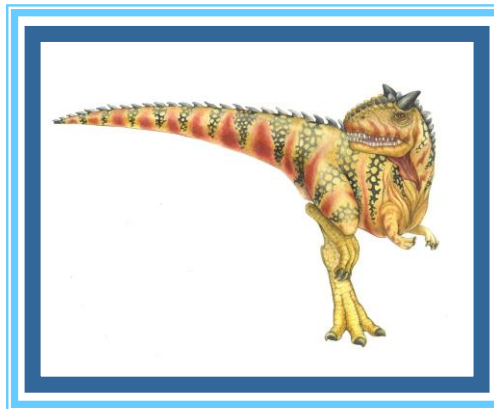
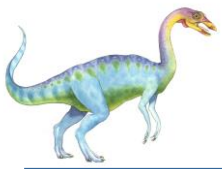


Chapter 15: Security

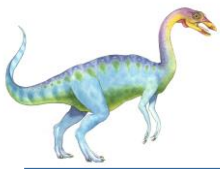




Chapter 15: Security

- The Security Problem
- Program Threats
- System and Network Threats
- Cryptography as a Security Tool





Objectives

- To discuss security threats and attacks
- To explain the fundamentals of encryption, authentication, and hashing
- To examine the uses of cryptography in computing
- To describe the various countermeasures to security attacks





The 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





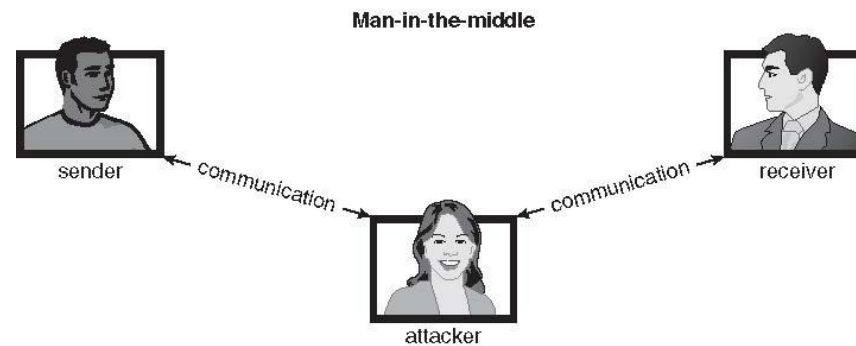
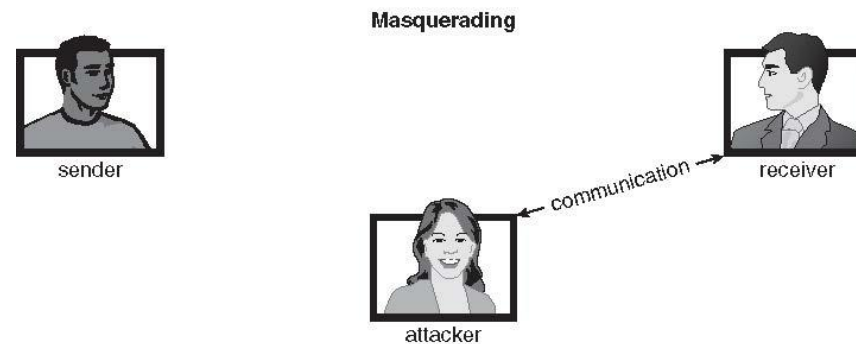
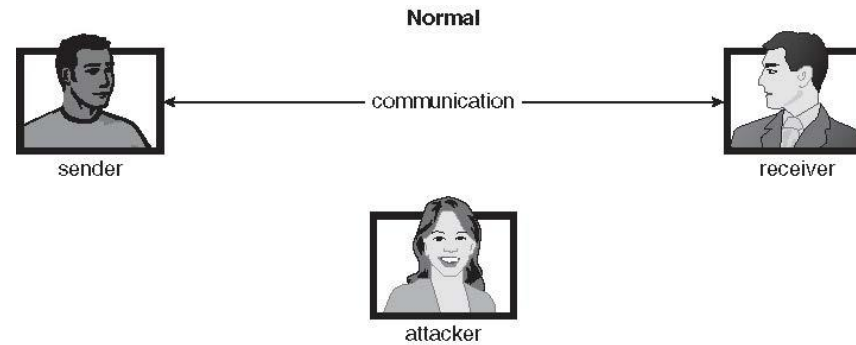
Security Violation Methods

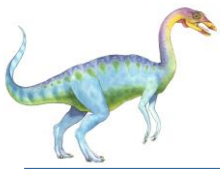
- **Masquerading** (breach **authentication**)
 - Pretending to be an authorized user to escalate privileges
- **Replay attack**
 - As is or with **message modification**
- **Man-in-the-middle attack**
 - Intruder sits in data flow, masquerading as sender to receiver and vice versa
- **Session hijacking**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





Standard Security Attacks





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?





