CSMC 412

Operating Systems

Prof. Ashok K Agrawala

Set 17

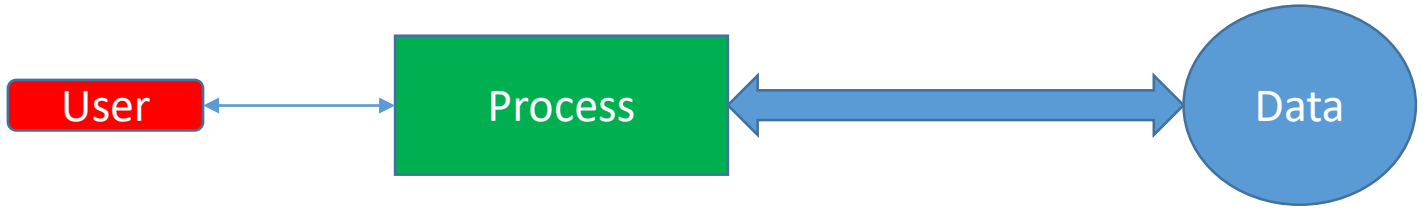
Security

- 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 7

Objectives

- Discuss security threats and attacks
- Explain the fundamentals of encryption, authentication, and hashing
- Examine the uses of cryptography in computing
- Describe the various countermeasures to security attacks

Structure



- User Authentication
- Access Control
- Intrusion

The Security Problem

- Security –
  - Guarding computer resources against unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency.
- Protection –
  - Controlling the access of processes and users to the resources defined by a computer system.
- System is **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

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
  - Does encrypting passwords solve the exposure problem?
    - Might solve **sniffing**
    - Consider **shoulder surfing**
    - Consider Trojan horse keystroke logger
    - How are passwords stored at authenticating site?

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 fob to generate the password
    - Changes very frequently
- Biometrics
  - Some physical attribute (fingerprint, hand scan)
- Multi-factor authentication
  - Need two or more factors for authentication
    - i.e. USB "dongle", biometric measure, and password

Passwords (cont.)

STRONG AND EASY TO REMEMBER PASSWORDS

It is extremely important to use strong (hard to guess and hard to shoulder surf) passwords on critical systems like bank accounts. It is also important to not use the same password on lots of systems, as one less important, easily hacked system could reveal the password you use on more important systems. A good technique is to generate your password by using the first letter of each word of an easily remembered phrase using both upper and lower characters with a number or punctuation mark thrown in for good measure. For example, the phrase "My girlfriend's name is Katherine" might yield the password "Mgn.isK!". The password is hard to crack but easy for the user to remember. A more secure system would allow more characters in its passwords. Indeed, a system might also allow passwords to include the space character, so that a user could create a **passphrase** which is easy to remember but difficult to break.

Biometric Techniques

- Finger Print
- Iris Scan
- Facial Recognition

Two Factor Authentication

- Second independent means
  - SMS
  - Email
  - Phone Call

Security Violation Categories

- **Breach of confidentiality**
  - Unauthorized reading of data
    - Capturing secret data e.g. credit card information
- **Breach of integrity**
  - Unauthorized modification of data
- **Breach of availability**
  - Unauthorized destruction of data
    - Website defacement
- **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
- **Privilege escalation**
  - Common attack type with access beyond what a user or resource is supposed to have

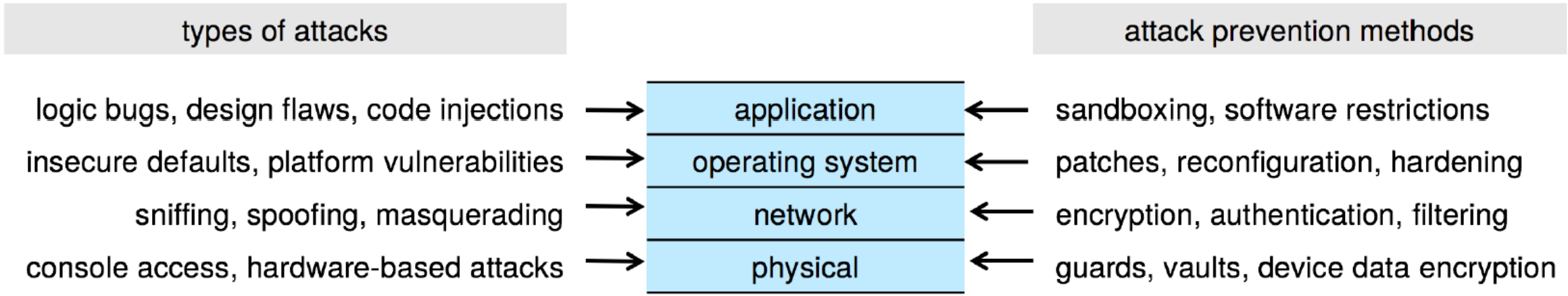
Security Measure Levels

- 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
  - **Application**
    - Benign or malicious apps can cause security problems
  - **Operating System**
    - Protection mechanisms, debugging
  - **Network**
    - Intercepted communications, interruption, DOS
- Security is as weak as the weakest link in the chain
- Humans a risk too via **phishing** and **social-engineering** attacks
- 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?

Four-layered Model of Security



Program Threats (Cont.)

- **Malware -** Software designed to exploit, disable, or damage computer
- **Trojan Horse** – Program that acts in a clandestine manner
  - **Spyware** – Program frequently installed with legitimate software to display adds, capture user data
  - **Ransomware** – locks up data via encryption, demanding payment to unlock it
- Others include trap doors, logic boms
- All try to violate the Principle of Least Privilege

> *THE PRINCIPLE OF LEAST PRIVILEGE*
>
> "The principle of least privilege. Every program and every privileged user of the system should operate using the least amount of privilege necessary to complete the job. The purpose of this principle is to reduce the number of potential interactions among privileged programs to the minimum necessary to operate correctly, so that one may develop confidence that unintentional, unwanted, or improper uses of privilege do not occur."—Jerome H. Saltzer, describing a design principle of the Multics operating system in 1974: https://pdfs.semanticscholar.org/1c8d/06510ad449ad24fbdd164f8008cc730cab47.pdf.

- Goal frequently is to leave behind Remote Access Tool (RAT) for repeated access

