CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Fall 2025 L28

Final Review

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

Project Peer Reviews

- D5: Each student will need to view/evaluate by Dec 12.
  - Use spreadsheet provided
  - Detailed review of 2 assigned D3 project reports in Canvas
  - At least 14 project D4 videos/slides (7-8 research, 6-7 development)
  - Team members
- Please finish course survey (Available in Canvas) by ASAP, if not already done.

Final

- Final: Comprehensive but mostly from the second half. 2 Hours.
- Must have laptop with Respondus Lockdown browser installed and tested test quiz available
- Sec 001: Tu 12/16/2025, 2-4 PM
  - may not sit next to usual neighbors or fellow team members. May not leave the room without permission.
  - SDC: must take at SDC
- Sec 801: Wed 12/17/2022, Two hours, 12:10 AM-11:50 PM window (must start at 9:50 PM, those with accommodation must start earlier to finish by 11:50 PM)

Grading

- Project D1, D2, D3, D4, D5 (raw/adjusted)
- Participation (raw/adjusted)
- Final (raw/adjusted)
- Letter Grades
  - Default: Given on course website
    - ≥ 90 is an A, ≥ 88 is an A-, ≥86 is a B+, ≥80 is a B, ≥78 is a B-, ≥76 is a C+, ≥70 is a C, ≥60 is a D, and <60 is an F.
  - *may* cut lower

Study/Resources

- Terms, concepts, implementations, algorithms, problems
- Lecture slides
  - Also see Midterm Review Slides on website
  - Possible questions not limited to Review Slides
- Quizzes, assignments
- Textbook

HW6

- Discuss after the review.

Deadlock Prevention

- If any one of the conditions for deadlock (with reusable resources) is denied, deadlock is impossible.
- Restrain ways in which requests can be made
  - Mutual Exclusion - cannot deny (important)
  - Hold and Wait - guarantee that when a process requests a resource, it does not hold other resources.
  - No Preemption
    - If a process that is holding some resources requests another resource that cannot be immediately allocated to it, the process releases the resources currently being held.
  - Circular Wait
    - Impose a total ordering of all resource types.

Deadlock Avoidance

- Requires that the system has some additional apriori information available.
  - Simplest and most useful model requires that each process declare the maximum number of resources of each type that it may need.
- Computation of Safe State
  - When a process requests an available resource, system must decide if immediate allocation leaves the system in a safe state. Sequence <P1, P2, …Pn> is safe, if for each Pi, the resources that Pi can still request can be satisfied by currently available resources + resources held by Pj with j<i.
  - Safe state - no deadlocks, unsafe state - possibility of deadlocks
  - Avoidance - system will never reach unsafe state.

Example: 12 Tape drives available in the system

	Max need	Current need
P0	10	5
P1	4	2
P2	9	2

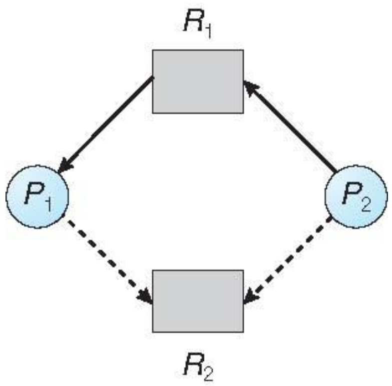
At T0:
3 drives available

Safe sequence
<P1, P0 , P2>

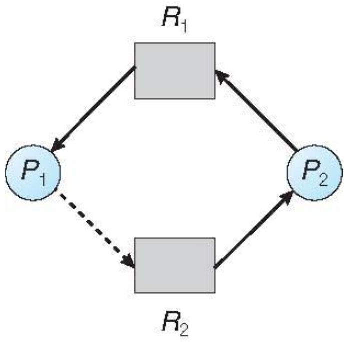
- At time **T0** the system is in a safe state because
  - P1 can be given 2 tape drives
  - When P1 releases its resources; there are 5 drives
  - P0 uses 5 and subsequently releases them (# 10 now)
  - P2 can then proceed.

Algorithms for Deadlock Avoidance

- Resource allocation graph algorithm
  - only one instance of each resource type



- Banker’s algorithm
  - Used for multiple instances of each resource type.
  - Data structures required
    - Available, Max, Allocation, Need
  - Safety algorithm
  - resource request algorithm for a process.

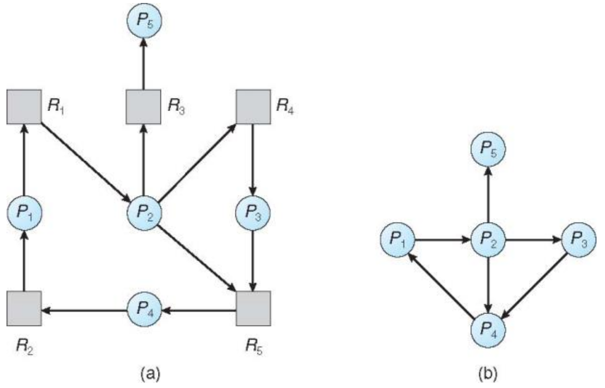


Unsafe state

Suppose *P2* requests *R2.* Although *R2* is currently free, we cannot allocate it to *P2,* since this action will create a cycle getting system is in an unsafe state. If *P1* requests *R2,* and *P2* requests *R1,* then a deadlock will occur.

Deadlock Detection

- Allow system to enter deadlock state
- Detection Algorithm
  - Single instance of each resource type
    - use wait-for graph
  - Multiple instances of each resource type
    - variation of banker's algorithm
- Recovery Scheme
  - Process Termination
  - Resource Preemption



Resource-Allocation Graph

Corresponding wait-for graph

Has cycles. Deadlock.

Binding of instructions and data to memory

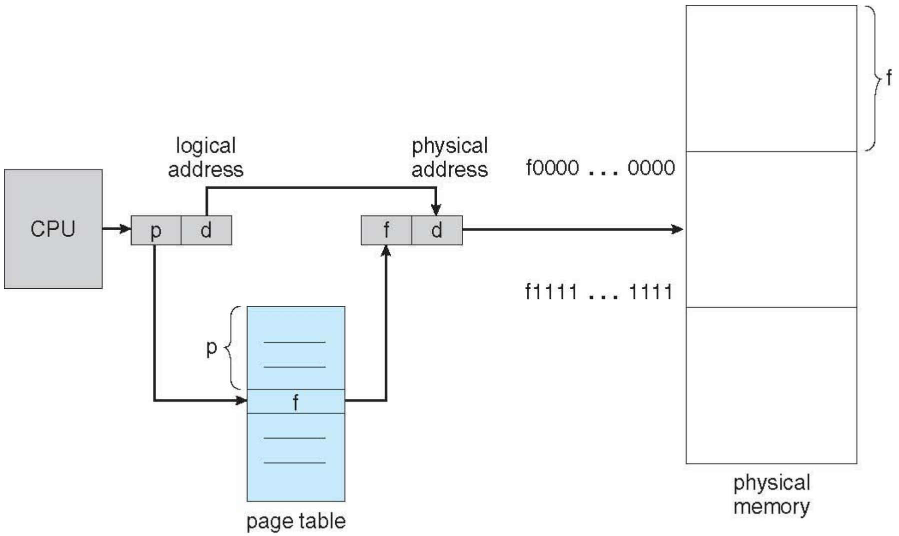
- Address binding of instructions and data to memory addresses can happen at three different stages.
  - Compile time, Load time, Execution time
- Other techniques for better memory utilization
  - Dynamic Loading - Routine is not loaded until it is called.
  - Dynamic Linking - Linking postponed until execution time
  - Swapping - A process can be swapped temporarily out of memory to a backing store and then brought back into memory for continued execution
- MMU - Memory Management Unit
  - Hardware device that maps virtual to physical address.

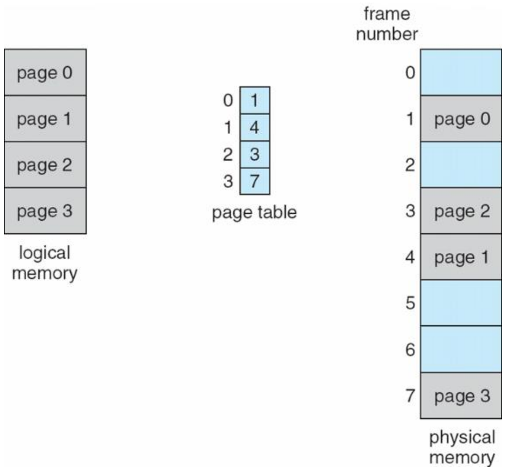
Dynamic Storage Allocation Problem

- How to satisfy a request of size n from a list of free holes.
    - First-fit
    - Best-fit
    - Worst-fit
- Fragmentation
    - External fragmentation
        - total memory space exists to satisfy a request, but it is not contiguous.
    - Internal fragmentation
        - allocated memory may be slightly larger than requested memory; this size difference is memory internal to a partition, but not being used.
    - Reduce external fragmentation by compaction

Page Table Implementation

- Page table is kept in main memory
    - Page-table base register (PTBR) points to the page table.
    - Page-table length register (PTLR) indicates the size of page table.
  - Every data/instruction access requires 2 memory accesses.
    - One for page table, one for data/instruction
    - Two-memory access problem solved by use of special fast-lookup hardware cache (i.e. cache page table in registers)
      - associative registers or translation look-aside buffers (TLBs)





Effective Access Time

Effective Access Time (**EAT**)

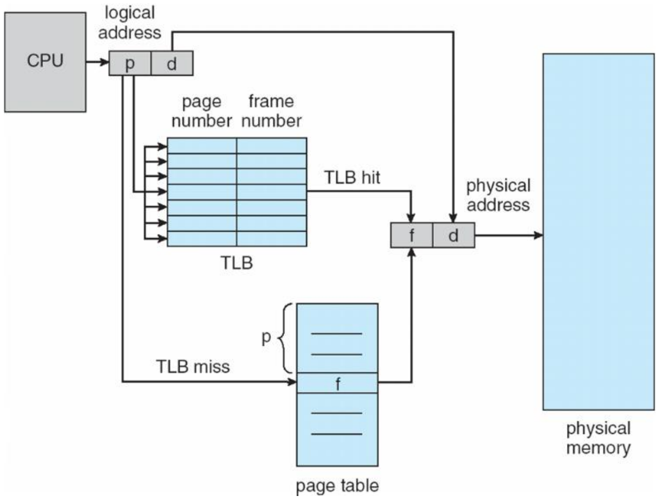
- Item in faster unit or in slower unit
- How often it is found in the faster unit?
  - Access time less if in the faster medium
  - Access time higher if in the slower medium

- Simplification: only two layers considered
- Approximation: some overhead may be ignored

Case 1: Need: page number to frame number mapping

- Faster unit: TLB
- Slower unit: full Page table in memory

Should you understand the process or memorize the formula?



