Week 16 - Monday

COMP 4290

Last time

- What did we talk about last time?
- Ethical case studies

Questions?

Project 3

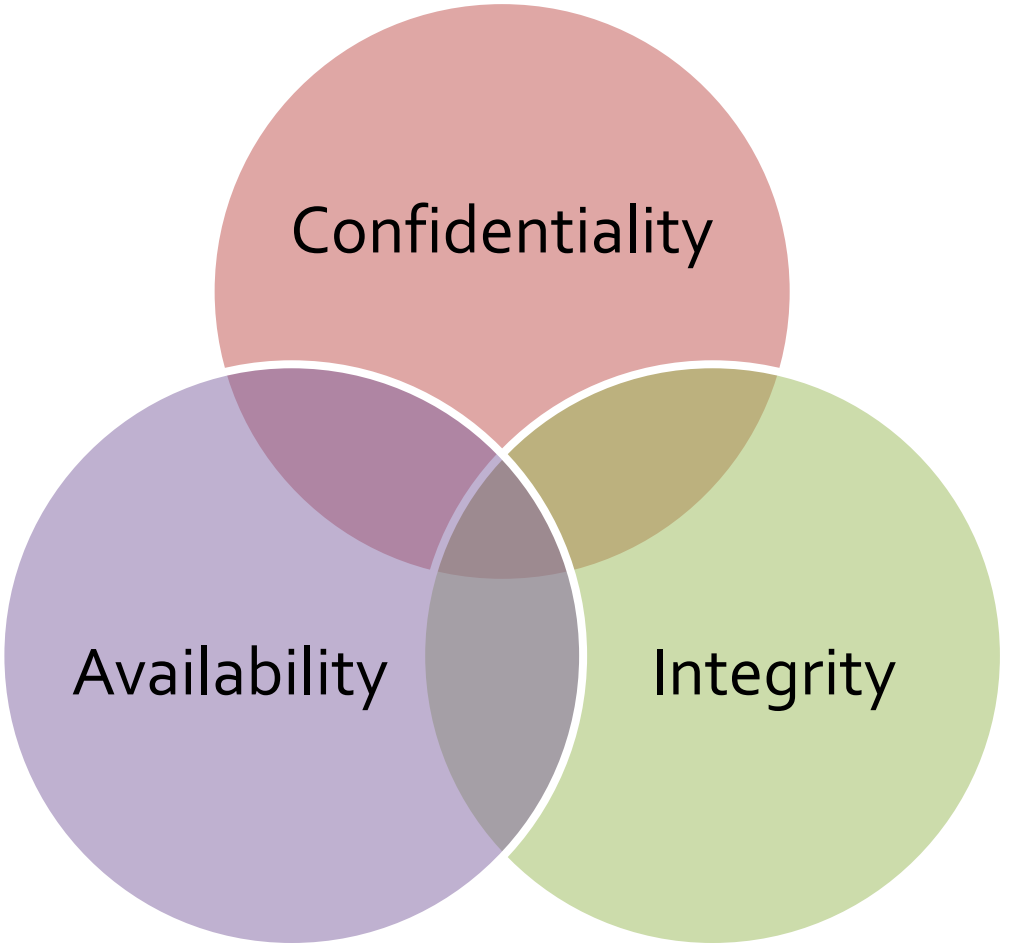
Anu Regmi Presents

Review

Final Exam

- Final exam:
  - Wednesday, 12/10/2025
  - 12:30 - 2:30 p.m.

The basics of computer security



Confidentiality

- You don't want other people to be able to read your stuff
  - Some of your stuff, anyway
- Cryptography, the art of encoding information so that it is only readable by those knowing a secret (key or password), is a principle tool used here
- Confidentiality is also called **secrecy** or **privacy**

Integrity

- You don't want people to mess up your stuff
- You want to know:
  - That your important data cannot be easily changed
  - That outside data you consider trustworthy cannot be easily changed either
- There are many different ways that data can be messed up, and every application has different priorities

Availability

- You want to be able to use your stuff
- Many attacks are based on **denial of service**, simply stopping a system from functioning correctly
- Availability can mean any of the following:
  - The service is present in usable form
  - There is enough capacity for authorized users
  - The service is making reasonable progress
  - The service completes in an acceptable period of time

Harm

Malicious, human-caused threats often involve one or more of the following kind of harm:

Interception	• Someone read something they weren't supposed to
Interruption	• Something became unavailable or unusable
Modification	• Someone changed something they weren't supposed to
Fabrication	• Someone created fake things

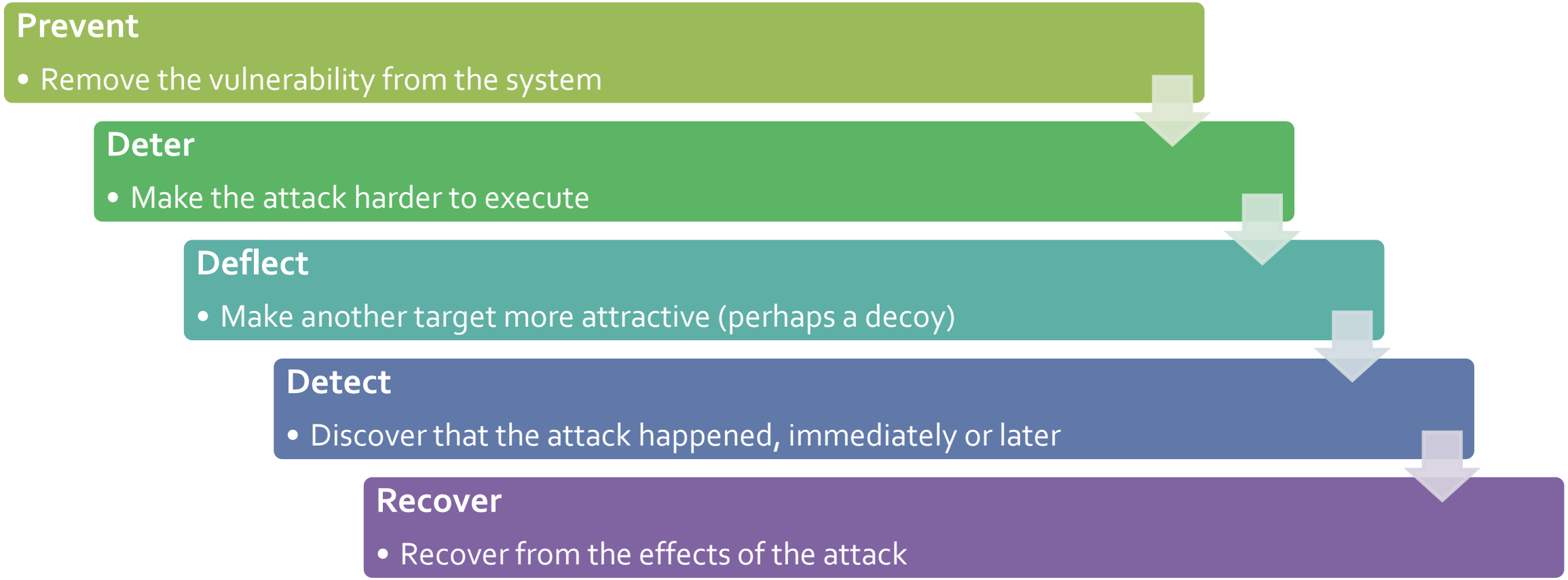
Method, opportunity, motive

- As with traditional crime, a computer attacker must have three things:

Method	• Skills and tools to perform the attack
Opportunity	• Time and access to accomplish the attack
Motive	• A reason to perform the attack

Controls

- There are five common ways of controlling attacks, many of which can be used together



Authentication

Definition of authentication

- **Authentication** is the binding of an identity to a subject
  - Example: Bill Gates (external entity) is a registered user whose identity on this system is **`gatesw`** (identity of system subject)
- The external identity must provide information to authenticate based on
  1. What the entity knows (passwords)
  2. What the entity has (security badge)
  3. What the entity is (fingerprints or voice ID)

Passwords

- Passwords are one of the most common forms of authentication mechanisms based on what the entity knows
- The password represents **authentication information** that the user must know
- The system keeps **complementation information** that can be used to check the password
- Real systems generally do not store passwords in the clear but store hashes of them
- Unix chooses one of 4,096 different hash functions, hashes the password into an 11-character string, and then prepends 2 characters specifying which hash function was used

Attacking a password system

- A **dictionary attack** is an attack based on guessing the password from trial and error
  - A dictionary attack can work on the complementary information (hashes of passwords)
  - If this information is unavailable, a dictionary attack can directly attack the authentication functions (literally trying to log in repeatedly)
- Let ***P*** be the probability that an attacker guesses the password over a certain span of time
- Let ***G*** be the number of guesses that can be made per unit time
- Let ***T*** be the number of time units of guessing
- Let ***N*** be the number of possible passwords
- Then,

$$P \geq \frac{TG}{N}$$

Defending authentication functions

- **Backoff**
  - Force the user to wait longer and longer between failed authentication techniques
  - Exponential backoff means that the first time waits 1 second before allowing a user to log in, the second waits 2 seconds, the third waits 4 seconds, etc.
- **Disconnection**
  - If the connection is remote and requires significant time to connect (dialing, VPN, etc.), the system can simply break connection after a number of failed attempts
- **Disabling**
  - With ***n*** failed attempts, an account is locked until an administrator resets the account
- **Jailing**
  - In jailing, the user is allowed to enter a fake system that looks like the real one
  - In theory, jailing can be used to learn more about an attacker's goals
  - Attractive data (called honeypots) can be made available, tempting the attacker to spend more time on the system (until he can be caught)