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;
  }
}
```

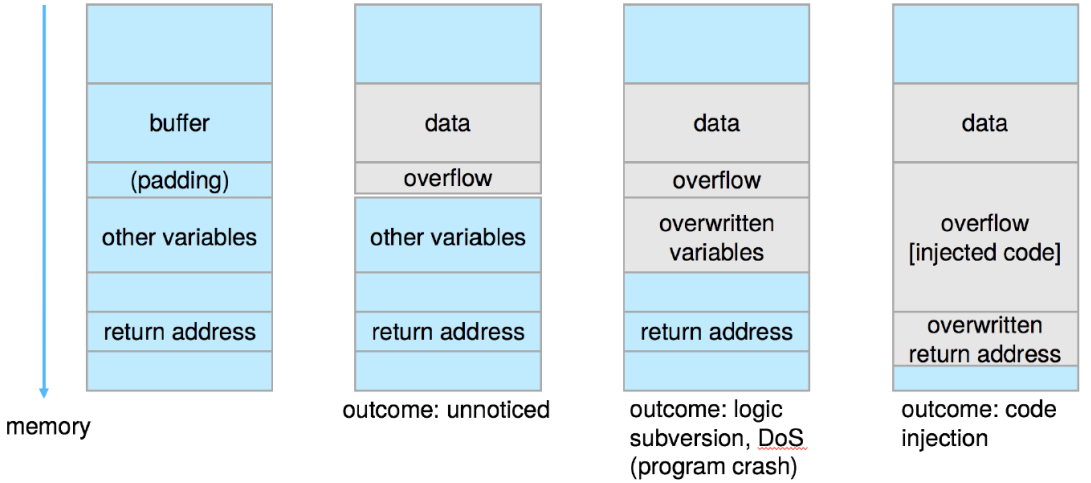
- **Code review** can help – programmers review each other's code, looking for logic flows, programming flaws

Code Injection

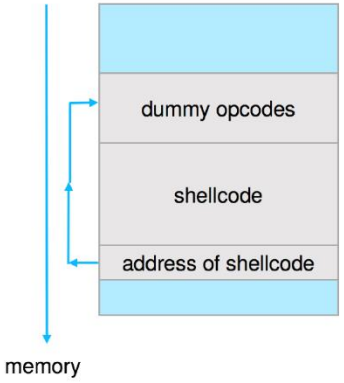
- **Code-injection attack** occurs when system code is not malicious but has bugs allowing executable code to be added or modified
  - Results from poor or insecure programming paradigms, commonly in low level languages like C or C++ which allow for direct memory access through pointers
  - Goal is a buffer overflow in which code is placed in a buffer and execution caused by the attack
  - Can be run by script kiddies – use tools written but exploit identifiers

Code Injection (cont.)

- Outcomes from code injection include:



- Frequently use trampoline to code execution to exploit buffer overflow:



Great Programming Required?

- For the first step of determining the bug, and second step of writing exploit code, yes
- **Script kiddies** can run pre-written exploit code to attack a given system
- Attack code can get a shell with the processes' owner's permissions
  - Or open a network port, delete files, download a program, etc
- Depending on bug, attack can be executed across a network using allowed connections, bypassing firewalls
- Buffer overflow can be disabled by disabling stack execution or adding bit to page table to indicate "non-executable" state
  - Available in SPARC and x86
  - But still have security exploits

Program Threats (Cont.)

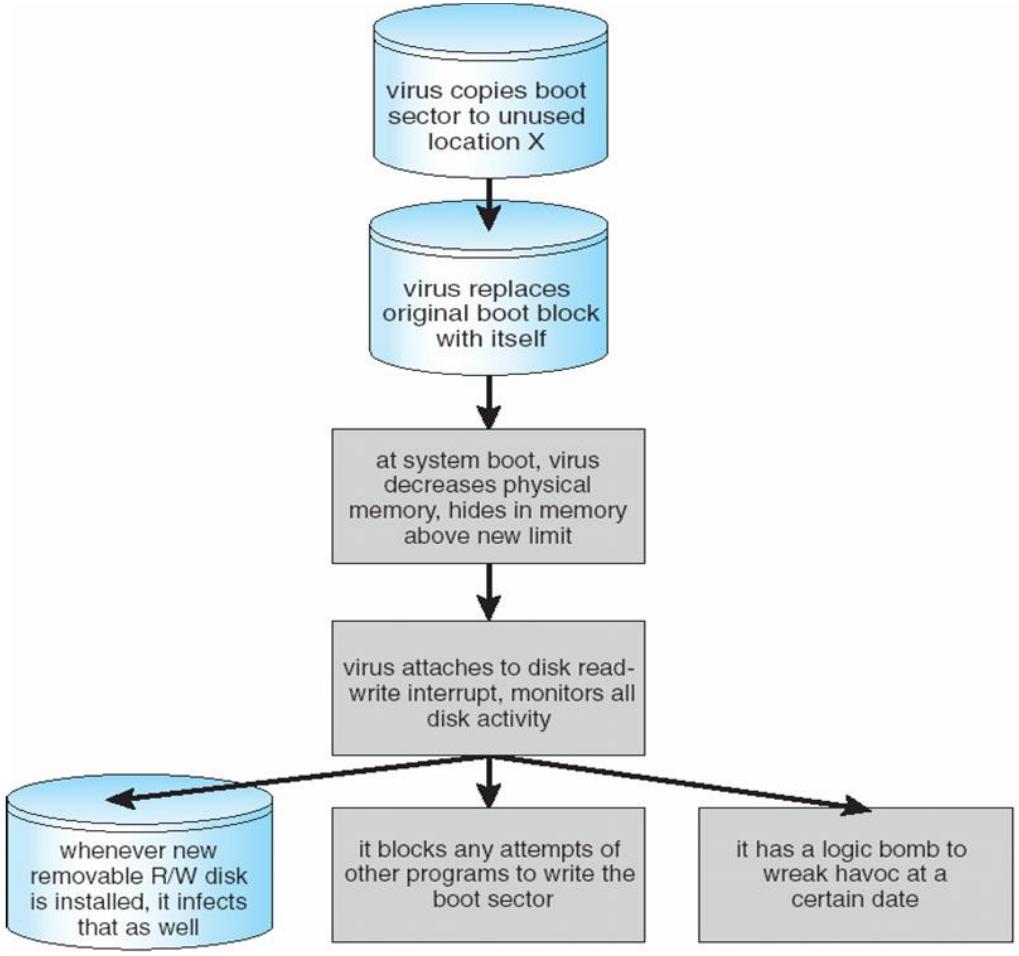
- **Viruses**
  - Code fragment embedded in legitimate program
  - Self-replicating, designed to infect other computers
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
  - Visual Basic Macro to reformat hard drive

```
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
  - Rootkit
  - Source code
  - Polymorphic to avoid having a **virus signature**
  - Encrypted
  - Stealth
  - Tunneling
  - Multipartite
  - Armored

A Boot-sector Computer Virus



The Threat Continues

- Attacks still common, still occurring
- Attacks moved over time from science experiments to tools of organized crime
  - Targeting specific companies
  - Creating botnets to use as tool for spam and DDOS delivery
  - **Keystroke logger** to grab passwords, credit card numbers
- Why is Windows the target for most attacks?
  - Most common
  - Everyone is an administrator
    - Licensing required?
  - **Monoculture** considered harmful

System and Network Threats

- Some systems “open” rather than **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
  - 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 7^{th} time

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

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
- Port scanning
  - Automated tool to look for network ports accepting connections
  - Used for good and evil

Standard Security Attacks

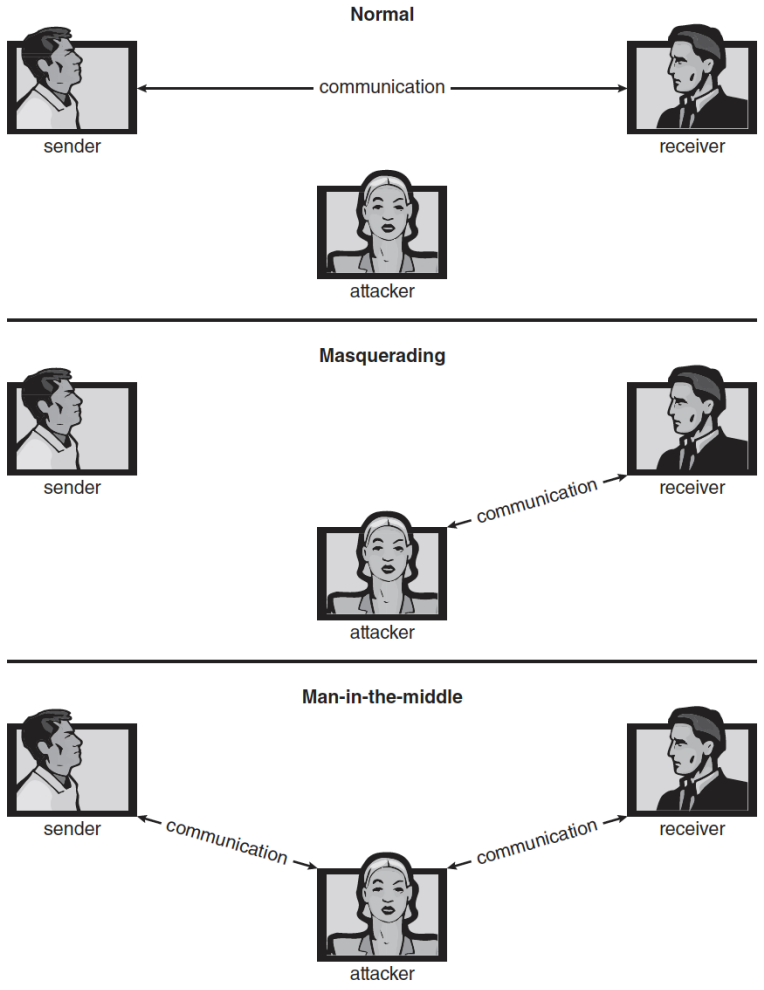


Figure 16.6 Standard security attacks.

Cryptography

Cryptography is a fundamental building block for security mechanisms.

- Introduction »
- Substitution ciphers »
- Transposition ciphers »
- One-time pads »
- Fundamental cryptographic principles »