Effective Access Time

- Hit ratio = α
  - Hit ratio – percentage of times that a page number is found in the TLB
- Associative Lookup = ε time unit
- Memory access time = 100 ns

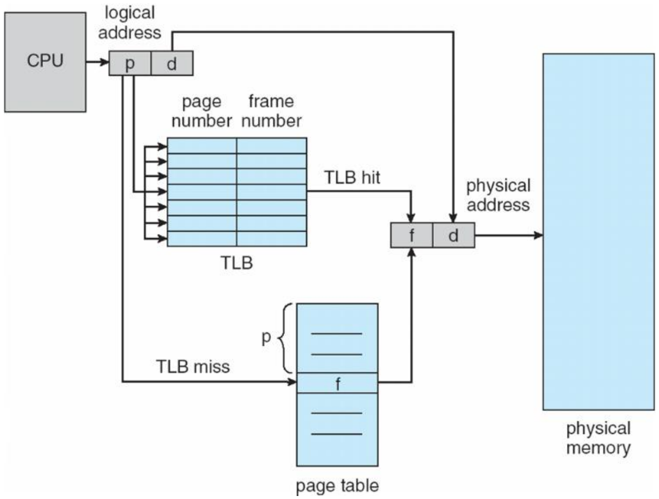
- **Effective Access Time** (**EAT**)

  EAT = $(100 + \varepsilon)\,\alpha + (200 + \varepsilon)(1 - \alpha)$

Consider α = 80%, ε = 20ns for TLB search, 100ns for memory access

- EAT = 120 x 0.80 + 220 x 0.20 = 140ns

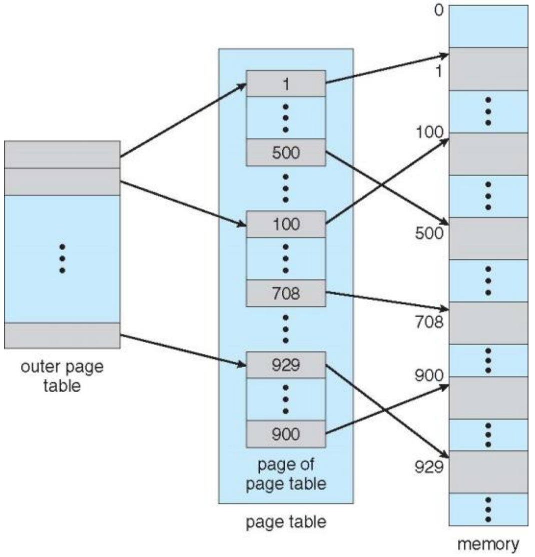
- Consider higher hit ratio -> α = 99%, ε = 20ns for TLB search, 100ns for memory access
  - EAT = 120 x 0.99 + 220 x 0.01 = 121ns

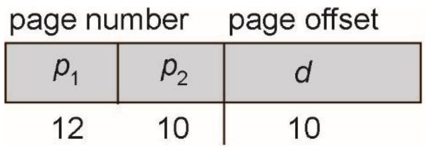


Colorado State University

Paging Methods

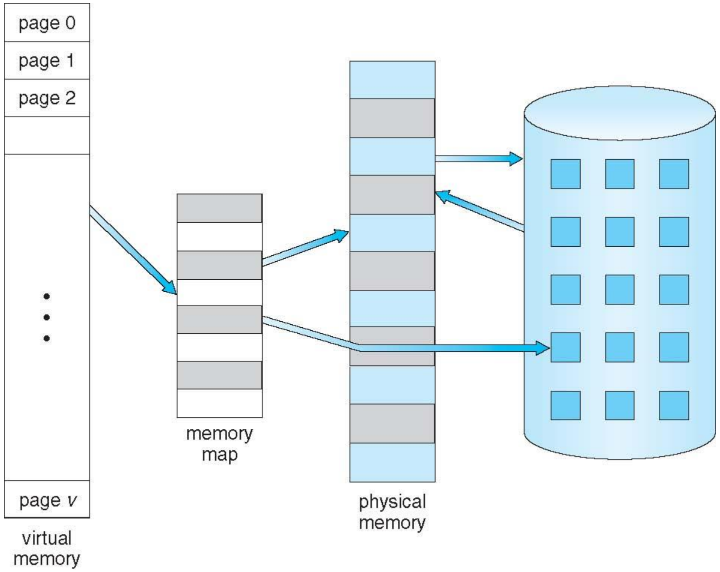
- Multilevel Paging
  - Each level is a separate table in memory
  - converting a logical address to a physical one may take 4 or more memory accesses.
  - Caching can help performance remain reasonable.
- Hashed page table
- Inverted Page Tables
  - One entry for each real page of memory. Entry consists of virtual address of page in real memory with information about process that owns page.





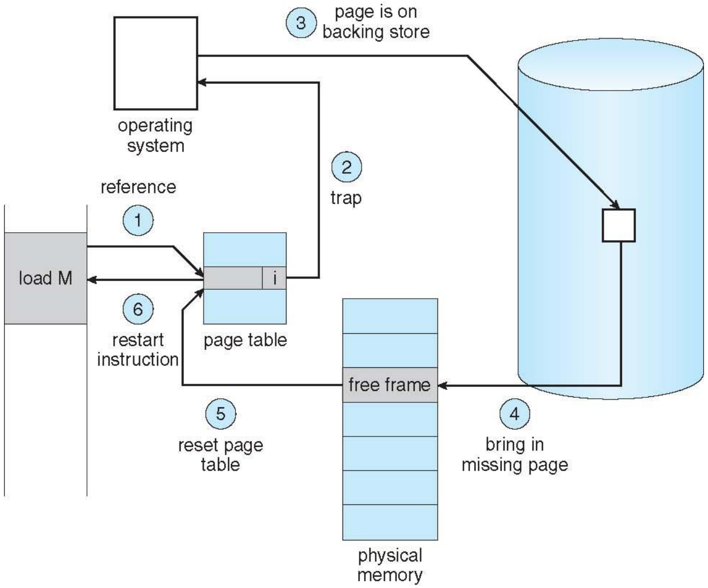
Virtual Memory

- Virtual Memory
  - Separation of user logical memory from physical memory.
  - Only *PART* of the program needs to be in memory for execution.
  - Logical address space can therefore be much larger than physical address space.
  - Need to allow pages to be swapped in and out.
- Virtual Memory can be implemented via
  - Paging
  - Segmentation



Demand Paging

- Bring a page into memory only when it is needed.
  - Less I/O needed
  - Less Memory needed
  - Faster response
  - More users
- The first reference to a page will trap to OS with a page fault.
- OS looks at another table to decide
  - Invalid reference - abort
  - Just not in memory.



Page fault:

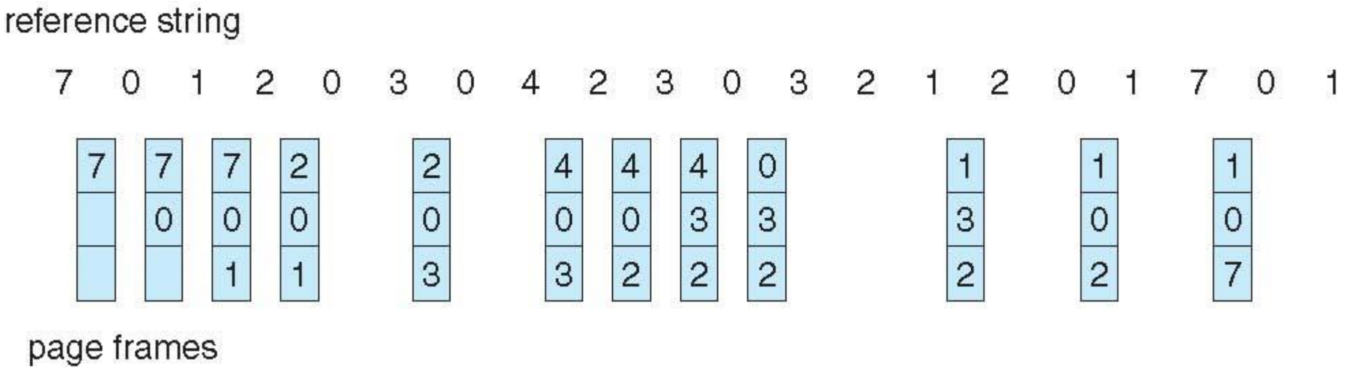
1. Find free frame
2. Get page into frame via scheduled disk operation
3. Reset tables to indicate page now in memory
   Set validation bit = **v**
4. Restart the instruction that caused the page fault

Page Replacement Strategies

- The Principle of Optimality
  - Replace the page that will not be used again the farthest time into the future.
- FIFO - First in First Out
  - Replace the page that has been in memory the longest.
- LRU - Least Recently Used
  - Replace the page that has not been used for the longest time.
  - LRU Approximation Algorithms - reference bit, second-chance etc.
- Working Set
  - Keep in memory those pages that the process is actively using

Least Recently Used (LRU) Algorithm

- Use past knowledge rather than future
- Replace page that has not been used in the most amount of time
- Associate time of last use with each page



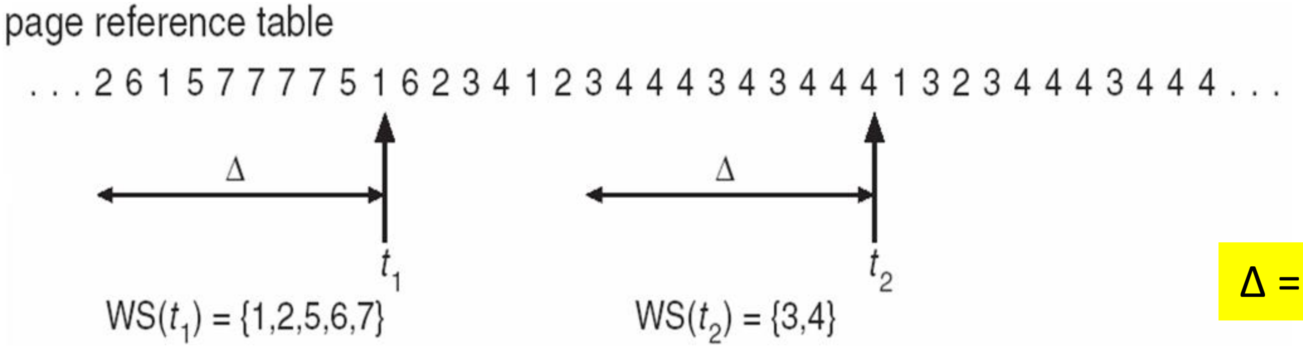
- Blank: implies a Hit with no page fault.
- 12 faults – better than FIFO but worse than OPT
- Generally good algorithm and frequently used
- Approximate Implementations:
  - Counter implementation time of use field
  - Stack implementation
  - Reference bit
  - Second chance

Allocation of Frames

- Single user case is simple
    - User is allocated any free frame
- Problem: Demand paging + multiprogramming
    - Each process needs minimum number of pages based on instruction set architecture.
    - Two major allocation schemes:
        - Fixed allocation - (1) equal allocation (2) Proportional allocation.
        - Priority allocation - May want to give high priority process more memory than low priority process.
    - Global vs local allocation

Working-Set Model

- $\Delta \equiv$ **working-set window** $\equiv$ a fixed number of page references
  Example: 10,000 instructions