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
 - Could be included in a compiler
 - How to detect them?





Program Threats (Cont.)

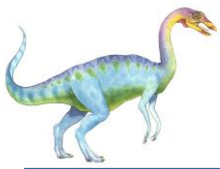
■ Logic Bomb

- 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





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
```

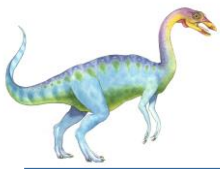




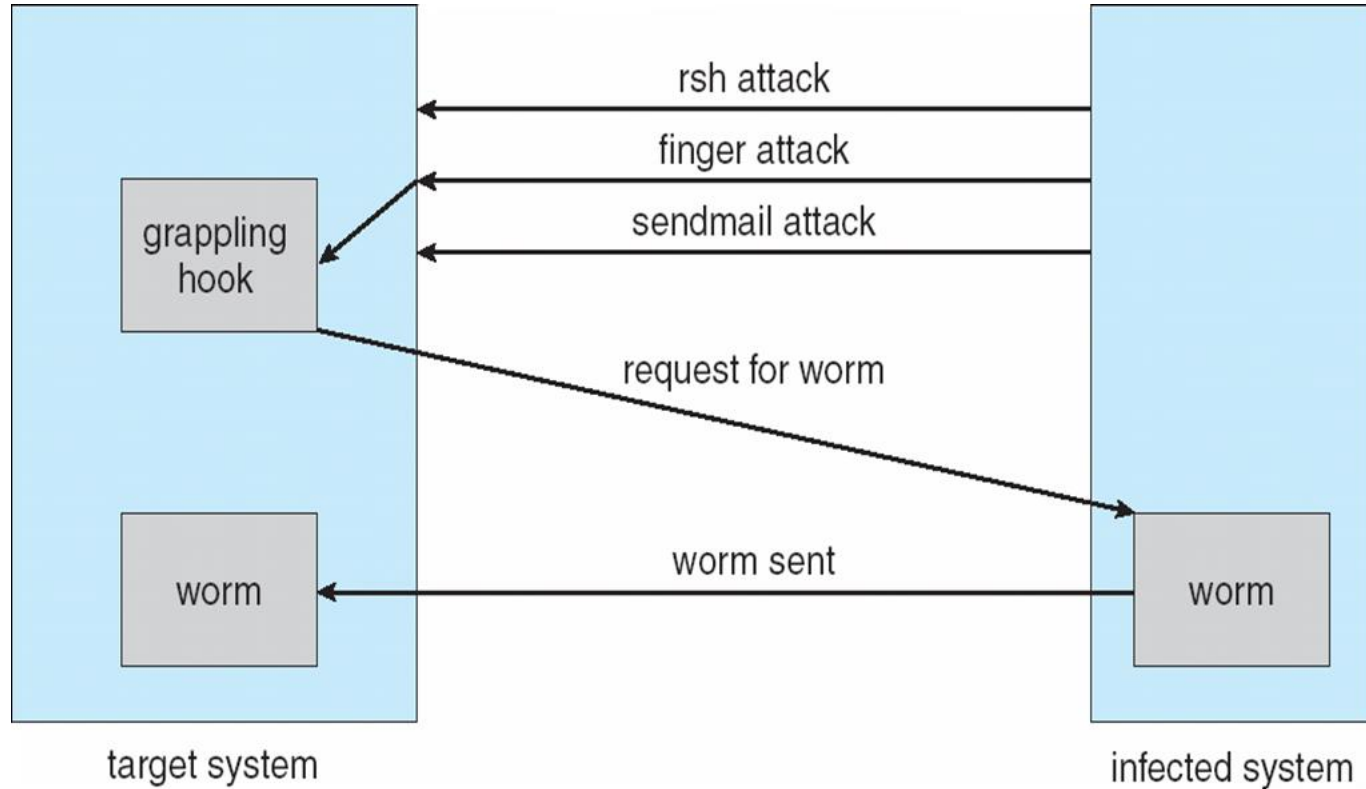
System and Network Threats

- **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
 - 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.)

■ 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 the start of the IP-connection handshake (SYN)
 - ▶ How many started-connections can the OS handle?
- Consider traffic to a web site
 - ▶ How can you tell the difference between being a target and being really popular?
- Accidental – CS students writing bad `fork()` code
- Purposeful – extortion, punishment

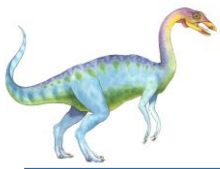




Cryptography

- Means to constrain potential senders (*sources*) and / or receivers (*destinations*) of *messages*
 - Based on secrets (**keys**)
 - Enables
 - ▶ Confirmation of source
 - ▶ Receipt only by certain destination
 - ▶ Trust relationship between sender and receiver





Encryption

- Constrains the set of possible receivers of a message
- **Encryption** algorithm consists of
 - Set K of keys
 - Set M of Messages
 - Set C of ciphertexts (encrypted messages)