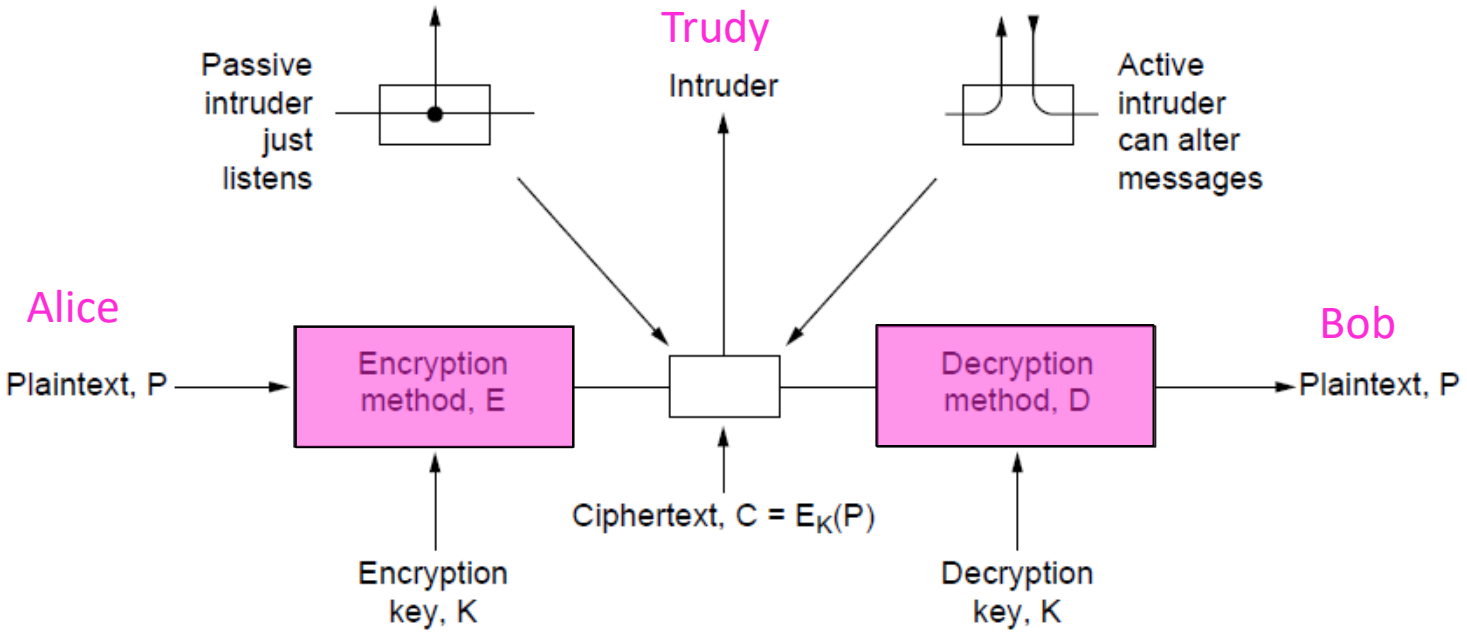
Cryptography as a Security Tool

- Broadest security tool available
  - Internal to a given computer, source and destination of messages can be known and protected
    - OS creates, manages, protects process IDs, communication ports
  - Source and destination of messages on network cannot be trusted without cryptography
    - Local network – IP address?
      - Consider unauthorized host added
    - WAN / Internet – how to establish authenticity
      - Not via IP address

Introduction

The encryption model (for a symmetric-key cipher)

- Kerckhoff's principle: Algorithms (E, D) are public; only the keys (K) are secret



Substitution Ciphers

Substitution ciphers replace each group of letters in the message with another group of letters to disguise it

plaintext:	a b c d e f g h i j k l m n o p q r s t u v w x y z
ciphertext:	Q W E R T Y U I O P A S D F G H J K L Z X C V B N M

Simple single-letter substitution cipher

Transposition Ciphers

Transposition ciphers reorder letters to disguise them

M	E	G	A	B	U	C	K
7	4	5	1	2	8	3	6
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

← Key gives column order

Plaintext

pleasetransferonemilliondollarsto
myswissbankaccountsixtwotwo

Ciphertext

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

Column 5 6 7 8

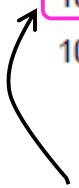
Simple column transposition cipher

One-Time Pads (1)

Simple scheme for perfect secrecy:

- XOR message with secret pad to encrypt, decrypt
- Pad is as long as the message and can't be reused!
  - It is a "one-time" pad to guarantee secrecy

Message 1:	1001001	0100000	1101100	1101111	1110110	1100101	0100000	1111001	1101111	1110101	0101110
Pad 1:	1010010	1001011	1110010	1010101	1010010	1100011	0001011	0101010	1010111	1100110	0101011
Ciphertext:	0011011	1101011	0011110	0111010	0100100	0000110	0101011	1010011	0111000	0010011	0000101
Pad 2:	1011110	0000111	1101000	1010011	1010111	0100110	1000111	0111010	1001110	1110110	1110110
Plaintext 2:	1000101	1101100	1110110	1101001	1110011	0100000	1101100	1101001	1110110	1100101	1110011



Different secret pad decrypts to the wrong plaintext

Fundamental Cryptographic Principles

1. Messages must contain some redundancy
   - All encrypted messages decrypt to something
   - Redundancy lets receiver recognize a valid message
   - But redundancy helps attackers break the design

2. Some method is needed to foil replay attacks
   - Without a way to check if messages are fresh then old messages can be copied and resent
   - For example, add a date stamp to messages

Symmetric-Key Algorithms

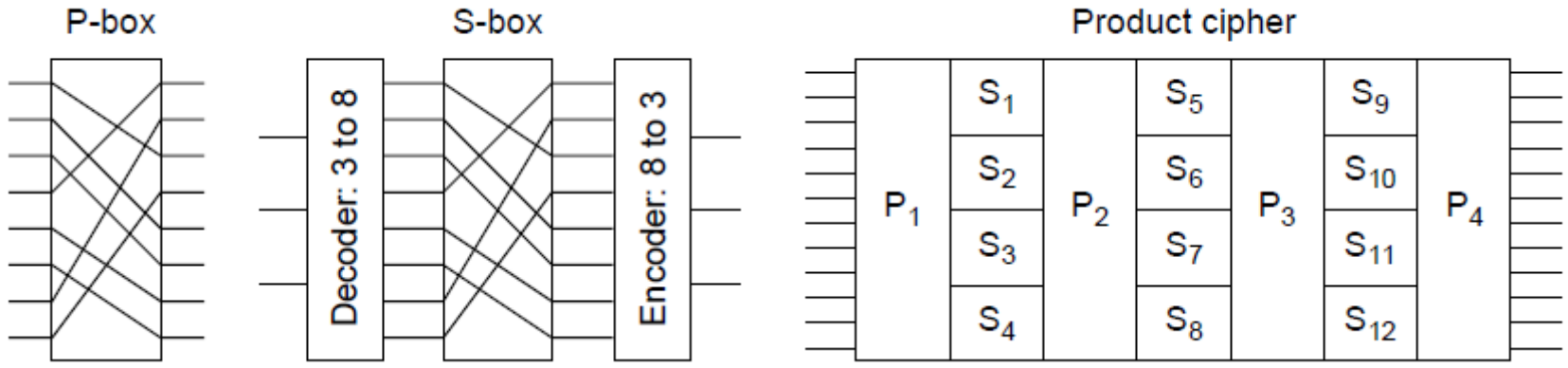
Encryption in which the parties share a secret key

- DES – Data Encryption Standard »
- AES – Advanced Encryption Standard »
- Cipher modes »
- Other ciphers »
- Cryptanalysis »

Symmetric-Key Algorithms (1)

Use the same secret key to encrypt and decrypt; block ciphers operate a block at a time

- Product cipher combines transpositions/substitutions



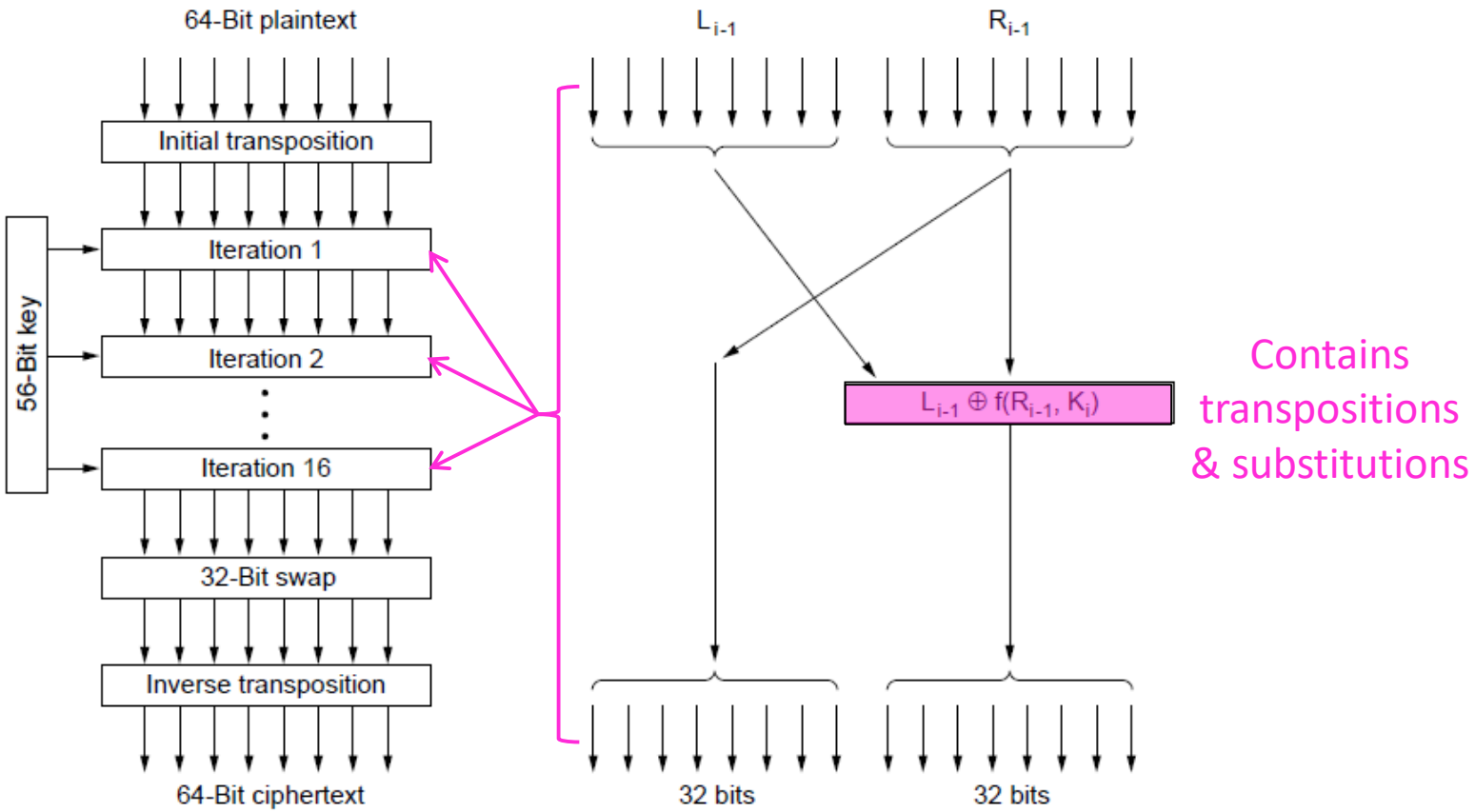
Permutation (transposition) box

Substitution box

Product with multiple P- and S-boxes

Data Encryption Standard (1)

