Classical Problem

- Bounded-buffer problem
- Readers and Writers Problem – ยอมให้มี reader หลายตัวได้ แต่ writer ควรมีตัวเดียว
- Dining-Philosophers Problem

Solution: อย่างน้อยคนใช้เต่าเท่ากับทรัพยากร
ถ้าจะหยิบตะเกียบต้องหยิบ 2 อันพร้อมกัน ถ้าหยิบไม่ได้ก็วางยังอันเดิม
ให้มีคนหยิบซ้ายบ้างขวาบ้าง (asymmetric)

Problems with Semaphore

- ปล่อยก่อนขอ
- ขอแล้วไม่ปล่อย ปล่อยโดยไม่ขอ } leads to deadlock and starvation
- ขอแล้วขออันเดิมซ้ำ

OS
MIDTERM – END

Memory Management - Strategies

Segmentation fail = access unallocate memory

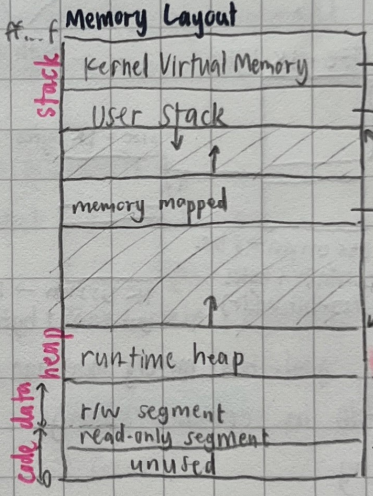
Memory Hierarchy

- taking advantage of the **locality**

Register
On-Chip Cache
Second level cache (SRAM)
Main Memory (DRAM)
Secondary Storage (Disk)
Tertiary Storage (Tape)
(CPU can access register and main memory)

speed ↓ size ↓

Memory Layout



Invisible to user code
Create at runtime
%esp (stack pointer)
for shared libraries
Create at runtime (malloc)
loaded from the executable file
(constant, global vars)
known size at complie time

Address

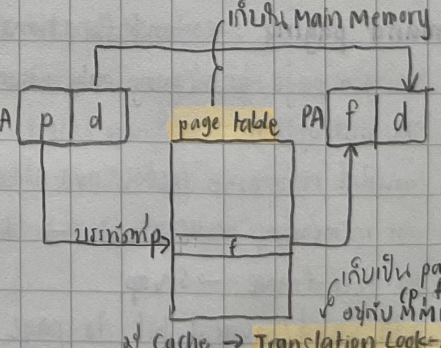
- Logical address = address in program instruction (virtual address)
- Physical address = address used by memory unit to access physical memory

* translate by memory management unit (MMU)
user doesn't have to know real physical address

Memory Allocation – เฟลเข้าของร้านอาหาร

- Continuous Allocation – มาด้วยกันนั่งด้วยกัน
- Paging – แยกกันนั่งคนละโต๊ะ

Frame = fixed size block in physical memory ~กรอบรูป
Page = block of same size in logical memory ~กระดาษห่อรูป
OS keeps track of all free frames (link list)
use **page table** to translate logical to physical address
Logical address = page number + Page offset, use page number as index of page table



Effective Access Time (EAT)

