

# IT 4823

## Information Security Administration

### Review for Final Exam



Notice: This session is  
being recorded.

Some lecture slides prepared by Dr Lawrie Brown for *Computer Security: Principles and Practice*.



Copyright © 2016 by Bob Brown



# Coming Attractions

- **Final Exam: July 26, 2:00 – 4:00**  
Study the prior exams!
- No office hours after today
- Final exam will not be returned.
- I will put your exam grades into D2L.
- Final course grades are released **only** by the Registrar.

# What Does “Secure” Mean?

- “A system that does what it is intended to do and *nothing else.*” -- *Pfleeger*
- “A computer is secure if you can depend on it and its software to behave as you expect.”  
– *Garfinkle and Spafford*

# Properties of Information Security

- **Confidentiality**
  - Keeping data and resources hidden from unauthorized personnel
- **Integrity**
  - Data integrity (integrity)
  - Origin integrity (authentication)
- **Availability**
  - Enabling access to data and resources when and where they are needed.

**Your book calls this the Security Requirements Triad.**

# Specifying Security

- Security is specified by *policy*.
- Policies are organizational laws; they state what must happen, what may happen, and what must not happen.
- A system the behavior of which does not violate organizational policies is, by definition, secure.
- So, effective security is impossible in the absence of policy.

# Organizational Security Policy

- “A formal statement of rules by which people given access to organization’s technology and information assets must abide”
- Policy defines “security.”

# Organizational Security Policy

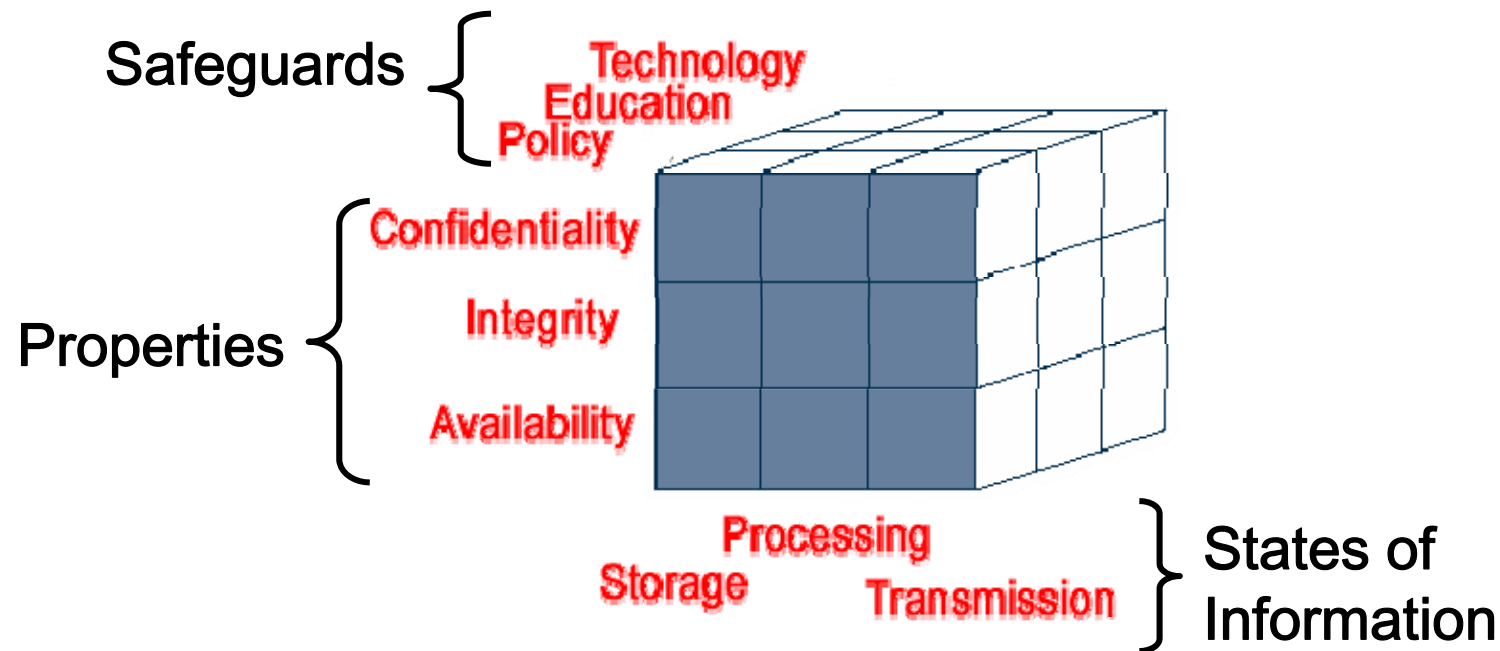
- Every organization needs a written security policy document...
- ...to define acceptable behavior, expected practices, and responsibilities
  - makes clear what is protected and why
  - articulates security procedures / controls
  - states responsibility for protection
  - provides basis to resolve conflicts
- Must reflect executive security decisions
  - protect information
  - comply with law
  - meet organizational goals

# Developing the Security Policy

- First examine organization's IT security:
  - objectives - wanted IT security outcomes
  - strategies - how to meet objectives
  - security policies - identify what needs to be done
- Maintained and updated regularly
  - using periodic security reviews
  - reflect changing technical / risk environments
- Examine role of IT systems in organization



# McCumber Security Model



*National Security Telecommunications and Information Systems  
Security Committee*

# Assets to be Protected

- Hardware
- Software
- Data
- Infrastructure (including communications facilities)
- People

# The Attacker's Triad: DAD

- Disclosure: compromises confidentiality
  - Outside attackers
  - Insiders
  - Programming or other errors
- Alteration: compromises integrity
  - Accidental or malicious alteration
  - Programming or equipment failure
- Denial: compromises availability
  - Deliberate attacks
  - Failures of systems or environment

# Vulnerabilities, Threats, Risks

- **Vulnerability:** a weakness that could allow a system to enter a state not permitted by policy.
- **Exploit:** a mechanism for taking advantage of a vulnerability.
- **Threat:** a circumstance that could allow a vulnerability to be taken advantage of.
- **Risk:** the circumstance that both a threat and a corresponding vulnerability exist. Risk is the *probability* of the threat being realized.