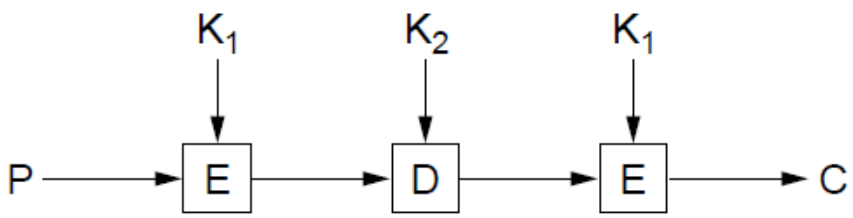
DES encryption was widely used (but no longer secure)



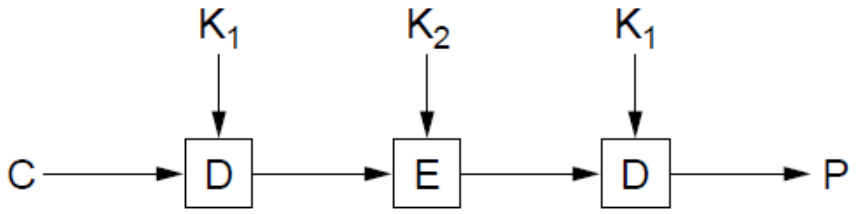
Data Encryption Standard (2)

Triple encryption ("3DES") with two 56-bit keys

- Gives an adequate key strength of 112 bits
- Setting $K_1 = K_2$ allows for compatibility with DES



Triple DES encryption



Triple DES decryption

Advanced Encryption Standard (1)

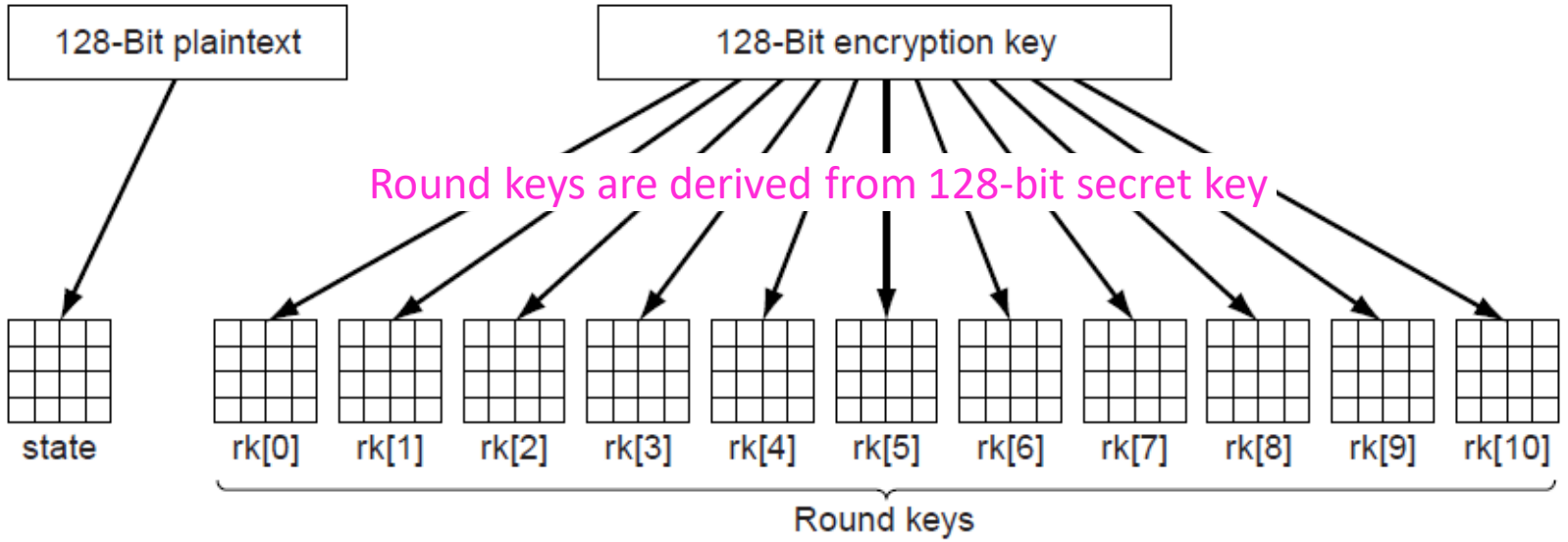
AES is the successor to DES:

- Symmetric block cipher, key lengths up to 256 bits
- Openly designed by public competition (1997-2000)
- Available for use by everyone
- Built as software (e.g., C) or hardware (e.g., x86)
- Winner was Rijndael cipher
- Now a widely used standard

Advanced Encryption Standard (2)

AES uses 10 rounds for 128-bit block and 128-bit key

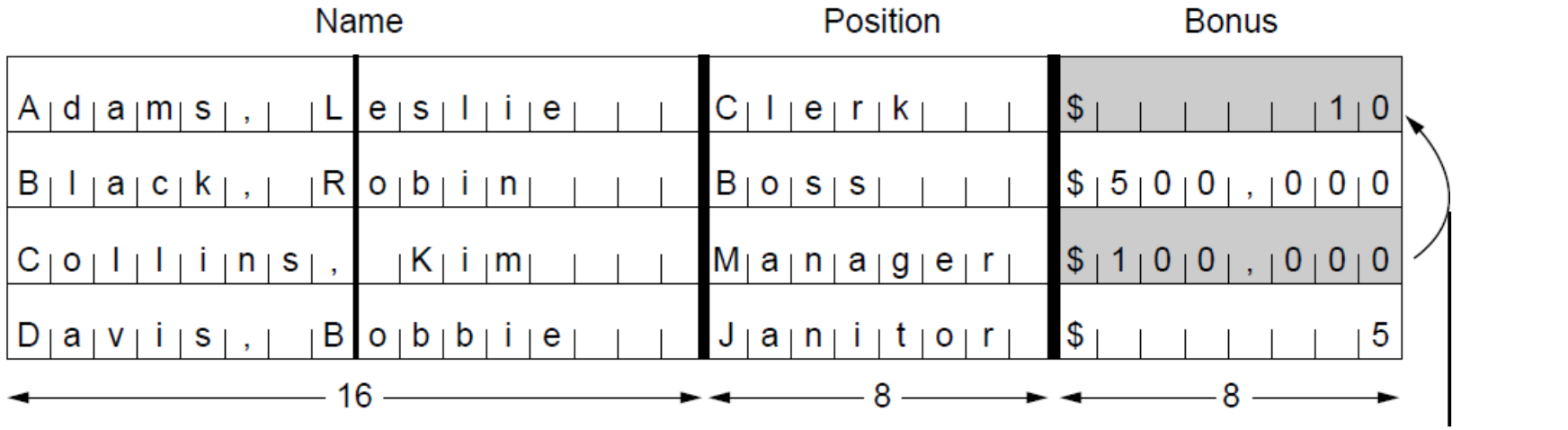
- Each round uses a key derived from 128-bit key
- Each round has a mix of substitutions and rotations
- All steps are reversible to allow for decryption



Cipher Modes (1)

Cipher modes set how long messages are encrypted

- Encrypting each block independently, called ECB (Electronic Code Book) mode, is vulnerable to shifts



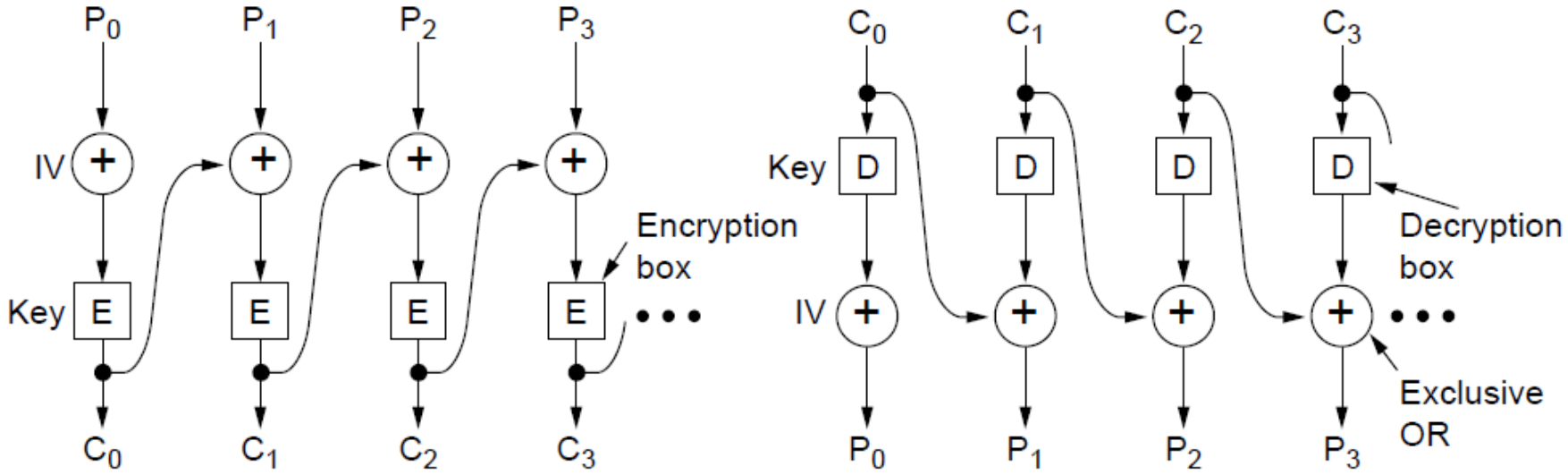
With ECB mode, switching encrypted blocks gives a different but valid message

Leslie gets a large bonus!

