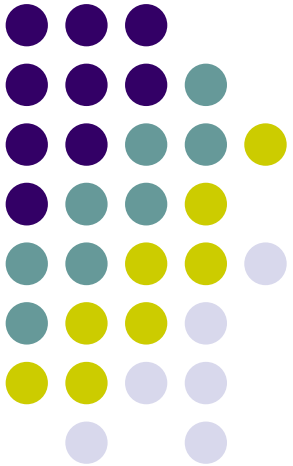
Chapter 14: System Security

WHAT'S AHEAD:

- The Security Problems and approaches
- Program Threats
- System & Network Threats
- Protection based on Access Control
- Cryptography as a Security Tool
- User Authentication
- Implementing Security Defenses
- An Example: Windows

WE AIM:

- Discuss the principles and approaches of system security in a modern computer system
- To discuss security threats and attacks
- Explain how access control is exercised
- To explain the fundamentals of cryptography and authentication



Note: These lecture materials are based on the lecture notes prepared by the authors of the book titled *Operating System Concepts*, 9e (Wiley), and by William Stallings for his book titled *Operating Systems: Internals and Design Principles*, 7e

핵심•요점

보안 위협요소 및 보안 기법

프로그램/시스템/
네트워크 위협
비밀 누설
데이터 훼손
자원 도용/도난
파괴
도청
위장
메시지 훼손
서비스 거부 공격
재사용

해결방안의 기초

- 접근제어
- 암호화 기법
- 인증

인증
권한 부여
서비스 가용성
자원 보호

사용자 프라이버시
데이터 무결성
데이터 프라이버시
데이터 인증

Core Ideas

Security Threats and Techniques

Program/System/
Network Threats
leakage
data tampering
resource stealing
vandalism
eavesdropping
masquerading
message tampering
denial of service
replay

Basic approaches

- access control
- cryptography
- authentication

authentication
authorization
service availability
resource protection

user privacy
data integrity
data privacy
data authentication

Security Problems and Approaches

- System **secure** if resources used and accessed as intended under all circumstances
  - Unachievable
- Security problems
  - 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 approaches
  - Protection based on access control
  - Cryptography
  - Authentication

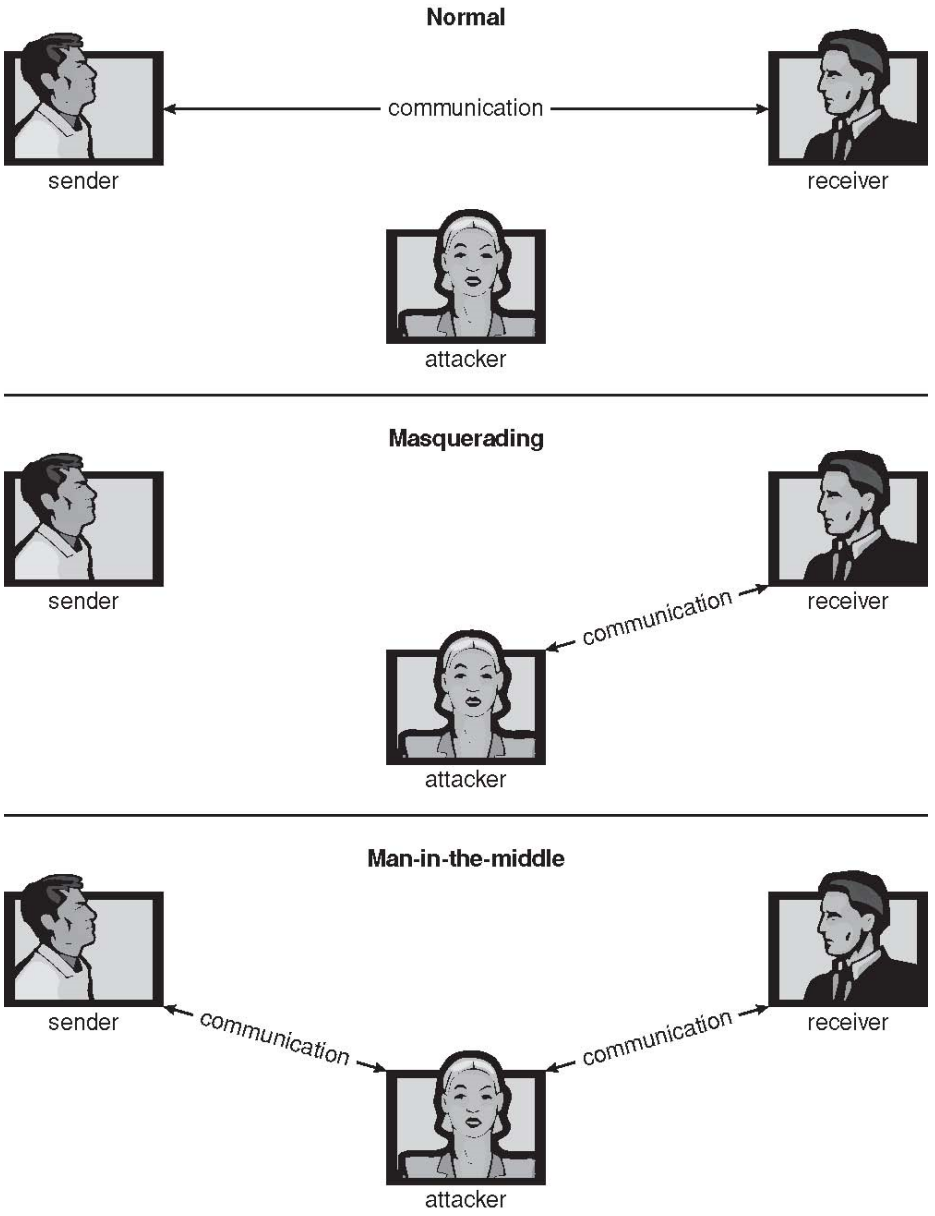
Security Violation Categories

- **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
  - Unauthorized prevention of service availability

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

Standard Security Attacks (Examples)



Normal

Masquerading

Man-in-the-middle

Security Measure Levels

- Impossible to have absolute security, but make cost to perpetrator sufficiently high to deter most intruders
- Security must occur at four levels to be effective:
  - **Physical**
    - Data centers, servers, connected terminals
  - **Human**
    - Avoid **social engineering**, **phishing**, **dumpster diving**
  - **Operating System**
    - Protection mechanisms, debugging
  - **Network**
    - Intercepted communications, interruption, DOS
- Security is as weak as the weakest link in the chain
- But can too much security be a problem?
  - System overhead, performance degradation, etc.
  - Bothers legitimate users

Social engineering attack: phishing, dumpster diving, etc.