# Goals of Information Security

- **Prevention**
  - Prevent attackers from violating security policy
- **Detection**
  - Detect attackers' violation of security policy
- **Response and Recovery**
  - Stop attack, assess and repair damage
  - Continue to function correctly even if attack succeeds
  - Return system to a state consistent with policy

# Trust and Assumptions

- Underlie *all* aspects of security
- Policies are assumed to:
  - Unambiguously partition system states
  - Correctly capture security requirements
- Mechanisms are assumed to:
  - Enforce policy
  - Work correctly

# Trust Comes From Assurance

- Specification assurance
  - Requirements analysis
  - Statement of desired functionality
- Design assurance
  - How system will meet specification
- Implementation assurance
  - Programs/systems carry out the design
  - A system does what is was designed to do...
  - *...and nothing else!*

# Cryptography

- **Cryptography:** making secret ciphers
  - crypt (*kryptós*): hidden, secret
  - graph (*gráphein*): writing
- **Cryptology:** the art and science of making and breaking secret ciphers.
- **Cryptanalysis:** analysis of cipher text (and other information) to attempt to derive plain text.



# Goals of Cryptosystems

- Protection of **confidentiality**
- Protection of **integrity**
  - Message integrity
  - Non-repudiation (origin integrity)
  - Authentication (origin integrity)
  - Can detect, but not prevent, integrity breaches
- Do not address availability.

# Cryptography = Algorithm + Key

**Algorithm:** A procedure for transforming plain text and a key to cipher text. The algorithm is generally assumed to be well-known. (Kerckhoffs' principle.)

**Key:** A secret, often random, collection of data that is combined with a plain text message to produce cipher text.

# Three Major Areas

- **Symmetric key (classical) cryptography:** Security depends upon a secret (the key) shared between sender and recipient
- **Cryptographic hash functions:** transform an input into a fixed-size output with properties that depend on the function.
- **Public key cryptography:** Depends upon a pair of keys, one public and one private

# Symmetric Key Cryptography

Encrypt by applying the key to the plaintext using an algorithm.

Decrypt by reversing the process using the *same* key and the inverse algorithm.

Mnemonic: *symmetric, shared key* and *secret key* are all the same thing and they all start with S.

# Key Strength and Security

Encryption is a mathematical operation

*The strength is in the key, not the algorithm!*

(Assume that the bad guys know the algorithm.

That is *Kerckhoffs' Principle*.)

However, the algorithm must be free from

“shortcut attacks.”

Keys should be *long* and *random*.

Keys are applied over blocks of data. Brute force

attack: try all possible key combinations. 40-bit

key:  $2^{40}$  combinations.

# Computational vs. Absolute Security

- *Absolutely secure*: There is no way to recover the plaintext from the cipher text without the key. (And the algorithm)

The one-time pad is provably secure.

- *Computationally secure*: The time to recover the plaintext from the cipher text is so great that the message has little or no value by the time it can be decrypted.

How long does “*We attack at dawn!*” need to remain secure?

# Symmetric (Secret Key) Encryption

- DES 56 bits (1973 – obsolete!)
- 3DES 3 keys, multiple encryption (1998)
- AES (Rijndael) 128 (and more) bit keys (2001)
- Others

# Public Key Cryptography

- With public key cryptography, one creates a pair of keys,  $k_v$  (private) and  $k_b$  (public).
- A ciphertext message,  $C$  is formed from a plaintext message  $M$ :  $C = f(k_b, M)$
- Deriving  $M$  from  $C$  and  $k_b$  is intractable.
- Deriving  $k_v$  from  $k_b$  is intractable.
- However,  $M = f(k_v, C)$  is straightforward to compute, but computationally intensive.



# Computational Effort

- Public key encryption can be 10,000 times more computational work than secret key encryption.
- Why? We are dealing with difficult arithmetic (exponentiation) on very large (hundreds of digits) numbers.
- The strength of public key cryptography lies in the difficulty of finding prime factors of very large numbers.

# Public Key Digital Signature

Public key cryptography works both ways; the keys are cryptographic inverses of one another!

Bob encrypts a message with his private key

*Anyone* can decrypt it using Bob's public key.

Only Bob could have encrypted it, so, it is "signed" by Bob. *Bob Brown*

*Uhhh*, but *anyone* can decrypt it; there is no confidentiality. (This is exactly analogous to a handwritten signature.)

# Cryptographic Checksums

- Idea: Detect tampering with messages.
- Needed: A function that “characterizes” a message in some way.
- A strong hash function:
  - $h(m)$  is easy to compute
  - Deriving  $m$  from  $h(m)$  is infeasible
  - Given  $m$ , finding  $m'$  such that  $h(m') = h(m)$  is infeasible (protects integrity by resisting collision attacks)

# Digital Signature Improved

- Bob computes a “digest” of a message using a hash function; this is smaller than the message and of fixed size, but has the property that changing the message changes the computed digest.
- Bob encrypts the digest with his private key
- Anyone can decrypt the digest using Bob’s public key, but only Bob could have encrypted it. The message is digitally signed.
- If the digest matches the message, the message has not been altered.