Biometrics

- Biometrics means identifying humans by their physical and biological characteristics
- This technology is often seen in spy and science fiction movies
  - It does exist, but it is far from perfect
- Like passwords, the actual biometric scans are usually not stored
  - Instead specific features are stored for later comparison
- Biometrics pose unique privacy concerns because the information collected can reveal health conditions

Problems with biometrics

- People assume that they are more secure than they are
- Attacks:
  - Fingerprints can be lifted off a champagne glass
  - Voices can be recorded
  - Iris recognition can be faked with special contact lenses
- Both false positives and false negatives are possible
- Disabilities can prevent people from using some kinds of biometrics
- It's possible to tamper with transmission from the biometric reader
- Biometric characteristics can change
- Identical twins sometimes pose a problem

Tokens

- Tokens are physical objects you possess
  - Keys
  - Badges
  - Cell phones
  - RFIDs
- **Passive tokens** take no action and do not change
  - Example: photo ID
- **Active tokens** change or interact with surroundings
  - Examples: RFID or magnetic card

Access Control

Access control

- **Subjects** are human users or programs that are executing on their behalf
- **Objects** are things that actions can be performed on
  - Files
  - Database fields
  - Directories
  - Hardware devices
- **Access modes** are the different ways that access can be done: read, write, modify, delete, etc.
- **Access control** is the process of managing the access modes that subjects can have on objects

Access control goals

- Check every access
  - The user may no longer have rights to a resource
  - The user may have gained rights
- Enforce least privilege
  - **Least privilege** means you get the bare minimum to get your job done
- Verify acceptable usage
  - Access to an object is not enough: Some actions might be legal and others illegal

Access control matrix example

	Objects			
Subjects	file 1	file 2	process 1	process 2
process 1	*read, write, own*	*read*	*read, write, execute, own*	*write*
process 2	*append*	*read, own*	*read*	*read, write, execute, own*

Cryptography

Cryptography

- "Secret writing"
- The art of encoding a message so that its meaning is hidden
- **Cryptanalysis** is breaking those codes

Encryption and decryption

- **Encryption** is the process of taking a message and encoding it
- **Decryption** is the process of decoding the code back into a message
- A **plaintext** is a message before encryption
- A **ciphertext** is the message in encrypted form
- A **key** is an extra piece of information used in the encryption process

Notation

- A plaintext is M (sometimes P)
- A ciphertext is C
- The encryption function $E(x)$ takes M and converts it into C
  - $E(M) = C$
- The decryption function $D(x)$ takes C and converts it into M
  - $D(C) = M$
- We sometimes specify encryption and decryption functions $E_k(x)$ and $D_k(x)$ specific to a key k

Attacks

- Cryptography is supposed to prevent people from reading certain messages
- Thus, we measure a **cryptosystem** based on its resistance to an **adversary** or **attacker**
- Kinds of attacks:
  - **Ciphertext only:** Attacker only has access to an encrypted message, with a goal of decrypting it
  - **Known plaintext:** Attacker has access to a plaintext and its matching ciphertext, with a goal of discovering the key
  - **Chosen plaintext:** Attacker may ask to encrypt any plaintext, with a goal of discovering the key
  - Others, less common

Terminology remix

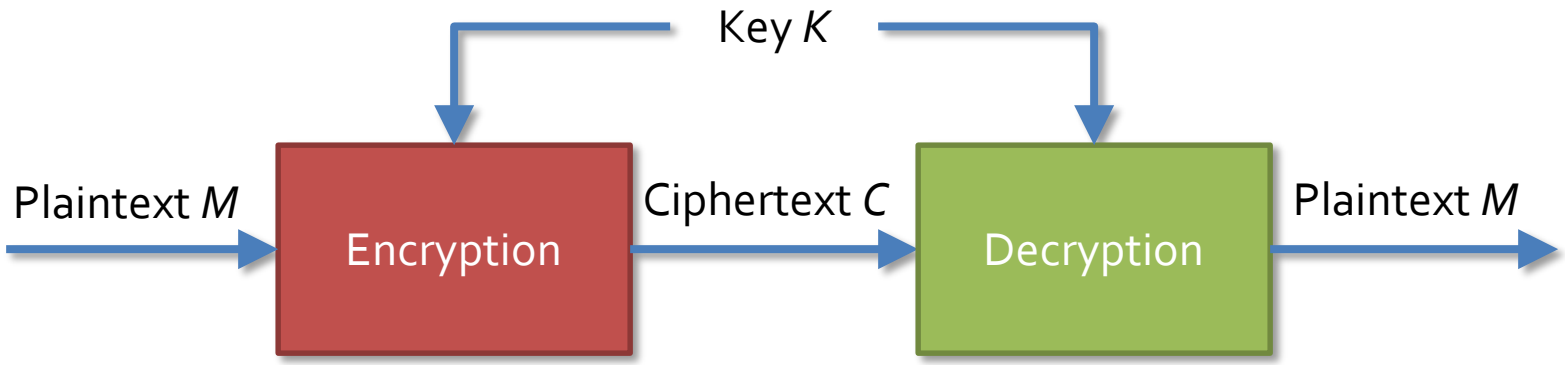
- The previous slide gives the book's terminology
- Rather than use letters, a system popularized by Ron Rivest is to use **Alice** and **Bob** as the two parties communicating
  - **Carl** or another "C" name can be used if three people are involved
- **Trent** is a trusted third party
- **Eve** is used for an evil user who often eavesdrops
- **Mallory** is used for a malicious user who is usually trying to modify messages

Encryption algorithms

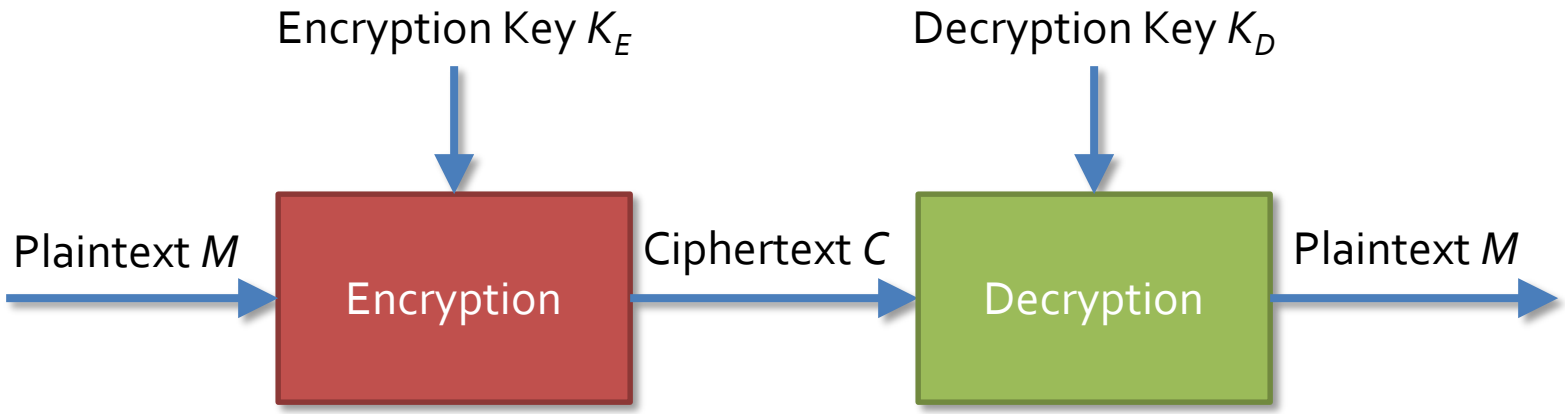
- The algorithms for encryption often rely on a secret piece of information, called a key
- We can notate the use of a specific key in either of the two following ways:
  - $C = E_K(M)$
  - $C = E(K, M)$
- In symmetric (or private key) encryption, the encryption key and the decryption key are the same
- In asymmetric (or public key) encryption, the encryption key and the decryption key are different

Symmetric vs. asymmetric

Symmetric Encryption



Asymmetric Encryption



Cryptanalysis

- There are two kinds of security for encryption schemes
  - **Unconditionally secure**
    - No matter how much time or energy an attacker has, it is impossible to determine the plaintext
  - **Computationally secure**
    - The cost of breaking the cipher exceeds the value of the encrypted information
    - The time required to break the cipher exceeds the useful lifetime of the information
- We focus on computationally secure, because there is only one practical system that is unconditionally secure

- "I want them to remain secret for as long as men are capable of evil" -Avi from Cryptonomicon

Review of Modular Arithmetic

- Modulo operator takes the remainder
- Two numbers are said to be congruent modulo n if they have the same remainder when divided by n
- For example,

  $39 \equiv 3 \pmod{12}$
- Addition, subtraction, and multiplication:
  - $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
  - $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
  - $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

Divided and Conquered

- We can't actually divide
- Instead, we have to find the multiplicative inverse
- The multiplicative inverse of ***x*** exists if and only if ***x*** is relatively prime to ***n***
- $13 \cdot 5 \equiv 65 \equiv 1 \pmod{16}$
- So, 13 and 5 are multiplicative inverses mod 16
- But, 0, 2, 4, 6, 8, 10, and 12 do not have multiplicative inverses mod 16

Definition

- A shift cipher encrypts a message by shifting all of the letters down in the alphabet
- Using the Latin alphabet, there are 26 (well, 25) possible shift ciphers
- We can model a shift cipher by numbering the letters A, B, C, … Z as 0, 1, 2, … 25
- Then, we let the key ***k*** be the shift
- For a given letter ***x***:

$E_k(x) = (x + k) \bmod 26$

Cryptanalysis of a Shift Cipher

- Cryptanalysis of a shift cipher is incredibly easy
- You just have to try 26 possibilities to be sure you have the right one
- A shift cipher is a simplified version of a **substitution cipher**

Definition

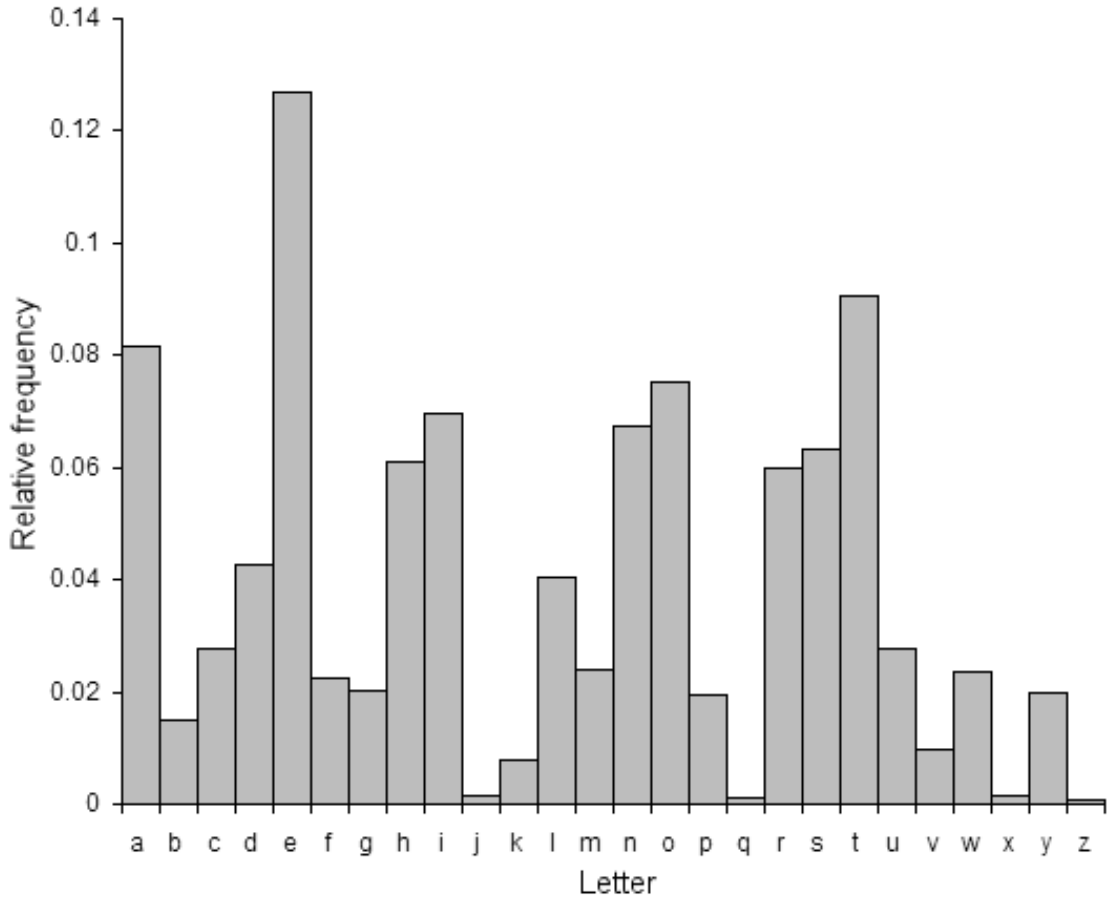
- In a transposition cipher, the letters are reordered but their values are not changed
- Any transposition cipher is a permutation function of some kind

Substitution ciphers

- **Substitution ciphers** cover a wide range of possible ciphers, including the shift cipher
- In a substitution cipher, each element of the plaintext is substituted for some corresponding element of the ciphertext
- **Monoalphabetic** substitution ciphers always use the same substitutions for a letter (or given sequence of letters)
- **Polyalphabetic** substitution ciphers use different substitutions throughout the encryption process

Frequency Attack

- English language defeats us
- Some letters are used more frequently than others: ETAOINSHRDLU
- Longer texts will behave more consistently
- Make a histogram, break the cipher



Vigenère cipher

- The Vigenère cipher is a form of polyalphabetic substitution cipher
- In this cipher, we take a key word and repeat it, over and over, until it is as long as the message
- Then, we add the repetitions of keywords to our message mod 26

Cryptanalysis of Vigenère

- The index of coincidence measures the differences in the frequencies in the ciphertext
- It is the probability that two randomly chosen letters from the ciphertext are the same
- IC = $$\frac{1}{N(N-1)} \sum_{i=0}^{25} F_i(F_i - 1)$$

Period	1	2	3	4	5	10	Large
Expected IC	0.066	0.052	0.047	0.045	0.044	0.041	0.038

Normalized index of coincidence

- Some systems look at a "normalized" index of coincidence, which is found by multiplying the formula given on the previous page by the number of letters in the language
  - 26 for English
  - When reading the literature, both normalized and unnormalized versions can be called index of coincidence
- Here are index of coincidence values for a few common languages

Language	Index
English	1.73
French	2.02
German	2.05
Italian	1.94
Portuguese	1.94
Russian	1.76
Spanish	1.94

After the length is known …

- The rest is easy
- Try various shifts for each letter of the key so that high frequency letters (E, T, A) occur with high frequency and low frequency letters (Q, X, Z) occur with low frequency
- Guess and check

Perfect secrecy

- A One-Time Pad has the property of **perfect secrecy** or **Shannon secrecy**
- Perfect secrecy means that $\mathbf{P}(\boldsymbol{M}) = \mathbf{P}(\boldsymbol{M}|\boldsymbol{C})$
  - Remember that it is possible to find a key that would decrypt a ciphertext to **any** plaintext
- Thus, learning the ciphertext tells you **nothing** about the plaintext

One-Time Pad weaknesses

- You can only use it one time
  - Otherwise, recovering the key is trivial
  - Completely vulnerable to known plaintext attack
- The key is as long as the message
- If you have a way of sending a key that long securely, why not send the message the same way?
- Generating keys with appropriate levels of randomness presents a problem

Stream and block ciphers

- A common way of dividing ciphers is into **stream ciphers** and **block ciphers**
- Block ciphers divide messages into fixed length parts (or blocks) and encipher each part with the same key
- Stream ciphers encipher each message character by character
  - Some other authors define a stream cipher to be like a block cipher except that the key changes with each block based on the message

Confusion and Diffusion

- Confusion is the property of a cryptosystem that changing a single character in the plaintext should not have a predictable effect
- Diffusion is the property of a cryptosystem that each character in the plaintext should impact many characters in the ciphertext
- Examples:
  - Caesar cipher has poor confusion and no diffusion
  - One time pad has good confusion but no diffusion
  - Auto-key ciphers may have poor confusion but good diffusion
  - AES and DES have good confusion and diffusion within a block

DES

- **D**ata **E**ncryption **S**tandard
- DES is a typical block cipher
- It was chosen as the government's standard for encryption in 1976 (but has since been deprecated)
- DES works on blocks 64 bits in size
- DES uses a 56-bit key
- NSA helped design it … amidst some controversy

DES strengths

- DES is fast
- Easy to implement in software or hardware
- Encryption is the same as decryption
- Triple DES is still standard for many financial applications
- Resistant to differential and linear cryptanalysis (2^{47} and 2^{43} known pairs required, respectively)

DES weaknesses

- Short key size
  - Brute force attack by EFF in 1998 in 56 hours then in 1999 in just over 22 hours
  - Brute force attack by University of Bochum and Kiel in 9 days in 2006 (but, using a machine costing only $10,000)
- If you could check 1,000,000,000 keys per second (which is unlikely with a commodity PC), it would take an average of 417 days to recover a key

Triple DES

- Although susceptible to a brute force attack, DES has no other major weaknesses
  - Double DES can be defeated by an extension of the brute force attack
  - What about triple DES?
- Let $E_K(X)$ and $D_K(X)$ be encryption and decryption using DES with key K
- Triple DES uses keys $K1$, $K2$, and $K3$
  - $C = E_{K1}(D_{K2}(E_{K3}(M)))$
  - Setting $K1 = K2 = K3$ allows for compatibility with single DES systems
- Triple DES is still a standard for financial transactions with no known practical attacks

AES

- **A**dvanced **E**ncryption **S**tandard
- Block cipher designed to replace DES
- Block size of 128 bits
- Key sizes of 128, 192, and 256 bits
- Like DES, has a number of rounds (10, 12, or 14 depending on key size)
- Originally called Rijndael, after its Belgian inventors
- Competed with 14 other algorithms over a 5 year period before being selected by NIST

AES pros and cons

- Strengths
  - Strong key size
  - Fast in hardware and software
  - Rich algebraic structure
  - Well-studied, open standard
- Weaknesses
  - Almost none
  - A few theoretical attacks exist on reduced round numbers of AES
  - No practical attacks other than side channel attacks

Public key cryptography

- Sometimes, we need something different
- We want a **public key** that anyone can use to encrypt a message to Alice
- Alice has a **private key** that can decrypt such a message
- The **public key** can only encrypt messages; it cannot be used to decrypt messages

RSA Algorithm

- Named for **R**ivest, **S**hamir, and **A**dleman
- Take a plaintext ***M*** converted to an integer

- Create an ciphertext ***C*** as follows:
  $C = M^e \bmod n$

- Decrypt ***C*** back into ***M*** as follows:
  $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$

The pieces

Term	Details	Source
M	Message to be encrypted	Sender
C	Encrypted message	Computed by sender
n	Modulus, $n = pq$	Known by everyone
p	Prime number	Known by receiver
q	Prime number	Known by receiver
e	Encryption exponent	Known by everyone
d	Decryption exponent	Computed by receiver
$\phi(n)$	Totient of n	Known by receiver