Program Threats

- Many variations, many names
- **Trojan Horse**
  - Code segment that misuses its environment
  - Exploits mechanisms for allowing programs written by users to be executed by other users
  - **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 (e.g. test-test)
  - Could be included in a compiler
  - Done by a program designer
  - How to detect them? Very hard. Need to analyze entire source code

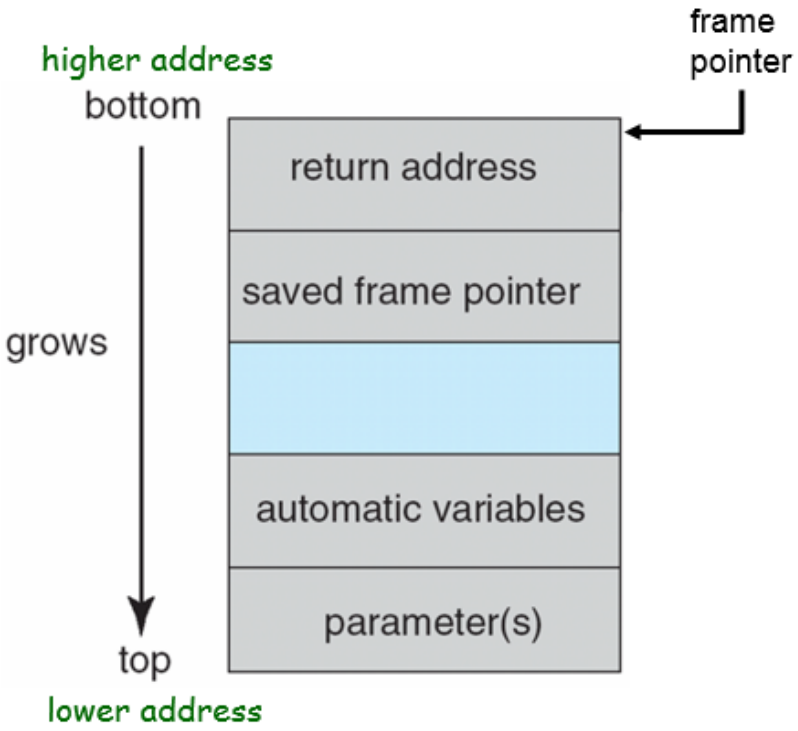
Program Threats (Cont.)

- **Logic Bomb**
  - Program that initiates a security incident under certain circumstances
- **Stack** and **Buffer Overflow**
  - Exploits a bug in a program (overflow either the stack or memory buffers)
  - Failure to check bounds on inputs, arguments
  - Write past arguments on the stack into the return address on stack
  - When routine returns from call, returns to hacked address
    - Pointed to code loaded onto stack that executes malicious code
  - Unauthorized user or privilege escalation

C Program with Buffer-overflow Condition

```
#include <stdio.h>
#define BUFFER_SIZE 256
int main(int argc, char *argv[])
{
    char buffer[BUFFER_SIZE];
    if (argc < 2)
       return -1;
    else {
       strcpy(buffer,argv[1]);
       return 0;
    }
}
```

Layout of Typical Stack Frame



- **`strcpy(dest, src)`:** Copies the *null*-terminated byte string pointed to by `src` to byte string, pointed to by `dest`.

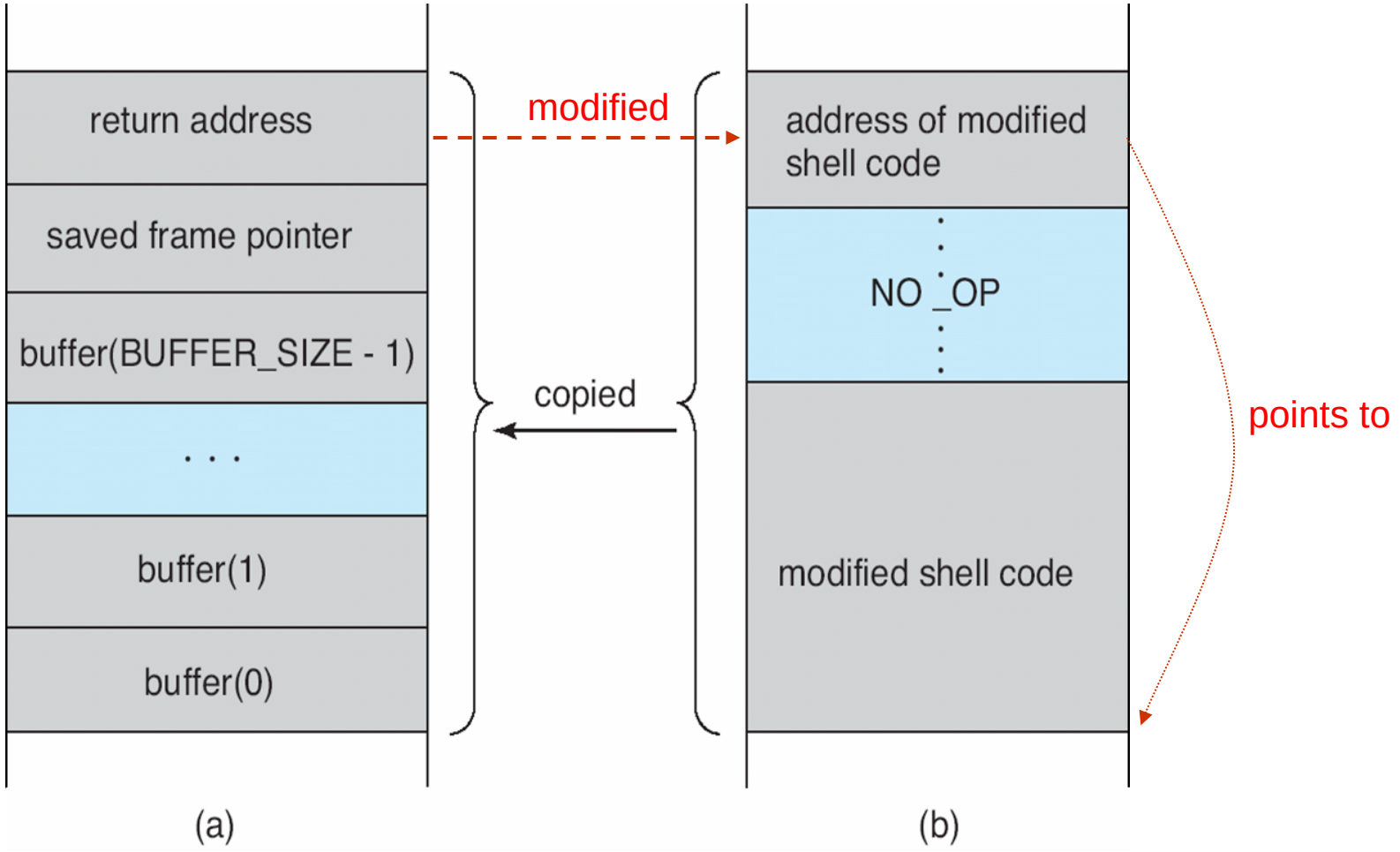
- What if the parameter provided on the command line is longer than the length BUFFER_SIZE?

