



Operating System Practice

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information
Engineering, Chang Gung University

Advanced Operating System Concepts

- Chapter 10: File System
- Chapter 11: Implementing File-Systems
- Chapter 12: Mass-Storage Structure
- Chapter 13: I/O Systems
- ● Chapter 14: System Protection
- Chapter 15: System Security



Study Items

- ▶ Goals of Protection
- ▶ Principles of Protection
- ▶ Domain of Protection
- ▶ Access Matrix
- ▶ Implementation of Access Matrix
- ▶ Access Control
- ▶ Revocation of Access Rights
- ▶ Capability-Based Systems
- ▶ Language-Based Protection



Goals of Protection

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

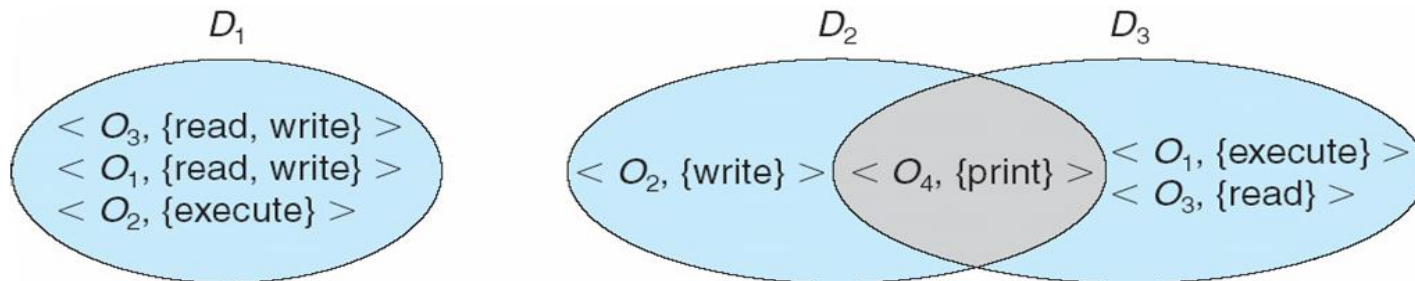
Principles of Protection

- ▶ 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 or gets abused
- ▶ Principle of Need-to-Know
 - At any time, a process should be able to access only those resources that it currently requires to complete its task



Domain Structure

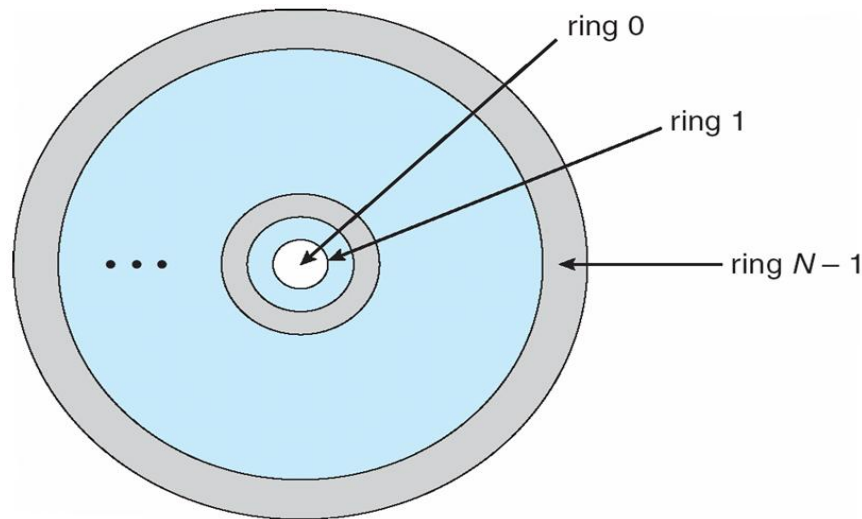
- ▶ A domain = a set of access-rights
- ▶ Access-right = $\langle \text{object-name}, \text{rights-set} \rangle$
where *rights-set* is a subset of all valid operations that can be performed on the object



- ▶ An example in Unix
 - Domain = user-id
 - When `setuid = on`, then user-id is set to owner of the file being executed
 - Domain switch accomplished via passwords
 - `su` command temporarily switches to another user's domain
 - `sudo` command prefix executes specified command in another