Associative Lookup = ε

Memory Access Time = m

Hit Ratio = α

EAT = $(\varepsilon + m)\alpha + (\varepsilon + 2m)(1-\alpha)$

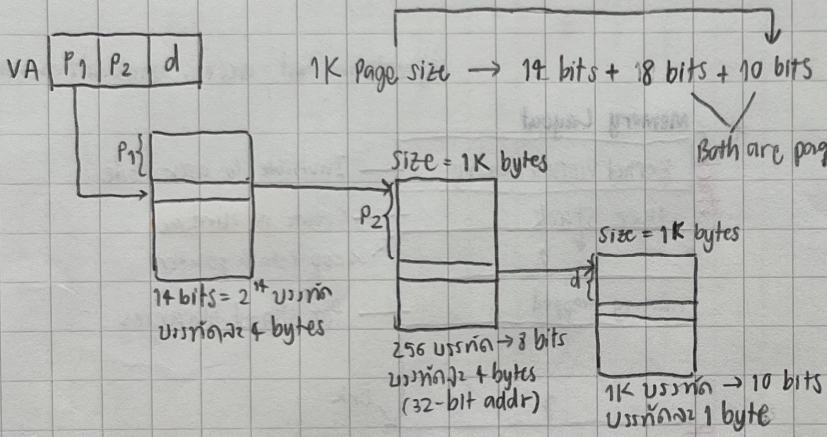
- $\varepsilon + m$: TLB access + real data access
- $\varepsilon + 2m$: more mem access of page table = m

Structure of the Page Table

goal: reduce page table size (เพราะ VA ยาวขึ้น)

① Hierarchical Page Table ⭐ ② Hashed Page Table ③ Inverted Page Table

- ที่ page table ของ page Table อีกที
- root = outer page table
- dynamically allocated page table for only used pages (อันที่ไม่ใช้ก็ไม่ต้องจอง) = null pointer



⑨ Virtual-Memory Management

goal: total memory more than physical memory

- swap partition or swap file in secondary storage → extend memory hierarchy
- Demand paging mechanism in OS
  - caching in software
  - shared address space by serveral processes → shared code, libraries, memory
- memory-mapped file, memory-mapped I/O

Demand paging : extend functionality of MMU

- bring a page to memory only when it is needed
- CPU instruction
  - invalid reference (access non-allocated) → abort
  - not in memory (allocated) → bring to memory
  - no-free-frame → swap
- add valid/invalid bit to page table

Page Fault

MMU generates hardware interrupt page fault interrupt

OS called page fault handler

1. OS looks at another table
   1. Invalid reference → abort
   2. Not in memory
2. Get empty frame ——— **Page Replacement**
   1. if there is no free frame use page replacement algorithm to select victim frame
   2. write victim frame to disk = swap out + set invalid
3. Swap page into frame = swap in
4. Update tables (TLB, page table, etc)
5. Set valid bit
6. return continue signal

Benefit

1. shared library → Memory-mapped shared library (read-only)
2. memory-mapped shared memory (R/W)
3. memory-mapped files = mapping disk block to page
   use ~~read() write()~~ memory command
   written data to disk periodically or close() time

⑪ File-system Interface – ดูที่กว้างๆ จะจัดการแฟ้มอย่างไร

File concept

File = Contiguous logical address space (logical → abstract: can be both block and char device)

- binary string + pointer

File = สมุด

File-system = ตู้เก็บหนังสือที่มีหน้าปก

File Attribute

- Name – human-readable form
- Identifier
- Type – extension (_⑧_._③_) in DOS (windows)
- Location
- Size
- Protection – who to read, write, execution
- Time, Date and user identification

File Operations

- Create
- Write } at pointer location
- Read }
- Seek – move pointer – operator for block device
- Delete
- Truncate
- [Open – move file from disk to memory
- Close – move file from memory to disk]
  - → open-file table – tracks open files
  - File-open count – นับว่าไฟล์ถูกเปิดกี่ครั้ง
  - → Locking → shared lock, Exclusive lock
    - Mandatory
    - Advisory
  - → Mode R/W Text/Binary

File Structure

- Line, Fixed length, variable length
- Formatted document, Relocatable load file
- OS และ Program เป็นคนจัดการ

Access method

- Sequential access (at pointer location)
- Direct access (at some point in file)
  - keep index in memory
  - if large, index of index

Directory structure

- collection of nodes, containing information about all files

Disk structure

- แบ่งได้เป็น partition → volume
- สร้างเป็น RAID ได้
- แต่ละ volume/disk สามารถมีหลาย file system ได้
- general-purpose file systems, special-purpose file systems
  ufs, zfs, APFS procfs, tmpfs, ctfs

Operations performed on Directory

- Search, create, delete, rename file
- List a directory
- Traverse the file system