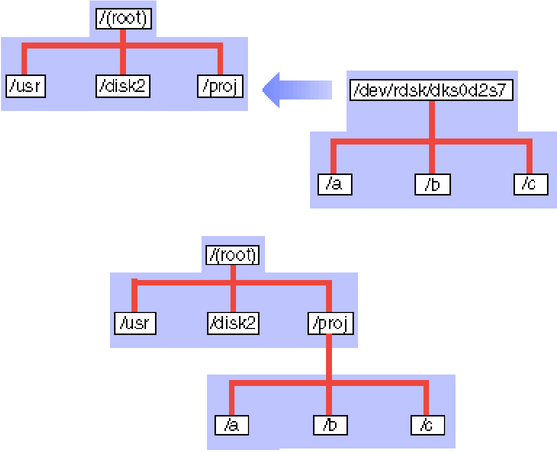


Δ = 10

- ***WSS$_i$* (working set of Process *P$_i$*)** =
  total number of pages referenced in the most recent Δ (varies in time)
  - if Δ too small will not encompass entire locality
  - if Δ too large will encompass several localities
  - if $\Delta = \infty \Rightarrow$ will encompass entire program
- ***D* = Σ *WSS$_i$* $\equiv$ total demand frames**
  - Approximation of locality
- **if *D* > *m* $\Rightarrow$ Thrashing**
- **Policy** if *D* > m, then suspend or swap out one of the processes

File-System Implementation

- File System Structure
  - File System resides on secondary storage (disks).
  - To improve I/O efficiency, I/O transfers between memory and disk are performed in blocks. Read/Write/Modify/Access each block on disk.
  - **File System Mounting** - File System must be mounted before it can be available to process on the system. The OS is given the name of the device and the mount point.
- Allocation Methods
- Free-Space Management
- Directory Implementation
- Efficiency and Performance, Recovery



Colorado State University

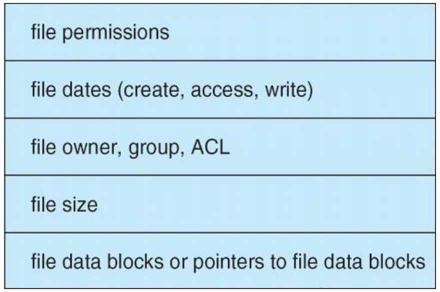
File Systems

- Many file systems, sometimes several within an operating system
  - Each with its own format
    - Windows has FAT (1977), FAT32 (1996), NTFS (1993)
    - Linux has more than 40 types, with **extended file system** (1992) ext2 (1993), ext3 (2001), ext4 (2008);
    - plus distributed file systems
    - floppy, CD, DVD Blu-ray
  - New ones still arriving –GoogleFS, xFAT, HDFS

On-disk File-System Structures

1. **Boot control block** contains info needed by system to boot OS from that volume
   - Needed if volume contains OS, usually first block of volume
2. **Volume control block** (**superblock UFS or master file tableNTFS**) contains volume details
   - Total # of blocks, # of free blocks, block size, free block pointers or array
3. Directory structure organizes the files
   - File Names and inode numbers UFS, master file table NTFS
4. Per-file **File Control Block** (**FCB or "inode"**) contains many details about the file
   - Indexed using inode number; permissions, size, dates UFS
   - master file table using relational DB structures NTFS

Volume: logical disk drive, perhaps a partition



file permissions

file dates (create, access, write)

file owner, group, ACL

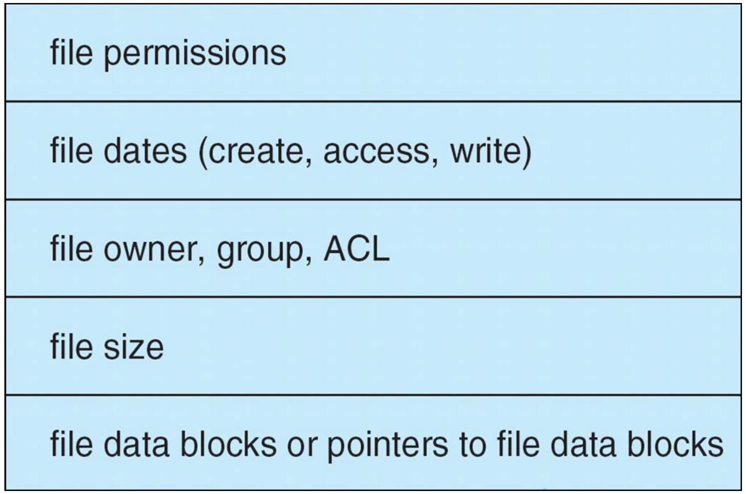
file size

file data blocks or pointers to file data blocks

File-System Implementation (Cont.)

4. Per-file **File Control Block** (**FCB or “inode”**) contains many details about the file

- Indexed using inode number; permissions, size, dates UFS



file permissions

file dates (create, access, write)

file owner, group, ACL

file size

file data blocks or pointers to file data blocks

In-Memory File System Structures

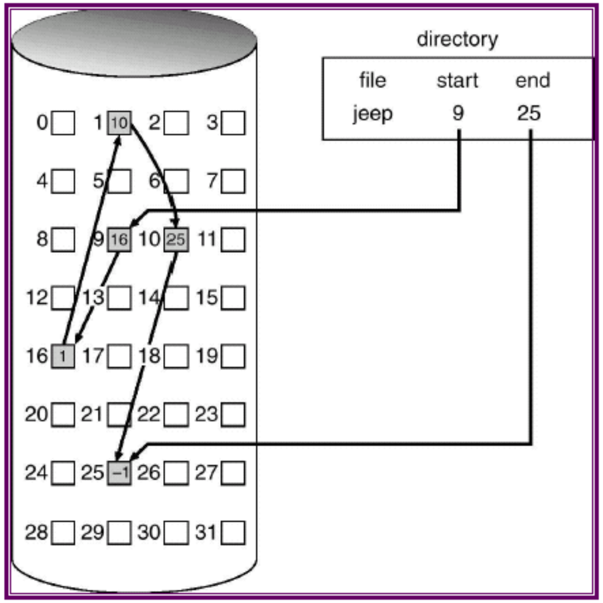
- An in-memory **mount table** contains information about each mounted volume.
- An in-memory directory-structure cache holds the directory information of recently accessed directories.
- The system-wide open-file table contains a copy of the FCB of each open file, as well as other information.
- The per-process open file table contains a pointer to the appropriate entry in the system-wide open-file table
- Plus buffers hold data blocks from secondary storage

Open returns a file handle (file descriptor) for subsequent use

- Data from read eventually copied to specified user process memory address

Allocation of Disk Space

- Low level access methods depend upon the disk allocation scheme used to store file data
  - Contiguous Allocation
    - Each file occupies a set of contiguous blocks on the disk. Dynamic storage allocation problem. Files cannot grow.
  - Linked List Allocation
    - Each file is a linked list of disk blocks. Blocks may be scattered anywhere on the disk. Not suited for random access.
    - Variation - FILE ALLOCATION TABLE (FAT) mechanisms
  - Indexed Allocation
    - Brings all pointers together into the index block. Need index table. Can link blocks of indexes to form multilevel indexes.

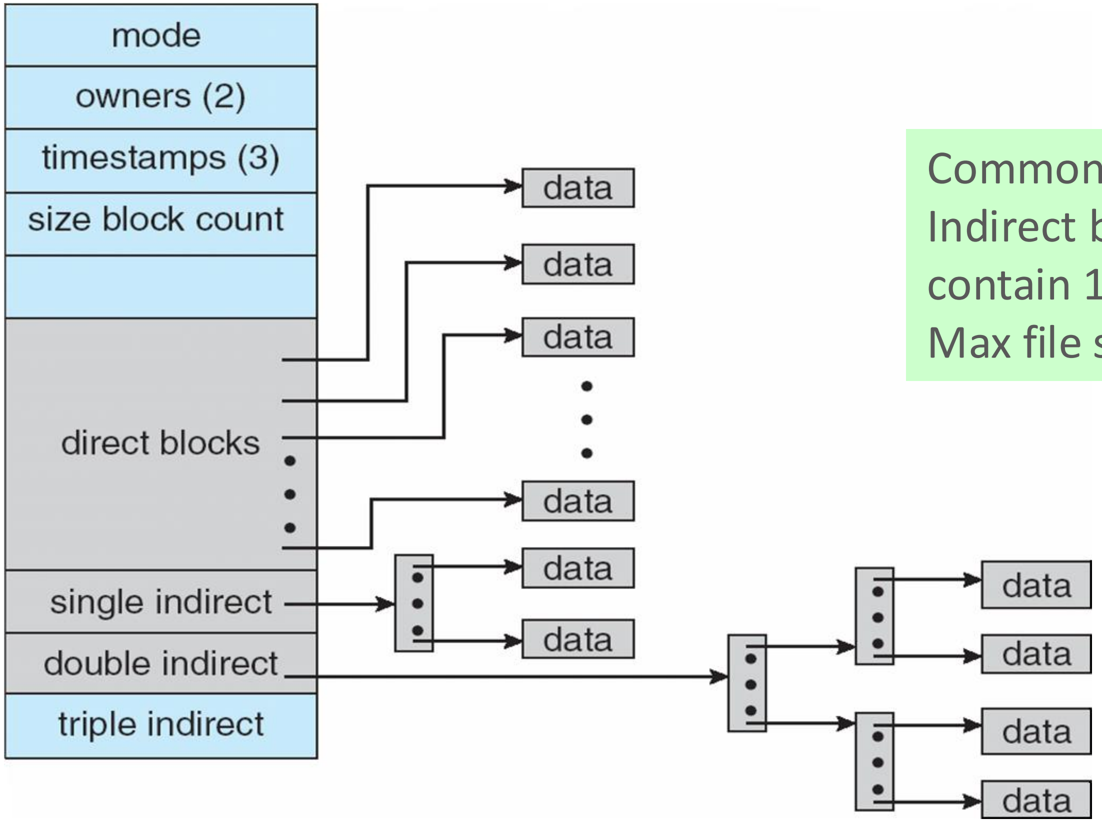


Combined Scheme: UNIX UFS

4K bytes per block, 32-bit addresses

Volume block:
Table with file names
Points to this

Inode (file control block)

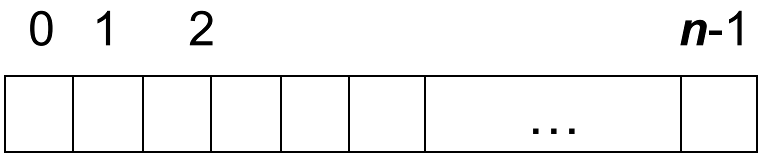


Common: 12+3
Indirect block could contain 1024 pointers.
Max file size: k.k.k.4k+

More index blocks than can be addressed with 32-bit file pointer

Free-Space Management

- File system maintains **free-space list** to track available blocks/clusters
  - (Using term "block" for simplicity)
- **Approaches: i. Bit vector ii. Linked list iii. Grouping iv. Counting**
- **Bit vector** or **bit map** (***n*** blocks)



$$\text{bit}[\boldsymbol{i}] = \begin{cases} 1 \Rightarrow \text{block}[\boldsymbol{i}] \text{ free} \\ 0 \Rightarrow \text{block}[\boldsymbol{i}] \text{ occupied} \end{cases}$$