Cipher Modes (2)

CBC (Cipher Block Chaining) is a widely used mode

- Chains blocks together with XOR to prevent shifts
- Has a random IV (Initial Value) for different output

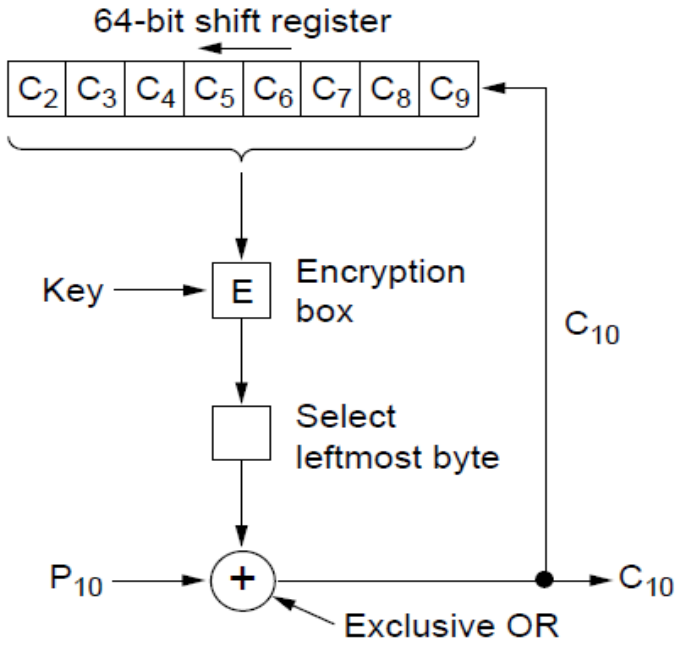


CBC mode encryption

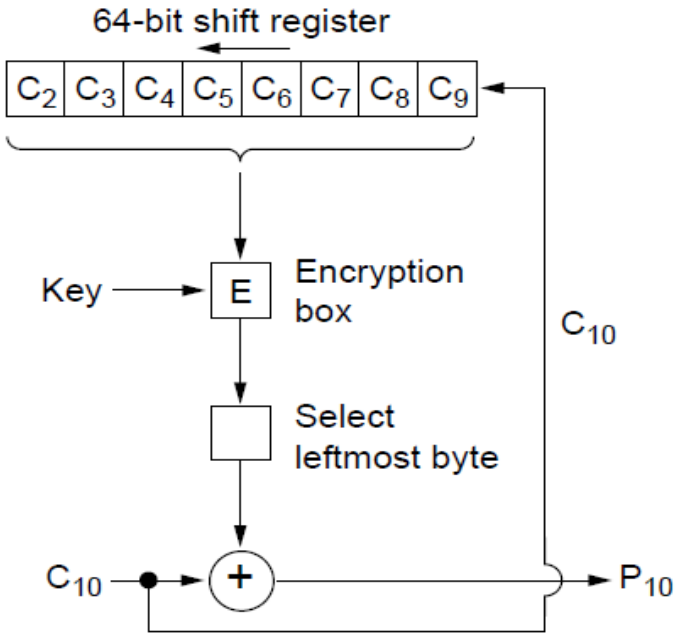
CBC mode decryption

Cipher Modes (3)

There are many other modes with pros / cons, e.g., cipher feedback mode is similar to CBC mode but can operate a byte (rather than a whole block) at a time



Encryption

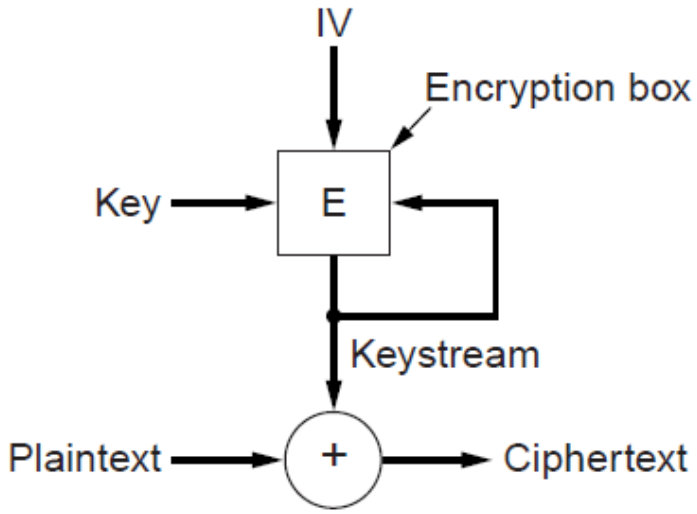


Decryption

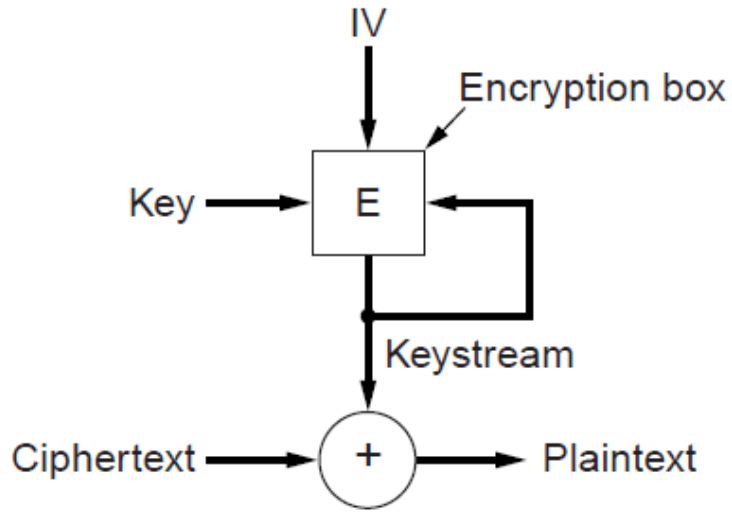
Cipher Modes (4)

A stream cipher uses the key and IV to generate a stream that is a one-time pad; can't reuse (key, IV) pair

- Doesn't amplify transmission errors like CBC mode



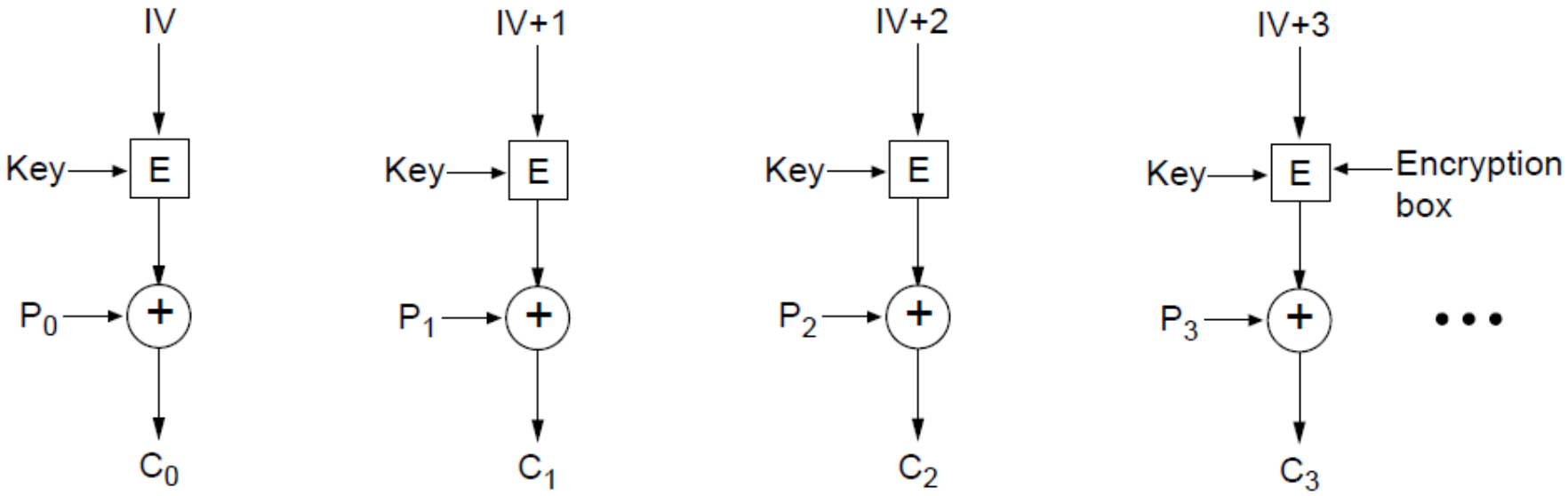
Encryption



Decryption

Cipher Modes (5)