Organize the directory (logically) to obtain

- Efficiency - locating a file quickly (search only within user)
- Naming - 1 file มีหลาย name, ต่าง user สามารถตั้งชื่อไฟล์ซ้ำกันได้
- Grouping - logical group by properties

Single-level directory : map directory to files, 1 dir for whole system (Naming and Grouping problem)

Two-level directory : ขั้นแรกแบ่ง user แล้วแต่ละ user มี user file directory (still no grouping)

Tree-Structured directory

- Absolute or relative path name
- current directory
- Acyclic-graph directory : shared subdirectory and file (for different user)
  use Link as a pointer to physical file
- do garbage collection

File System Mounting

- Single root — mount = sub node from root (in /mnt in Linux)
  Multiple root (drive A, B, C, ...) — mount = create new root

File sharing to multi-user

- userID or GroupID
  grant permission
- NFS (Network File System) — Unix protocol
  CIFS — Windows protocol
  } remote file system

Protection — who can do what

Read (R) Write (W) Execute (X) Append, Delete, List

Class of users on Unix	R	W	X		
owner (u: user)	1	1	1	=	7 (bin to dec)
group (g: group)	1	1	0	=	6
public (o: other)	0	0	1	=	1

chmod 761 game = RWX RW- --X permission
chmod u+x game = grant X to user

(12) File System Implementation

File-System Structure

- มีการแยกกันระหว่าง Virtual (Abstract) กับ Physical (Device Driver) ด้วย File System ทำงานกลาง (mapping)
- แบ่ง disk เป็น block / sector
- มี File control block — เก็บ information ของ file

Layered file system

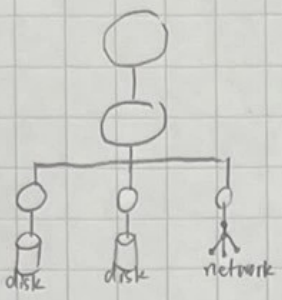
application program
| logical file system — manage metadata information (translate file name to file number) inodes in Unix
OS | file-organization module — understand files, logical addr, physical blocks (translate logical to physical)
| basic file system — translate code to device driver
| I/O control — Device driver, Specific cmd to hardware controller
HW | device — file-like device — ทุก device ถูกทำให้เป็นเหมือน file

File-System Implementation

- Boot control block — to boot OS
- Volume control block — volume detail
- File control block — file detail (inode number, permission, size, date)
  open (file) = open inode
  read (file) = read information mapped from inode

Virtual File System (VFS)

- implement vnode to hold inode or network file detail
  - API for different types of file systems / network file systems
  - API to VFS interface (not type-specific)



Network File System (NFS)

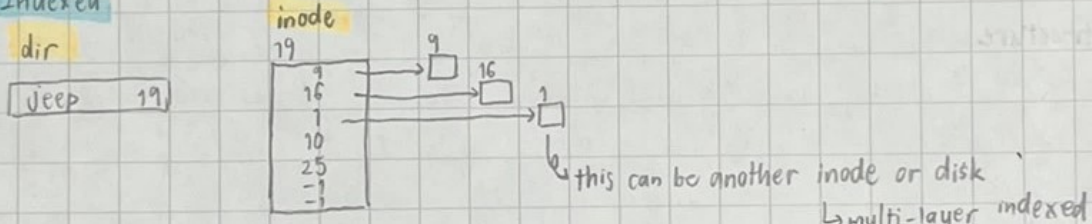
- access remote file across LAN in a transparent manner but not fully transparent (provide hostname)
- independent from environment (machine, OS, network arch)
- distinguish between the services provided by a mount mechanism and the actual remote-file-access services
- NFS servers are stateless

Directory Implementation

- Linear list
- Hash table

Allocation Methods

- Contiguous – ของ block ที่ติดกันต่อ 1 file (บอกจุดเริ่ม + ขนาด) เก็บข้อมูลใน dir (ยากขึ้นเมื่อใช้)
  - เร็ว ง่าย
  - ที่ว่างกระจาย → defragmentation เอาที่ว่างไปอยู่ด้วยกัน
- Linked – เก็บ block เริ่มต้นกับ block สุดท้าย แล้วแต่ละ block เก็บ pointer ชี้ไปที่ block ถัดไป
  - แก้ปัญหา fragment แต่โปรแกรมจะช้าลง
- Indexed