MULTICS Ring Structure

- ▶ Let D_i and D_j be any two domain rings
 - If $j < i$ then $D_i \subseteq D_j$
- ▶ Problem:
 - Process A can access object X but should not access object Y
 - Process B can access object Y but should not access object X



Access Matrix

- ▶ The entry $\text{access}(i,j)$ defines the set of operations that a process executing in domain D_i can invoke on object O_j
- ▶ Switching from domain D_i to domain D_j is allowed if and only if the access right $\text{switch} \in \text{access}(i, j)$

domain \ object	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			

Access Matrix Operations

- ▶ Allowing controlled change in the contents of the access-matrix entries requires three additional operations: copy, owner, and control
- ▶ Copy: can copy the access methods to other domains
- ▶ Owner: can change the access methods of all domains
- ▶ Control: can remove the access methods of all domains



Implementation of Access Matrix (1 / 2)

- ▶ Option 1 – Global table
 - Store ordered triples $\langle domain, object, rights-set \rangle$ in table
 - The table could be large \rightarrow won't fit in main memory
 - Difficult to group objects
 - Consider an object that all domains can read
- ▶ Option 2 – Access lists for objects
 - Resulting per-object list consists of ordered pairs $\langle domain, rights-set \rangle$ defining all domains with non-empty set of access rights for the object
 - Easily extended to contain default set
 - All domains can read an object \rightarrow set the read operation in the default set



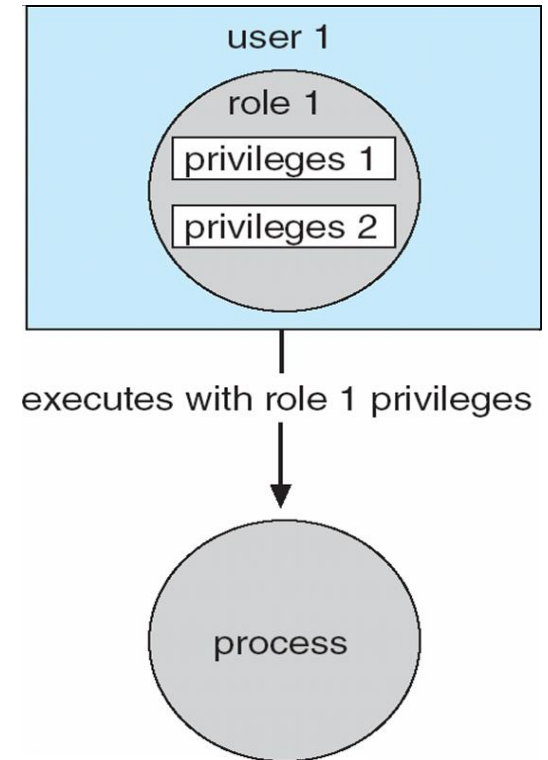
Implementation of Access Matrix (2/2)

- ▶ Option 3 – Capability list for domains
 - Instead of object-based, the list is domain based
 - Capability list for domain is a list of objects together with operations allows on them
- ▶ Option 4 – Lock-key
 - Compromise between access lists and capability lists
 - Each object has a list of unique bit patterns, called **locks**
 - Each domain has a list of unique bit patterns called **keys**
 - A process in a domain can access an object if the domain has a key that matches one of the locks



Access Control

- ▶ Solaris 10 provides role-based access control (RBAC) to implement least privilege
 - Privilege is the right to execute a system call or use an option within a system call
 - Role is a collection of privilege
 - Enable role via password to gain its privileges
 - This implementation of privileges decreases the security risk associated with superusers and setuid programs



Revocation of Access Rights

- ▶ Various options to remove the access right of a domain to an object
 - Immediate vs. delayed
 - Selective vs. general
 - Partial vs. total
 - Temporary vs. permanent
- ▶ Access List – Delete access rights from access list
 - Simply search access list and remove entry
- ▶ Capability List – Scheme required to locate capability in the system before capability can be revoked



Language-Based Protection

- ▶ Specification of protection in a programming language allows the high-level description of policies for the allocation and use of resources
- ▶ Language implementation can provide software for protection enforcement when automatic hardware-supported checking is unavailable
- ▶ Interpret protection specifications to generate calls on whatever protection system is provided by the hardware and the operating system



