Chapter 9

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
- Allows address spaces to be shared by several processes
- Allows for more efficient process creation
- More programs running concurrently
- Less I/O needed to load or swap processes
- Pages can be shared during fork (), speeding process creation
- **Virtual memory** **can be implemented via:**
  - Demand paging
  - Demand segmentation

page fault: results from the first time there is a reference to a specific page → traps the OS
Must decide to abort if the reference is invalid, or if the desired page is just not in memory yet

1. Get empty frame
2. Swap 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

Demand Paging: only brings a page into memory when it is needed → less I/O and memory needed

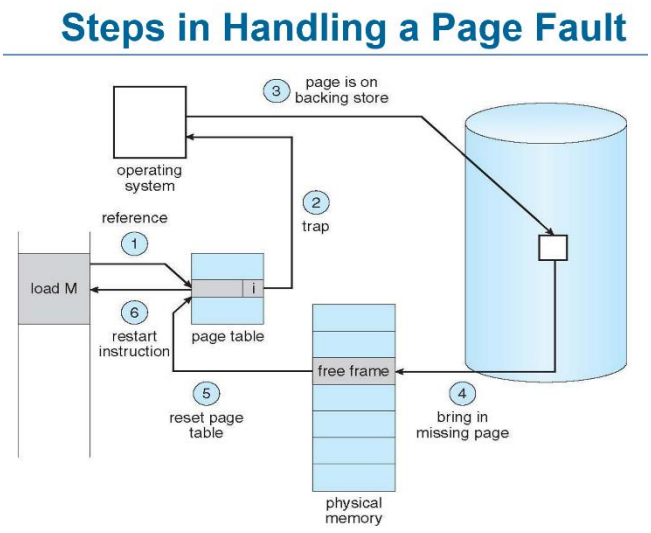
Lazy swapper: never swaps a page into memory unless page will be needed

- Swapper that deals with pages is a pager

Pure Demand Paging: process starts with no pages in memory

Valid-Invalid Bit:

- v ⇒ in-memory
- i ⇒ not-in-memory



Performance of Demand Paging:

- Trap to the operating system
- Save the user registers and process state
- Determine that the interrupt was a page fault
- Check that the page reference was legal and determine the location of the page on the disk
- Issue a read from the disk to a free frame:
    - Wait in a queue for this device until the read request is serviced
    - Wait for the device seek and/or latency time
    - Begin the transfer of the page to a free frame

Copy-on-Write (COW) allows both parent and child processes to initially share the same pages in memory

- If either process modifies a shared page, only then is the page copied
- **COW** allows more efficient process creation as only modified pages are copied
- free pages are allocated from a **pool** of **zero-fill-on-demand** pages

Thrashing: a process is busy swapping pages in and out
Increase the Page Size: This may lead to an increase in fragmentation as not all applications require a large page size
Pages can be replaced using different algorithms: FIFO, LRU
Fixed page allocation: Proportional allocation – Allocate according to size of process
Global replacement: process selects a replacement frame from set of all frames
Local replacement: each process selects from only its own set of allocated frames
Page-fault rate is very high if a process does not have "enough" pages
Memory-mapped file I/O allows file I/O to be treated as routine memory access by mapping a disk block to a page in memory
I/O Interlock: Pages must sometimes be locked into memory

1-First-In-First-Out (FIFO) Algorithm
https://www.youtube.com/watch?v=_0CJNfltKLM&index=84&list=PLO6V6DeYbPNErUCcbZOjzfG_9Y_hHsuh6

2-Optimal Algorithm
https://www.youtube.com/watch?v=NzKwoKFT78A&list=PLO6V6DeYbPNErUCcbZOjzfG_9Y_hHsuh6&index=85

3-Least Recently Used (LRU) Algorithm
https://www.youtube.com/watch?v=RXfQ0FISjBs&index=86&list=PLO6V6DeYbPNErUCcbZOjzfG_9Y_hHsuh6

Chapter 10

File: Contiguous logical address space

- Types:
  - Data (numeric, character, binary)
  - Program
- Types Contents defined by file's creator
  - text file, source file, executable file

File Attributes:

- **Name**: only information kept in human-readable form
- **Identifier**: unique tag (number) identifies file within file system
- **Type**: needed for systems that support different types
- **Location**: pointer to file location on device
- **Size**: current file size
- **Protection**: controls who can do reading, writing, executing
- **Time, date, and user identification**: data for protection, security, and usage monitoring
- Information about files are kept in the directory structure, which is maintained on the disk

File Operations

- File is an **abstract data type**
- **Create**
- **Write:** at **write pointer** location
- **Read:** at **read pointer** location
- **Reposition within file: seek**
- **Delete**
- **Truncate**
- **Open(F_i):** search the directory structure on disk for entry F_i, and move the content of entry to memory
- **Close (F_i):** move the content of entry F_i in memory to directory structure on disk