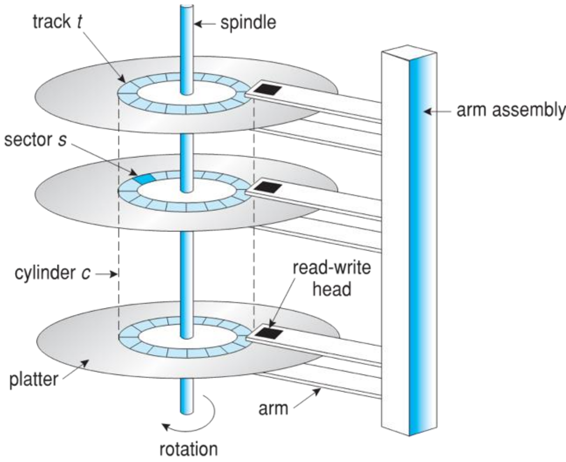
Block number calculation

(number of bits per word) *(number of 0-value words) + offset of first 1 bit

```
00000000
00000000
00111110
..
```

CPUs have instructions to return offset within word of first "1" bit

Hard Disk Performance



- Average I/O time = average access time + (amount to transfer / transfer rate) + controller overhead
- **Average access time** = average seek time + average latency
- Example: to transfer a 4KB block on a 7200 RPM disk with a 5ms average seek time, 1Gb/sec transfer rate with a .1ms controller overhead.
  - average latency = 0.5 x 1/(7200/60) = 0.00417 sec
  - Transfer time = 4KB / 1Gb/s = 4x8K/G = 0.031 ms
  - Average I/O time for 4KB block

    = 5ms + 4.17ms + transfer time + 0.1ms

    = 9.27ms + .031ms = 9.301ms

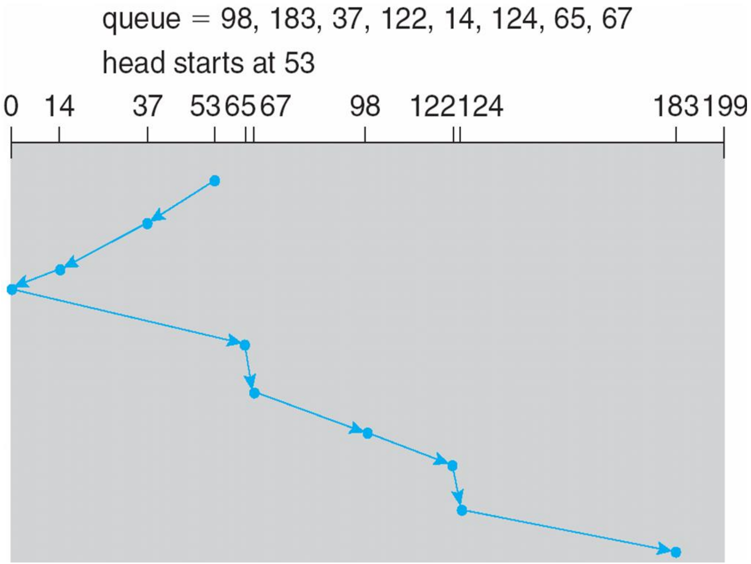
Colorado State University

Disk Scheduling

- Several algorithms to schedule the servicing of disk I/O requests
  - The analysis is true for one or many platters
  - SCAN, C-SCAN, C-LOOK,
- We illustrate scheduling algorithms with a request queue (cylinders 0-199) 98, 183, 37, 122, 14, 124, 65, 67

  Head pointer 53 (head is at cylinder 53)

Scan



queue = 98, 183, 37, 122, 14, 124, 65, 67

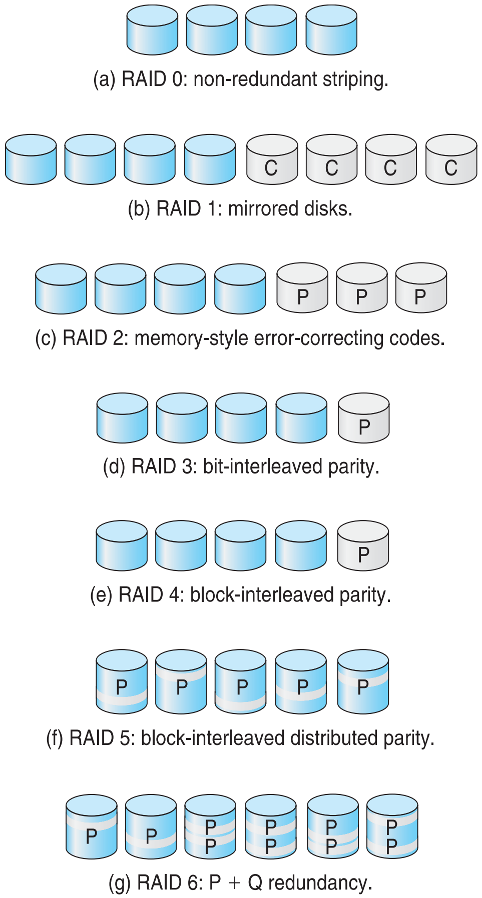
head starts at 53

RAID Techniques

- **Striping** uses multiple disks in parallel by splitting data: higher performance, no redundancy (ex. RAID 0)
- **Mirroring** keeps duplicate of each disk: higher reliability (ex. RAID 1)
- **Block parity: One Disk hold** parity block for other disks. A failed disk can be rebuilt using parity. Wear leveling if interleaved (RAID 5, double parity RAID 6).
- Ideas that did not work: Bit or byte level level striping (RAID 2, 3) Bit level Coding theory (RAID 2), dedicated parity disk (RAID 4).
- Nested Combinations:
  - RAID 01: Mirror RAID 0
  - RAID 10: Multiple RAID 1, striping
  - RAID 50: Multiple RAID 5, striping
  - others

Parity: allows rebuilding of a disk

Not common: RAID 2, 3,4
Most common: RAID 5



(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c) RAID 2: memory-style error-correcting codes.

(d) RAID 3: bit-interleaved parity.

(e) RAID 4: block-interleaved parity.

(f) RAID 5: block-interleaved distributed parity.

(g) RAID 6: P + Q redundancy.

Colorado State University

Parity

- Parity block: Block1 xor block2 xor block3 …

10001101	block1
01101100	block2
~~11000110~~	block3

00100111	parity block (*ensures even number of 1s*)

- Can reconstruct any bad block using all others

Read Errors and RAID recovery

- Example: RAID 5
  - 10 one-TB disks, and 1 fails
  - Read remaining disks to reconstruct missing data
- Probability of an error in reading 9 TB disks =

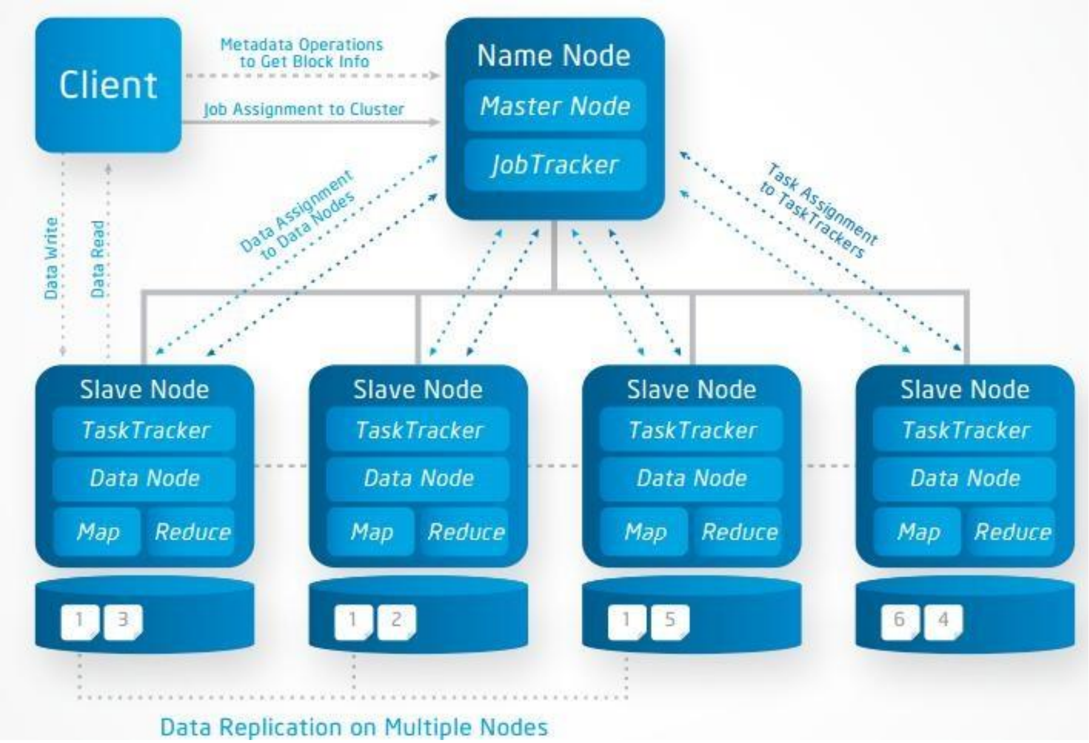
  10^{-15}*(9 disks * 8 bits * 10^{12} bytes/disk)

  = 7.2% Thus recovery probability = 92.8%
- Even better:
  - RAID-6: two redundant disk blocks
  - Can work even in presence of one bad disk
  - Scrubbing: read disk sectors in background to find and fix latent errors

Hadoop: Core components

- Hadoop (originally): MapReduce + HDFS
- For **Big Data** applications.
- **MapReduce**: A programming framework for processing parallelizable problems across huge datasets using a large number of commodity machines.
- **HDFS**: A **d**istributed **f**ile **s**ystem designed to efficiently allocate data across multiple machines, and provide self-healing functions when some of them go down

HDFS Architecture


Client
Metadata Operations to Get Block Info
Job Assignment to Cluster
Name Node
Master Node
JobTracker
Data Assignment to Data Nodes
Task Assignment to TaskTrackers
Data Write
Data Read
Slave Node
TaskTracker
Data Node
Map
Reduce
1
3
1
2
1
5
6
4
Data Replication on Multiple Nodes

HDFS Block size: 64-128 MB
ext4: 4KB
HDFS is on top of a local file system.