How it Works

- To encrypt:
  $C = M^e \bmod n$
- e could be 3 and is often 65537, but is always publically known
- To decrypt:
  $M = C^d \bmod n = M^{ed} \bmod n$
- We get d by finding the multiplicative inverse of e mod $\phi(n)$
- So, $ed \equiv 1 \pmod{\phi(n)}$

Why it Works

- We know that $ed \equiv 1 \pmod{\phi(n)}$
- This means that $ed = k\phi(n) + 1$ for some nonnegative integer k
- $M^{ed} = M^{k\phi(n)+1} \equiv M \cdot (M^{\phi(n)})^k \pmod{n}$
- By Euler's Theorem
  $M^{\phi(n)} \equiv 1 \pmod{n}$
- So, $M \cdot (M^{\phi(n)})^k \equiv M \pmod{n}$

Notation for sending

- We will refer to several schemes for sending data
- Let ***X*** and ***Y*** be parties and ***Z*** be a message
- { ***Z*** } ***k*** means message Z encrypted with key k
- Thus, our standard notation will be:
  - $X \rightarrow Y$: { ***Z*** } ***k***
  - Which means, ***X*** sends message ***Z***, encrypted with key ***k***, to ***Y***
- ***X*** and ***Y*** will be participants like Alice and Bob and ***k*** will be a clearly labeled key
- ***A*** || ***B*** means concatenate message ***A*** with ***B***

Kinds of keys

- Typical to key exchanges is the idea of interchange keys and session keys
- An **interchange key** is a key associated with a particular user over a (long) period of time
- A **session key** is a key used for a particular set of communication events
- Why have both kinds of keys?

Key exchange criteria

- To be secure, a key exchange whose goal is to allow secret communication from Alice to Bob must meet this criteria:
  1. Alice and Bob cannot transmit their key unencrypted
  2. Alice and Bob may decide to trust a third party (Cathy or Trent)
  3. Cryptosystems and protocols must be public, only the keys are secret

Classical exchange: Attempt 0

- If Bob and Alice have no prior arrangements, classical cryptosystems require a trusted third party Trent
- Trent and Alice share a secret key k_{Alice} and Trent and Bob share a secret key k_{Bob}
- Here is the protocol:
  1. Alice $\rightarrow$ Trent: {request session key to Bob} k_{Alice}
  2. Trent $\rightarrow$ Alice: $\{ k_{session} \} k_{Alice} \,||\, \{ k_{session} \} k_{Bob}$
  3. Alice $\rightarrow$ Bob: $\{ k_{session} \} k_{Bob}$

What's the problem?

- Unfortunately, this protocol is vulnerable to a replay attack
- (Evil user) Eve records $\{ k_{session} \} k_{Bob}$ sent in step 3 and also some message enciphered with $k_{session}$ (such as "Deposit $500 in Dan's bank account")
- Eve can send the session key to Bob and then send the replayed message
- Maybe Eve is in cahoots with Dan to get him paid twice
- Eve may or may not know the contents of the message she is sending
- The real problem is no **authentication**

Public key exchange

- Suddenly, the sun comes out!
- Public key exchanges should be really easy
- The basic outline is:
  1. Alice $\rightarrow$ Bob: $\{ k_{session} \} e_{Bob}$
- e_{Bob} is Bob's public key
- Only Bob can read it, everything's perfect!
- Except …
- There is still no authentication

Easily fixable

- Alice only needs to encrypt the session key with her private key
- That way, Bob will be able to decrypt it with her public key when it arrives
- New protocol:
  1. Alice $\rightarrow$ Bob: $\{\{ \boldsymbol{k}_{session} \} \boldsymbol{d}_{Alice} \} \boldsymbol{e}_{Bob}$
- Man in the middle attacks are still possible if Alice gets the wrong public key for Bob

Hash Functions

Catch-22

- Your computer needs to be able read the password file to check passwords
- But, even an administrator shouldn't be able to read everyone's passwords
- Hash functions to the rescue!

Definition

- A **cryptographic** (or one-way) hash function (called a cryptographic checksum in the book) takes a variable sized message ***M*** and produces a fixed-size hash code **H**(***M***)
- **Not the same as hash functions from data structures**
- The hash code produced is also called a **digest**
- It can be used to provide authentication of both the integrity and the sender of a message
- It allows us to store some information about a message that an attacker cannot use to recover the message

Crucial properties

Preimage Resistance

- Given a digest, should be hard to find a message that would produce it
- One-way property

Second Preimage Resistance

- Given a message m, it should be hard to find a different message that has the same digest

Collision Resistance

- Should be hard to find any two messages that hash to the same digest (collision)

Additional properties

Avalanching	• A small change in input should correspond to a large change in output
Applicability	• Hash function should work on a block of data of any size
Uniformity	• Output should be a fixed length
Speed	• It should be fast to compute a digest in software and hardware • No longer than retrieval from secondary storage

Salt

- If you are the administrator of a large system, you might notice that two people have the same password hash
- With people's password habits, the odds are very high that their passwords are the same
- To add to the semantic security of such schemes extra data called **salt** is added to the end of a password
- The salt is usually based on the time the account was created or the account name

MD5

- Message Digest Algorithm 5
- Very popular hashing algorithm
- Designed by Ron Rivest (of RSA fame)
- **Digest size:** 128 bits
- **Security**
  - Completely broken
  - Reasonable size attacks (2^{32}) exist to create two messages with the same hash value
- MD5 hashes are still commonly used to check to see if a download finished without error

SHA family

- **Secure Hash Algorithm**
- Created by NIST
- SHA-0 was published in 1993, but it was replaced in 1995 by SHA-1
- The difference between the two is only a single bitwise rotation, but the NSA said it was important
- Digest size: 160 bits
- Security
  - Broken if you have the resources
  - Theoretical attacks running in 2^{51} - 2^{57} time exist
  - Google generated two PDF files with the same hash in just over 2^{63} hashes in 2017
- SHA-2 is a successor family of hash functions
  - 224, 256, 384, 512 bit digests
  - Much better security
  - Designed by the NSA

Keccak (SHA-3)

- Keccak uses a completely different form of hashing than SHA-0, SHA-1, and SHA-2
- Although there are only theoretical attacks on SHA-1 and no real attacks on SHA-2, the attacks on SHA-0 made people nervous about hash functions following the same design
- Keccak also allows for variable size digests, for added security
  - 224, 256, 384, and 512 are standard for SHA-3, but it is possible to go arbitrarily high in Keccak

General case

- If we care about a group of *k* items which can have a value between 1 and *n*, the probability that two are the same is:

$$P(n,k) = 1 - \frac{n!}{(n-k)!\,n^k}$$

- Because this form is a little unwieldy, we have an approximation that is easier to punch into a calculator:

$$P(n,k) > 1 - e^{\frac{-k(k-1)}{2n}}$$

Count it up

- If we want to find the number of items needed before there is greater than a $\frac{1}{2}$ probability of collision we get:

$$\frac{1}{2} = 1 - e^{\frac{-k(k-1)}{2n}}$$

$$2 = e^{\frac{k(k-1)}{2n}}$$

$$\ln(2) = \frac{k(k-1)}{2n}$$

- For large k, $k(k\text{-}1) \approx k^2$, giving:

$$k \approx \sqrt{2\ln(2)\, n} \approx 1.18\sqrt{n}$$

The lesson?

- Use hash functions with a long digest
- A hash function with an m-bit digest can produce about 2^m different hashes
  - You need to hash about 2^{m-1} messages for a 50% chance of finding a preimage or second preimage if the hash function can't be attacked another way
  - Because of the birthday paradox, to have a 50% chance of finding a collision (*any* two messages with the same hash), you only need to hash around $2^{m/2}$ different messages

Quantum Cryptography

Quantum cryptography

- When people talk about quantum computers in the context of cryptography, they're usually talking about one of two very different things:
  - Breaking cryptography with quantum computers
  - Using quantum mechanics to send secret messages

Shor's algorithm

- Shor's algorithm is an algorithm invented by Peter Shor in 1994 to factor integers
- With enough qubits, Shor's algorithm can factor an integer in $\mathbf{O}((\log N)^2(\log\log N)(\log\log\log N))$ time
  - In other words, polynomial in the length of the number N that is being factored
- RSA depends on the difficulty of factoring
- Shor's algorithm can be adapted to solve the discrete log problem as well

What a quantum computer could do

- Break all popular **public key** cryptography
  - The RSA, El Gamal, and Elliptic Curve public key systems can all be broken by Shor's algorithm
  - But there are some other systems out there that are not known to be vulnerable to quantum algorithms
- Get a quadratic speedup when trying to brute force symmetric ciphers like AES
  - Meaning that $\sqrt{2^n} = 2^{\frac{n}{2}}$ attempts might be needed instead of 2^n
  - Suggests that key lengths should be doubled