Consequences of Buffer Overflow

- If the parameter provided on the command line is longer than the length BUFFER_SIZE
  - **strcpy()** will copy from **argv[1]** untill it encounters a null terminator (**\0**) or until the program crashes
  - → will lead to buffer overflow
  - → modify the return address on the stack so the new address may point to an "exploit"
  - exploit: program, data, or sequence of commands that takes advantage of a bug, glitch or vulnerability in order to cause unintended or unanticipated behavior in a computer
- Example of an exploit: Modified shell code

```
#include <stdio.h>
int main(int argc, char *argv[])
{
   execvp(''\bin\sh'',''\bin \sh'', NULL);
   return 0;
}
```

Hypothetical Stack Frame



Before attack

After attack

Program Threats (Cont.)

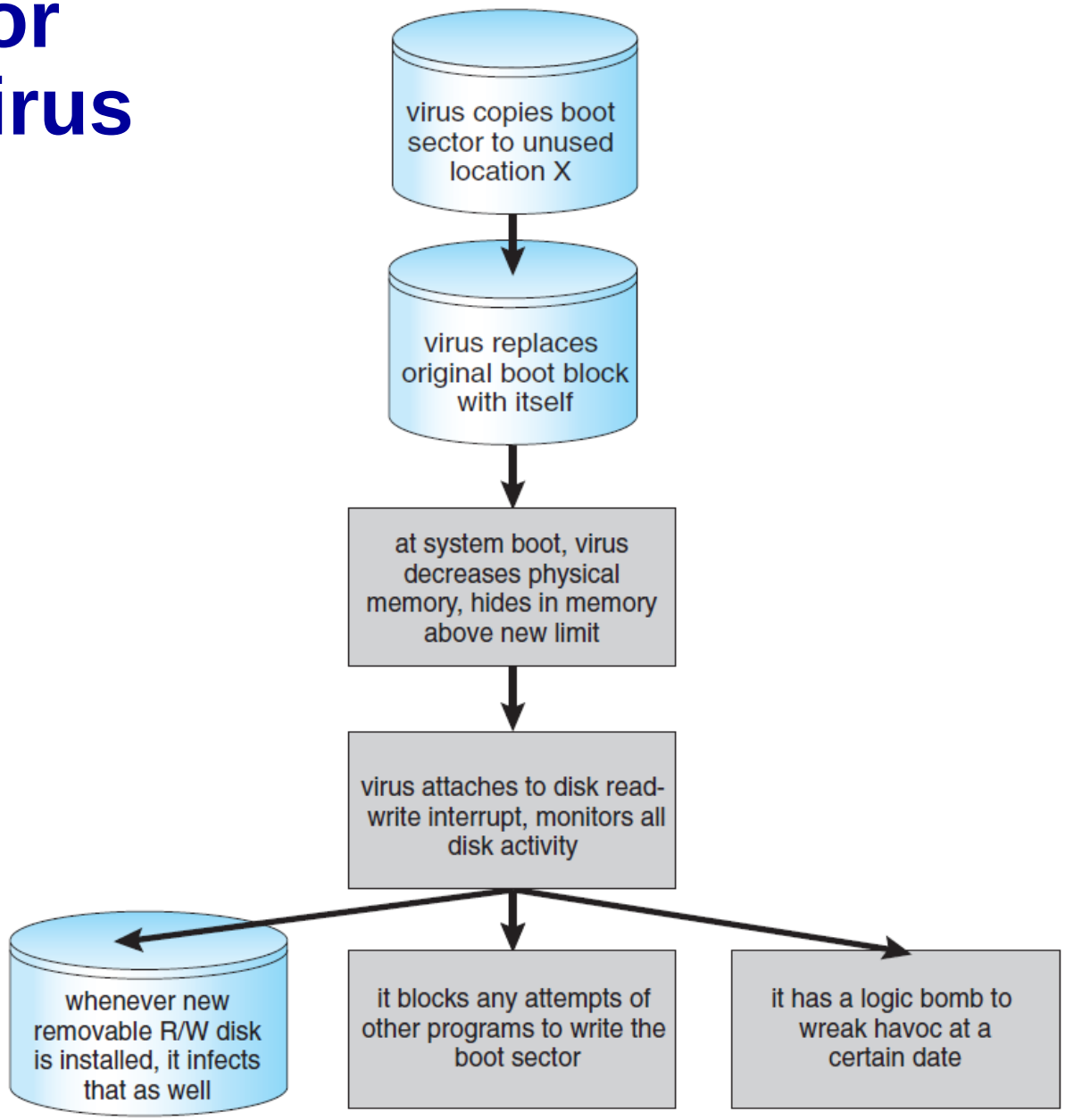
- Viruses
  - Code fragment embedded in legitimate program
  - Self-replicating, designed to infect other computers
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
    - Visual Basic Macro to reformat hard drive

```
Sub AutoOpen()
Dim oFS
  Set oFS =
  CreateObject(''Scripting.FileSystemObject'')
  vs = Shell(''c:command.com /k format
  c:'',vbHide)
End Sub
```

Program Threats (Cont.)

- Virus dropper inserts virus onto the system
- Many categories of viruses, literally many thousands of viruses
  - File / parasitic
  - Boot / memory
  - Macro
  - Source code
  - Polymorphic to avoid having a virus signature
  - Encrypted
  - Stealth
  - Tunneling
  - Multipartite
  - Armored