# Digital Signature

**Message**

**Message Digest**

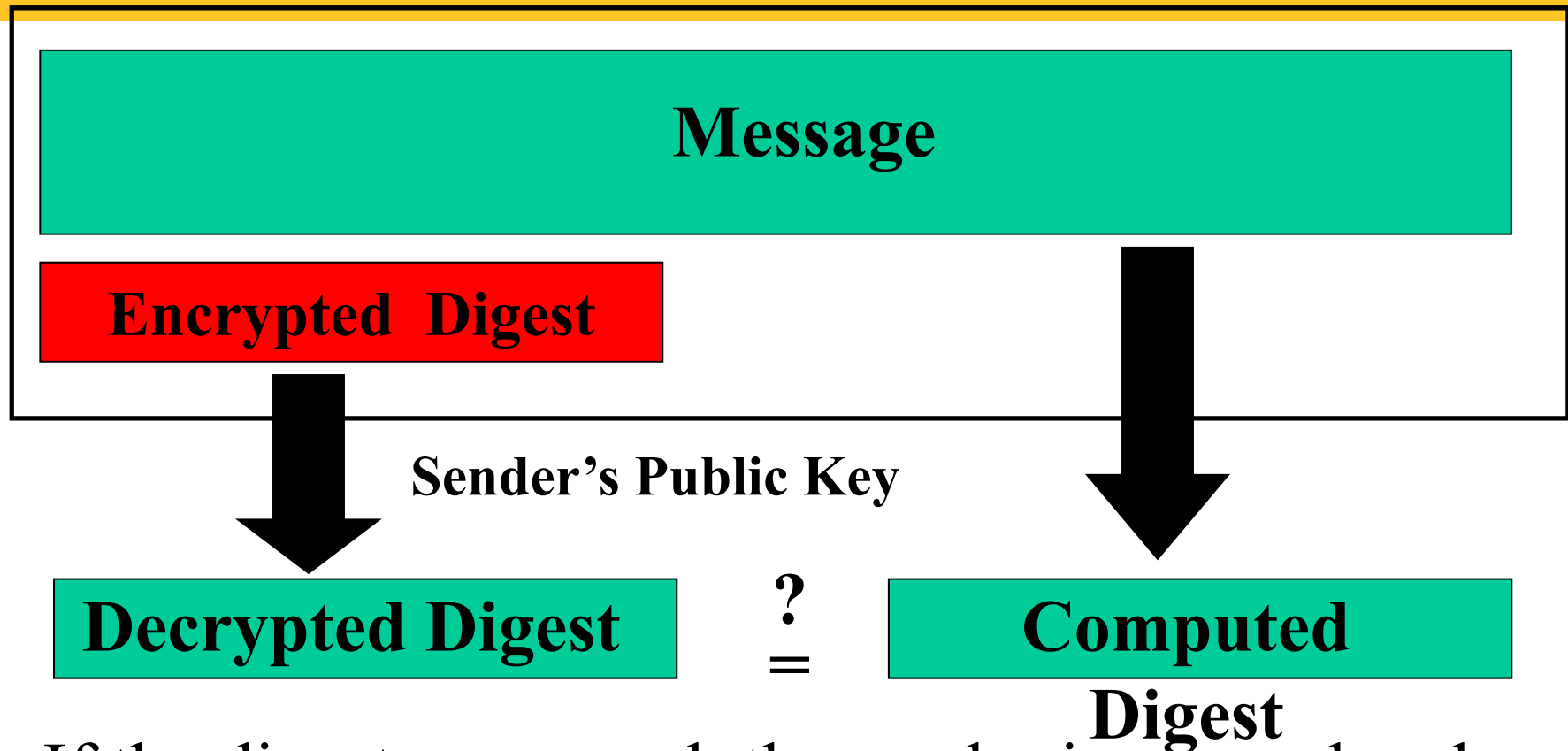
**Encrypt with Sender's Private Key**



**Message**

**Encrypted Digest**

# Digital Signature

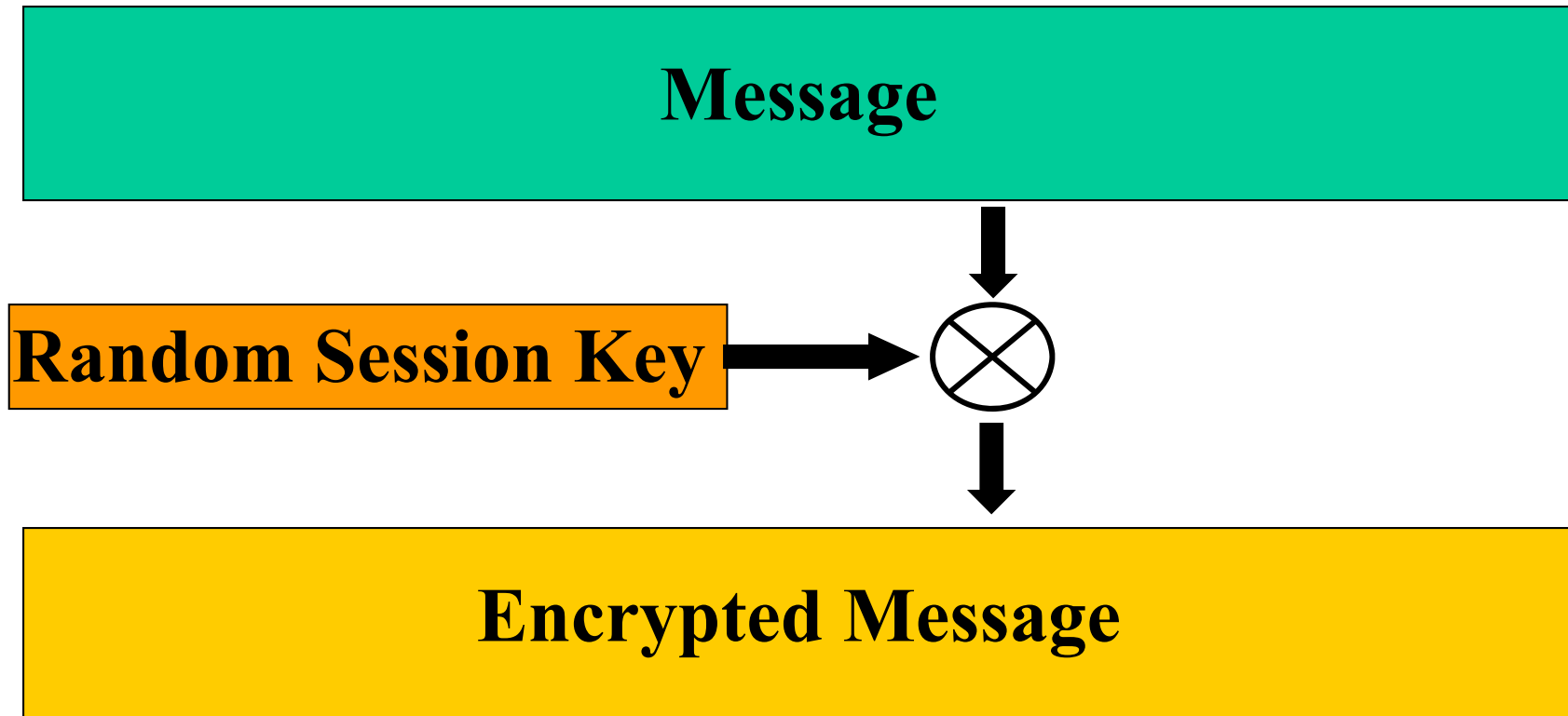


If the digests are equal, the sender is assured and no tampering has occurred.