Name Node: metadata, where blocks are physically located
Data Nodes: hold blocks of files (files are distributed)

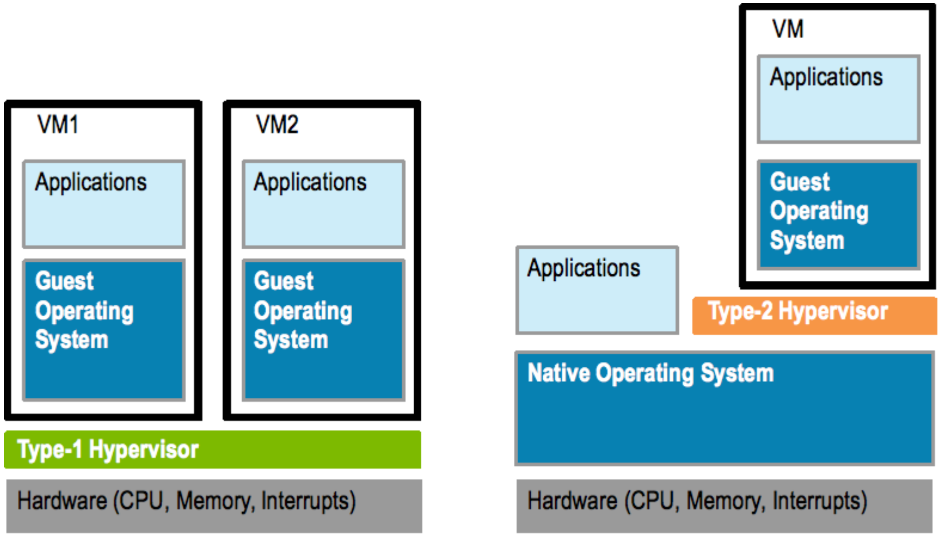
http://a4academics.com/images/hadoop/Hadoop-Architecture-Read-Write.jpg

HDFS Fault-tolerance

- Individual node/rack may fail.
  - Disks use error detecting codes to detect corruption.
- Data Nodes (on slave nodes):
  - data is replicated. Default is 3 times. Keep a copy far away.
  - Send periodic heartbeat (I'm OK) to Name Nodes. Perhaps once every 10 minutes.
  - Name node creates another copy if no heartbeat.
- Name Node (on master node) Protection:
  - Transaction log for file deletes/adds, etc (only metadata recorded).
  - Creation of more replica blocks when necessary after a DataNode failure
- Standby name node: namespace backup
  - In the event of a failover, the Standby will ensure that it has read all of the edits from the Journal Nodes and then promotes itself to the Active state

Implementation of VMMs

- **Type 1 hypervisors** - Operating-system-like software built to provide virtualization. Runs on ‘bare metal”.
    - Including VMware ESX, Joyent SmartOS, and Citrix XenServer
- Also includes general-purpose operating systems that provide standard functions as well as VMM functions
    - Including Microsoft Windows Server with HyperV and RedHat Linux with KVM
- **Type 2 hypervisors** - Applications that run on standard operating systems but provide VMM features to guest operating systems
    - Including VMware Workstation and Fusion, Parallels Desktop, and Oracle VirtualBox



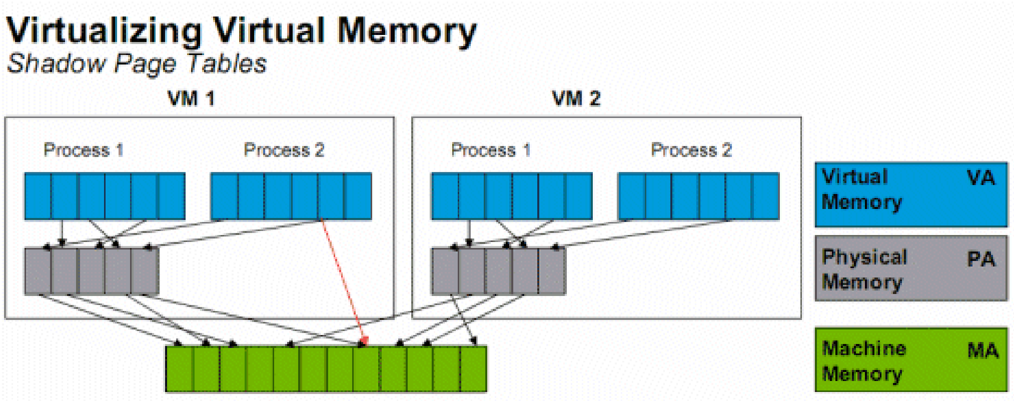
Memory Management

Memory mapping:

- On a bare metal machine:
  - VPN -> PPN
- VMM: Real physical memory (*machine memory*) is shared by the OSs. Need to map PPN of each VM to MPN (Shadow page table)

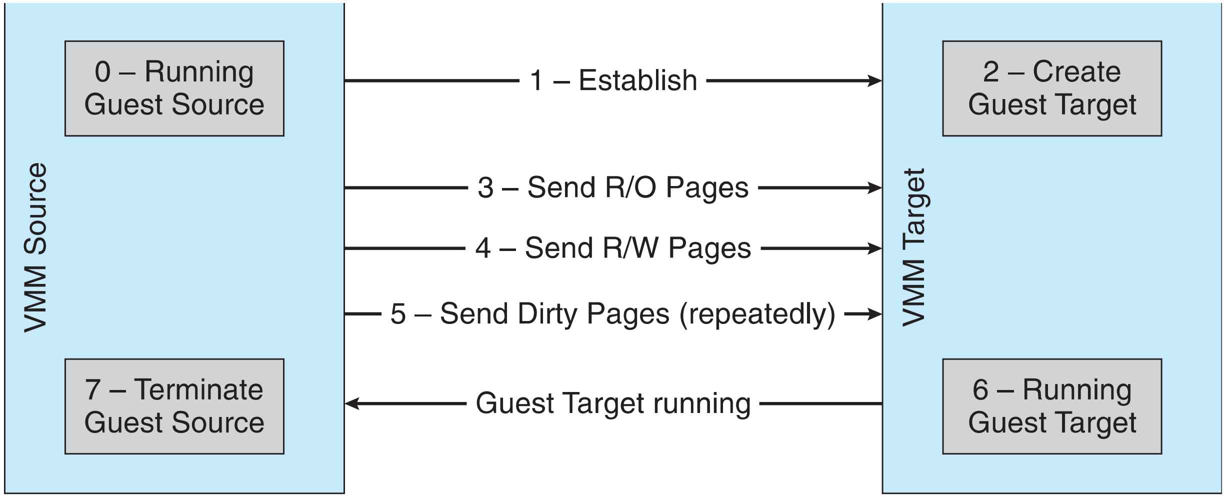
  PPN ->MPN

- Where is this done?
  - In Full virtualization?



Colorado State University

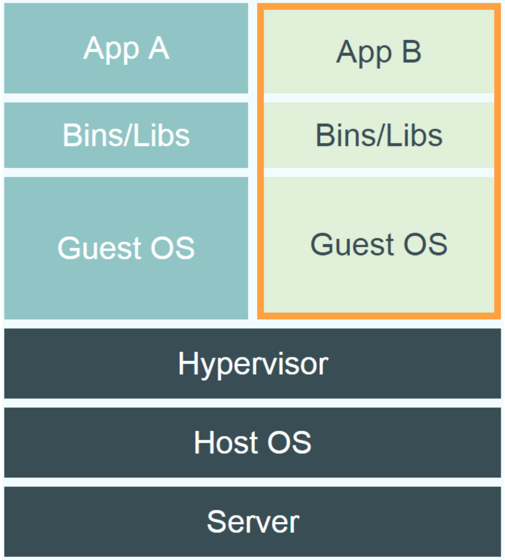
Live Migration

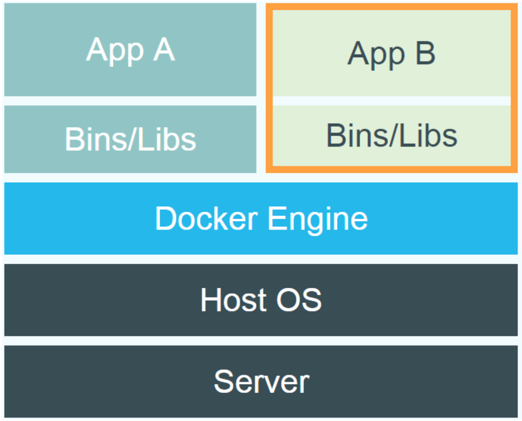


- Migration from source VMM to target VMM
  - Source establishes a connection with the target
  - Target creates a new guest
  - Source sends all read-only memory pages to target
  - Source starts sending all read-write pages
  - Source VMM freezes guest, sends final stuff,
  - Once target acknowledge

Linux Containers and Docker

- Linux containers (LXC) are “lightweight” VMs
- Comparison between LXC/docker and VM





- Containers provide “OS-level Virtualization” vs “hardware level”.
- Containers can be deployed in seconds.
- Very little overhead during execution, just like Type 1.

Colorado State University

Microservices Characteristics

- Many smaller (fine grained), clearly scoped services
  - Single Responsibility Principle
  - Independently Managed
- Clear ownership for each service
  - Typically need/adopt the "DevOps" model
- 100s of MicroServices
  - Need a Service Metadata Registry (Discovery Service)
- May be replicated as needed
- A microservice can be updated without interruption

Cloud Capacity provisioning

User has a variable need for capacity. User can choose among

Fixed resources: Private data center

- Under-provisioning when demand is too high, or
- Provisioning for peak

Variable resources:

- Use more or less depending on demand
- Public Cloud has elastic capacity (i.e. way more than what the user needs)
- User can get exactly the capacity from the Cloud that is actually needed

Why does this work for the provider?

- – Varying demand is statistically smoothed out over many users, their peaks may occur at different times
- – Prices set low for low overall demand periods

Cloud Instance types/Service/Management models

Instance types

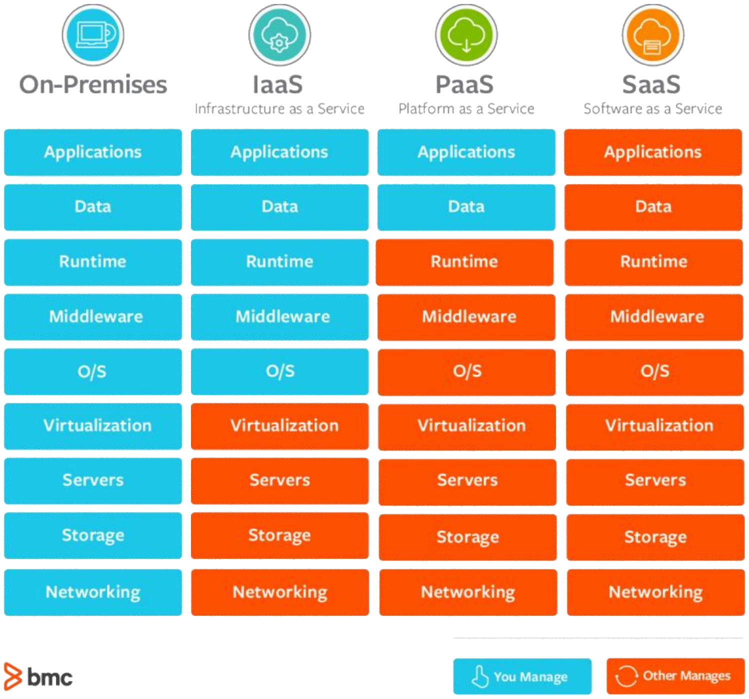
- On-Demand instances
- Spot Instances
- Reserved Instances
- Dedicated Hosts

Service models

- IaaS: Infrastructure as a Service
- PaaS: Platform as a Service
- SaaS: Software as a Service