A Boot-sector Computer Virus



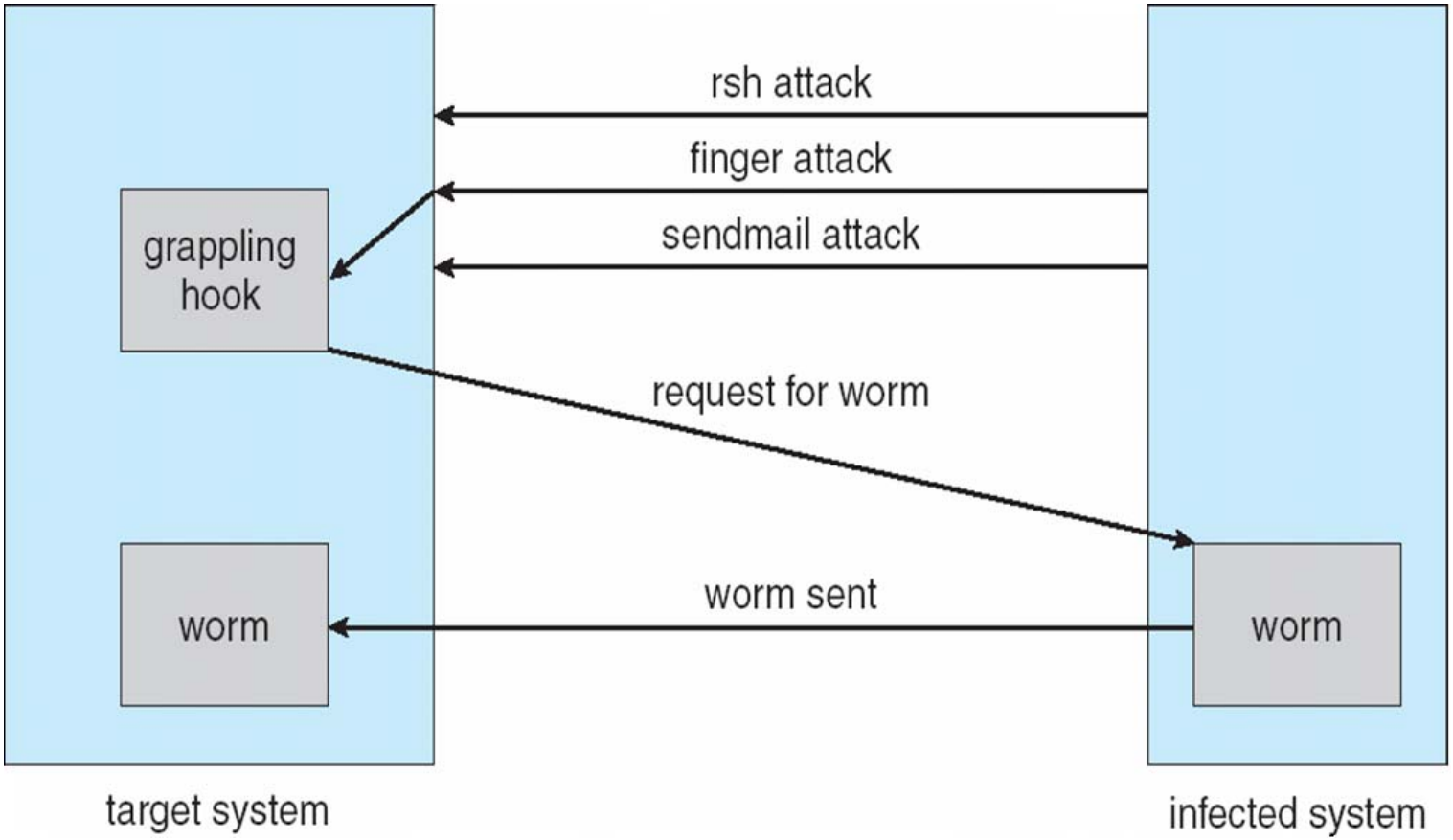
System and Network Threats

- Some systems "open" rather than secure by default. If secure by default:
  - Reduce attack surface
  - But harder to use, more knowledge needed to administer
- Network threats harder to detect, prevent
  - Protection systems weaker
  - More difficult to have a shared secret on which to base access
  - No physical limits once system attached to internet
    - Or on network with system attached to internet
  - Even determining location of connecting system difficult
    - IP address is only knowledge

System and Network Threats (Cont.)

- **Worms** – use spawn mechanism; standalone program
- Internet worm
  - Exploited UNIX networking features (remote access) and bugs in *finger* and *sendmail* programs
  - Exploited trust-relationship mechanism used by *rsh* to access friendly systems without use of password
  - Grappling hook program uploaded main worm program
    - 99 lines of C code (Robert Morris, Jr. at Cornell Univ.)
  - Hooked system then uploaded main code, tried to attack connected systems
  - Also tried to break into other users accounts on local system via password guessing
  - If target system already infected, abort, except for every 7th time

The Morris Internet Worm



System and Network Threats (Cont.)

- **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
  - **nmap** scans all ports in a given IP range for a response
  - **nessus** has a database of protocols and bugs (and exploits) to apply against a 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
    - "Botnet"

Protection Based on Access Control

- System protection approach
  - Computer consists of a collection of objects, HW or SW
  - 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 are allowed to 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
  - Must consider "grain" aspect
    - Rough-grained privilege management easier, simpler, but least privilege now done in large chunks
      - E.g., traditional Unix processes either have abilities of the associated user
    - Fine-grained management more complex, more overhead, but more protective
      - File ACL lists, RBAC

Access Control

- Dictates what types of access are permitted, under what circumstances, and by whom
- Access control policies are generally grouped into the following categories:
  - Discretionary access control (DAC)
    - controls access based on the identity of the requestor and on access rules stating what requestors are (or are not) allowed to do
  - Mandatory access control (MAC)
    - controls access based on comparing security labels with security clearances
  - Role-based access control (RBAC)
    - controls access based on the roles that users have within the system and on rules stating what accesses are allowed to users in given roles