Protection in Java 2

- ▶ Protection is handled by the Java Virtual Machine (JVM)
- ▶ A class is assigned a protection domain when it is loaded by the JVM
- ▶ The protection domain indicates what operations the class can (and cannot) perform
- ▶ If a library method is invoked that performs a privileged operation, the stack is inspected to ensure the operation can be performed by the library



Advanced Operating System Concepts

- Chapter 10: File System
- Chapter 11: Implementing File-Systems
- Chapter 12: Mass-Storage Structure
- Chapter 13: I/O Systems
- Chapter 14: System Protection
- ➔ ● Chapter 15: System Security



Study Items

- ▶ The Security Problem
- ▶ Program Threats
- ▶ System and Network Threats
- ▶ Cryptography as a Security Tool
- ▶ Firewalling to Protect Systems and Networks



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
 - Up to 80% of spam delivered by spyware-infected systems
- ▶ Trap Door
 - Specific user identifier or password that circumvents normal security procedures
 - Could be included in a compiler
- ▶ Logic Bomb
 - Program that initiates a security incident under certain circumstances
- ▶ Stack and Buffer Overflow
 - Failure to check bounds on inputs, arguments
 - Pointed to code loaded onto stack that executes malicious code



System and Network Threats

▶ Port Scanning

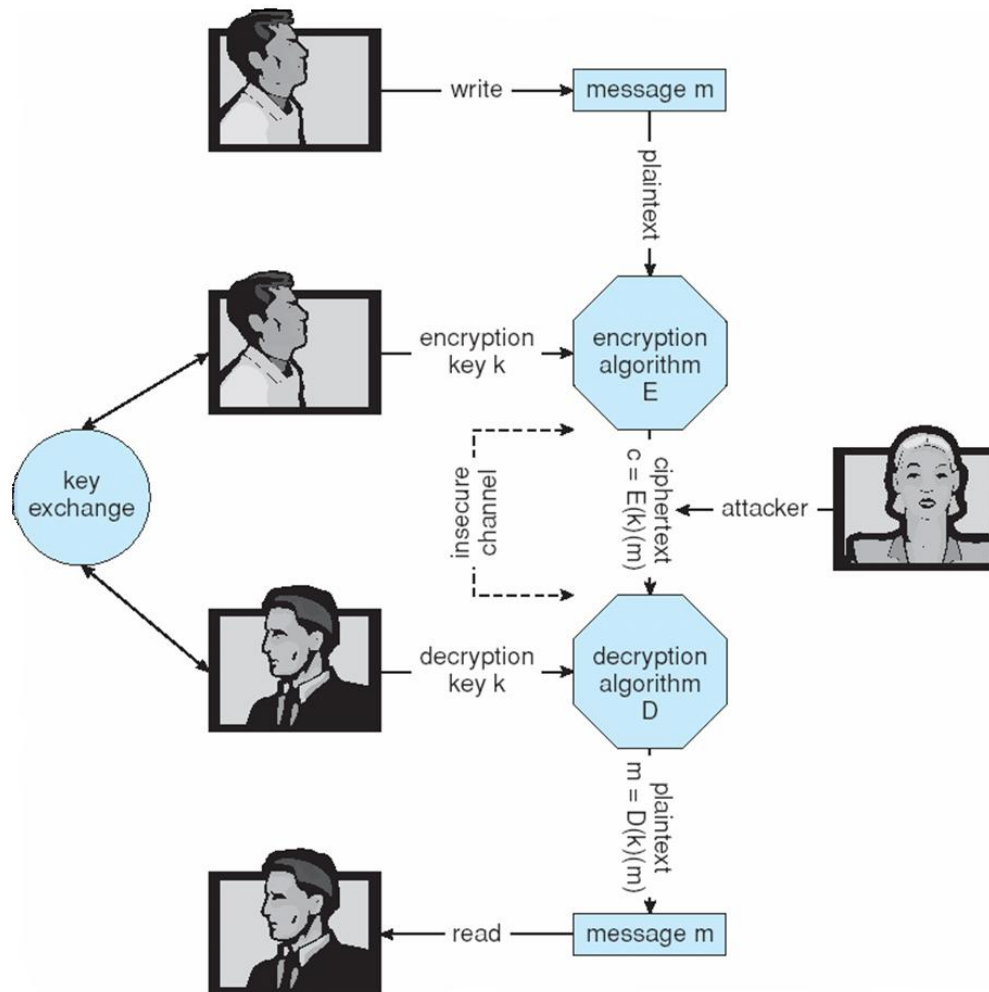
- Automated attempt to connect to a range of ports on one or a range of IP addresses
- Detection of answering service protocol
- Frequently launched from zombie systems