# Public Keys with Less Computation

- Public key encryption is time-consuming... up to 10,000 times more work than secret key encryption.
- Solution: generate a random, one-time secret key.
- Encrypt the message with the secret key.
- Encrypt the secret key with the recipient's public key.
- Send encrypted message and encrypted key.

# Hybrid Cryptography

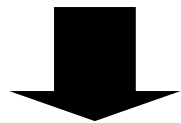


The session key is a shared key; computation for encryption is (relatively) easy.



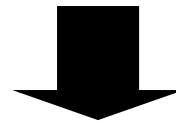
# Packaging a Hybrid Crypto Message

**Message Digest**



Encrypt with  
Sender's *Private* Key

**Session Key**



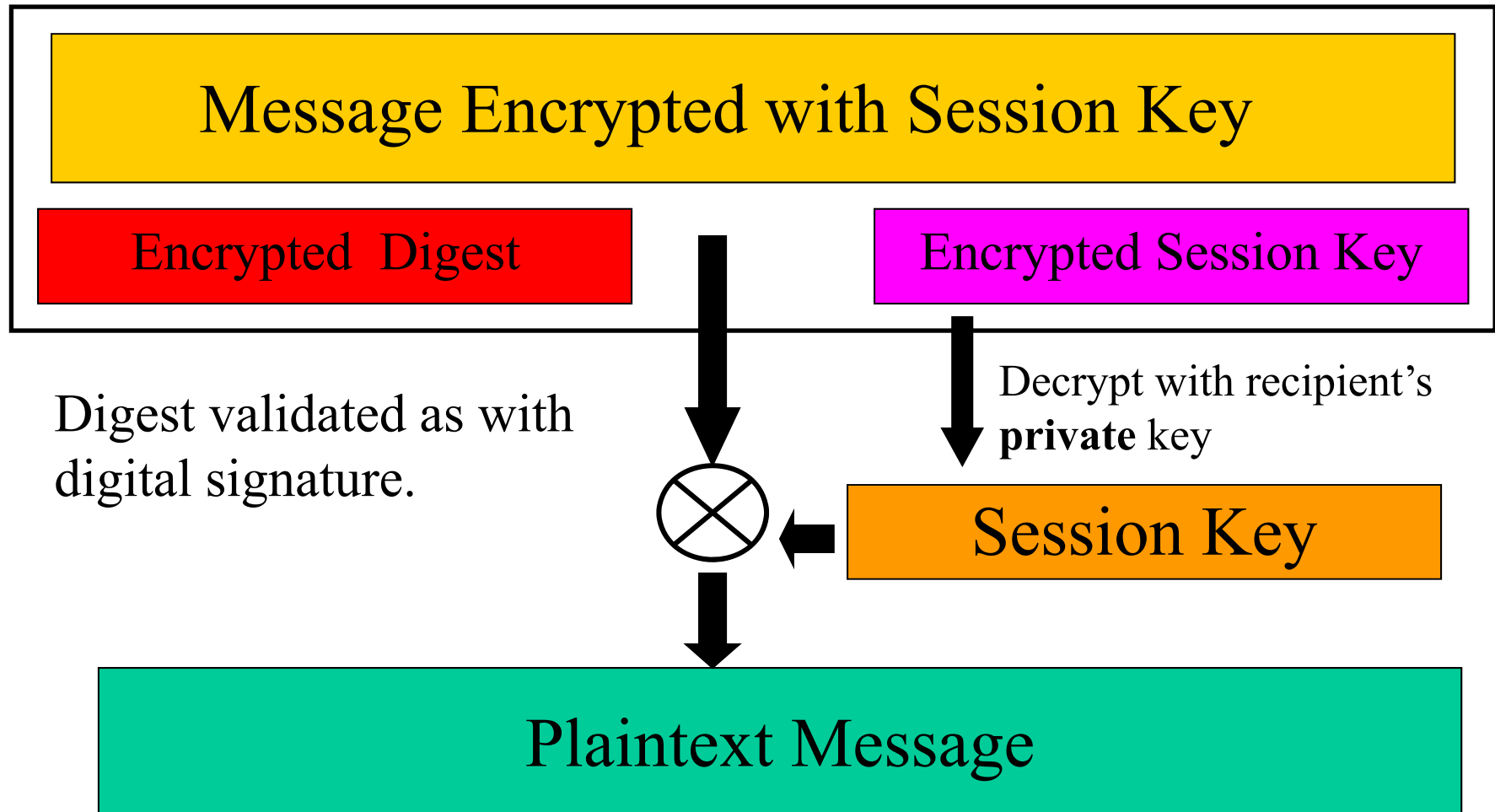
Encrypt with  
Recipient's *Public* Key

**Encrypted Digest**

**Encrypted Key**

**Encrypted Message**

# Decrypting a Hybrid Message



# How TLS Works

Generate a “master secret” at the browser

Server sends browser a digital certificate

- Identifies the server

- Includes the server’s public key

Browser encrypts the “master secret” with server’s public key, sends it to the server.

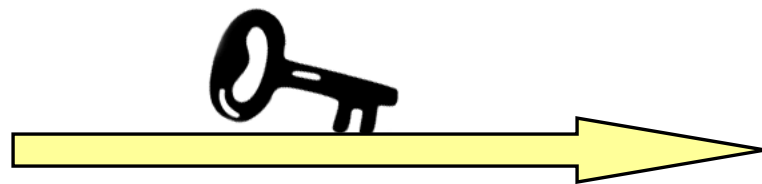
Both browser and server generate a session key from the master secret.

The session key is used for one communication session only.

# Public Key Cryptography

Bob wants to communicate securely with George.  
He gets George's public key from a key repository  
and uses it to encrypt his message.