Access Control Policies

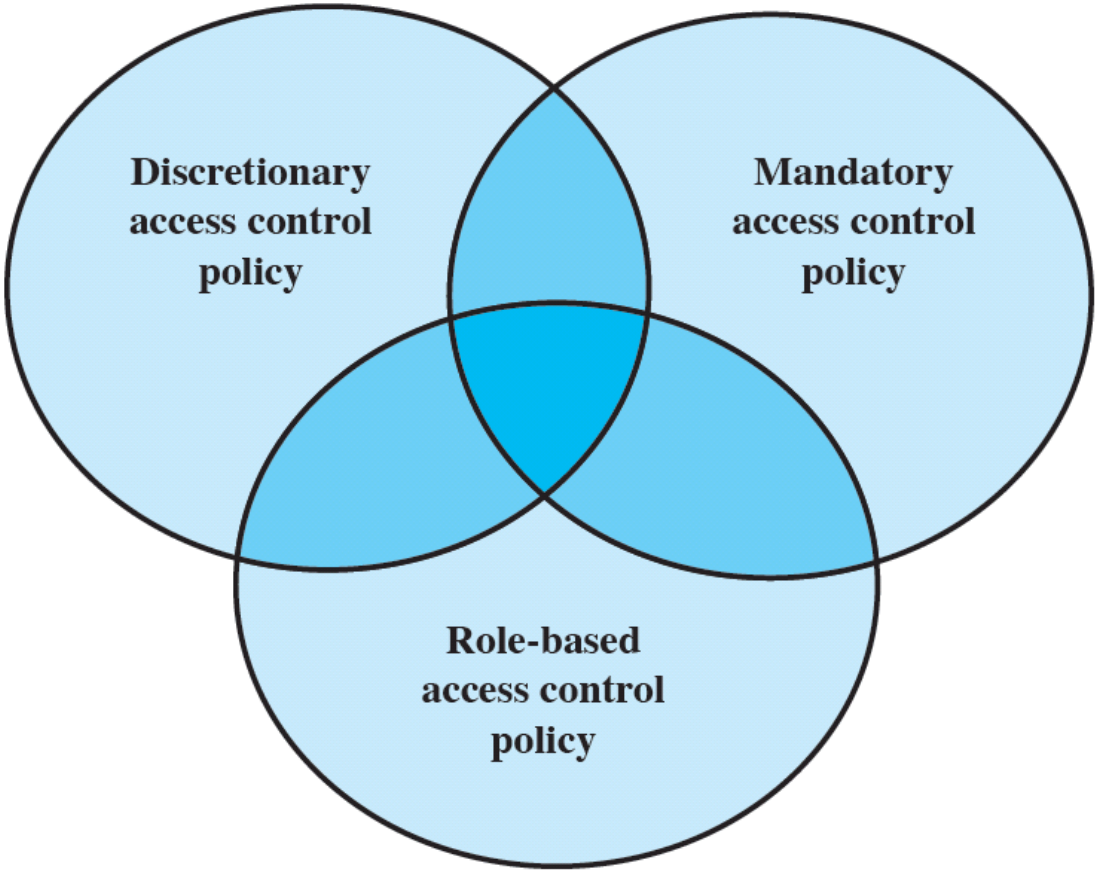


Figure 15.3 Access Control Policies

Extended Access Control Matrix

		OBJECTS								
		subjects			files		processes		disk drives	
		S_1	S_2	S_3	F_1	F_2	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

* - copy flag set

Figure 15.4 Extended Access Control Matrix

Role-Based Access Control

- Based on the roles that users assume in a system rather than the user's identity
- Models define a role as a job function within an organization
- Systems assign access rights to roles instead of individual users
  - in turn, users are assigned to different roles, either statically or dynamically, according to their responsibilities
  - NIST has issued a standard that requires support for access control and administration through roles

Users,

Roles,

Resources

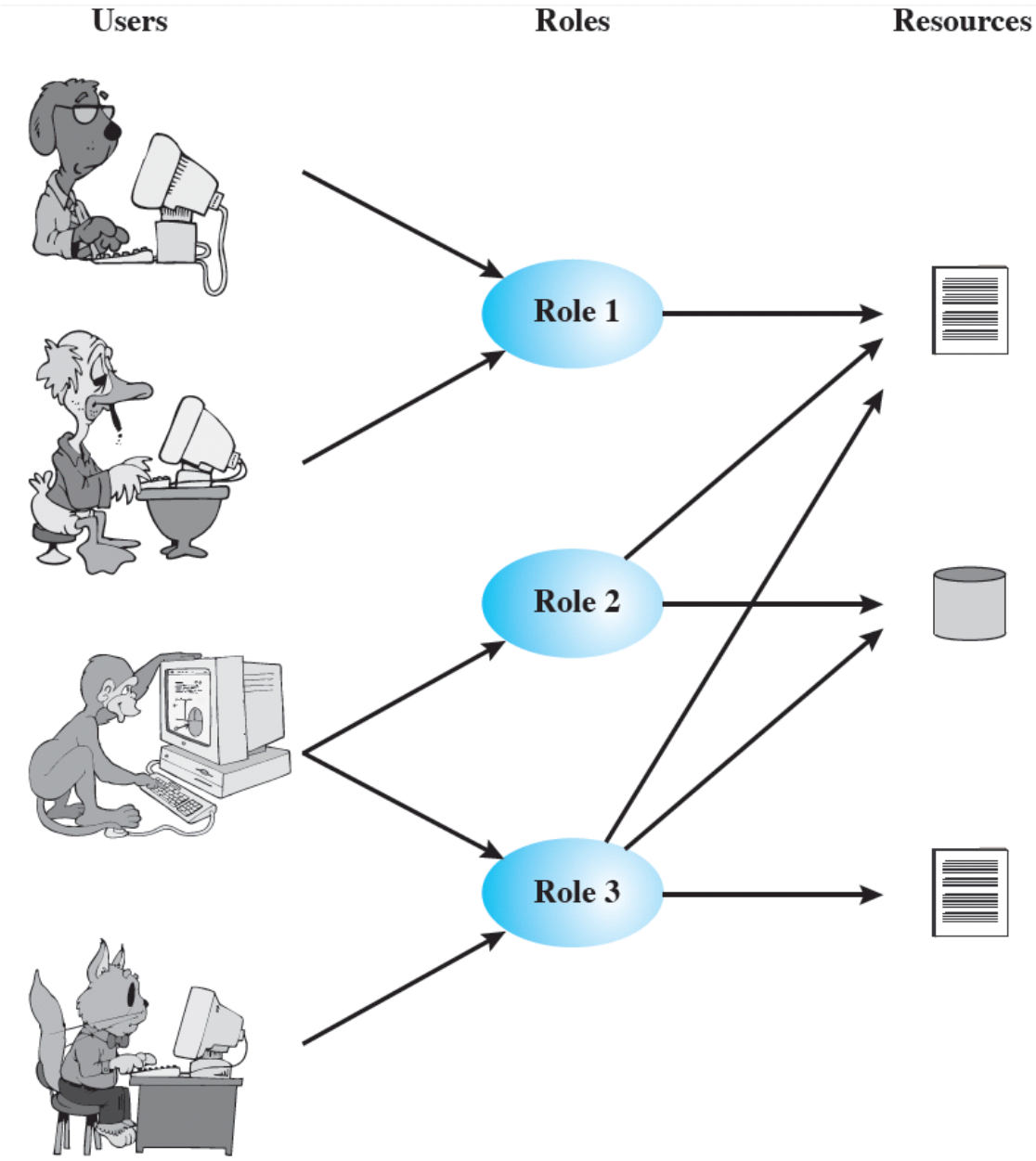
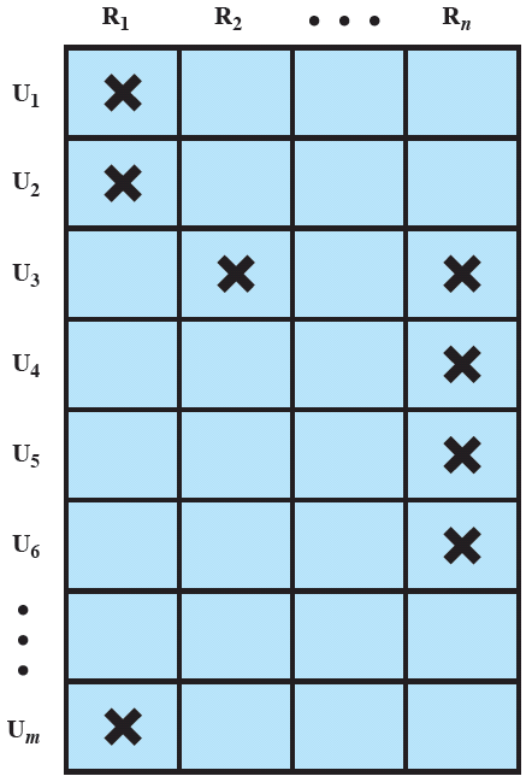