 - A function $E : K \rightarrow (M \rightarrow C)$. That is, for each $k \in K$, E_k is a function for generating ciphertexts from messages
 - ▶ Both E and E_k for any k should be efficiently computable functions
 - A function $D : K \rightarrow (C \rightarrow M)$. That is, for each $k \in K$, D_k is a function for generating messages from ciphertexts
 - ▶ Both D and D_k for any k should be efficiently computable functions

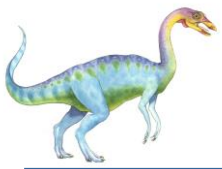




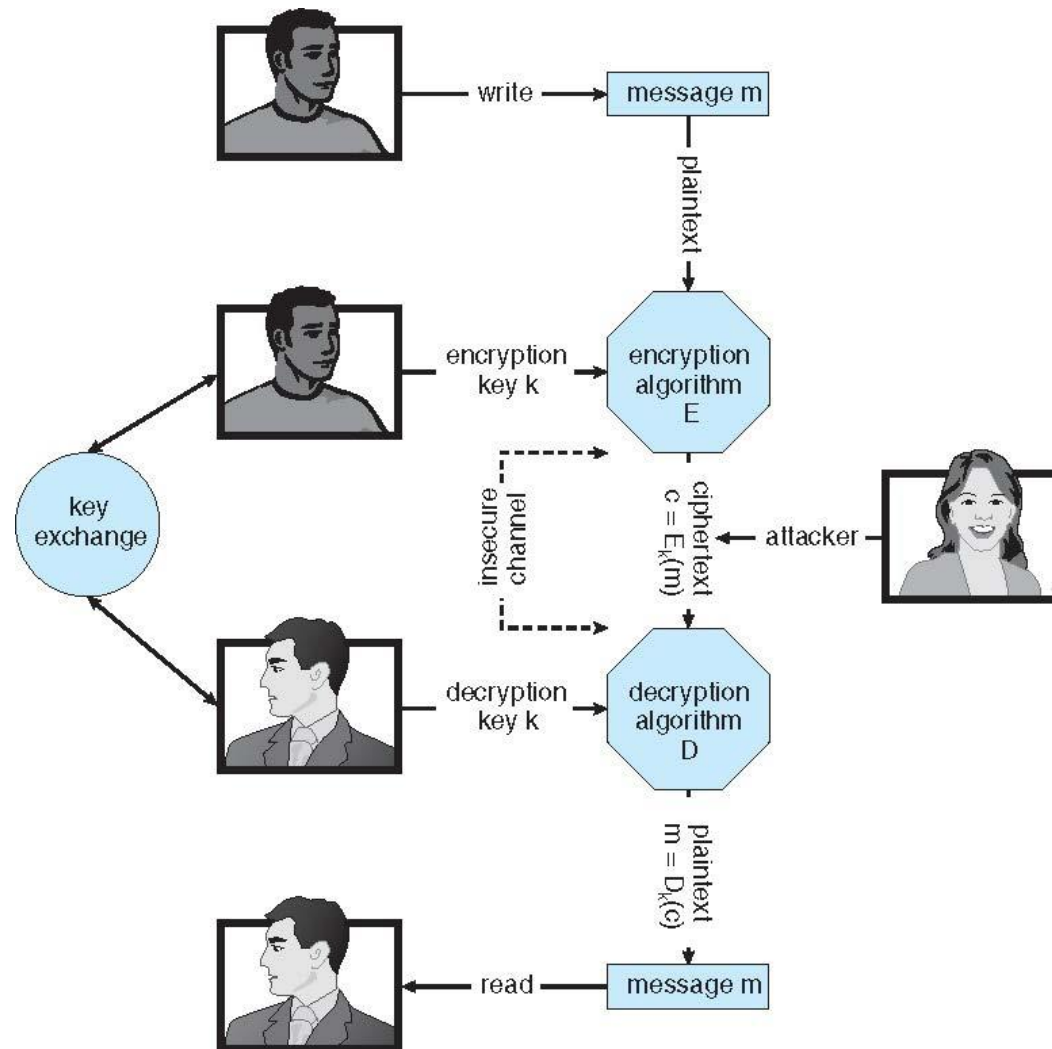
Symmetric Encryption

- Same key used to encrypt and decrypt
 - Therefore k must be kept secret
- DES was most commonly used symmetric block-encryption algorithm (created by US Govt)
 - Encrypts a block of data at a time
 - Keys too short so now considered insecure
- Triple-DES considered more secure
 - Algorithm used 3 times using 2 or 3 keys
 - For example $c = E_{k3}(D_{k2}(E_{k1}(m)))$
- 2001 NIST adopted new block cipher - Advanced Encryption Standard (**AES**)
 - Keys of 128, 192, or 256 bits, works on 128 bit blocks
- RC4 is most common symmetric stream cipher, but known to have vulnerabilities
 - Encrypts/decrypts a stream of bytes (i.e., wireless transmission)
 - Key is a input to pseudo-random-bit generator
 - ▶ Generates an infinite **keystream**





Secure Communication over Insecure Medium





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
- Must be an encryption scheme that can be made public without making it easy to figure out the decryption scheme