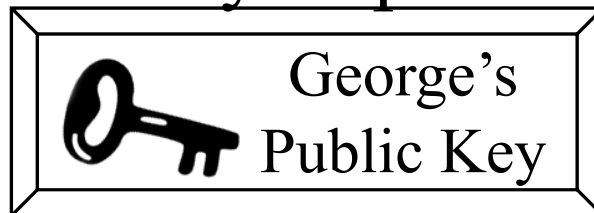
Bob



George



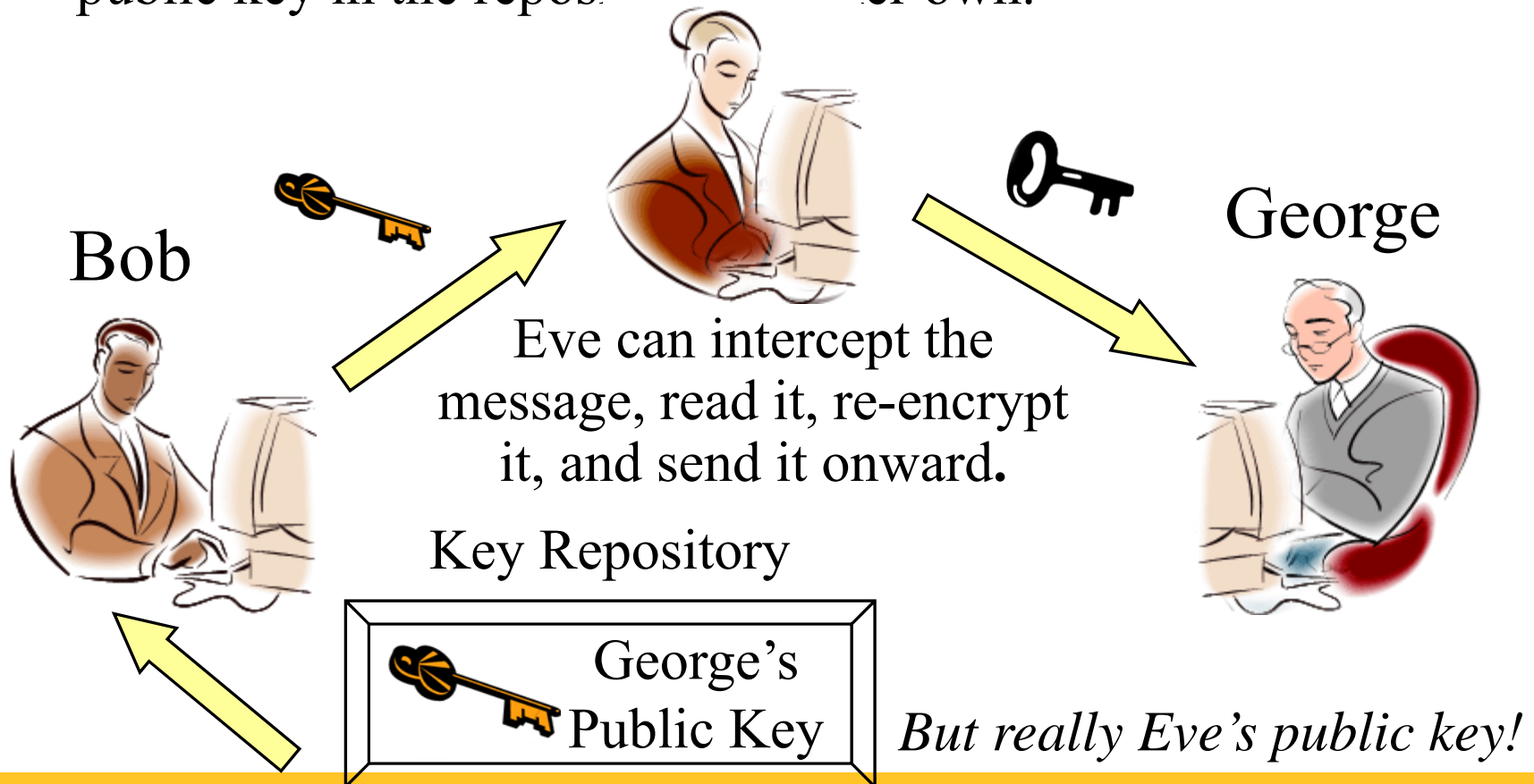
Key Repository



George decrypts  
the message with  
his private key.

# Man-in-the-Middle Attack

Eve (the man in the middle) sneakily replaces George's public key in the repository with her own.



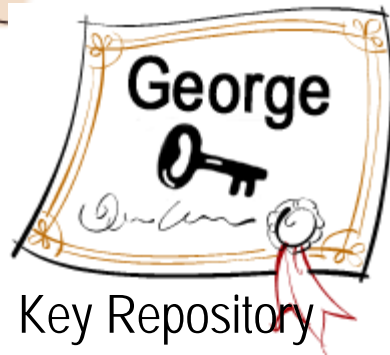
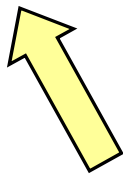
# Public Key Infrastructure

- Goal: **bind identity to a public key**
  - Crucial as people will use key to communicate with principal whose identity is bound to key
  - Erroneous binding means no secrecy between principals
  - Assumes each principal identified by an acceptable name (such as email address)

# Public Key in a Certificate

If George's public key is contained in a digital certificate, signed by someone Bob trusts, it is *much harder* to tamper with the key.

Bob



Key Repository



George



ID:	George@example.com
Pub Key:	etaoin#\$\$%shrdlu
CA:	Verisign
Digital signature over above	

# Reminder: Digital Certificate

Contains:

- Principal's identity
- Principal's public key
- Identity of signer and other info
- Digital Signature

Everything is plain text except the digital signature, which is a cryptographic hash (digest) encrypted by the signer's *private* key.

Principal's Identity George@spsu.edu
Principal's Public Key mQGiBD9aAvwRB
Hash, other information SHA-256
Signer's Identity Verisign
Digital Signature B157 ACE3 9788



# Changing the Public Key

- Evil Eve substitutes her public key for George's, a man-in-the-middle attack.
- The digital signature is no longer valid (because the data have changed.)
- Evil Eve must gain access to the signer's private key (signing key) to forge a new digital signature. We hope this is hard!