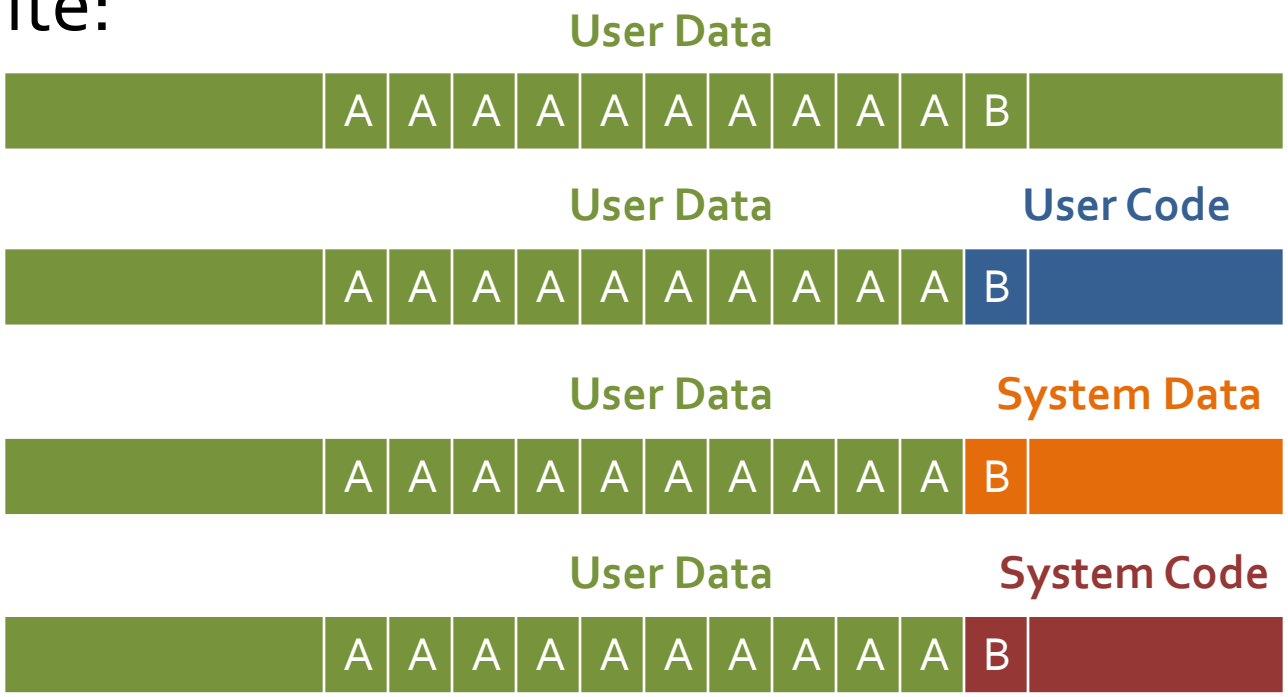
Quantum communication

- The other (and possibly more useful) way to use quantum mechanics is as a way to send secret information
- To do so, Sam (the sender) sends Ruth (the receiver) photons
- What's really remarkable about this kind of quantum cryptography is that no one can eavesdrop on it

Program Security

Buffer overflow

- A **buffer overflow** happens when data is written past the end (or beginning) of an array
- It could overwrite:
  - User data
  - User code
  - System data
  - System code



Incomplete mediation

- **Incomplete mediation** happens with a system does not have complete control over the data that it processes
- Example URL:
  - `http://www.security.com/query.php?date=2012March20`
- Wrong URL:
  - `http://www.security.com/query.php?date=2000Hyenas`
- The HTML generates the URL, but the URL can be entered manually

Time-of-check to time-to-use

- A **time-of-check to time-to-use flaw** is one where one action is requested, but before it can be performed, the data related to the action is changed
- The book's example is a man who promises to buy a painting for $100 who puts five $20 bills on the counter and pulls one back when the clerk is turning to wrap up the painting
- In this flaw, the first action is authorized, but the second may not be

Viruses

- Terminology is inconsistent
- Popular culture tends to call lots of things a virus
- Sometimes we will too, but here are some other terms
- Almost all of these are, by definition, Trojan horses
- Worms differ from viruses primarily because they spread across networks

Type	Characteristics
Virus	Attaches itself to a program and propagates copies of itself to other programs
Trojan horse	Contains unexpected, additional functionality
Logic bomb	Triggers action when condition occurs
Time bomb	Triggers action when specified time occurs
Trapdoor	Allows unauthorized access to functionality
Worm	Propagates copies of itself through a network
Rabbit	Replicates itself without limit to exhaust resources
Spyware	Covertly communicates user data or user activities
Ransomware	Transfers data offsite or encrypts it, demanding money for the data or decryption key

Where Viruses Live

- One-time execution
- Boot sector
  - The part of a hard drive that says what code to load to start your OS
- Memory resident
  - Sometimes called TSR (terminate and stay resident)
- Inside documents
- A few other places that are sensible:
  - Applications
  - Libraries
  - Compilers (infect programs as you create them)
  - Antivirus software

Virus Signatures

- Storage patterns
  - The size of a file
  - Compare against a hash digest for the program
- Execution patterns
  - Viruses are also suspicious because of the way they execute
  - The functioning of the code compared to some standard
  - Suspicious execution patterns (weird JUMP commands)

Polymorphic viruses

- Because virus scanners try to match strings in machine code, virus writers design **polymorphic viruses** that change their appearances
- **No-ops**, code that doesn't have an impact on execution, can be used for simple disguises
- Clever viruses can break themselves apart and hide different parts in randomly chosen parts of code
  - Similar to code obfuscation
- Advanced polymorphic viruses called **encrypting viruses** encrypt parts of themselves with randomly chosen keys
  - A scanner would have to know to decrypt the virus to detect it
- Virus scanners cannot catch everything

Targeted malicious code

- Trapdoors
  - A way to access functionality that is not documented
  - Often inserted during development for testing purposes
- Salami attacks
  - Steal tiny amounts of money when a cent is rounded in financial transactions
  - Or steal a few cents from millions of people
- Rootkits
- Privilege escalation
- Keystroke logging

Testing to prevent programming flaws

- **Unit testing** tests each component separately in a controlled environment
- **Integration testing** verifies that the individual components work when you put them together
- **Function** and **performance tests** sees if a system performs according to specification
- **Acceptance testing** give the customer a chance to test the product you have created
- The final **installation testing** checks the product in its actual use environment

Testing methodologies

- **Regression testing** is done when you fix a bug or add a feature
  - We have to make sure that everything that used to work still works after the change
- **Black-box testing** uses input values to test for expected output values, ignoring internals of the system
- **White-box** or **clear box testing** uses knowledge of the system to design tests that are likely to find bugs
- **You can only prove there are bugs. It's impossible to prove that there aren't bugs.**

Secure design principles

- Saltzer and Schroeder wrote an important paper in 1975 that gave 8 principles that should be used in the design of any security mechanisms
  1. Least privilege
  2. Fail-safe defaults
  3. Economy of mechanism
  4. Complete mediation
  5. Open design
  6. Separation of privilege
  7. Least common mechanism
  8. Psychological acceptability

Web Security – User Side

Browser security issues

- Browsers are how most of the world interacts with the Internet
- There are lots of problems when trying to maintain security:
  - Browsers often connect to more than just the URL listed in the address bar
  - Fetching a page automatically fetches lots of other data
  - If the browser is corrupted, you have no protection
  - Most browsers support plug-ins, which can be malicious or badly implemented
  - Browsers can access data on the user computer
  - The user does not know what data the browser is sending

Man-in-the-Browser

- The browser controls all the interactions with the world wide web
- If your browser has been compromised, it doesn't matter how good your encryption is
- The browser sees all the data before it is encrypted
- SilentBanker is an example of a plug-in that stole bank information
  - The banking websites still worked!

Page-in-the-middle

- A page-in-the-middle attack is one in which you are redirected to a page that looks like the one you wanted
  - For example, a copy of your banking website
- Such a page might be arrived at because of a phishing link or DNS cache poisoning
- A browser-in-the-middle attack is worse, since your browser is compromised and no websites can be trusted

Program download substitution

- A page could trick you into downloading a file that appears to be an application you want
  - In reality, it's a virus, Trojan horse, or other malware
- How do you know what you're downloading?
- Often, there's no way to be sure

User-in-the-middle

- A user-in-the-middle attack tricks an unsuspecting user to do something only a human can do, like solve a CAPTCHA
- Spam and porn companies often have the same owners
- People get offers for free porn in their e-mail, provided that they fill out a CAPTCHA
- This attack is not very damaging to the individual, but it wastes time and fills the world with more spam

Browser authentication issues

- We've already talked about how people authenticate
- One of the problems here is that **computers** are failing to authenticate
  - You're not sure that the site you're connecting to is really your bank
- The problem is hard because computers authenticate based almost entirely on what they know
  - It's possible to eavesdrop on such information

- Some banks let you to pick a picture and a caption

GOAT POLITICS

Authentication approaches

- Web authentication can be done with approaches beyond or in addition to a password
- Shared secret
  - Secret questions asked earlier
- One-time password
  - Password provided by an app or a SecurID
- Out-of-band communication
  - Sending a PIN and a credit card in separate mailings
  - Texting a one-time password to a registered cell phone

Defaced web site

- **Website defacement** is when an attacker changes the content of a legitimate website
- Usually, this is done by exploiting a weakness in authentication of the people who are allowed to update content
- These attacks can be pranks
- They can be done to demonstrate that security is poor
  - Often to embarrass government websites
- They can be done to show political disagreement with the website or the agency behind the website
- The changes could be subtle enough that the change is not noticed for a while

Fake website

- Websites are easy to fake
- By their nature, the HTML, JavaScript, CSS, and images used to create a website are all publically available
  - It's even possible to link to current images on the real website
- This attack is usually designed to trick users into entering private information into the malicious website

Protecting websites

- Detecting that a change has occurred on a website can be difficult
- One approach is to make a hash value of the website
  - Store the hash elsewhere, securely
  - Hash the contents of the website periodically to see if it still matches
  - This approach only works if the data doesn't constantly change
- **Digital signatures** allows companies to sign code to verify that they did originate the code
  - Example: ActiveX controls
  - You shouldn't be running this kind of code anyway

Web bugs

- Only a website you visit can leave a cookie or run JavaScript, right?
  - Sure, but how many sites do you visit?
- Images that are linked to other websites (especially ads) count as visiting other websites
- Visiting a single page could store cookies from every ad on the page (and more!)
- **Web bugs** are images that are usually 1 x 1 pixels and clear
  - They make it impossible to know how many sites could be storing cookies

Clickjacking

- **Clickjacking** is when you think you're clicking on one button, but you're really clicking on another
- It could be that you're agreeing to download or install a program that you don't think you are
  - Called a **drive-by download**
- It could be that you think you're entering data into a real website, but it's just a front for a malicious one
- These attacks are possible because web pages can have transparent frames, allowing you to see something that you're not really interacting with

Obtaining user or website data

- The inherently unsecure model used for web interactions has a number of weak points
- Some ways that website data can be leaked include:
  - Cross-site scripting
  - SQL injection
  - Dot-dot-slash
  - Server-side includes

Fake e-mail

- There is lots of fake e-mail out there
- The book calls **spam** fake or misleading e-mail
- Several kinds of spam are rising
  - Fake "Your message could not be delivered" messages
  - Fake social networking messages
  - Current events messages
  - Shipping notices
  - Threats to lose access to your account

Why do people send spam?