Figure 15.6 Users, Roles, and Resources



Access Control Matrix Representation of RBAC

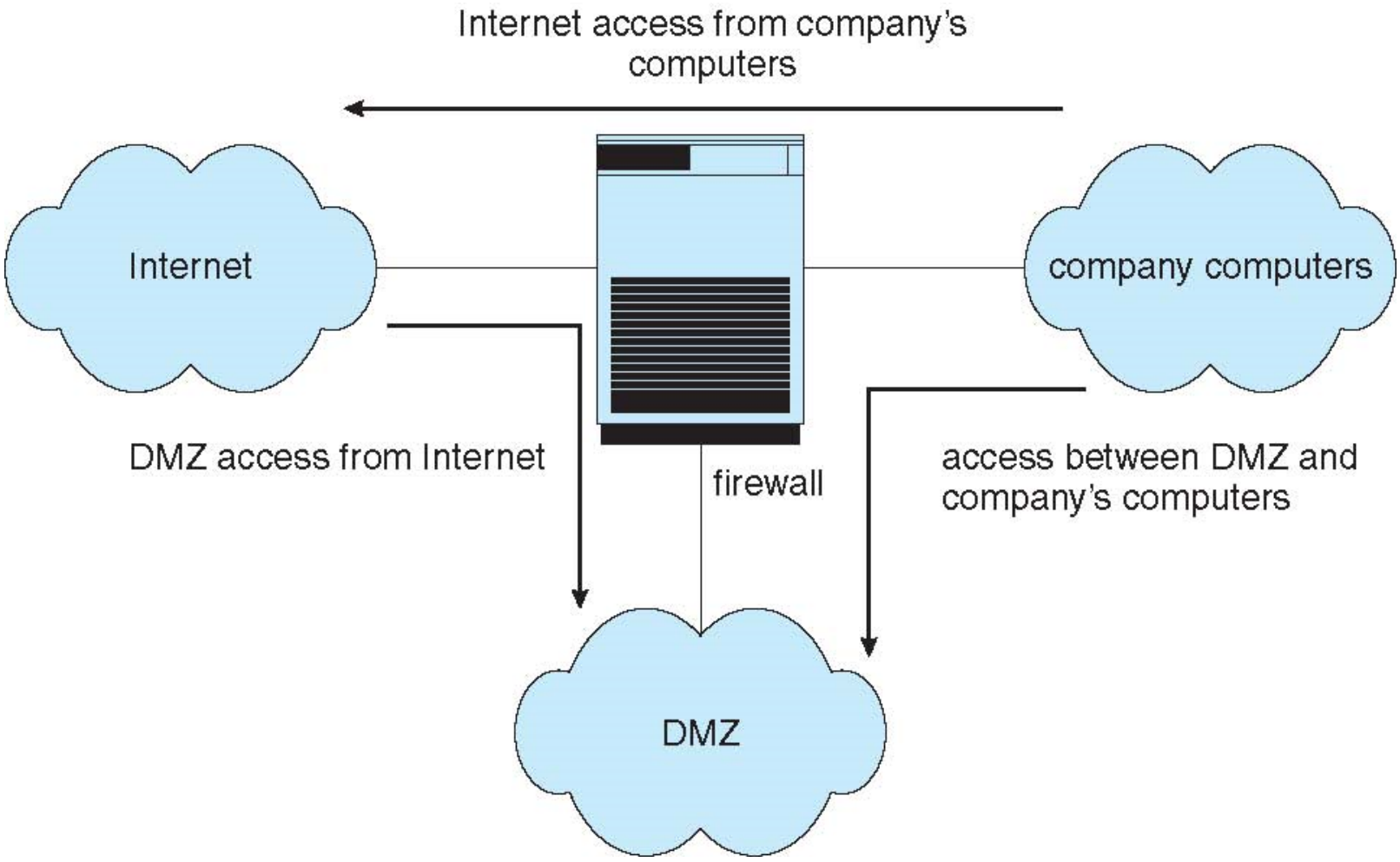
ROLES	OBJECTS								
	R_1	R_2	R_n	F_1	F_1	P_1	P_2	D_1	D_2
R_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
R_2		control		write *	execute			owner	seek *
•									
•									
•									
R_n			control		write	stop			

Figure 15.7 Access Control Matrix Representation of RBAC

Firewalls for System Protection

- A network firewall is placed between trusted and untrusted hosts
  - The firewall limits network access between these two security domains
- Can be tunneled or spoofed
  - Tunneling allows disallowed protocol to travel within allowed protocol (i.e., telnet inside of HTTP)
  - Firewall rules typically based on host name or IP address which can be spoofed
- Personal firewall is software layer on given host
  - Can monitor / limit traffic to and from the host
- Application proxy firewall understands application protocol and can control them (i.e., SMTP)
- System-call firewall monitors all important system calls and apply rules to them (i.e., this program can execute that system call)