Open Files: **Several pieces of data are needed to manage open files:**

- Open-file table: tracks open files
- File-open count: counter of number of times a file is open – to allow removal of data from open-file table when last processes closes it

Open File Locking: mediates access to a file

- **Shared lock** like reader lock – several processes can acquire concurrently

- **Exclusive lock** like writer lock
- **Mandatory:** access is denied depending on locks held and requested
- **Advisory:** processes can find status of locks and decide what to do

File Types:

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

File Structure

- None - sequence of words, bytes
- Simple record structure (Lines, Fixed length, Variable length)
- Complex Structures (Formatted document, Relocatable load file)

Access Methods:

- **Sequential Access**: Information in the file is processed in order, one record after the other
- **Direct Access:** a file is made up of fixed-length logical records that allow programs to read and write records rapidly in no order

Directory Structure:

- A collection of nodes containing information about all files
- Both the directory structure and the files reside on disk

Directory: is like symbol table – translating file names into their directory entries

Operations Performed on Directory:

- Search, Create, Delete, Rename and Traverse the file system, List a directory.

logical structure of a directory:

1. **Single-Level Directory:** A single directory for all users

- Naming problem
- Grouping problem

2. **Two-Level Directory:** Separate directory for each user

- a user name and a file name define a path name
- Efficient searching
- Can have the same file name for different user
- No grouping capability

3. **Tree-Structured Directories:** allows users to create their own subdirectories and to organize their files accordingly

- Efficient searching
- **Absolute** or **relative** path name
- Grouping Capability
- Creating a new file or subdirectory is done in current directory

4. **Acyclic-Graph Directories:** Have shared subdirectories and files between users
5. **General Graph Directory:** Allow only links to file not subdirectories
    - Garbage collection
    - Every time a new link is added use a cycle detection algorithm determine if it is OK

File System Mounting: A file system must be mounted before it can be accessed

- An unmounted file system is mounted at a **mount point**

File Sharing:

- Sharing may be done through a **protection** scheme
- On distributed systems, files may be shared across a **network**
- Network File System (**NFS**) is a common distributed file-sharing method
- **User IDs** identify users, allowing permissions and protections to be per-user
- **Group IDs** allow users to be in groups, permitting group access rights

Protection:

- **File owner/creator should be able to control**:
    - what can be done
    - by whom

Types of access (Read, Write, Execute, Append, Delete, List)

Chapter 11

File control block: storage structure consisting of information about a file (exist per-file)
Device driver: controls the physical device; manage I/O devices
Logical file system: manages metadata information – maintains file control blocks
Boot control block: contains info needed by system to boot OS from volume

Volume control block: contains volume details
File control block (FCB) : contains information about the file, including ownership, permissions, and location of the file contents.
Root partition: contains OS; mounted at boot time
Virtual file systems provide object-oriented way of implementing file systems
Directories can be implemented as:

- **Linear list:** of file names with pointer to data blocks – simple but slow
- **Hash table:** linear list with hash data structure – decreased search time
- Collisions can occur in hash tables when two file names hash to same location

Allocation Methods

- **Contiguous Allocation:** each file occupies set of contiguous blocks
- **Linked Allocation:** each file a linked list of blocks
- **Indexed Allocation:** Each file has its own **index block**(s) of pointers to its data blocks
- **Performance:** Best method depends on file access type

Efficiency dependent on:

- Disk allocation and directory algorithms
- Fixed-size or varying-size data structures

Performance

- Keeping data and metadata close together
- Buffer cache – separate section of main memory for frequently used blocks
- Synchronous writes sometimes requested by apps or needed by OS

Recovery:

- **Consistency checking:** compares data in directory structure with data blocks on disk, and tries to fix inconsistencies
  - Can be slow and sometimes fails
- **Use system programs to back up** data from disk to another storage device
- Recover lost file or disk by **restoring** data from backup

Chapter 12

Magnetic disks provide bulk of secondary storage of modern computers
Transfer rate is rate at which data flow between drive and computer
Positioning time (random-access time) is time to move disk arm to desired cylinder (**seek time**) and time for desired sector to rotate under the disk head (**rotational latency**)
Head crash results from disk head making contact with the disk surface
Drive attached to computer via **I/O bus, EIDE, ATA, SATA, USB**
Host controller in computer uses bus to talk to **disk controller**
Disk Scheduling