Free-Space Management

- use free-space list (bit vector / bit map) corresponding to block (1=ว่าง 0=ใช้อยู่)
- มี instruction ที่ชี้ไป first free-space
- use linked list

Efficiency and Performance

- ขึ้นอยู่กับการจัดเก็บข้อมูลใน disk และขนาดของ data structure (fixed/varies)
- cache / synchronous / free-behind and read-ahead (sequential access opt.)
- read slower than write

Recovery

- consistency checking
- backup to another storage device
- restore from backup
- capture transaction in the system to log file

⑬ Kernel Module – สิ่งที่ load ลง Kernal, upon demand

- Extend the functionality of the kernel without rebooting the system
- Hardware driver / special API
- ≈ resource manager

To write kernel in C
- must write init_module (to use when insmod) and module_exit (to use when rmmod)
- no standard C lib → use kernel function

Linux device
- view device as file (device file)
- character device : single char, cannot seek, serial/parallel port
- block device : entire block of data, can seek, disk/ usb camera
- major number and minor number

⑭ OS security

- ความปลอดภัย = สามารถเข้าถึง resource ได้เมื่อต้องการ (เข้าถึงไม่ได้ = ไม่ปลอดภัย)
- Intruder (cracker) → who breach security (≠ hacker)
- Threat (ยังไม่เกิด) and Attack (เกิดการโจมตี)
  ↳ Accidental / Malicious (จงใจ)

Security Measure Levels – no absolute security ป้องกันให้คนเจาะรู้สึกไม่อยากทำ

Physical	Data centers, Servers, Connected terminals
Application	Malicious apps
OS	Protection mechanism, logging
Network	Intercepted communications, การดักฟัง, Break 3-way handshake TCP, ARP, DNS, DDoS
Human	phishing, social-engineering (หลอกถาม password) (the weakest link)

* ความปลอดภัย ต่อรองกันกับความสะดวก

Program Threats

- **Trojan Horse** : code segment, misuses env ex. Spyware (ดักฟังข้อมูล), Pop-up browser
- **Trap Door** : Programmer สร้างเพื่อความสะดวก เช่น define username and password
- **Salami Attack** : Small Attack (ไม่ขึ้นสังเกตุ) ก่อตัวเป็นปัญหาใหญ่
  - เกิดได้ยาก เพราะเครื่อง detect ทัน, มักเกิดจากภายใน เพราะทำจากภายนอกได้ยาก
  - เช่น หักเงินทุกบัญชี บัญชีละ 0.01 บาท
- **Malware** : Software designed to exploit, disable, or damage computer
- **Ransomware** : ล็อคเครื่อง ขู่เรียกค่าไถ่เพื่อปลดล็อกข้อมูล – แก้ได้ด้วยการ backup
- **Logic bombs** : ทำลายข้อมูลเมื่อเกิด trigger บางอย่าง
  - ex. cookie monster : สร้างความรำคาญ
- **Viruses** : Code fragment, Self-replicating, Specific (เช่น virus บน Windows ไม่โจมตี Linux)
  - เจอบน Email attachment หรือ macro

The Threat Continues

- จากการทดลองสู่อาชญากรรม
- มีเงินเป็นแรงจูงใจ
- เป้าหมาย specific (โดยเฉพาะ FinTech)
- Keystroke logger = โปรแกรมดัก password ทำได้ทั้งบน sw และ hw
- OS Windows โดนโจมตีมากที่สุด (Most Common, Everyone is an admin)
  ฝั่งมือถือ = android

System and Network Threats

- Worms : standalone program เจาะ sourcecode
- port scanning : ตรวจสอบ port บน network ของคนอื่น
  - nmap : scan all ports for a response (เพื่อหาว่ามี port ไหนที่เปิดอยู่) [threat]
  - nessus
  - antiscanning : ดูว่ามีใครมา scan port เรา หรือเปล่า