Counter mode (encrypt a counter and XOR it with each message block)
allows random access for decryption



Encryption above; repeat the operation to decrypt

Other Ciphers

Some common symmetric-key cryptographic algorithms

- Can be used in combination. e.g., AES over Twofish

Cipher	Author	Key length	Comments
Blowfish	Bruce Schneier	1–448 bits	Old and slow
DES	IBM	56 bits	Too weak to use now
IDEA	Massey and Xuejia	128 bits	Good, but patented
RC4	Ronald Rivest	1–2048 bits	Caution: some keys are weak
RC5	Ronald Rivest	128–256 bits	Good, but patented
Rijndael	Daemen and Rijmen	128–256 bits	Best choice
Serpent	Anderson, Biham, Knudsen	128–256 bits	Very strong
Triple DES	IBM	168 bits	Second best choice
Twofish	Bruce Schneier	128–256 bits	Very strong; widely used

Public-Key Algorithms

Encryption in which each party publishes a public part of their key and keep secret a private part of it

- RSA (by Rivest, Shamir, Adleman) »

Public-Key Algorithms (1)

Downsides of keys for symmetric-key designs:

- Key must be secret, yet be distributed to both parties
- For N users there are N^2 pairwise keys to manage

Public key schemes split the key into public and private parts that are mathematically related:

- Private part is not distributed; easy to keep secret
- Only one public key per user needs to be managed

Security depends on the chosen mathematical property

- Much slower than symmetric-key, e.g., 1000X
- So use it to set up per-session symmetric keys

RSA (1)

RSA is a widely used public-key encryption method whose security is based on the difficulty of factoring large numbers

Key generation:

- Choose two large primes, p and q
- Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
- Choose d to be relatively prime to z
- Find e such that $e \times d = 1 \bmod z$
- Public key is (e, n), and private key is (d, n)

Encryption (of k bit message, for numbers up to n):

- $\text{Cipher} = \text{Plain}^{e} \pmod n$

Decryption:

- $\text{Plain} = \text{Cipher}^{d} \pmod n$

RSA (2)

Small-scale example of RSA encryption

- For p=3, q=11 → n=33, z=20 → d=7, e=3

Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E
			Sender's computation		Receiver's computation	

Encryption: $C = P^3 \bmod 33$

Decryption: $P = C^7 \bmod 33$

Digital Signatures

Lets receiver verify the message is authentic

- Symmetric-Key signatures »
- Public-Key signatures »
- Message digests »
- The birthday attack »

Digital Signatures (1)

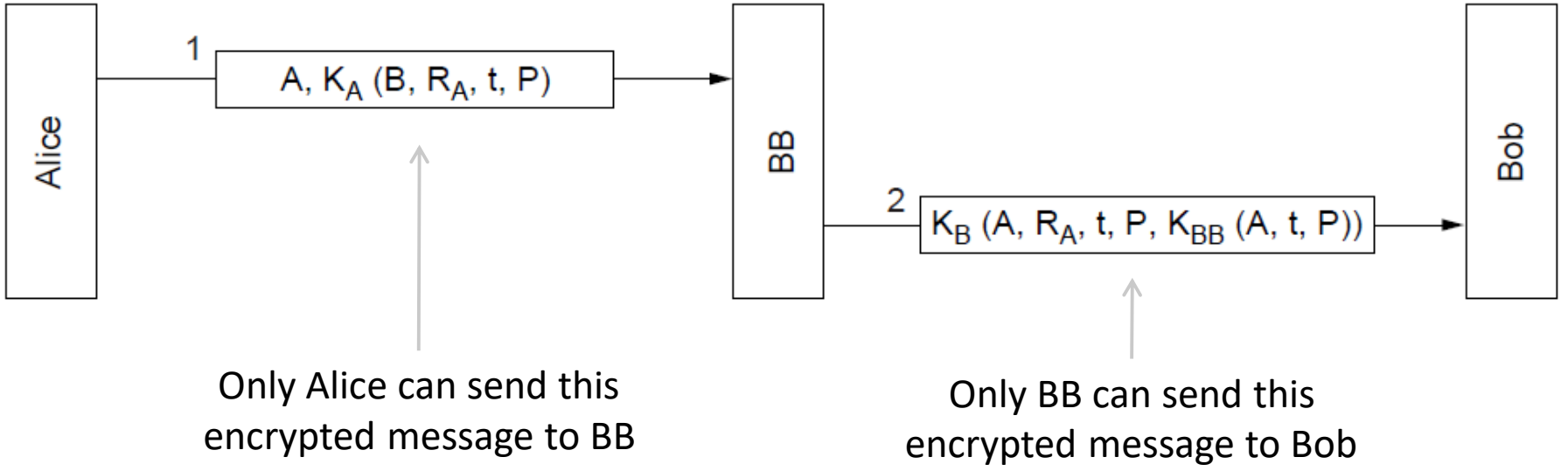
Requirements for a signature:

- Receiver can verify claimed identity of sender.
- Sender cannot later repudiate contents of message.
- Receiver cannot have concocted message himself.

Symmetric-key Signatures

Alice and Bob each trust and share a key with Big Brother; Big Brother doesn't trust anyone

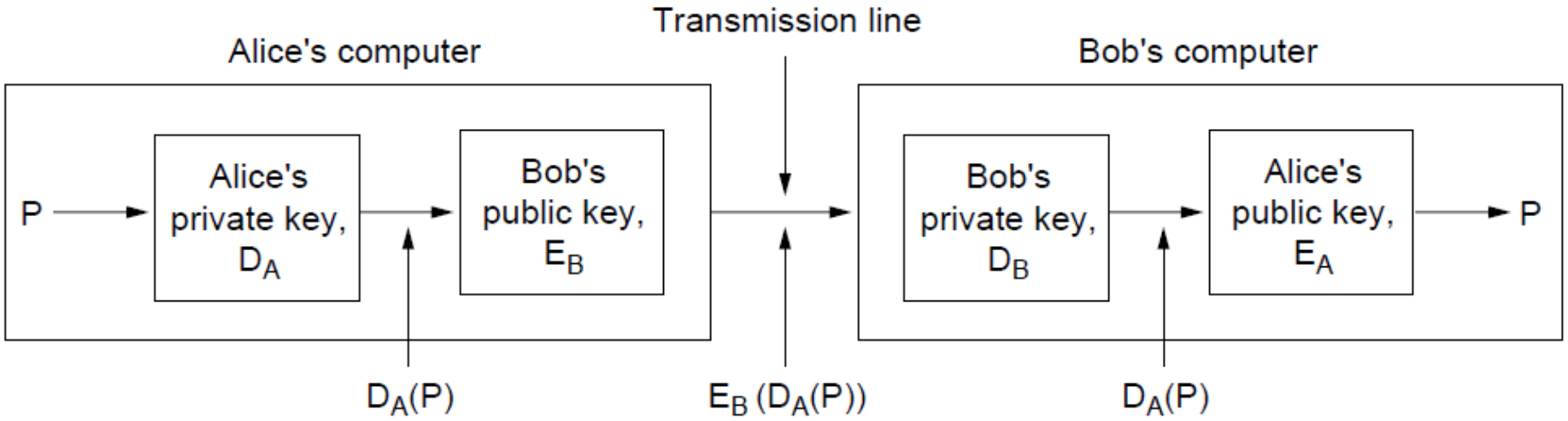
- A=Alice, B=Bob, P=message, R_A=random, t=time



Public-Key Signatures

No Big Brother and assumes encryption and decryption are inverses that can be applied in either order

- But relies on private key kept and secret
- RSA & DSS (Digital Signature Standard) widely used



Message Digests (1)

Message Digest (MD) converts arbitrary-size message (P) into a fixed-size identifier MD(P) with properties:

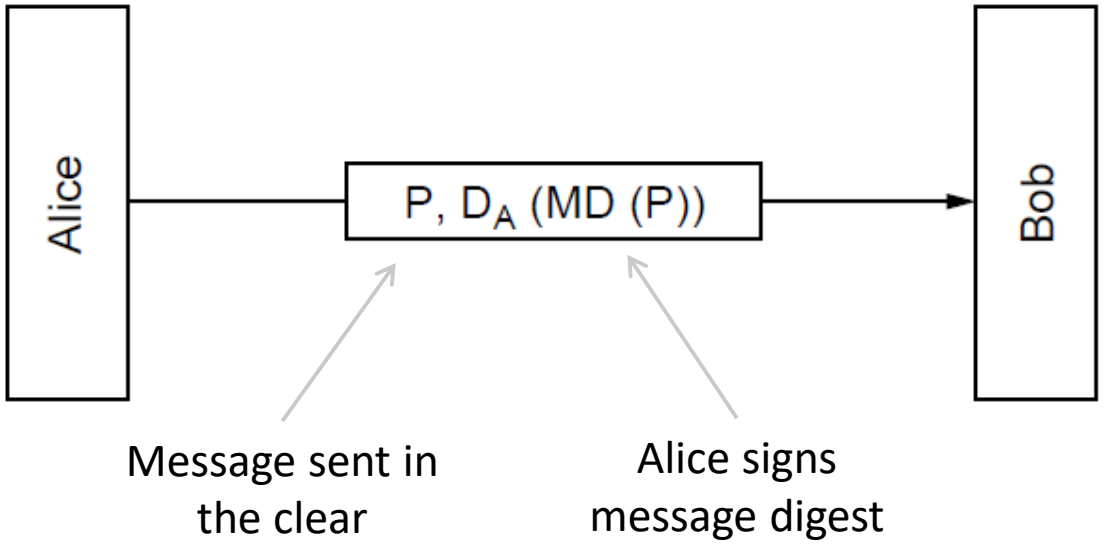
- Given P, easy to compute MD(P).
- Given MD(P), effectively impossible to find P.
- Given P no one can find P' so that MD(P') = MD(P).
- Changing 1 bit of P produces very different MD.