Cloud Management models

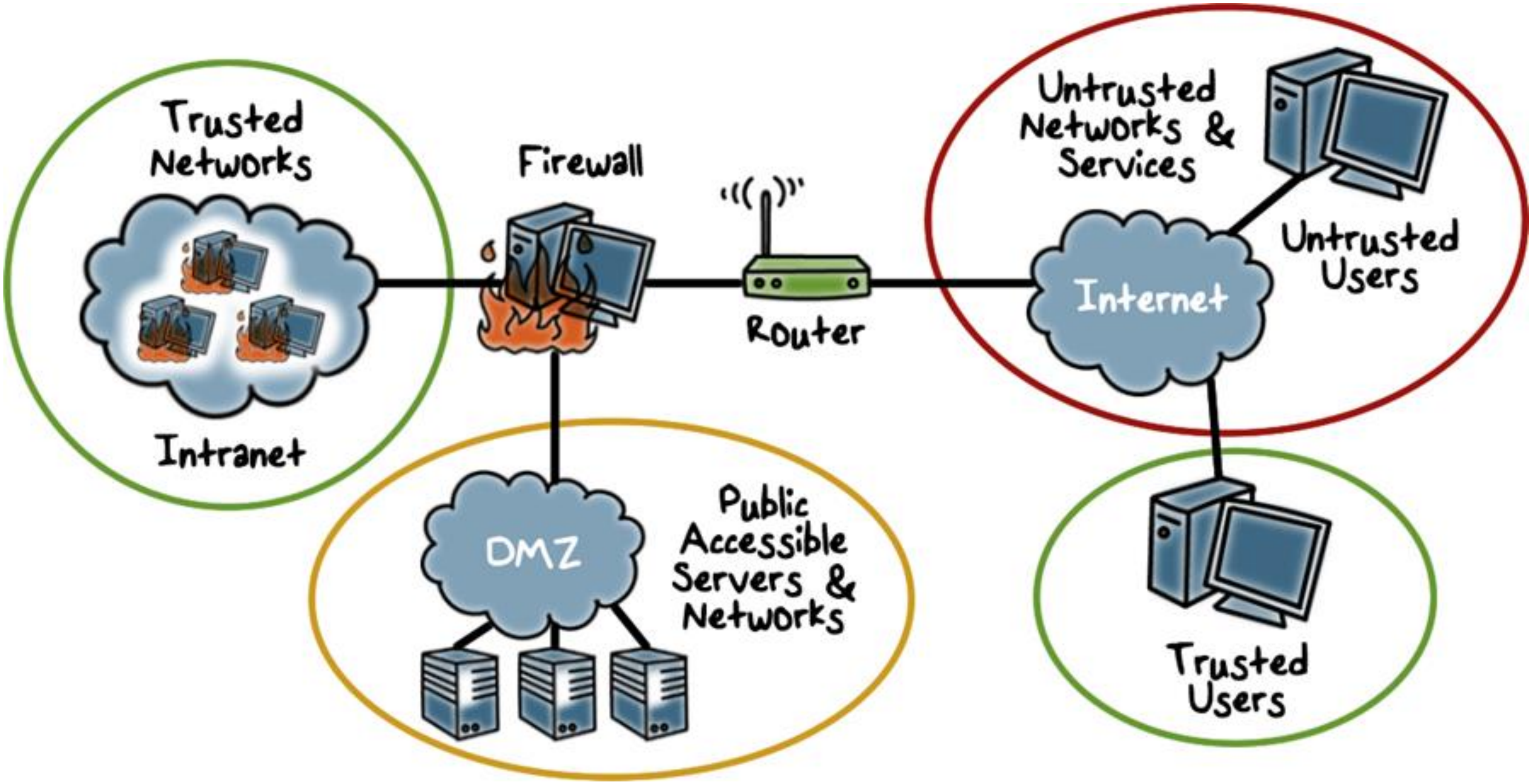
- Public clouds
- Private clouds
- Hybrid clouds:



On-Premises	IaaS Infrastructure as a Service	PaaS Platform as a Service	SaaS Software as a Service
🟦 Applications	🟦 Applications	🟦 Applications	🟧 Applications
🟦 Data	🟦 Data	🟦 Data	🟧 Data
🟦 Runtime	🟦 Runtime	🟧 Runtime	🟧 Runtime
🟦 Middleware	🟦 Middleware	🟧 Middleware	🟧 Middleware
🟦 O/S	🟦 O/S	🟧 O/S	🟧 O/S
🟦 Virtualization	🟧 Virtualization	🟧 Virtualization	🟧 Virtualization
🟦 Servers	🟧 Servers	🟧 Servers	🟧 Servers
🟦 Storage	🟧 Storage	🟧 Storage	🟧 Storage
🟦 Networking	🟧 Networking	🟧 Networking	🟧 Networking

🟦 You Manage 🟧 Other Manages

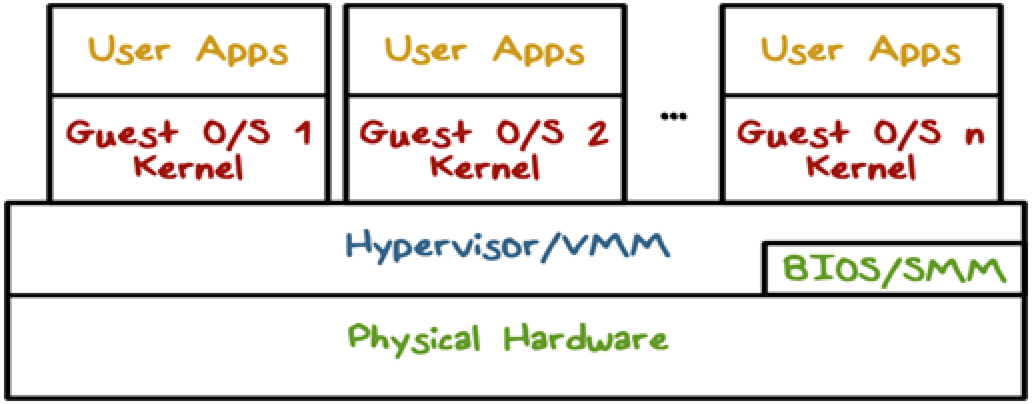
Firewalls



DMZ: “Demilitarized zone”, distributed firewalls, From Georgia Tech
Note multiple levels of trust.

Isolation in a system

- OS isolates address spaces of different processes using address translation. Also data vs code isolation.
  - Page tables governed by OS.
- In virtualization, hypervisor isolates virtual machines.
- Containers (Docker): Linux cgroups isolate process groups.



Colorado State University

Assets, Risk, Threat, Vulnerability

System Resource (Asset): what needs protection by the defenders.

Risk: A measure of the adverse impacts and the likelihood of occurrence.

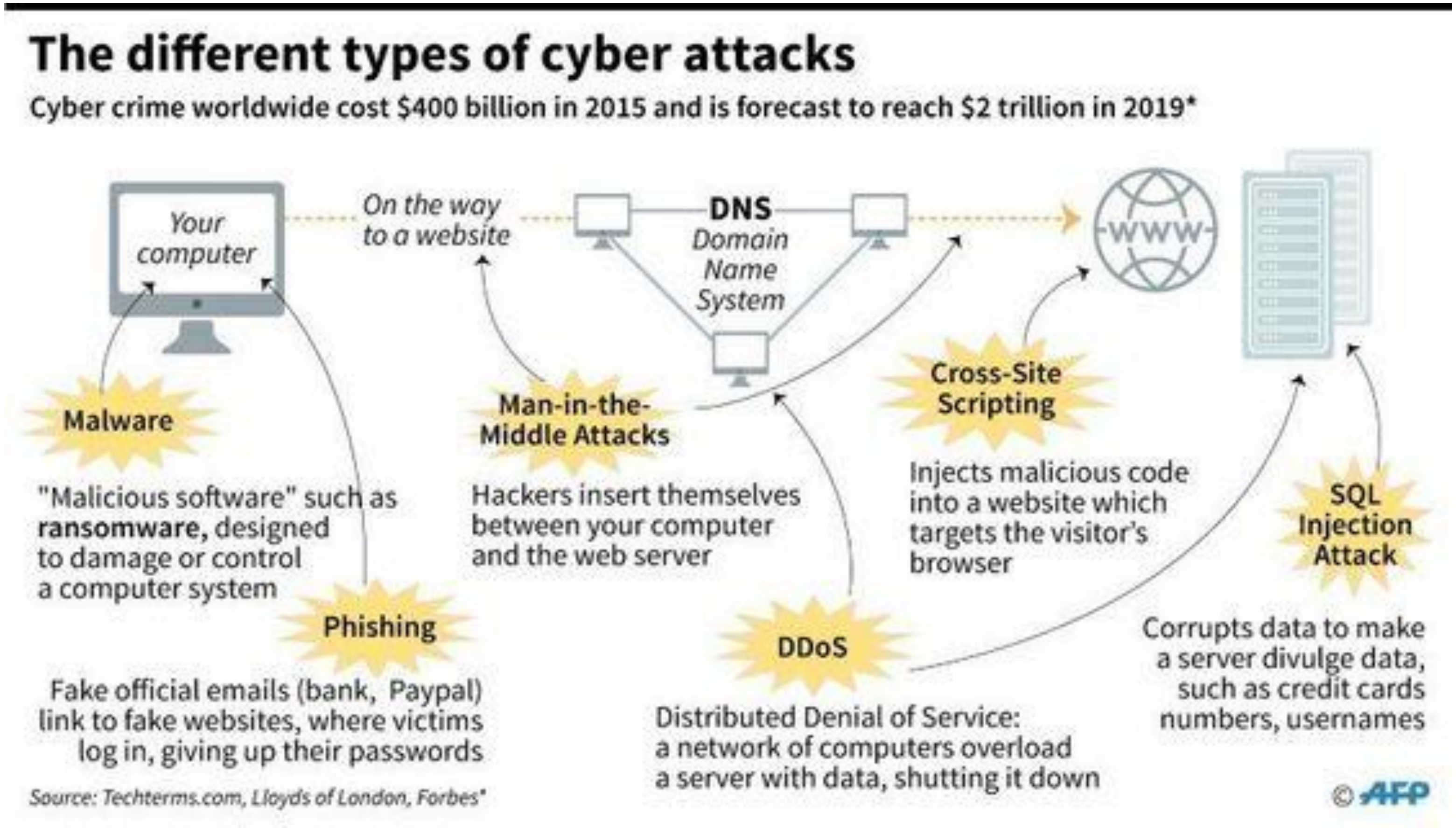
Threat: potential attempts by an adversary.

Vulnerability: Weakness in an information system that may be exploited.

Note of caution: In pre-cyber-security days, classical risk literature used the term vulnerability with a different meaning.