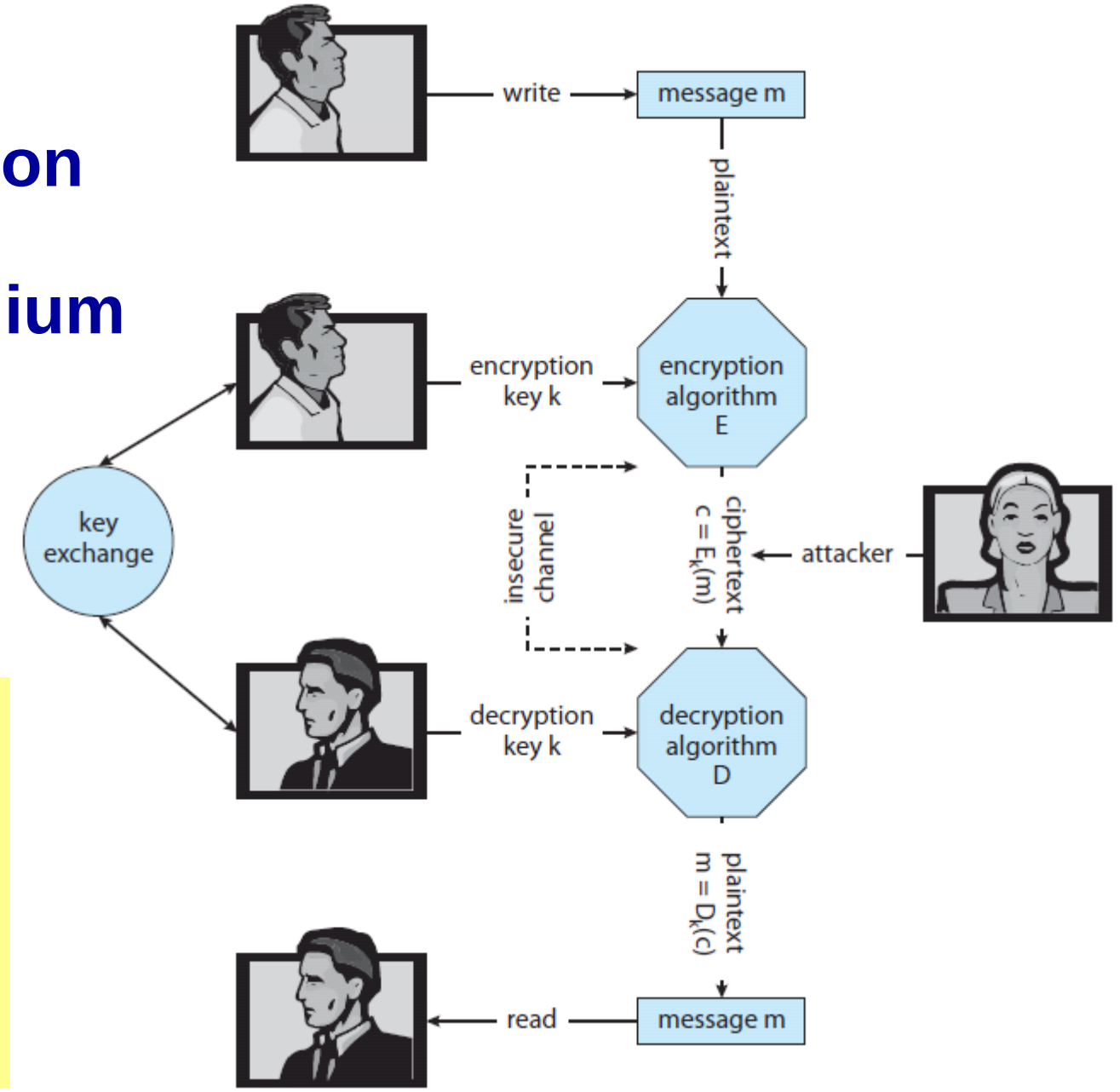
Network Security Through Domain Separation Via Firewall



Cryptography as a Security Tool

- Cryptography
  - basic approach to data and network security
  - encryption(encipherment) ↔ decryption(decipherment)
    - transformation of messages: plain text → cipher text
    - restoration of enciphered messages: cipher text → plain text
- Transformation method
  - defined by function and key
  - function
    - defines an encryption algorithm
    - combines a key and data so that resulting data may be altered as much as possible
  - distribution and storage of keys: important
    - key distribution service

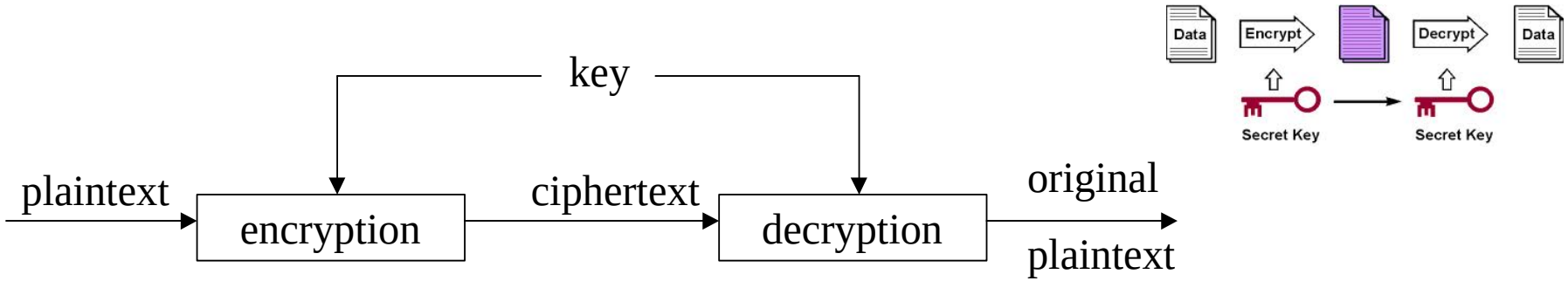
Secure Communication over Insecure Medium (Private Key)



m: message
c: ciphertext
k: key
E: encryption algorithm
E_k: function for generating ciphertext from *m* using *k*
D: decryption algorithm
D_k: function for generating *m* from ciphertext using *k*

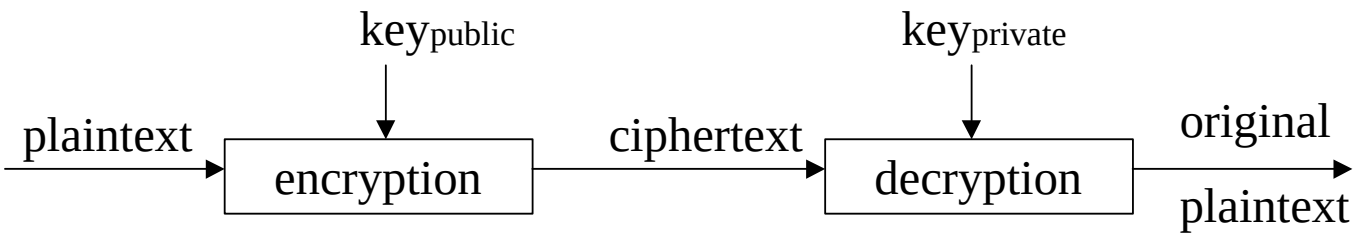
Data Encryption Methods – *Private Key*

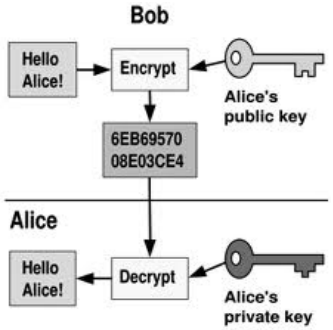
- Secret (or private) key algorithms
  - symmetric encryption algorithms
  - sender and receiver share a single key
  - computationally efficient, but initial key agreement is an issue
  - popular example is DES (Data Encryption Standard) which uses the 56-bit key
    - currently, can be deciphered in 3.5 hours with an inexpensive (< $500,000) computer
  - NIST has improved DES: AES (Advanced Encryption Standard) allows 128-, 192-, and 256-bit keys