* Kali Linux : Linux with security tools

* honeypot : เครื่องหลอก hacker ที่มาโจมตีเครื่องนี้แทน

Standard Security Attacks

- Masquerading : ปลอมตัวเป็นผู้รับ
- Man-in-the-middle : ส่งสารที่ผิดๆ ไปยังผู้รับจริง / ดักฟังเฉยๆ

User Authentication (การพิสูจน์ตัวตน)

- password ล้อจี้ที่พบบ่อยที่สุด — ใช้สิ่งที่รู้ / สิ่งที่มี (object) ในการตรวจสอบตัวตน
  - เก็บเป็นความลับ, เปลี่ยนบ่อย, เดายาก
  - ถ้าใช้ผิดมากๆ ก็ล็อกระบบ
  - encrypted + add salt
  - OTP
  - ความเสี่ยง : ใช้ระบบดัก / แอบมองขณะผู้ใช้ป้อนตรงๆ
- biometrics — ใช้สิ่งที่เป็น ในการตรวจสอบตัวตน
  - ลายนิ้วมือ, ม่านตา (retina)
  - ใช้ hardware โดยตรง
  - ถ้าโดนขโมย จะเปลี่ยนไม่ได้ (ต่างจาก password ที่เปลี่ยนได้)

System Admin Code of Ethics
by usenix
- เข้าถึง private info เมื่อจำเป็นเท่านั้น
- ปกป้องความลับของข้อมูล
- ดูแล NFR ที่เกี่ยวข้องกับ Security เช่น integrity, reliability

Security Defenses

- ช่วงชิงการป้องกันให้เหนือกว่า
  - educate users about safe computing, prevent phishing attacks
  - use secure communication
  - physically protect hardware
  - config OS, disable unused module
  - use modern sw and HW (up-to-date) / patch

- ใช้เทคนิคที่น่าเชื่อถือ
- antivirus, firewall
- หมั่นตรวจสอบ (log and audit)
- จ้างคนมาตรวจสอบ
- encrypt
- มี policy ที่ชัดเจน

⑮ Container — Docker

- Program depends on OS (dependency)
- Hard to do hardware-map • Scale/mange/operate
- สร้าง standard ของการขนส่ง (deliver) ผ่าน pack
- พยายามนำของใส่ container — self contain
  - runtime environment
- Build once, run anywhere
  configure once, run anything — ลองงานที่ต้องทำซ้ำ
- shared kernel, local environment

- Condor : run only when resource is available
  ถ้าไม่เพียงพอ ก็เปลี่ยนเครื่อง
- Matrix from hell → container
- Docker = code container

หลักการทำงานของ Docker container

- Build code + เอา code ใส่ตู้ } Dev / Ops } Separation of concerns
- define plugin (log, network, CPU, RAM)
- if it can run on the host, it can run in the container
- lightweight VM (เร็วกว่า เพราะไม่ต้องใช้ guest OS) — layer file system
  - เก็บเฉพาะส่วนที่ต่างกันใน Dockerfile
  - อัพเดทง่าย
  - ใช้พื้นน้อย
- 1 container = 1 chroot — isolated processes
  closed environment
- Stateless (ปิดแล้วลืมค่า)
  อยากจำได้ — plug storage
  ป้องกัน ; replica ช่วยได้

Docker Basic

- Docker abstracts storage network CPU RAM Management
- Virtual Env

Architecture
- daemon
- client
- inside : image, containers, registry (dockerhub), services

-v : ต่อ volume ภายนอก container
--link : สร้าง docker network ขึ้นมา
access ตัวที่ link ภายในด้วยกันได้
(เช่น รัน DB แล้วรัน service —link DB)

Docker Compose

- Compose = a tool for defining and running multi-container
- deploy with single command
~ เอาสายไฟต่อกัน

Docker Swarm — distrubed by Kubernetes

- orchestration for docker engine
- manager and workers → cluster