- Advertising black- or graymarket pharmaceuticals
- Pump and dump – artificially inflating the price of a stock
- General advertising
- Malware in the e-mail or in links from the e-mail
- Advertising sites (such as porn) that might be illegal
- Cost is virtually nothing

Dealing with spam

- Legal approaches
  - US CAN-SPAM act
  - Directive 2002/58/EC in Europe
  - It's hard to define what is and isn't spam
  - Most laws require an opt-out mechanism, but enforcement is hard
- IP addresses are easy to spoof, but the next generation Internet might change that
- Screening programs try to filter out spam (with both false positives and false negatives)
- Some web hosting companies enforce volume limitations on how many e-mails can be sent per day
- Paying postage per e-mail?

E-mail spoofing

- SMTP is the protocol for sending e-mail
- It's very straight-forward
- The **from** field is easy to spoof
- There are protocols with authentication built in, but regular SMTP is entrenched how
- You can never trust header information in an e-mail

Phishing

- **Phishing** is when an e-mail tries to trick someone into giving out private data or doing something else unsafe
- **Spear phishing** is phishing that targets a specific individual
  - Details about that user's life or accounts might be included
- **Whaling** is a term used for spear phishing of rich people or celebrities
  - They have more money
  - Many of their personal details could be public

OS Security

Separation

- OS security is fundamentally based on separation
  - **Physical separation:** Different processes use different physical objects
  - **Temporal separation:** Processes with different security requirements are executed at different times
  - **Logical separation:** Programs cannot access data or resources outside of permitted areas
  - **Cryptographic separation:** Processes conceal their data so that it is unintelligible

Memory protection

- Protecting memory is one of the most fundamental protections an OS can give
  - All data and operations for a program are in memory
  - Most I/O accesses are done by writing memory to various locations
- Techniques for memory protection
  - Fence
  - Base/bounds registers
  - Tagged architectures
  - Segmentation
  - Paging

Storing access control information

- Directory based approaches
  - Create a directory that lists all the objects a given user can access and their associated rights:
  - Problems:
    - Directories can become large
    - How is access revoked?
    - What if two files in different locations in the system have the same name?
- Access control lists
  - List all the users that have rights for a specific object
  - Most objects only have a few legal users
  - Wild cards can make the situation easier
- Access control matrices
  - Both directories and access control lists are equivalent
  - We can also imagine a matrix that holds all subjects and all objects
  - It is too inefficient for most systems to be implemented this way, but security researchers sometimes use this model for theoretical purposes

Access control matrix example

	Objects			
Subjects	file 1	file 2	process 1	process 2
process 1	*read, write, own*	*read*	*read, write, execute, own*	*write*
process 2	*append*	*read, own*	*read*	*read, write, execute, own*

Bell-LaPadula overview

- Confidentiality access control system
- Military-style classifications
- Uses a linear clearance hierarchy
- All information is on a **need-to-know** basis
- It uses **clearance** (or **sensitivity**) levels as well as project-specific **compartments**

Security clearances

- Both subjects (users) and objects (files) have security clearances
- Below are the clearances arranged in a hierarchy

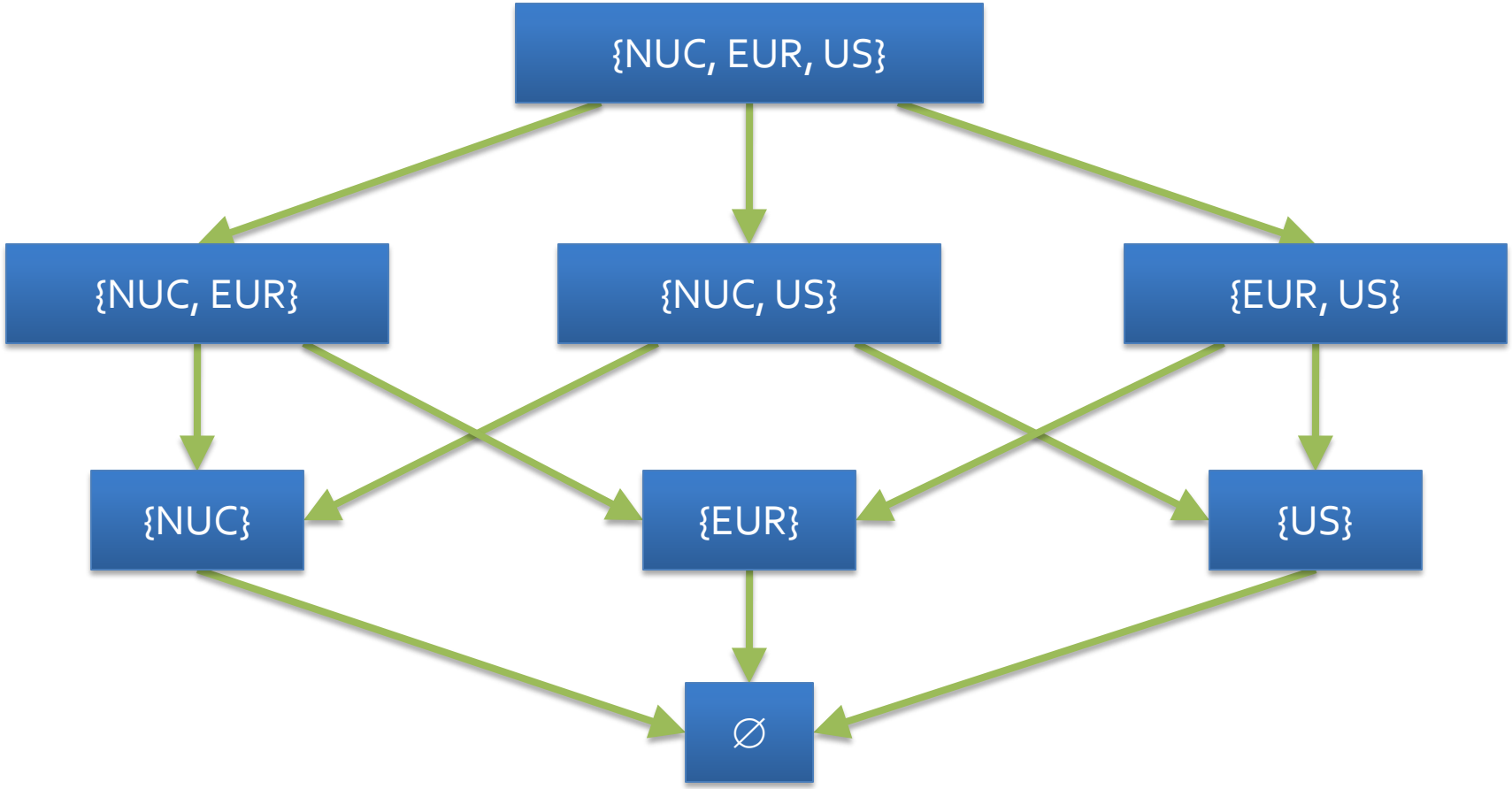
Clearance Levels	Sample Subjects	Sample Objects
Top Secret (TS)	Tamara, Thomas	Personnel Files
Secret (S)	Sally, Samuel	E-mail Files
Confidential (C)	Claire, Clarence	Activity Log Files
Restricted (R)	Rachel, Riley	Telephone List Files
Unclassified (UC)	Ulaley, Ursula	Address of Headquarters

Adding compartments

- We add compartments such as NUC = Non-Union Countries, EUR = Europe, and US = United States
- The possible sets of compartments are:
  - ∅
  - {NUC}
  - {EUR}
  - {US}
  - {NUC, EUR}
  - {NUC, US}
  - {EUR, US}
  - {NUC, EUR, US}
- Put a clearance level with a compartment set and you get a **security level**
- The literature does not always agree on terminology