- FCFS
- SSTF

- SCAN
- C-SCAN
- C-LOOK
- Selecting a Disk-Scheduling Algorithm

Selecting a Disk-Scheduling Algorithm

- **SSTF** is common and has a natural appeal
- **SCAN and C-SCAN** perform better for systems that place a heavy load on the disk
  - Less starvation
- **Performance** depends on the number and types of requests
- Requests for disk service can be influenced by the file-allocation method
  - And metadata layout
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary
- Either **SSTF or LOOK** is a reasonable choice for the default algorithm
- What about rotational latency?
  - Difficult for OS to calculate

LINK:
https://www.youtube.com/watch?v=qvoJixqfIBk&feature=youtu.be

Chapter 13

I/O Hardware:

- **Port** – connection point for device
- **Bus - daisy chain** or shared direct access
- **Controller** (**host adapter**) – electronics that operate port, bus, device

Blocking and Nonblocking I/O:

- **Blocking** - process suspended until I/O completed
  - Easy to use and understand
  - Insufficient for some needs
- **Nonblocking** - I/O call returns as much as available
  - User interface, data copy (buffered I/O)
  - Implemented via multi-threading
  - Returns quickly with count of bytes read or written
  - select() to find if data ready then read() or write() to transfer
- **Asynchronous** - process runs while I/O executes
  - Difficult to use
  - I/O subsystem signals process when I/O completed

Kernel I/O Subsystem

- **Scheduling**
- **Buffering** - store data in memory while transferring between devices
- **Caching** - faster device holding copy of data
- **Spooling** - hold output for a device
- **Device reservation** - provides exclusive access to a device

STREAM: a full-duplex communication channel between a user-level process and a device in Unix System V and beyond

Chapter 14

Goals of Protection:

- In one protection model, computer consists of a collection of objects, hardware or software
- Each object has a unique name and can be accessed through a well-defined set of operations
- Protection problem - ensure that each object is accessed correctly and only by those processes that can do so

Principles of Protection:

- Guiding principle – **principle of least privilege**
  - Programs, users and systems should be given just enough **privileges** to perform their tasks
  - Limits damage if entity has a bug, gets abused
  - Can be static (during life of system, during life of process)
  - Or dynamic (changed by process as needed) – **domain switching**, **privilege escalation**
  - "Need to know" a similar concept regarding access to data
- Must consider "grain" aspect
  - Rough-grained privilege management easier, but least privilege now in large chunks
    - example, Unix processes either have abilities of the associated user, or of root
  - Fine-grained management more complex, more overhead, but more protective
- Domain can be user, process, procedure

Access Matrix:

- View protection as a matrix (access matrix)
- Rows represent domains
- Columns represent objects
- Access (i, j) is the set of operations that a process executing in Domain can invoke on Object

Use of Access Matrix (Special access rights) :

- **owner of O_i**
- **copy op from O_i to O_j (denoted by "*")**

- **control – D_i can modify D_j access rights**
- **transfer – switch from domain D_i to D_j**

Access matrix design separates mechanism from policy

- **Mechanism**
  - Operating system provides access-matrix + rules
  - If ensures that the matrix is only manipulated by authorized agents and that rules are strictly enforced
- **Policy**
  - User dictates policy
  - Who can access what object and in what mode

Chapter 15

Security Problem

- **System secure** if resources used and accessed as intended under all circumstances
  - **Unachievable**
- Intruders (**crackers**) attempt to breach security
- **Threat** is potential security violation
- **Attack** is attempt to breach security
- **Attack** can be accidental or malicious
- Easier to protect against accidental than **malicious** misuse

Security Violation Categories

- **Breach of confidentiality**
  - Unauthorized reading of data
- **Breach of integrity**
  - Unauthorized modification of data
- **Breach of availability**
  - Unauthorized destruction of data
- **Theft of service**
  - Unauthorized use of resources
- **Denial of service (DOS)**

- Prevention of legitimate use

Security Violation Methods

- **Masquerading** (breach **authentication**)
  - Pretending to be an authorized user to escalate privileges
- **Replay attack**
  - As is or with message modification
- **Man-in-the-middle attack**
  - Intruder sits in data flow, masquerading as sender to receiver and vice versa
- **Session hijacking**
  - Intercept an already-established session to bypass authentication

Security Measure Levels:

- **Physical**
  - Data centers, servers, connected terminals
- **Human**
  - Avoid **social engineering**, **phishing**, **dumpster diving**
- **Operating System**
  - Protection mechanisms, debugging
- **Network**
  - Intercepted communications, interruption, DOS

Program Threats:

- **Trojan Horse:**
  - Code segment that misuses its environment
  - Spyware, pop-up browser windows, covert channels
  - Up to 80% of spam delivered by spyware-infected systems
- **Trap Door**: Specific user identifier or password that circumvents normal security procedures
- **Logic Bomb:** Program that initiates a security incident under certain circumstances
- **Stack** and **Buffer Overflow**: Exploits a bug in a program
- **Viruses:** Code fragment embedded in legitimate program
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
- **Virus dropper:** inserts virus onto the system
- **Attacks:** still common, still occurring
  - Attacks moved over time from science experiments to tools of organized crime
  - Targeting specific companies
  - Creating botnets to use as tool for spam and DDOS delivery
  - Keystroke logger to grab passwords, credit card numbers

Why is Windows the target for most attacks?