Message digests (also called cryptographic hash) can "stand for" messages in protocols, e.g., authentication

- Example: SHA-1 160-bit hash, widely used
- Example: MD5 128-bit hash – now known broken

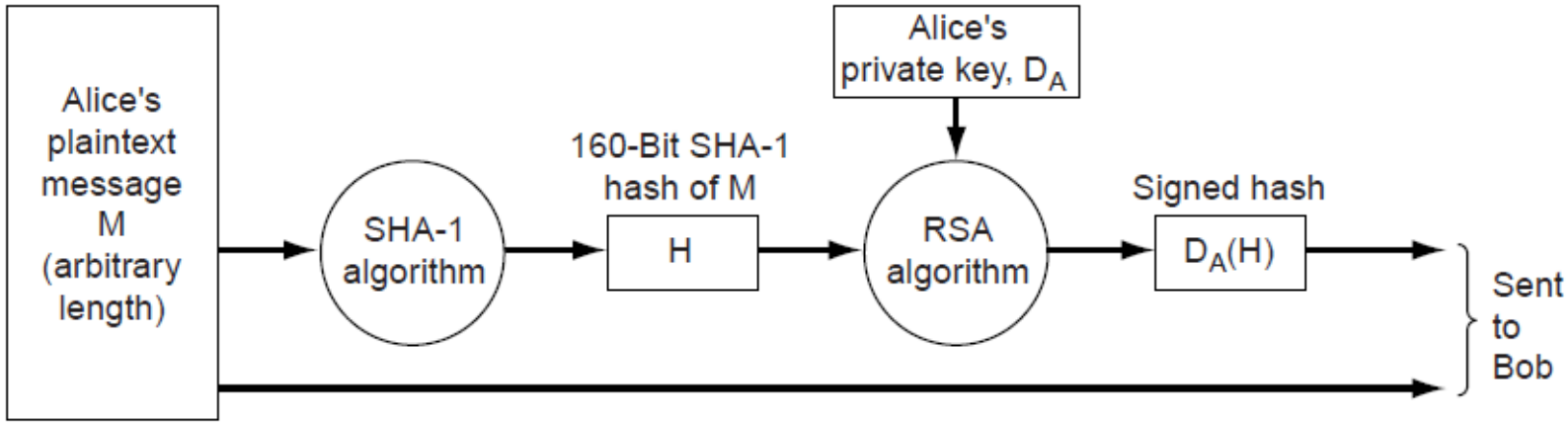
Message Digests (2)

Public-key signature for message authenticity but not confidentiality with a message digest



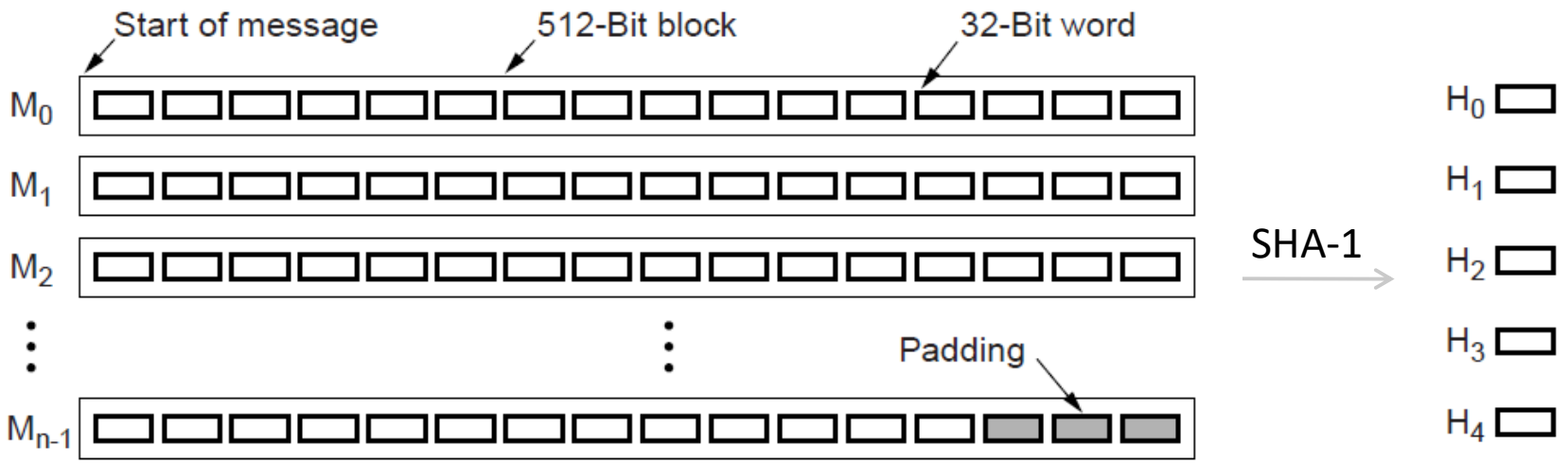
Message Digests (3)

In more detail: example of using SHA-1 message digest and RSA public key for signing nonsecret messages



Message Digests (4)

SHA-1 digests the message 512 bits at a time to build a 160-bit hash as five 32-bit components



Message in 512-bit blocks

Five 32-bit hashes output

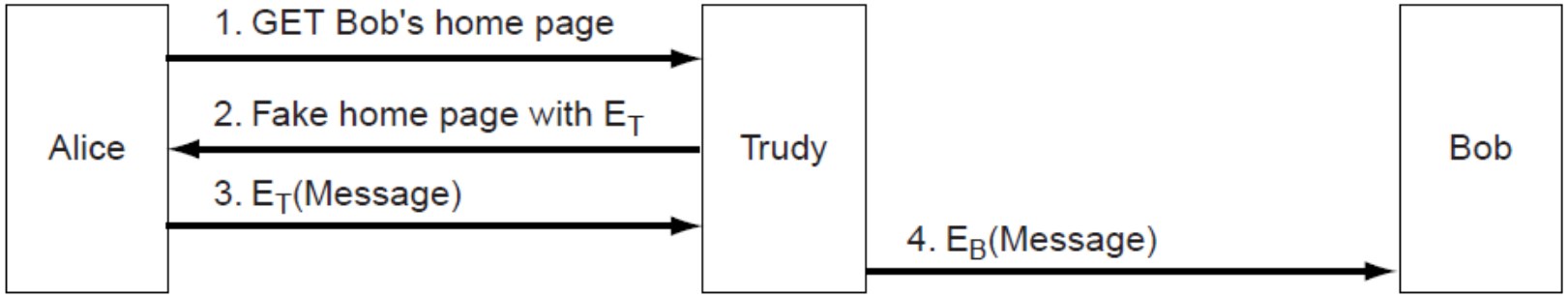
Management of Public Keys

We need a trusted way to distribute public keys

- Certificates »
- X.509, the certificate standard »
- Public Key infrastructures »

Management of Public Keys (1)

Trudy can subvert encryption if she can fake Bob's public key;
Alice and Bob will not necessarily know



Trudy replaces E_B with E_T and acts
as a "man in the middle"

Certificates

CA (Certification Authority) issues signed statements about public keys; users trust CA and it can be offline

I hereby certify that the public key 19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A belongs to Robert John Smith 12345 University Avenue Berkeley, CA 94702 Birthday: July 4, 1958 Email: bob@superdupernet.com
SHA-1 hash of the above certificate signed with the CA's private key