Romaine lattice

- The subset relationship induces a **lattice**



Bell-La Padula properties

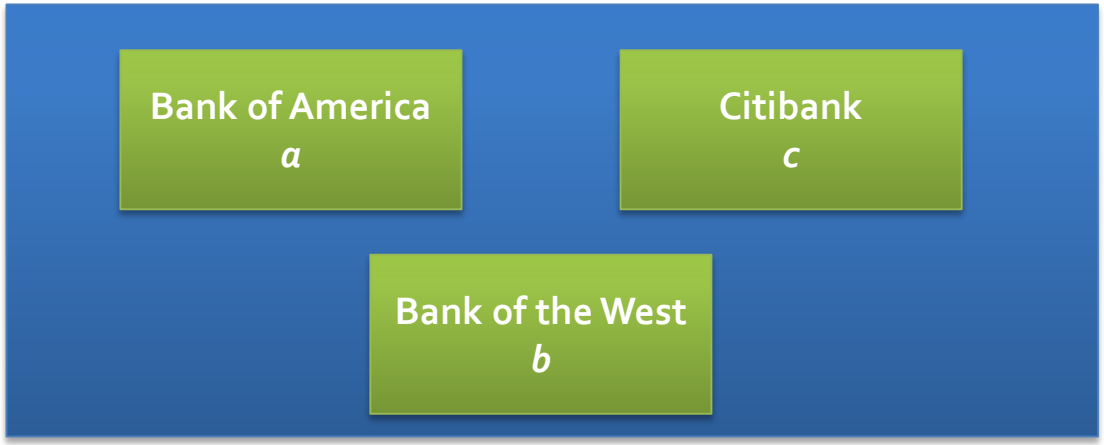
- Let L be a security level and C be a category
- We say that security level (L, C) **dominates** security level (L', C') if and only if $L' \leq L$ and $C' \subseteq C$
- Simple security requires (L_S, C_S) to dominate (L_O, C_O) and S to have read access
  - **Read down**
- *-property now requires (L_O, C_O) to dominate (L_S, C_S) and S to have write access
  - **Write up**

Chinese Wall model

- The Chinese Wall model respects both confidentiality and integrity
- It's very important in business situations where there are conflict of interest issues
- Real systems, including British law, have policies similar to the Chinese Wall model
- Most discussions around the Chinese Wall model are couched in business terms

COI Examples

Bank COI Class



Gasoline Company COI Class

Chinese Wall overview

- We can imagine the Chinese Wall model as a policy controlling access in a database
- The **objects** of the database are items of information relating to a company
- A **company dataset** (CD) contains objects related to a single company
- A **conflict of interest** (COI) class contains the datasets of companies in competition
- Chinese Wall rules prevent people from reading and writing data from CDs in different COIs

Biba model

- Integrity based access control system
- Uses integrity levels, similar to the clearance levels of Bell-LaPadula
- Precisely the **dual** of the Bell-LaPadula Model
- That is, we can only read up and write down
- Note that integrity levels are intended only to indicate integrity, **not** confidentiality
- Actually a measure of accuracy or reliability

Mandatory and discretionary access control

- **Mandatory access control** (**MAC**) means that the controls are enforced by rules in the system, not by user choices
  - Bell-La Padula is a perfect example of MAC
- **Discretionary access control** (**DAC**) means that the user has control over who can access the objects he or she owns
  - Linux and Windows are largely DAC systems
- Most real systems have elements of both

Network Security

Packet switched vs. circuit switched

- The Internet is a packet switched system
- Individual pieces of data (called packets) are sent on the network
  - Each packet knows where it is going
  - A collection of packets going from point A to point B might not all travel the same route
- Phone lines are circuit switched
  - This means that a specific circuit is set up for a specific communication
  - Operators used to do this by hand
  - Now it is done automatically
  - Only one path for data

Network strength

- If a single cut can case a network to go down, that network is vulnerable to a **single point of failure**
- Most important networks like electrical systems have redundancy so that this doesn't happen to a whole city
  - **Resilience** or **fault tolerance**

Terminology

- A **computer network** is at least two computers connected together
  - Often one is a **server** and the other is a **client**
- A computer system in a network is called a **node**
- The processor in a node is called a **host**
- A connection between two hosts is a **link**

Network characteristics

- **Anonymity**: We don't know who we're dealing with
- **Automation**: Communication may be entirely between machines without human supervision
- **Distance**: Communications are not significantly impacted by distance
- **Opaqueness**: It is hard to tell how far away other users are and to be sure that someone claiming to be the same user as before is

Transmission media

- Copper wire
  - **Twisted pair** is a pair of insulated copper wires
  - **Coaxial cable** has a single wire surrounded by an insulation jacket covered by a grounded braid of wire
  - **Repeaters** or **amplifiers** are needed periodically to prevent signal degradation
- Optical fiber
  - Carries light instead of electricity
  - Higher bandwidth and less signal degradation than copper
  - Replacing aging copper lines
- Wireless
  - Good for short distance
  - Uses radio signals
- Microwave
  - Strong signals
  - Requires line of sight
- Infrared
  - Similar to microwave but weaker signals
- Satellites
  - Need **geosynchronous** orbits
  - Secure applications need smaller **footprints** than broadcasts

Layers

- Protocols and standards define each layer
- Not every layer is always used
- Sometimes user errors are referred to as Layer 8 problems

Layer	Name	Activity	Example
7	**Application**	User-level data	HTTP
6	**Presentation**	Data appearance, some encryption	TLS
5	**Session**	Sessions, sequencing, recovery	IPC and part of TCP
4	**Transport**	Flow control, end-to-end error detection	TCP
3	**Network**	Routing, blocking into packets	IP
2	**Data Link**	Data delivery, packets into frames, transmission error recovery	Ethernet
1	**Physical**	Physical communication, bit transmission	Electrons in copper

TCP/IP

- The OSI model is conceptual
- Most network communication uses TCP/IP
- We can view TCP/IP as four layers:

Layer	Action	Responsibilities	Protocol
Application	Prepare messages	User interaction	HTTP, FTP, etc.
Transport	Convert messages to packets	Sequencing, reliability, error correction	TCP or UDP
Internet	Convert packets to datagrams	Flow control, routing	IP
Physical	Transmit datagrams as bits	Data communication	

TCP/IP

- **Transmission Control Protocol (TCP)**
  - Creates a reliable communication session
  - Wraps information into packets
  - Uses **port** numbers to connect processes to information streams
- **Internet Protocol (IP)**
  - Allows for unreliable transport
  - Wraps packets into datagrams
  - Uses IP addresses for routing
- **User Datagram Protocol (UDP)**
  - Alternative to TCP that is unreliable but has low overhead

Reconnaissance

- A smart attacker learns everything he or she can about the system before attacking it
- Useful methods for reconnaissance of a network include:
  - Port scans
  - Social engineering
  - Dumpster diving
  - OS and application fingerprinting
  - Background research

Eavesdropping and wiretapping

- **Eavesdropping** means overhearing private information without much effort
  - Administrators need to periodically monitor network traffic
- **Wiretapping** implies that more effort is being used to overhear information
- **Passive wiretapping** is only listening to information
- **Active wiretapping** means that you may adding or changing information in the stream

Wiretapping

- If you are on the same LAN, you can use a **packet sniffer** to analyze packets
- **Inductance** allows you to measure the signals inside of a wire without a direct physical connection
- Wireless is broadcast
  - Easy to intercept, but can be protected by WPA or WPA2 encryption (and hardly at all by WEP)
- Microwave is easy to intercept
  - Heavy multiplexing makes it hard to untangle individual signals
- Satellites are similar (unsecure but heavily multiplexed)
- Optical fiber is very difficult to tap
  - Cutting a single fiber means recalibrating the network
  - Repeaters and taps that connect the fiber are the best places to attack

Authentication issues

- Passwords are often easy to guess
  - Because we're bad at picking passwords
  - Because the user may not have realized that the machine would be exposed to network attacks
- Passwords are sent in the clear
- Bad hashes can give information about the password
- Sometimes buffer overflows can crash the authentication system
- Sometimes authentication is not needed
  - **`.rhosts`** and **`.rlogin`** files in Unix
  - Guest accounts
- Default passwords on routers and other devices that never get changed

Authentication attacks

- **Spoofing** is when an attacker carries out one end of a networked exchange
- A **masquerade** is spoofing where a host pretends to be another host
  - URL confusion: someone types **`hotmale.com`** (don't go there!) or **`gogle.com`**
- **Phishing** is a form of masquerading
- **Session hijacking** (or **sidejacking**) is carrying on a session started by someone else
  - Login is encrypted, the rest of the data often isn't
  - Firesheep allows you to log on to other people's Facebook and Twitter accounts in, say, the same coffeeshop
- **Man-in-the-middle** attacks

Confidentiality threats

- Misdelivery
  - Data can have bad addresses, occasionally because of computer error
  - Human error (e.g. James Hughes (student) instead of James Hughes (professor)) is more common)
- Exposure of data can happen because of wiretapping or unsecure systems anywhere along the network
- Traffic flow analysis
  - Data might be encrypted
  - Even so, it is very hard to hide where the data is going to and where it is coming from
  - Tor and other anonymization networks try to fix this

Integrity threats

- Attackers can falsify some or all of a message, using attacks we've talked about
  - Parts of messages can be combined
  - Messages can be redirected or deleted
  - Old messages can also be replayed
- Noise can degrade the signals
  - All modern network protocols have error correction built in
- Malformed packets can crash systems
- Protocols often have vulnerabilities

Denial of service

- Networks are one of the best places to launch an attack on availability
- In this setting, these are usually called **denial of service** (DoS) attacks
- Transmission failure can happen because a line is cut or because there is too much noise
- Flooding is a common technique
  - Ask for too many connections
  - Request too many of some other service
- **Distributed denial of service** (DDoS) attacks are common (often using zombies or botnets) to make a more damaging and hard to trace attack