- **Most common**
- **Everyone is an administrator**
- **Licensing required?**
- **Monoculture considered harmful**

System and Network Threats

- **Worms:** use spawn mechanism; standalone program
  - **Grappling hook** program uploaded main worm program
- **Port scanning:**
  - Automated attempt to connect to a range of ports on one or a range of IP addresses
  - Detection of answering service protocol
  - Detection of OS and version running on system
  - Frequently launched from **zombie systems** to decrease trace-ability
- **Denial of Service**
  - Overload the targeted computer preventing it from doing any useful work
  - **Distributed denial-of-service (DDOS)** come from multiple sites at once
  - Consider traffic to a web site

Cryptography:

- Means to constrain potential senders (sources) and / or receivers (destinations) of messages
- Based on secrets (**keys**)

Encryption:[K of keys],[M of **messages**],[C of ciphertexts],function E:K to encrypt,function D:K to decrypt

Symmetric Encryption:

- **Same key** used to **encrypt** and **decrypt**
- **DES** is most commonly used symmetric block-encryption algorithm
- **Triple-DES** considered more secure
- **Advanced Encryption Standard** (**AES**), **twofish** up and coming
- **RC4** is most common symmetric stream **cipher**, but known to have **vulnerabilities**

Asymmetric Encryption

- **Public-key** encryption based on each user having two keys:
  - **public key** – published key used to **encrypt** data
  - **private key** – key known only to individual user used to **decrypt** data
- Most common is RSA block cipher
- Keys can be stored on a **key ring**

Cryptography:

- **symmetric** cryptography based on **transformations**
- **asymmetric** based on **mathematical** functions
  - **Asymmetric** much more **compute** intensive

Authentication:

- Constraining set of potential senders of a message
- Helps to prove that the message is **unmodified**
- **Hash** functions are basis of authentication
- Creates small, fixed-size block of data (message digest, hash value)
- Symmetric encryption used in **message-authentication code (MAC)**
- Authenticators produced from authentication algorithm are **digital signatures**
- **Digital Certificates:** proof of who or what owns a public key

Why authentication if a subset of encryption?

- Fewer computations (except for RSA digital signatures)
- Authenticator usually shorter than message
- Sometimes want authentication but not confidentiality

Digital Certificates: proof of who or what owns a public key

Defense in depth: most common security theory – multiple layers of security

- **Can attempt to detect intrusion:**
- **Signature-based:** detect "bad patterns"
- **Anomaly detection:** spots differences from normal behavior
- Can detect **zero-day** attacks
- **False-positives** and **false-negatives** a problem
- Auditing, accounting, and logging specific system or network activity

Firewalling to Protect Systems and Networks

- A network firewall is placed between **trusted** and **untrusted** hosts
- The firewall limits network **access** between these two security domains
- **Personal firewall** is software layer on given host
- **Application proxy firewall** understands application protocol and can control them
- **System-call firewall** monitors all important system calls and apply rules to them

Important Question:

What is the difference between symmetric and asymmetric encryption?

symmetric encryption	asymmetric encryption
the same key is used to encrypt and to decrypt	are different encryption and decryption keys
based on transformations	based on mathematical functions
much more computationally expensive to execute	

What is port scanning and how is it typically launched?

Port scanning is not an attack but rather is a means for a cracker to detect a system's vulnerabilities to attack. Port scanning typically is automated, involving a tool that attempts to create a TCP/IP connection to a specific port or a range of ports. Because port scans are detectable, they are frequently launched from zombie systems.

Bootstrap program?

is loaded at power-up or reboot

1. Typically stored in ROM or EPROM, generally known as firmware
2. Initializes all aspects of system
3. Loads operating system kernel and starts execution

What is the difference between Deleting and truncate file operations?

Delete: release all file space, so that it can be reused by other files, and erase the directory entry.
Truncate: erase the contents of a file but keep its attributes except for file length which reset to length zero

Types of System Calls:

- Process control
- File management
- Device management
- Information maintenance
- Communications
- Protection

- Asymmetric clustering has one machine in hot-standby mode
- Symmetric clustering has multiple nodes running applications, monitoring each other

- Asymmetric Multiprocessing: only one processor accesses the system data structures, alleviating the need for data sharing.
- Symmetric Multiprocessing(SMP): each processor is self-scheduling, all processes in common ready queue, or each has its own private queue of ready processes.