RFC 2828, Internet Security Glossary

Cyber attack types



The different types of cyber attacks

Cyber crime worldwide cost $400 billion in 2015 and is forecast to reach $2 trillion in 2019*

Malware

"Malicious software" such as **ransomware,** designed to damage or control a computer system

Phishing

Fake official emails (bank, Paypal) link to fake websites, where victims log in, giving up their passwords

Man-in-the-Middle Attacks

Hackers insert themselves between your computer and the web server

DDoS

Distributed Denial of Service: a network of computers overload a server with data, shutting it down

Cross-Site Scripting

Injects malicious code into a website which targets the visitor's browser

SQL Injection Attack

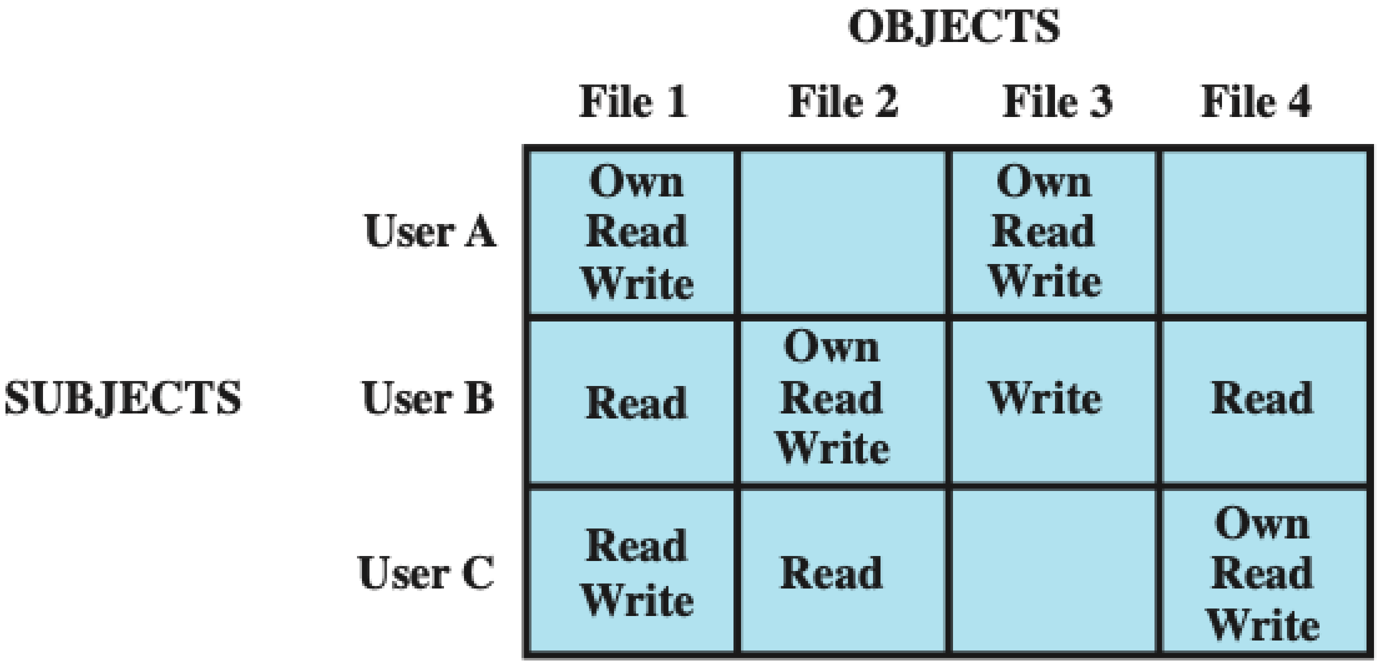
Corrupts data to make a server divulge data, such as credit cards numbers, usernames

Source: Techterms.com, Lloyds of London, Forbes*

© AFP

Colorado State University

Example: Access Control Matrix



		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix

Access Control List (ACL): Every object has an ACL that identifies what operations subjects can perform. Each access to object is checked against object's ACL.

May be kept in a relational database. Access recorded in file metadata (inode).

Authentication Methods

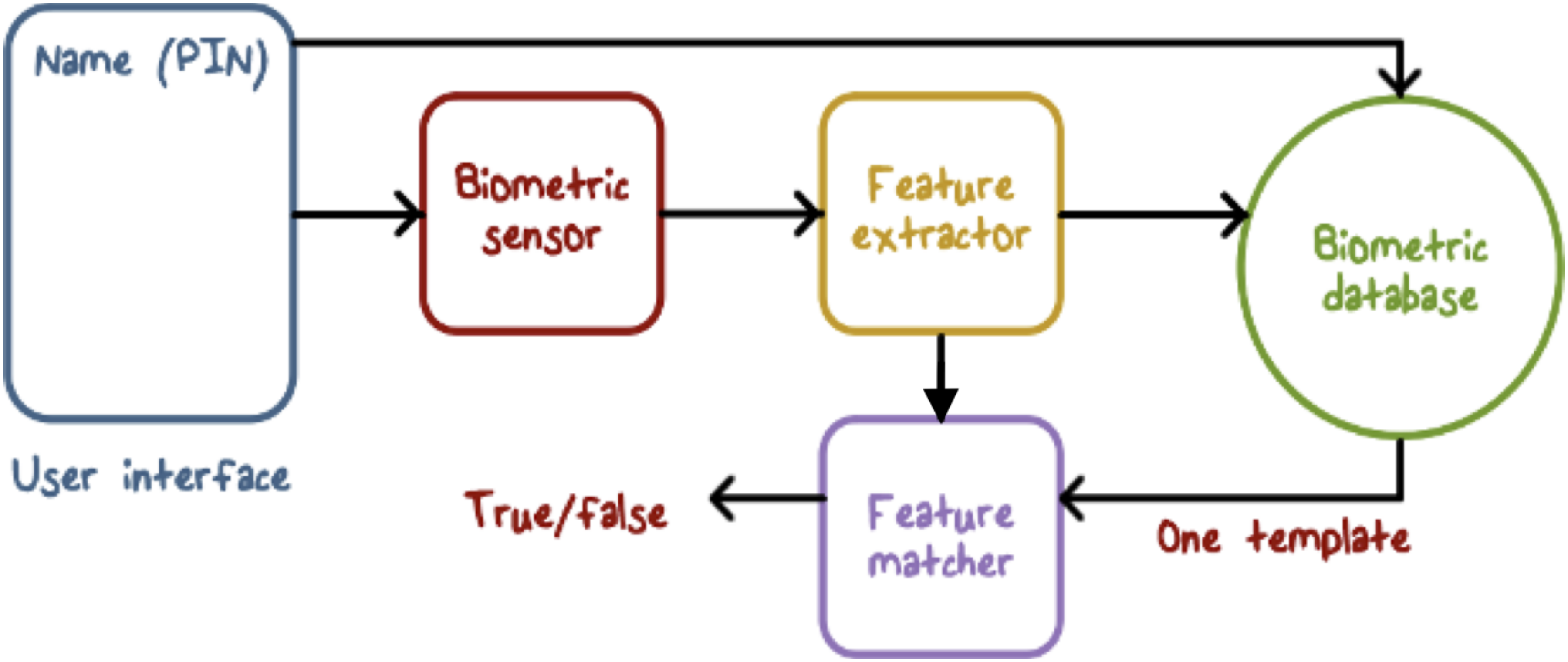
Three existing and two new.

- Something a user knows
  - Password, answers to questions
- Something a user has
  - Ex. Id card, Phone
- Something a user is
  - Biometric (face, iris, fingerprint)
- Somewhere you are geographically
- Something you do based on recognizable patterns of behavior

- Can be multifactor to reduce false positives
- After-access confirmation

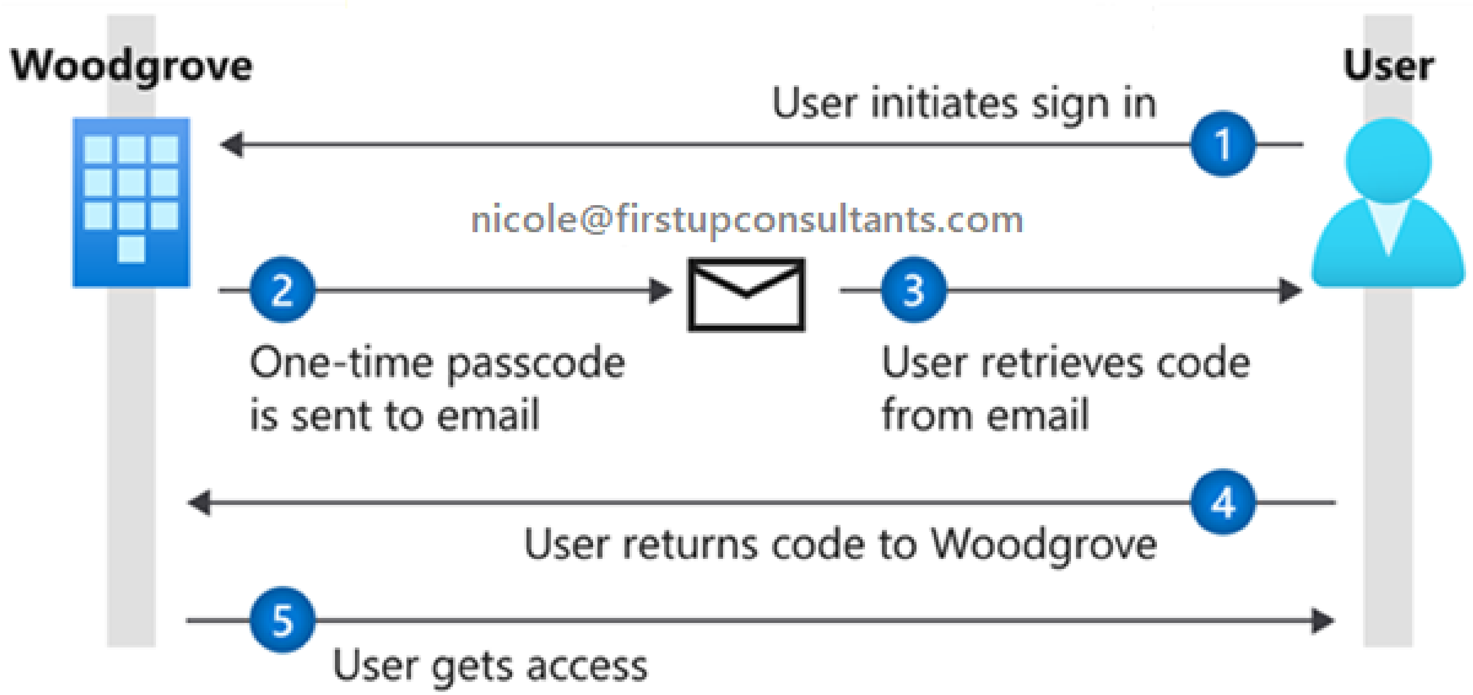
Colorado State University

Implementing Biometric Authentication



Colorado State University

Smartphone OTP Authentication


Woodgrove
User
User initiates sign in
1
nicole@firstupconsultants.com
2
One-time passcode is sent to email
3
User retrieves code from email
4
User returns code to Woodgrove
5
User gets access

OTP (one-time passcode authentication) texted or email to unique phone number/email address.