Denial of service attacks

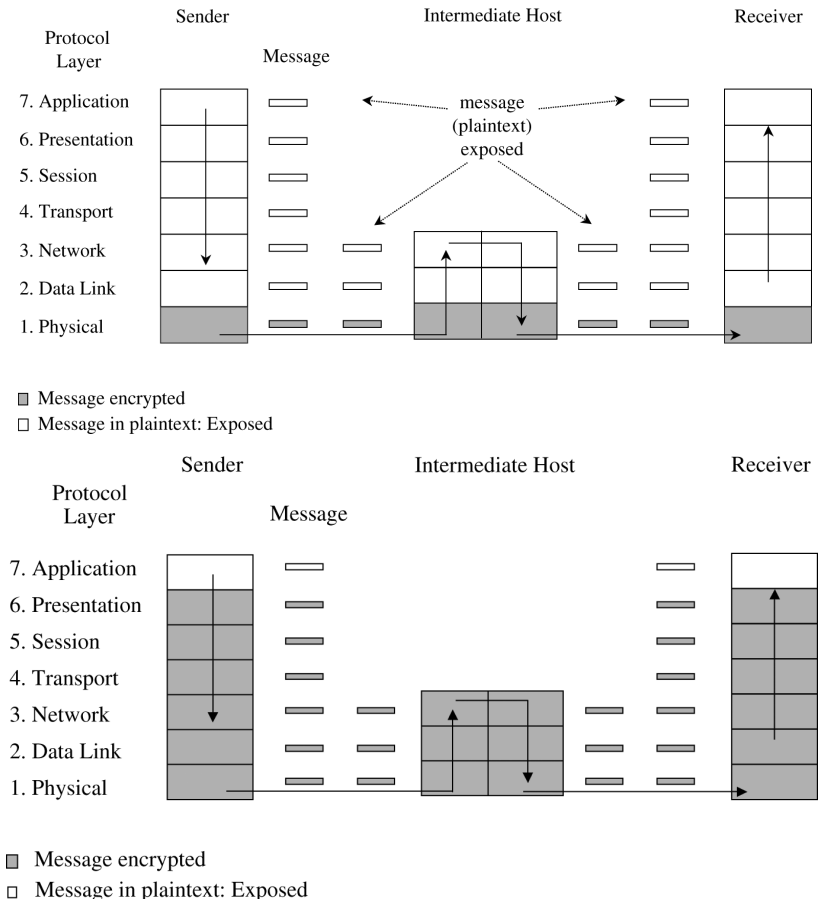
- TCP SYN floods
  - Exploit the three-way handshake
- Echo-chargen
  - Chargen sets up a stream of packets for testing
  - Echo packets are supposed to be sent back to the sender
  - If you can trick a server into sending echo packets to itself, it will respond to its own packets forever
- Ping of death
  - A ping packet requests a reply
  - If you can send more pings than a server can handle, it goes down
  - Only works if the attacker has more bandwidth than the victim (DDoS helps)
- Smurf
  - A ping packet is broadcast to everyone, with the victim spoofed as the originator
  - All the hosts try to ping the victim
  - The real attacker is hidden
- Teardrop
  - A teardrop attack uses badly formed IP datagrams
  - They claim to correspond to overlapping sequences of bytes in a packet
  - There's no way to put them back together and the system can crash

DNS attacks

- The Domain Name System (DNS) uses Domain Name Servers (also DNS) to convert user readable URLs like **`google.com`** to IP addresses
- Taking control of a server means that you get to say where google.com is
- For efficiency, servers cache results from other servers if they didn't know the IP
  - **DNS cache poisoning** is when an attacker gives a good server a bad IP address

Network encryption

- Encryption is important for network security
- **Link encryption** encrypts data just before going through the physical communication layer
  - Each link between two hosts could have different encryption
  - Message are in plaintext within each host
  - Link encryption is fast and transparent
- **End-to-end encryption** provides security from one end of the transmission to the other
  - Slower
  - Responsibility of the user
  - Better security for the message in transit



Database Security

What is a database?

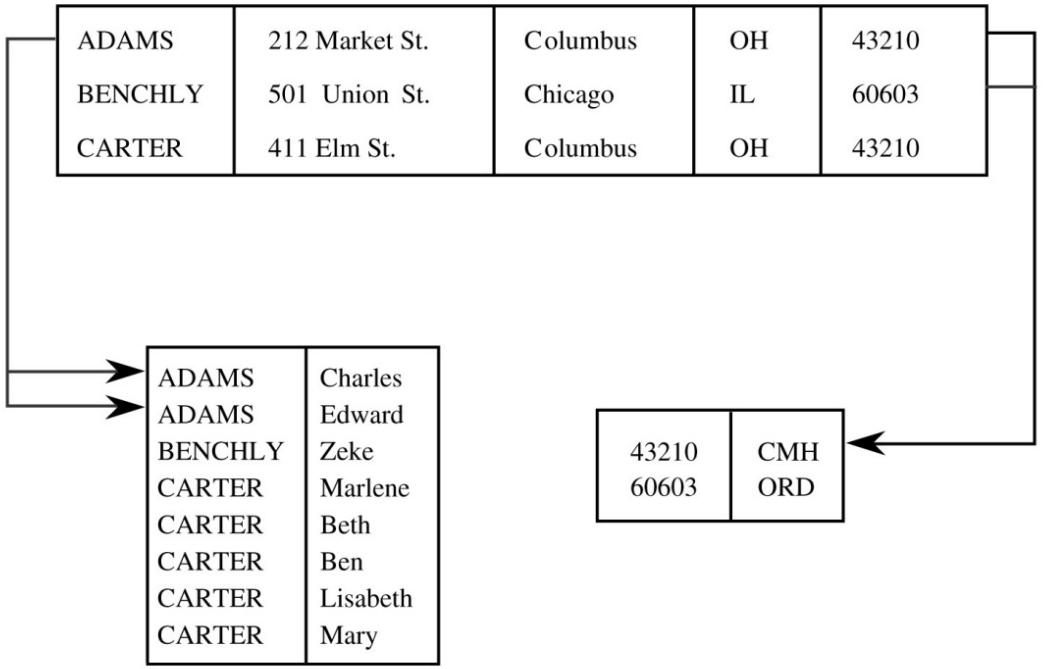
- A **database** is a collection of data and a set of rules to organize the data by relationships
- A **database administrator** makes the rules and controls access
- A **database management system** (**DBMS**) is the program through which the user interacts with the database

Database components

- Almost all modern databases use the relational database model
  - The fundamental unit of organization is a table
  - An older format for databases was hierarchical, like a tree
- A table consists of **records**
- A record consists **fields** or **elements**, which are each a specific item of data

Schemas

- The tables in a database are usually related to each other in some way
- The logical structure of a database is called a **schema**
- A user may only see part of it, called a **subschema**
- An **attribute** is the name of a column
- A **relation** is a set of columns



Queries

- A **query** is the name of a command given to a database by a user
- Queries can:
  - Retrieve
  - Modify
  - Add
  - Delete
- Most databases allow commands to be issued through a variant of SQL

Database security requirements

- Because they are a central part of modern business, several aspects of database security are crucial:
  - Physical database integrity
  - Logical database integrity
  - Element integrity
  - Access control
  - User authentication
  - Availability

Reliability and integrity

- **Reliability** is a measure of how long a software system can run without failing
  - Reliability is often quoted in terms of uptime percentage
  - Or mean time between failures
- Database reliability and integrity has three aspects:
  - Database integrity
    - Is the database as a whole protected from disk failure or corruption
  - Element integrity
    - Are only authorized users allowed to change elements
  - Element accuracy
    - Are the values in the elements correct

Two-phase update

- A key problem for database integrity is what happens if the system fails in the middle of an update
  - Then the database is inconsistent
- A two-phase update is a common solution
  - During the **intent** phase, the DBMS computes the results needed for the update, but does not change the database
  - During the **commit** phase, it changes all of the fields to the values computed in the intent phase
  - If the intent phase fails, the DBMS can start over from the beginning
  - If the commit phase fails, the DBMS can try to write all the data from the intent phase again

Disclosure of sensitive data

- The most serious disclosure of sensitive data is its exact value
- Bounds can also be disclosed
  - Example: highest salary and lowest salary
  - If the user can manipulate the bounds, he or she can search for specific values
- Negative result
  - Felonies is not zero
  - Visits to the oncology ward is not zero
- Existence
  - Knowing that a field even exists means someone is using it
- Probable value
  - How many people are in Bob's dorm room? 2
  - How many people in Bob's dorm room pirate movies? 1
  - There's a 50% chance that Bob pirates movies

Direct attack

- In a direct attack on sensitive information, a user will try to determine the values of a sensitive field by finding the right query
- Sometimes an unusual query will be used to bypass checks

Indirect attack

- To avoid leaking sensitive data, some DBMSs allow statistics to be reported
- Each of the following statistics can be attacked in different ways:
  - Sum
  - Count
  - Mean
  - Median

Protecting against inference

- Suppress obviously sensitive information
  - Easy, but incomplete
- Track what the user knows
  - Expensive in terms of computation and storage requirements
  - Analysis may be difficult
  - Multiple users can conspire together
- Disguise the data
  - Data is hidden
  - Users who are not trying to get sensitive data get slightly wrong answers

Integrity and confidentiality

- Integrity is difficult, but we can assign levels of trust
  - It is necessarily not going to be as rigorous as Biba
- Confidentiality
  - Difficult and causes redundancies since top secret information cannot be visible in any way to low clearance users
  - Worse, we don't want to leak any information by preventing a record from being added with a particular primary key (because there is a hidden record that already has that primary key)
  - **Polyinstantiation** means that records with similar or identical primary keys (but different data) can exist at different security levels

Ticket Out the Door

Upcoming

Next time…

- Review after Exam 2

Reminders

- **Finish Project 3!**
  - Try to attack the other projects
  - Don't forget to include the documentation of your system, explaining how it meets (or does not meet) the eight secure design principles
  - **Due Wednesday**
- Study for final: 12:30 - 2:30 p.m., Wednesday, 12/10/2025