Data Encryption Methods – *Public Key*

- Public key algorithms
  - asymmetric encryption algorithms
  - each user has a private key and public key
    - a user keeps private key for decryption and makes public key for encryption, vice versa
    - sender encrypts with public key while receiver decrypts with private key
  - computationally impractical to derive private from public
    - one-way function: y=f(x), hard to determine x from known value of y
  - popular example is RSA and PGP





Example: Online Banking at BoA

- Online banking at Bank of America (BoA)
  - Data transfer using the SSL (Secure Socket Layer) protocol.
  - SSL uses public key cryptography to secure transmissions over the Internet
- Scenario
  - Your browser will send a message via SSL to the bank's server
  - The bank responds by sending a certificate, which contains the bank's public key
  - Your browser authenticates the certificate (agrees that the server is BoA)
  - Then the browser generates a random session key which is used to encrypt data
  - This session key is encrypted using the bank's public key and sent back to the server
  - The bank decrypts this message using its private key, and then uses the session key for the remainder of the communication.
- Secure Socket Layer (SSL) protects data in three key ways:
  - Authentication ensures that you are communicating with the correct server, preventing another computer from impersonating BoA
  - Encryption scrambles transferred data.
  - Data integrity verifies that the information sent to BoA wasn't altered during the transfer

https://www.bankofamerica.com/onlinebanking/online-banking-security-faqs.go

User Authentication

- Crucial to identify user correctly, as protection systems depend on user ID
- User identity most often established through passwords, can be considered a special case of either keys or capabilities
- Passwords must be kept secret
  - Frequent change of passwords
  - History to avoid repeats
  - Use of "non-guessable" passwords
  - Log all invalid access attempts (but not the passwords themselves)
  - Unauthorized transfer
- Passwords may also either be encrypted or allowed to be used only once

Authentication Methods

- Passwords: Encrypt to avoid having to keep secret
  - But keep secret anyway (i.e. Unix uses superuser-only readably file `/etc/shadow`)
  - Use algorithm easy to compute but difficult to invert
  - Only encrypted password stored, never decrypted
  - Add "salt" to avoid the same password being encrypted to the same value
- One-time passwords
  - Use a function based on a seed to compute a password, both user and computer
  - Hardware device / calculator / key tab to generate the password
    - Changes very frequently
- Biometrics
  - Some physical attribute (fingerprint, hand scan, retina, etc.)
- Two-factor and multi-factor authentication
  - "Factors": hardware-based (security token, smart card, etc.), knowledge-based (password, question-answer, etc.), biometric
  - Need two or more factors for authentication
    - i.e. USB "dongle", biometric measure, and password

Implementing Security Defenses

- **Defense in depth** is most common security theory – multiple layers of security
- Security policy describes what is being secured
- Vulnerability assessment compares real state of system / network compared to security policy
- Intrusion detection endeavors to detect attempted or successful intrusions
  - **Signature-based** detection spots known bad patterns
  - **Anomaly detection** spots differences from normal behavior
    - Can detect **zero-day** attacks
  - **False-positives** and **false-negatives** a problem
- Virus protection, vaccine
- Auditing, accounting, and logging of all or specific system or network activities

Example: Windows 7

- Security is based on user accounts
  - Each user has unique security ID
  - Login to ID creates **security access token**
    - Includes security ID for user, for user's groups, and special privileges
    - Every process gets copy of token
    - System checks token to determine if access allowed or denied
- Uses a subject model to ensure access security
  - A subject tracks and manages permissions for each program that a user runs
- Each object in Windows has a security attribute defined by a security descriptor
  - For example, a file has a security descriptor that indicates the access permissions for all users

Summary

- 보안
  - 컴퓨팅 자원이 의도된대로 사용됨
  - 안전한 상태 혹은 규정의 위반, 그 유형
  - 위협 및 그 유형
  - 공격 및 그 유형
  - 보안조치의 레벨
- 프로그램 위협
  - 트로이 목마
  - 트랩 도어
  - 논리 폭탄
  - 스택/버퍼 오버플로우
  - 바이러스
- 시스템과 네트워크 위협
  - 웜
  - 포트 스캐닝
  - 서비스 거부 (DoS)

- The security problem
  - computing resources used as intended
  - breach (violation) of secure state or rules, its types
  - threats and their types
  - attacks and their types
  - levels of security measures
- Program threats
  - Trojan horse
  - trap door
  - logic bomb
  - stack and buffer overflow
  - viruses
- System and network threats
  - worms
  - port scanning
  - denial of service (DoS)

Summary (Cont.)

- 접근제어에 의한 시스템 보호
  - 접근제어 정책
  - 접근행렬
  - 역할 기반의 접근제어
  - 방화벽 기능: 패킷 여과, 프락시, 감시
- 보안 도구로서의 암호학
  - 데이터의 암호화: 함수와 키 이용
  - 비밀키 알고리즘
  - 공개키 알고리즘
- 사용자의 인증
  - 아이디 - 비밀번호
  - 생체정보, 하드웨어 기반, 지식 기반, 이중 요소(two-factor) 인증
- 보안 기법의 구현
  - 보안 정책, 취약성 평가, 침입탐지, 백신, 등
- 예: Windows 7
  - id-pw 로 로그인 → 보안 토큰 생성
  - subject model 사용
  - 객체에 대한 보안 디스크립터

- Access control-based protection
  - access control policy
  - access matrix
  - Role-Based Access Control (RBAC)
  - Functions of firewalls: packet filtering, proxy, monitoring
- Cryptography as a security tool
  - data encryption: use function and key
  - secret-key (private-key) encryption
  - public-key encryption
- User authentication
  - id – password
  - biometrics, hardware-based, knowledge-based, two-factor authentication
- Implementing security defenses
  - security policy, vulnerability assessment, intrusion detection, etc.
- An example: Windows 7
  - login with id – pw → create security token
  - use subject model
  - security descriptor for objects