 - Most common is **RSA** block cipher
 - Efficient algorithm for testing whether or not a number is prime
 - No efficient algorithm is known for finding the prime factors of a number





Asymmetric Encryption (Cont.)

- Formally, it is computationally infeasible to derive $k_{d,N}$ from $k_{e,N}$, and so k_e need not be kept secret and can be widely disseminated
 - k_e is the **public key**
 - k_d is the **private key**
 - N is the product of two large, randomly chosen prime numbers p and q (for example, p and q are 512 bits each)
 - Encryption algorithm is $E_{k_e,N}(m) = m^{k_e} \bmod N$, where k_e satisfies $k_e k_d \bmod (p-1)(q-1) = 1$
 - The decryption algorithm is then $D_{k_d,N}(c) = c^{k_d} \bmod N$





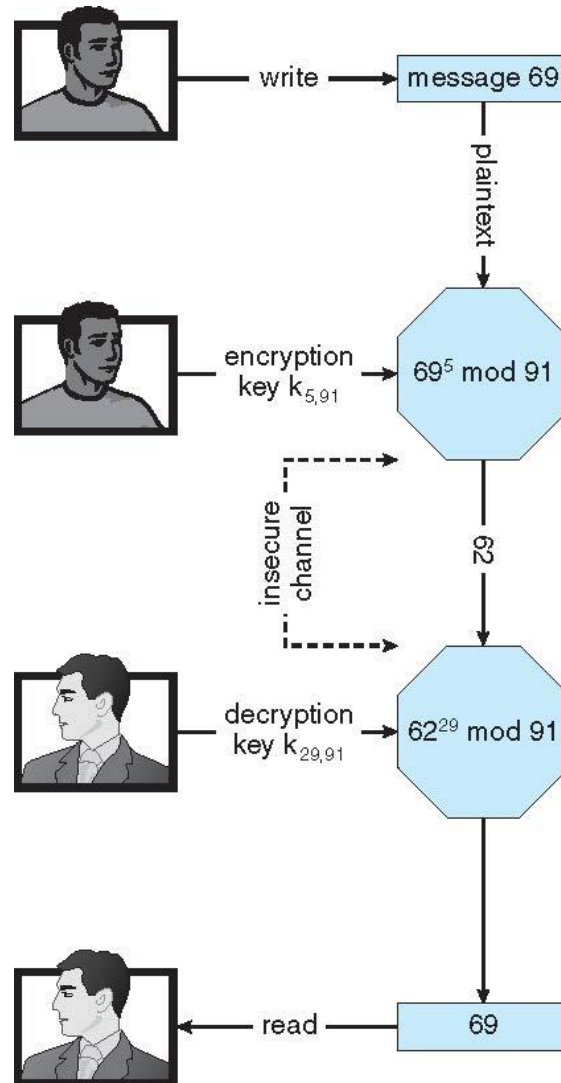
Asymmetric Encryption Example

- For example. make $p = 7$ and $q = 13$
- We then calculate $N = 7 * 13 = 91$ and $(p-1)(q-1) = 72$
- We next select k_e relatively prime to 72 and < 72 , yielding 5
- Finally, we calculate k_d such that $k_e k_d \bmod 72 = 1$, yielding 29
- We now have our keys
 - Public key, $k_{e,N} = 5, 91$
 - Private key, $k_{d,N} = 29, 91$
- Encrypting the message 69 with the public key results in the cyphertext 62
- Cyphertext can be decoded with the private key
 - Public key can be distributed in cleartext to anyone who wants to communicate with holder of public key





Encryption using RSA Asymmetric Cryptography



End of Chapter 15