Principal's Identity George@spsu.edu
Eve's Public Key DfIZOkhURN
Hash, other information SHA-256
Signer's Identity Verisign
Signature is invalid! B157 ACE3 9788

# User Identification and Authentication

- Fundamental security building block
  - basis of access control and user accountability
- The process of verifying an identity claimed by or for a system entity
- Two steps:
  - identification – who are you?
  - authentication – prove it!  
(And proving it essentially binds the identifier to a principal, *i.e.* a person.)

# Means of User Authentication

- Three means of authenticating user's identity
- Based on something the principal...
  - knows - *e.g.* password, PIN
  - has - *e.g.* key, token, smartcard
  - is (biometrics)
- Can be used alone or combined (multi-factor authentication)
- All can provide user authentication

# Multi-Factor Authentication

- The three factors are:
  - Something you know
  - Something you have
  - Something you are
- Using two or all three factors improves security.
  - Loss of a token is less bad if a password is also required.
  - A guessed or compromised password is less bad if a fingerprint is also required. Etc.

# Basics of Access Control

- Access control is a collection of methods and components
  - Supports confidentiality (protects information from unauthorized disclosure)
  - Supports integrity (protects information from unauthorized modification)
- ***Authorization***: allow only authorized subjects to access objects that they are permitted to access
- ***Least privilege*** philosophy:  
A subject is granted permissions needed to accomplish the required tasks and nothing more.

# Authentication is a Prerequisite

- Access control mechanisms assume the subject (person, principal) has been authenticated.
- If another can masquerade as the subject, that person will be granted the subject's level of access.

# Goals of Access Control

- **Complete mediation:** Check every access.  
(What happens if access is removed while I am using a file? What *should* happen?)
- **Least privilege:** In granting access to an object, do not also grant more rights than needed, nor rights to other objects.
- **Acceptable use:** Permitted operations depend upon the nature of the object and access granted.

# Definitions

- **Subject** - entity that can access objects
  - a process representing user/application
  - often have 3 classes: owner, group, world
- **Object** - access controlled resource
  - *e.g.* files, directories, records, programs etc
  - number/type depend on environment
- **Access right** - way in which subject accesses an object
  - *e.g.* read, write, execute, delete, create, search



# Access Control Matrix

- Describes the protection state of a system, that is, the conditions under which it is secure.
- Access Control Matrix Model: simplest abstraction mechanism for representing protection state
- Protection State Transitions

# Example: Files

- Processes *Andy*, *Betty*
- Files *f*, *g*
- Rights *r*, *w*, *x*, *a*, *o*

	<i>f</i>	<i>g</i>	<i>Andy</i>	<i>Betty</i>
<i>Andy</i>	<i>rwo</i>	<i>r</i>	<i>rwxo</i>	<i>w</i>
<i>Betty</i>	<i>a</i>	<i>r</i>	<i>r</i>	<i>rwxo</i>

# Access Control Lists

Columns of access control matrix

	<i>file1</i>	<i>file2</i>	<i>file3</i>
<i>Andy</i>	rx	r	rwo
<i>Betty</i>	rwxo	r	
<i>Charlie</i>	rx	rwo	w

ACLs:

- file1: { (Andy, rx) (Betty, rwxo) (Charlie, rx) }
- file2: { (Andy, r) (Betty, r) (Charlie, rwo) }
- file3: { (Andy, rwo) (Charlie, w) }

# Access Control Techniques

- Choose techniques that fit the organization's needs (which are **defined by policy**)
- Considerations include
  - Level of security required
  - User and environmental impact of security measures
- Techniques differ in
  - The way objects and subjects are identified
  - How decisions are made to approve or deny access

# Access Control Designs

- Access control designs define rules for users accessing files or devices
- Three common access control designs
  - Mandatory access control
  - Discretionary access control
  - Non-discretionary (role-based) access control

# Bell-LaPadula Model, Preliminary

- A mandatory access control model
  - Security levels arranged in linear ordering
    - Top Secret: *highest*
    - Secret
    - Confidential
    - Unclassified: *lowest*
- ↓
- Levels ( $L$ ) consist of *security clearance levels*  $L(s)$  for subjects
  - Objects have *security classification levels*  $L(o)$

# Reading Information

- Information flows *up*, not *down*
  - “Reads up” disallowed, “reads down” allowed
- Simple Security Condition (Preliminary)
  - Subject  $s$  can read object  $o$  iff  $L(o) \leq L(s)$
  - Sometimes called “*no read up*” rule

# Writing Information

- Information flows *up*, not *down*
  - “Writes up” allowed, “writes down” disallowed
  - Prevents accidental or malicious leakage to lower levels of clearance.
- \*-Property (Preliminary)
  - Subject  $s$  can write object  $o$  iff  $L(s) \leq L(o)$
  - Sometimes called “*no write down*” rule.



# Discretionary Access Control

- All access to an object is defined by the object owner
- Most common design in commercial operating systems
  - Generally less secure than mandatory control
  - Generally easier to implement and more flexible
- Includes
  - Identity-based access control
  - Access control lists (ACLs)

# Role-Based Access Control

- Uses a subject's role or a task assigned to subject to grant or deny object access
  - Also called *non-discretionary* or task-based access control
- Works well in environments with high turnover of subjects since access is not tied directly to subject

# Database Leakage through Inference

- A cost-accounting analyst is allowed to know average salaries and numbers by job class.
- The analyst is not allowed to know individual salaries.
- The analyst asks the DBMS for the average salary for job class “President.”

# Traffic Analysis

- It's an inference attack.
- Can identify “centers of control”
- Can measure message volume (“chatter”)
  - Increase: a project is being planned
  - Decrease: planning complete, or fear of interception.

# Why Care About Traffic Analysis?

- As security professionals, we might use traffic analysis to identify problems, such as leaks from organizations where we work. (You would have your own mail server logs, for example.)
- We might want to defeat traffic analysis to protect our privacy for completely legal reasons, *e.g.* planning a merger.

# Security Intrusion and Detection

## **Security Intrusion**

a security event, or combination of multiple security events, that constitutes a security incident in which an intruder gains, or attempts to gain, access to a system (or system resource) without having authorization to do so.

