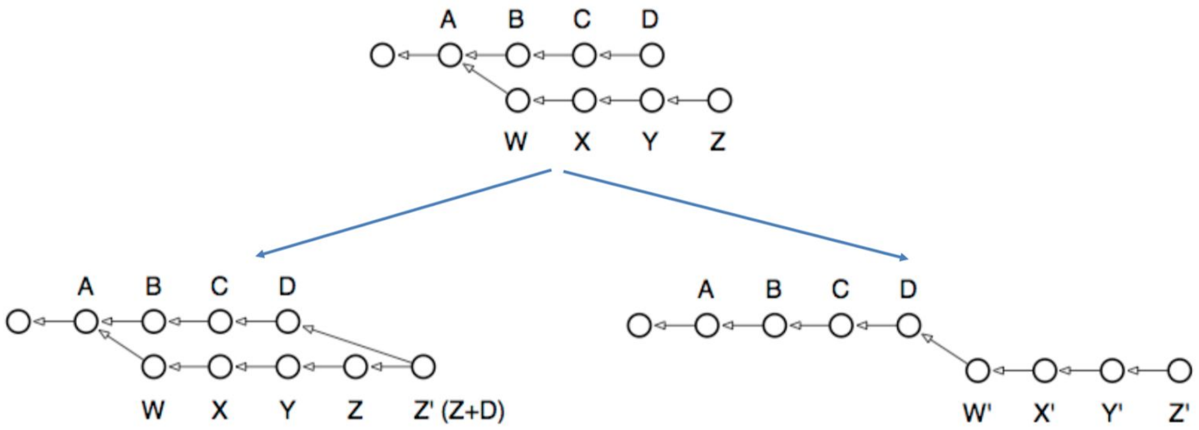


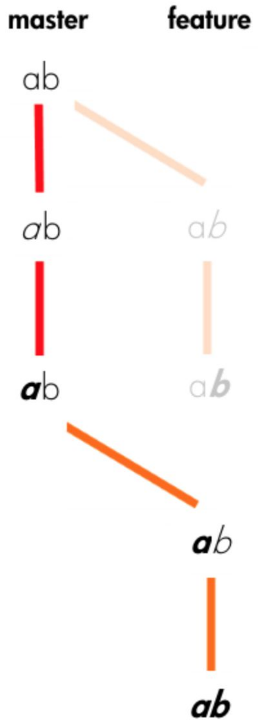


2. Git Objects


HEAD
triangles are **tree objects**, held by commits and other trees
circles are **commit objects,** which link to one or more parent commits — back to their original ancestor(s) — thus forming a "history"
squares are **blob objects**, which are why the whole system exists
every *commit* holds a *tree*, and every *tree* may contain any number of other *trees* and *blobs* in its leaves

3. Merge vs Rebase


A
B
C
D
W
X
Y
Z
A
B
C
D
W
X
Y
Z
Z' (Z+D)
A
B
C
D
W'
X'
Y'
Z'


master
feature
ab
ab
ab
ab
ab
ab
ab