A possible certificate

X.509

X.509 is the standard for widely used certificates

- Ex: used with SSL for secure Web browsing

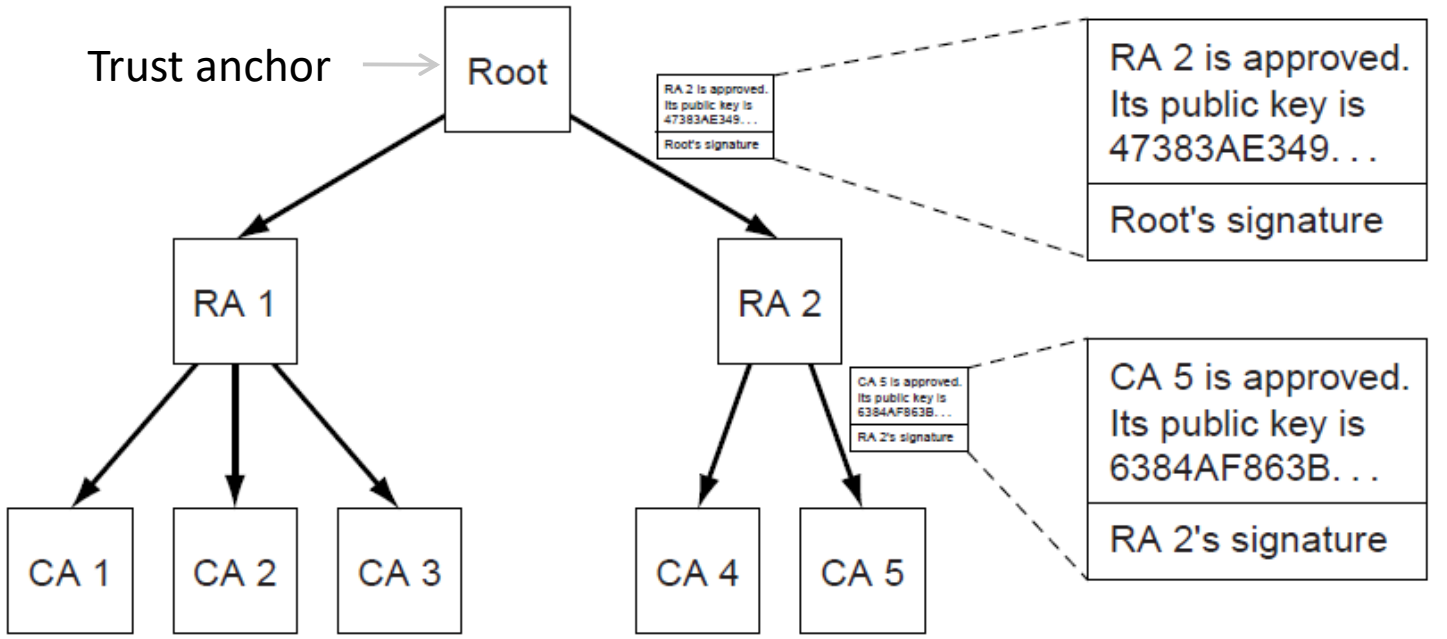
Field	Meaning
Version	Which version of X.509
Serial number	This number plus the CA's name uniquely identifies the certificate
Signature algorithm	The algorithm used to sign the certificate
Issuer	X.500 name of the CA
Validity period	The starting and ending times of the validity period
Subject name	The entity whose key is being certified
Public key	The subject's public key and the ID of the algorithm using it
Issuer ID	An optional ID uniquely identifying the certificate's issuer
Subject ID	An optional ID uniquely identifying the certificate's subject
Extensions	Many extensions have been defined
Signature	The certificate's signature (signed by the CA's private key)

Basic fields in X.509 certificates

Public Key Infrastructures (PKIs)

PKI is a system for managing public keys using CAs

- Scales with hierarchy, may have multiple roots
- Also need CRLs (Certificate Revocation Lists)



Hierarchical PKI

Chain of certificates for CA 5

Encryption Example - TLS

- 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 then uses symmetric key cryptography

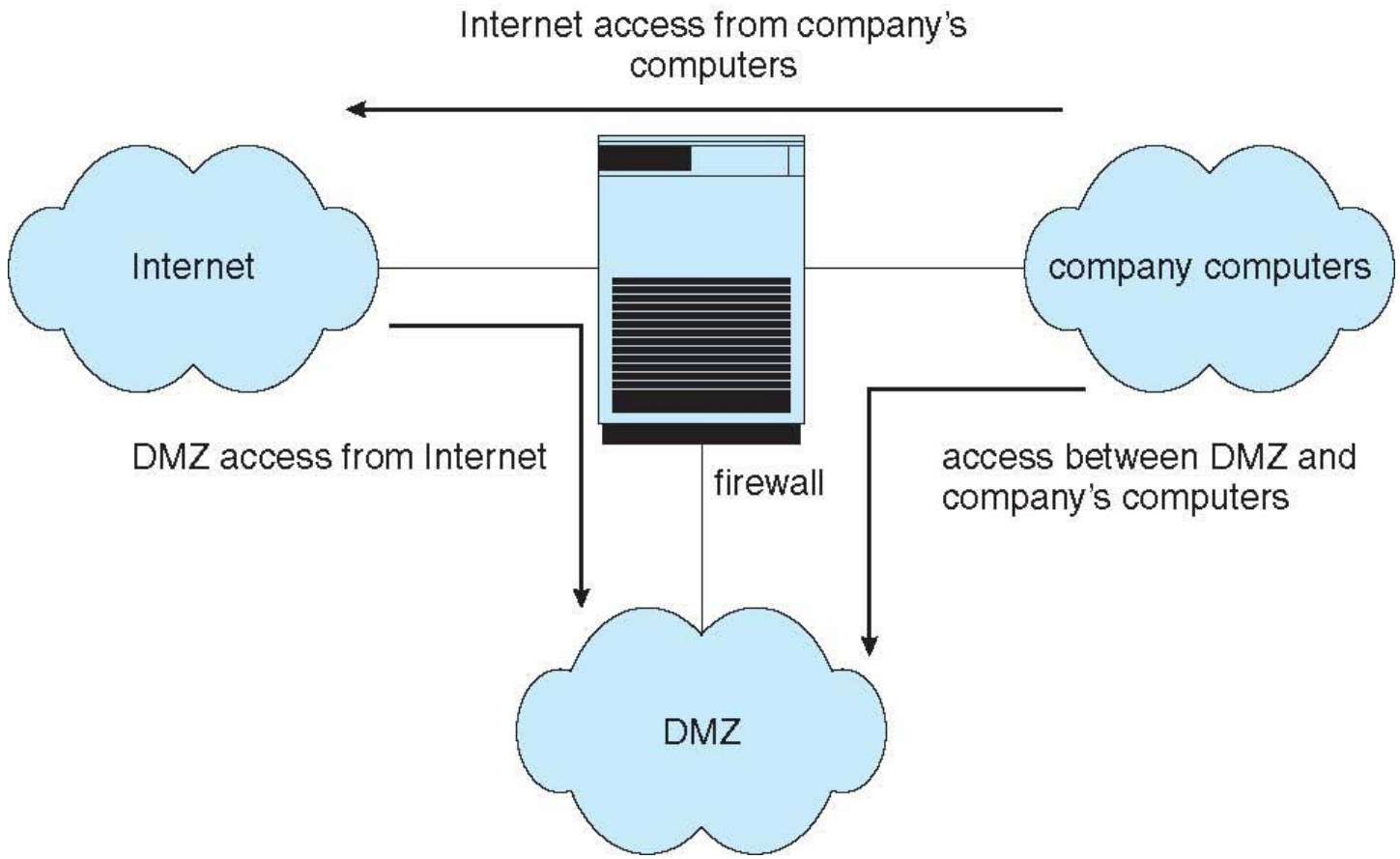
Implementing Security Defenses

- **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
  - Searching all programs or programs at execution for known virus patterns
  - Or run in **sandbox** so can't damage system
- Auditing, accounting, and logging of all or specific system or network activities
- Practice **safe computing** – avoid sources of infection, download from only "good" sites, etc

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



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

Security Defenses Summarized

- By applying appropriate layers of defense, we can keep systems safe from all but the most persistent attackers. In summary, these layers may include the following:
  - Educate users about safe computing—don't attach devices of unknown origin to the computer, don't share passwords, use strong passwords, avoid falling for social engineering appeals, realize that an e-mail is not necessarily a private communication, and so on
  - Educate users about how to prevent phishing attacks—don't click on email attachments or links from unknown (or even known) senders; authenticate (for example, via a phone call) that a request is legitimate
  - Use secure communication when possible
  - Physically protect computer hardware
  - Configure the operating system to minimize the attack surface; disable all unused services
  - Configure system daemons, privileges applications, and services to be as secure as possible

Security Defenses Summarized (cont.)

- Use modern hardware and software, as they are likely to have up-to-date security features
- Keep systems and applications up to date and patched
- Only run applications from trusted sources (such as those that are code signed)
- Enable logging and auditing; review the logs periodically, or automate alerts
- Install and use antivirus software on systems susceptible to viruses, and keep the software up to date
- Use strong passwords and passphrases, and don’t record them where they could be found
- Use intrusion detection, firewalling, and other network-based protection systems as appropriate
- For important facilities, use periodic vulnerability assessments and other testing methods to test security and response to incidents

Security Defenses Summarized (cont.)

- Encrypt mass-storage devices, and consider encrypting important individual files as well
- Have a security policy for important systems and facilities, and keep it up to date

Example: Windows 10

- 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

Example: Windows 7 (Cont.)

- Win added mandatory integrity controls – assigns **integrity label** to each securable object and subject
  - Subject must have access requested in discretionary access-control list to gain access to object
- Security attributes described by security descriptor
  - Owner ID, group security ID, discretionary access-control list, system access-control list
- Objects are either **container objects** (containing other objects, for example a file system directory) or **noncontainer objects**
  - By default an object created in a container inherits permissions from the parent object
- Some Win 10 security challenges result from security settings being weak by default, the number of services included in a Win 10 system, and the number of applications typically installed on a Win 10 system