▶ 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



Secure Communication over Insecure Medium



Encryption

- ▶ Encryption algorithm consists of
 - Set K of keys
 - Set M of Messages
 - Set C of ciphertexts (encrypted messages)
- ▶ A encryption function $E : K \rightarrow (M \rightarrow C)$ which is a function for generating ciphertexts from messages
- ▶ A function $D : K \rightarrow (C \rightarrow M)$ which is a function for generating messages from ciphertexts



Symmetric/Asymmetric Encryption

▶ Symmetric Encryption

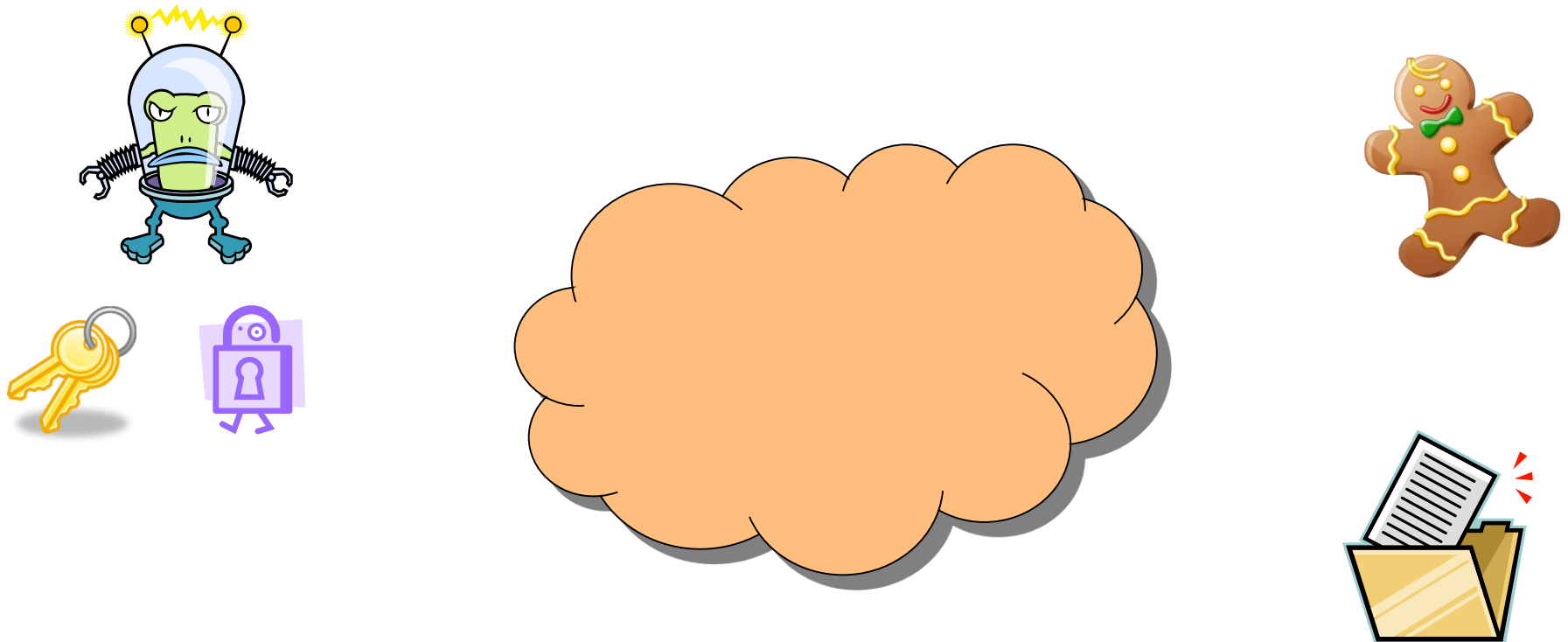
- Same key used to encrypt and decrypt
- Data Encryption Standard (DES) is most commonly used symmetric block-encryption algorithm
- Advanced Encryption Standard (AES)