## **Intrusion Detection**

a security service that monitors and analyzes system events for the purpose of finding, and providing real-time or near real-time warning of attempts to access system resources in an unauthorized manner.

# Intrusion Detection Systems

- We classify intrusion detection systems (IDSs) as:
  - Host-based IDS: monitor single host activity
  - Network-based IDS: monitor network traffic
- Logical components:
  - sensors - collect data
  - analyzers - determine if intrusion has occurred
  - user interface - manage / direct / view IDS

# Malicious Software

- Programs exploiting system vulnerabilities (or social engineering.)
- Also known as malware
  - program fragments that need a host program
    - *e.g.* viruses, logic bombs, and backdoors
  - independent self-contained programs
    - *e.g.* worms, bots
  - replicating or not
- These pose a sophisticated threat to computer systems



# Denial of Service

- **Denial of service (DoS):** an action that prevents or impairs the authorized use of networks, systems, or applications by exhausting resources such as central processing units (CPU), memory, bandwidth, and disk space
- attacks:
  - network bandwidth
  - system resources
  - application resources
- have been an issue for some time

# Source Address Spoofing

- Malicious packets can use forged source addresses:
  - given sufficient privilege to create “raw sockets”
  - easy to create, especially on MS operating systems
- generate large volumes of packets
- directed at target
- with different, random, source addresses
- cause same congestion
- responses are scattered across Internet
- real source is much harder to identify

# SYN Flooding

- Another common attack
- attacks ability of a server to respond to future connection requests by overflowing tables used to manage them
- hence an attack on system resource

# Distributed Denial of Service Attacks

- Have limited volume if single source used
- Multiple systems allow much higher traffic volumes to form a Distributed Denial of Service (DDoS) Attack
- Often compromised PC's / workstations
  - zombies with backdoor programs installed
  - forming a botnet

# Firewalls and Intrusion Prevention Systems

- Effective means of protecting LANs
- Internet connectivity is essential
  - for organization and individuals
  - but creates a threat
- Can also secure workstation and server subnets
- Use firewall as perimeter defense
  - single choke point to impose security

# Firewall Capabilities & Limits

- Capabilities:
  - defines a single choke point
  - provides a location for monitoring security events
  - convenient platform for some Internet functions such as NAT, usage monitoring, IPSEC VPNs
- Limitations:
  - cannot protect against attacks bypassing firewall
  - may not protect against internal threats
  - improperly secure wireless LAN
  - laptop, phone, portable storage device infected outside then used inside

# Virtual Private Networks

- Provide secure connections over public networks through encryption.
- Can connect two networks.
- Can connect a remote user to the corporate network. VPN software on the remote computer makes the network “appear local.”
- In other words, like “a tunnel through the Internet.”

# Intrusion Prevention Systems (IPS)

- Recent addition to security products which
  - inline net/host-based IDS that can block traffic
  - functional addition to firewall that adds IDS capabilities
- Can block traffic like a firewall, using IDS algorithms
- May be network or host based



# Formal Models for Computer Security

- Two fundamental computer security facts:
  - all complex software systems have flaws
  - is extraordinarily difficult to build computer hardware/software not vulnerable to attack
- Hence, the desire to prove design and implementation satisfy security requirements...
- led to development of formal security models
  - initially funded by US DoD
- Bell-LaPadula (BLP) model very influential

# Common Criteria (CC)

- ISO standards for security requirements and defining evaluation criteria to give:
  - greater confidence in IT product security
  - from formal actions during process of:
    - development using secure requirements
    - evaluation confirming meets requirements
    - operation in accordance with requirements
- evaluated products are listed for use

# Integrity Policy Models

- Requirements:
  - Very different from confidentiality policies
  - We are describing whether (and how much) we can trust data. (Not who can see it.)
- Integrity: the state that exists when records agree with the source from which they were derived and have not been incorrectly altered or destroyed.
- Biba's model (dual of Bell-LaPadula)
- Clark-Wilson model (transaction-based)
- Chinese Wall model (conflict of interest)

# Principles

- Separation of duties:  
Two or more people are required to perform critical functions. (May require separation into two or more steps)
- Separation of function:  
Development separated from production, etc.
- Auditing:  
It must be possible to reconstruct what happened, by whom, and when.

# Intuition for Integrity Levels

- The higher the level, the more confidence
  - That a program will execute correctly
  - That data is accurate and/or reliable
- Note: there is a relationship between integrity and trustworthiness
- Important point: *integrity levels are **not** confidentiality levels*. Integrity is concerned with trustworthiness, not data flow.

# Software Security

- Many vulnerabilities result from poor programming practices
- Often from insufficient checking / validation of program input
- Awareness of issues is critical

# Software Quality vs. Security

- Software quality and reliability
  - accidental failure of program
  - from theoretically random unanticipated input
  - improve using structured design and testing
  - not how many bugs, but **how often triggered**
- Software security is related
  - but *attacker* chooses input distribution, specifically targeting buggy code to exploit
  - often triggered by very unlikely inputs
  - which common tests don't identify

# Defensive Programming

- A form of defensive design to ensure continued function of software despite unforeseen usage
- Requires attention to all aspects of program execution, environment, data processed
- Also called secure programming
- Assume nothing, check all potential errors
- Rather than just focusing on solving task
- Must validate all assumptions
- “OK if OK” *vs.* “OK if not OK.”