Risk: Formal definition

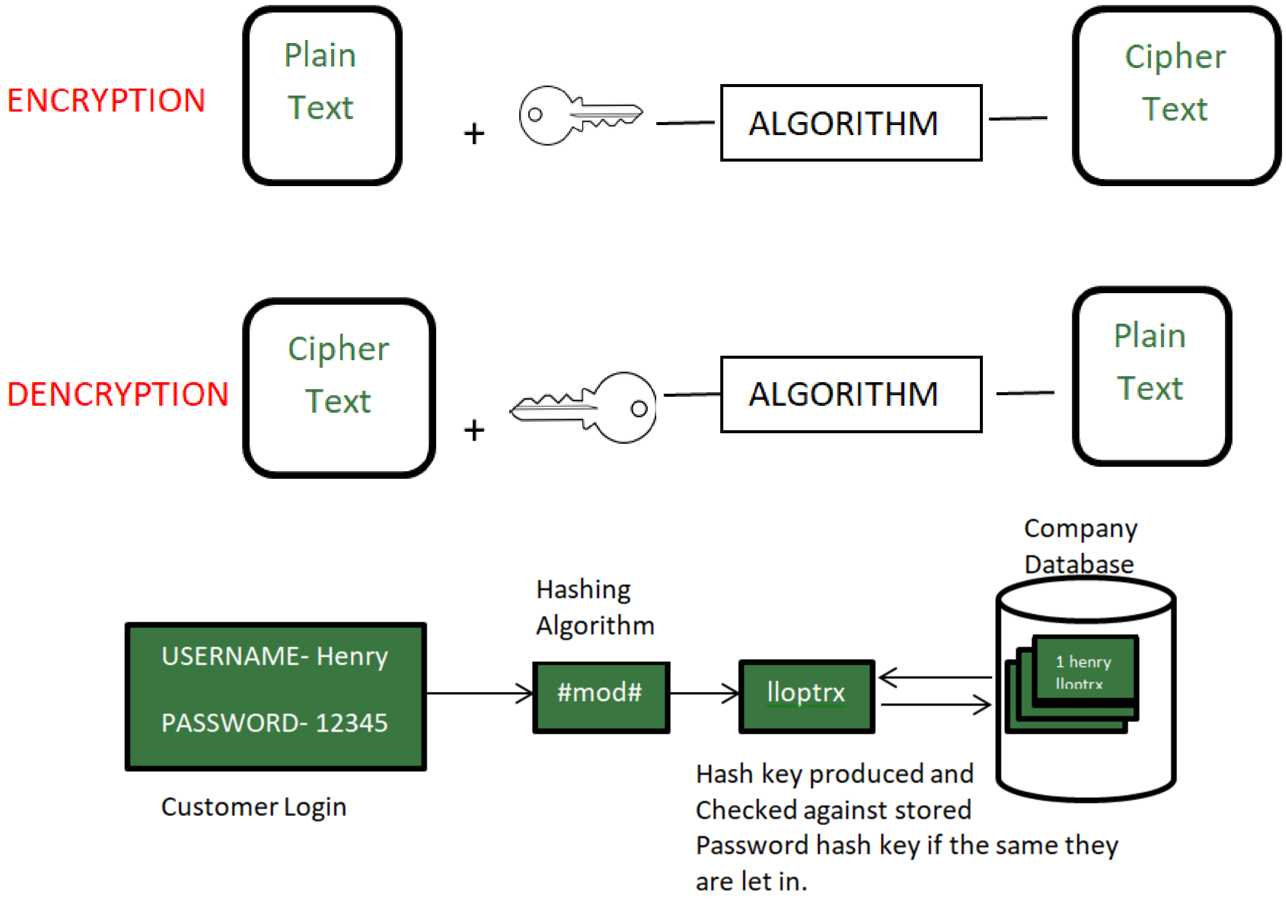
Definition: The Risk due to an adverse event e_i is

$$Risk_i = Likelihood_i \times Impact_i$$

- $Likelihood_i$: Probability of the adverse event i occurring within a specific time-frame.
  - The time-frame is often chosen to be a year. Note that the probability of an adverse event happening depends on the duration of the time-frame.
  - Probability is a number between 0 and 1.
- $Impact_i$: The impact of the adverse event, measured in monetary terms.
  - Note that impact me be direct or indirect.
  - Common units are dollars (US$)#.

US$ is a common and convenient scale. Non-monetary losses, including human life, can be converted into US$, if you are a business or insurance company .

Encryption vs Hashing



ENCRYPTION

Plain Text + — ALGORITHM — Cipher Text

DENCRYPTION

Cipher Text + — ALGORITHM — Plain Text

USERNAME- Henry

PASSWORD- 12345

Customer Login

Hashing Algorithm

#mod#

lloptrx

Company Database

1 henry

lloptrx

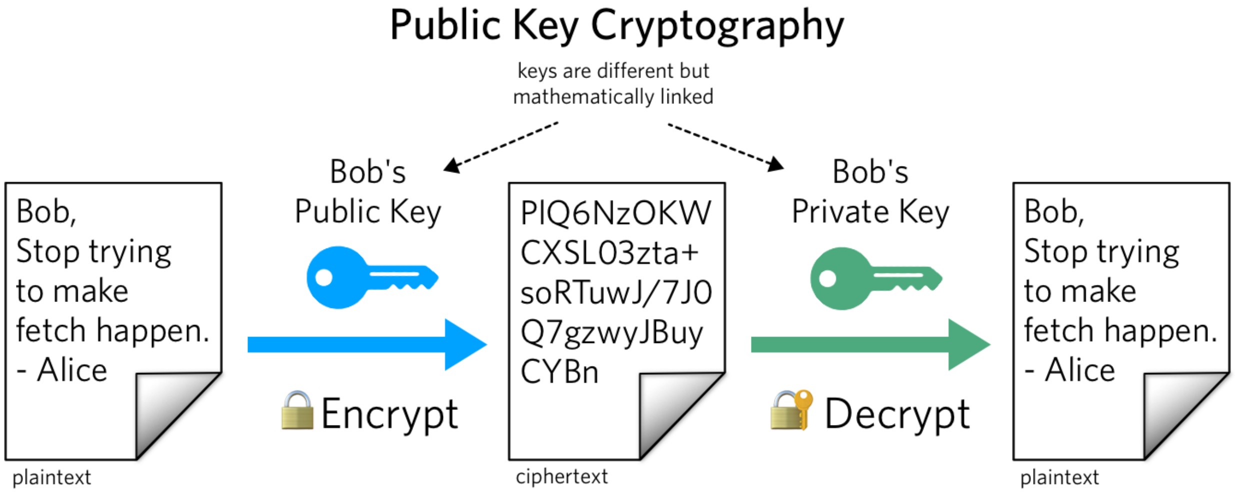
Hash key produced and Checked against stored Password hash key if the same they are let in.

- Impossible to reconstruct password from hash but ..

Public Key Cryptography?

In public-key cryptography, also known as asymmetric cryptography, each entity has two keys:

- Public Key — to be shared
- Private Key — to be kept secret
- These keys are generated at the same time using an algorithm and are mathematically linked. When using the RSA algorithm, the keys are used together in one of the following ways:


Public Key Cryptography
keys are different but mathematically linked
Bob's Public Key
Bob's Private Key
Bob, Stop trying to make fetch happen. - Alice
plaintext
Encrypt
PIQ6NzOKW CXSL03zta+ soRTuwJ/7J0 Q7gzwyJBuy CYBn
ciphertext
Decrypt
Bob, Stop trying to make fetch happen. - Alice
plaintext

https://www.twilio.com/en-us/blog/developers/tutorials/building-blocks/what-is-public-key-cryptography#What-is-Public-Key-Cryptography

Using the Public and the Private key

When using the RSA algorithm, the keys are used together in one of the following ways:

1. Encrypting with a public key

   Use: sending messages only the intended recipient can read.

   Bob encrypts a plaintext message with Alice's public key, then Alice decrypts the ciphertext message with her private key. Since Alice is the only one with access to the private key, the encrypted message cannot be read by anyone besides Alice.

2. Signing with your private key

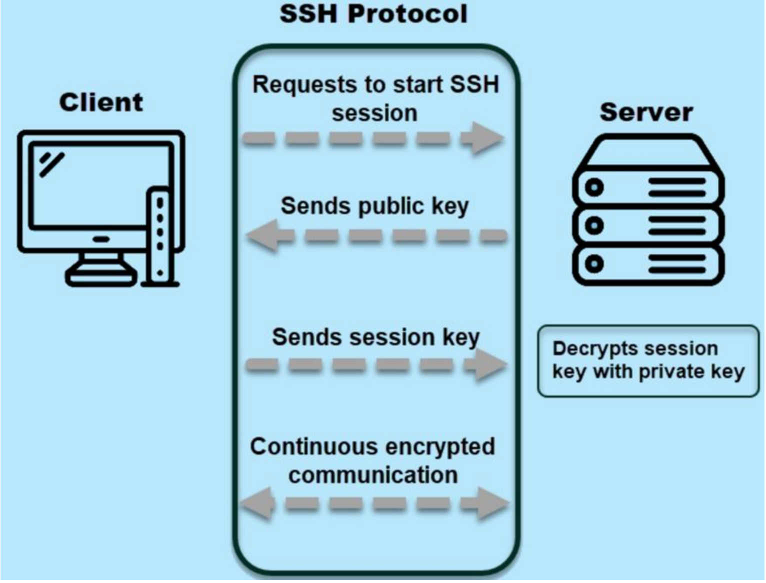
   Use: verifying that you're the one who sent a message.

   Alice encrypts a plaintext message with her private key, then sends the ciphertext to Bob. Bob decrypts the ciphertext with Alice's public key. Since the public key can only be used to decrypt messages signed with Alice's private key, we can trust that Alice was the author of the original message.

- These methods can also be combined to both encrypt and sign a message with two different key pairs.

SSH (Secure Shell Protocol) & HTTPS

- When a client initiates a connection to a server and sends a request to start an SSH session, the server responds with the public key. The client uses the public key to verify the server's identity and generates a session key (symmetric).
- Next, the client sends the key to the server in an encrypted form that only the server can decrypt. The server decrypts the code with the private key and uses it to encrypt all future communication with the client.
- Both the client and server use the session key to encrypt all communication, providing confidentiality and integrity. Once the session is complete, both the client and server exchange a final message and close the connection.
- HTTPS works the same way.


SSH Protocol
Client
Server
Requests to start SSH session
Sends public key
Sends session key
Decrypts session key with private key
Continuous encrypted communication

Colorado State University

Authentication Process
When the user returns to sign in later, the process is streamlined and passwordless

Authentication Process
When the user returns to sign in later, the process is streamlined and passwordless

How do passkeys work?

Passkeys use public key cryptography to generate an authentication key pair (a public key and a private key) also known as a credential.

- Public keys are stored on a backend server (either the *relying party* server or their authentication provider's server)
- Private keys are stored on the device where they were generated and in the passkey manager like the iCloud Keychain or Google's passkey manager.
- There are two process for using passkeys.
  - The **registration process** typically occurs after a user has signed into a website or application using an existing method. Key pair is generated and Public key is transmitted.
  - **Authentication Process** When the user returns to sign in later, the process is streamlined and passwordless.
  - Both involve several exchanges.

Colorado State University

HW6

HW6

- Quick Review
- Will not be posted