▶ Asymmetric Encryption

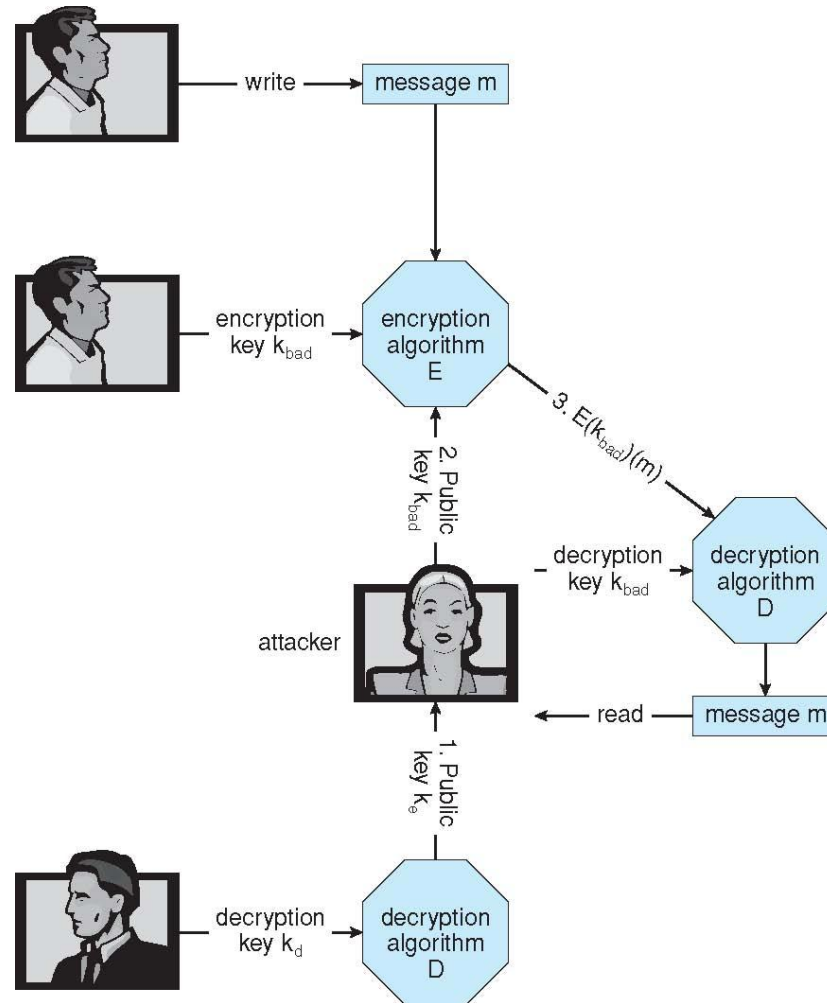
- Public key – published key used to encrypt data
- Private key – key known only to individual user used to decrypt data
- Most common is RSA block cipher



Scenario of Asymmetric Encryption



Man-in-the-middle Attack on Asymmetric Cryptography



Example

⚠ 不安全 | <https://163.25.101.35/cgi-bin/>



你的連線不是私人連線

攻擊者可能會試圖從 **163.25.101.35** 竊取你的資訊 (例如密碼、郵件或信用卡資料)。 [瞭解詳情](#)

NET::ERR_CERT_AUTHORITY_INVALID

自動傳送部分系統資訊和網頁內容給 Google，協助偵測危險的應用程式和網站。 [隱私權政策](#)

進階

返回安全性瀏覽



Encryption Example – SSL

- ▶ SSL – Secure Socket Layer
 - A newer version is Transport Layer Security (TLS)
 - But we usually just call it SSL
- ▶ Insertion of cryptography at the transport layer of the ISO network model
- ▶ Cryptographic protocol that limits two computers to only exchange messages with each other
- ▶ Used between web servers and browsers for secure communication
 - e.g., credit card numbers
- ▶ The server is verified with a certificate assuring client is talking to correct server
- ▶ Asymmetric cryptography used to establish a secure session key (symmetric encryption) for bulk of communication during session
- ▶ Communication between each computer then uses symmetric key cryptography



Firewall

- ▶ Firewall is a software or hardware-based network security system that controls the incoming and outgoing network traffic
 - Untrusted domain
 - Internal computers
 - Demilitarized zone