# Buffer Overflow

- A very common attack mechanism
  - from 1988 Morris Worm to Code Red, Slammer, Sasser and many others
- Prevention techniques are well known
- Still of major concern due to
  - legacy of widely deployed buggy
  - continued careless programming techniques
- Caused by programming error
- Allow more data to be stored than capacity available in a fixed sized buffer

# Human Factors

- An important, broad area
- We covered a few key topics:
  - security awareness, training, and education
  - organizational security policy
  - personnel security
  - E-mail and Internet use policies

# Risk Assessment

- Risk assessment means asking:
  - what assets do we need to protect?
  - how are those assets threatened?
  - what can we do to counter those threats?
- IT security management answers these:
  - determining security objectives and risk profile
  - perform security risk assessment of assets
  - select, implement, monitor controls
  - iterate the process

# Security Risk Assessment

- Critical component of process
  - else may have vulnerabilities or waste money
- Ideally examine every asset versus risk; not feasible in practice
- Choose one of possible alternatives based on the organization's resources and risk profile
  - baseline
  - informal
  - formal
  - combined

# Controls or Safeguards

- Controls or safeguards are
  - practices, procedures or mechanisms which may protect against a threat, reduce a vulnerability, limit the impact of an unwanted incident, detect unwanted incidents and facilitate recover
- Classes of controls:
  - Management
  - Operational
  - Technical
- Compare to McCumber Model (Policy, education, technology)

# IT Security Plan

- Provides details of:
  - what will be done
  - what resources are needed
  - who is responsible
- Should include:
  - risks, recommended controls, action priority
  - selected controls, resources needed
  - responsible personnel, implementation dates

# Incident Handling

- Need procedures specifying how to respond to a security incident
  - such an incident will most likely occur sometime
- Reflect range of consequences on organization
- Codify action to avoid panic
- Example: mass email worm
  - exploiting vulnerabilities in common apps
  - propagating via email in high volumes
  - should disconnect from Internet or not?

# Internet Security Protocols and Standards

- Transport Layer Security (TLS)
- IPv4 and IPv6 Security
- S/MIME (Secure/Multimedia Internet Mail Extension)



# Internet Authentication Applications

- Application-level authentication and digital signatures
- Implementations:
  - Kerberos symmetric key authentication service
  - X.509 public-key directory authentication
  - Public-key infrastructure (PKI)
  - Federated identity management

# Linux Security Model

- Linux's traditional security model is:
  - people or processes with “root” privileges can do anything
  - other accounts can do much less
- Hence, attackers want to get root privileges
- One can run robust, secure Linux systems
- The crux of the problem is use of **Discretionary Access Controls (DAC)**

# File System Security

- In Linux *everything* is treated as a file
  - *e.g.* memory, device-drivers, named pipes, and other system resources
  - hence why filesystem security is so important
- I/O to devices is via a “special” file
  - *e.g.* **/dev/cdrom**
- There are other special files like named pipes
  - a conduit between processes / programs

# Users and Groups

- A user-account (user)
  - represents someone capable of using files
  - associated both with humans and processes
- A group-account (group)
  - is a list of user-accounts
  - users have a main group
  - may also belong to other groups
- Users and groups are **not** files

# SetUID and SetGID

- setuid bit means program “runs as” owner
  - no matter who executes it
- setgid bit means run as a member of the group which owns it
  - again regardless of who executes it
- “run as” = “run with same privileges as”
- *are very dangerous* if set on file owned by root or other privileged account or group

# Mandatory Access Controls

- Linux uses a DAC security model
- But Mandatory Access Controls (MAC) impose a global security policy on all users
  - users may not set controls weaker than policy
  - normal admin done with accounts without authority to change the global security policy
  - but MAC systems have been hard to manage
- Use SELinux for high-sensitivity, high-security, applications

# Windows and Windows Security

- Windows is the world's most widely-used O/S
- An advantage is that security enhancements can protect millions of nontechnical users
- A challenge is that vulnerabilities in Windows can also affect millions of users
- We will review overall security architecture of Windows 2000 and later (but not Win9X)

# Windows Security Architecture

- Security Reference Monitor (SRM)
  - a kernel-mode component that performs access checks, generates audit log entries, and manipulates user rights (privileges)
- Local Security Authority (LSA)
  - responsible for enforcing local security policy
- Security Account Manager (SAM)
  - a database that stores user accounts and local users and groups security information
  - local logins perform lookup against SAM DB
  - passwords are stored using MD4



# Windows Security Architecture

- Active Directory (AD)
  - Microsoft's LDAP directory
  - all Windows clients can use AD to perform security operations including account logon
  - authentication uses AD when the user logs on using a domain rather than local account
  - user's credential information is sent securely across the network to be verified by AD
- WinLogon (local) and NetLogon (net) handle login requests

# Local vs. Domain Accounts

A networked Windows computer can be:

- domain joined
  - can login with either domain or local accounts
  - centrally managed and much more secure
- in a workgroup
  - a collection of computers connected together
  - only local accounts in SAM can be used
  - no infrastructure to support AD domain

# Mandatory Access Control

- Windows Vista has “Integrity Control” that limits operations changing an object’s state
- Objects and principals are labeled (using SID) as:
  - Low integrity (S-1-16-4096)
  - Medium integrity (S-1-16-8192)
  - High integrity (S-1-16-12288)
  - System integrity (S-1-16-16384)
- When a write operation occurs, first check whether subject’s integrity level dominates object’s integrity level
- Much of O/S marked medium or higher integrity

# Windows Vulnerabilities

- Windows, like all O/S's, has security flaws
  - and bugs have been exploited by attackers to compromise customer operating systems
- Microsoft now uses process improvement called the Security Development Lifecycle
- Net effect approx 50% reduction in errors
- Windows Vista used SDL start to finish
- IIS v6 (in Windows Server 2003) had only 3 vulnerabilities in 4 years, none critical

# Windows Security Defenses

- Attackers are now criminals rather than poorly-socialized adolescents, and are highly motivated by money
- Windows has categories of security defenses:
  - account defenses
  - network defenses
  - buffer overrun defenses.
  - browser defenses

# Securing Wireless Transmissions

- Principal threats are eavesdropping, altering or inserting messages, and disruption
- Countermeasures for eavesdropping:
  - signal-hiding techniques
  - **encryption**
- The use of encryption and authentication protocols is the standard method of countering attempts to alter or insert transmissions

# Wireless LAN Security Algorithms

- Wired Equivalent Privacy (WEP) algorithm
  - 802.11 privacy
  - **Trivial to crack**
- Wi-Fi Protected Access (WPA)
  - set of security mechanisms that eliminates most 802.11 security issues and was based on the current state of the 802.11i standard
- Robust Security Network (RSN) a.k.a WPA2
  - final form of the 802.11i standard
- Wi-Fi Alliance certifies vendors in compliance with the full 802.11i specification under the WPA2 program

# Questions





**All's Well That Ends!**