Week 8 - Friday

COMP 3400

Last time

- What did we talk about last time?
- Finished TCP
- Network security
  - CIA

Questions?

Assignment 5

Form Teams!

Project 2

Cryptography

Cryptography

- "Secret writing"
- The art of encoding a message so that its meaning is hidden
- **Cryptanalysis** is breaking those codes
- Cryptography is a powerful tool for *confidentiality* because modern encryption methods make it almost impossible to read an encrypted message
- Cryptographic hash functions provide a tool for *integrity* because they can make it obvious when a message has been changed

Cryptography and availability

- Although cryptography provides tools for confidentiality and integrity, there's no clear cryptographic tool for availability
- In fact, cryptography often makes availability **worse** because encryption puts more strain on a system
  - Making it more susceptible to various DoS attacks
- There's **always** tension between confidentiality, integrity, and availability
- Increasing confidentiality and integrity usually decreases availability

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 often specify encryption and decryption functions $E_k(x)$ and $D_k(x)$ specific to a key k

Terminology

- A **sender** *S* wants to send a message to a **recipient** *R*
- If *S* gives the message to *T* who gives it to *R*, *T* is a **transmission medium**
- If an outsider *O* wants to access the message (to read, change, or destroy it), we call *O* an **interceptor** or **intruder**
- The fear is that O will cause one of four security failures:
  - Blocking the message
  - Intercepting the message
  - Modifying the message
  - Fabricating a false message

Terminology remix

- The previous slide gives dry 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

Symmetric key cryptography

- Symmetric encryption is what you probably think of as encryption
- Two parties have a **key** which they use for both encrypting and decrypting messages
  - The key is also known as a **shared secret**
- We have excellent symmetric encryption algorithms, of which AES is the most used
- But how do we distribute keys between parties who want to communicate secretly?

AES

- **A**dvanced **E**ncryption **S**tandard
- Symmetric block cipher designed to replace DES
- Block size of 128-bits
- Key sizes of 128, 192, and 256 bits
- Like the older (and deprecated) 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
- No known attacks exist against good implementations of AES
  - It should take more than a billion billion years to break an AES encryption
  - Even quantum computers shouldn't change that much

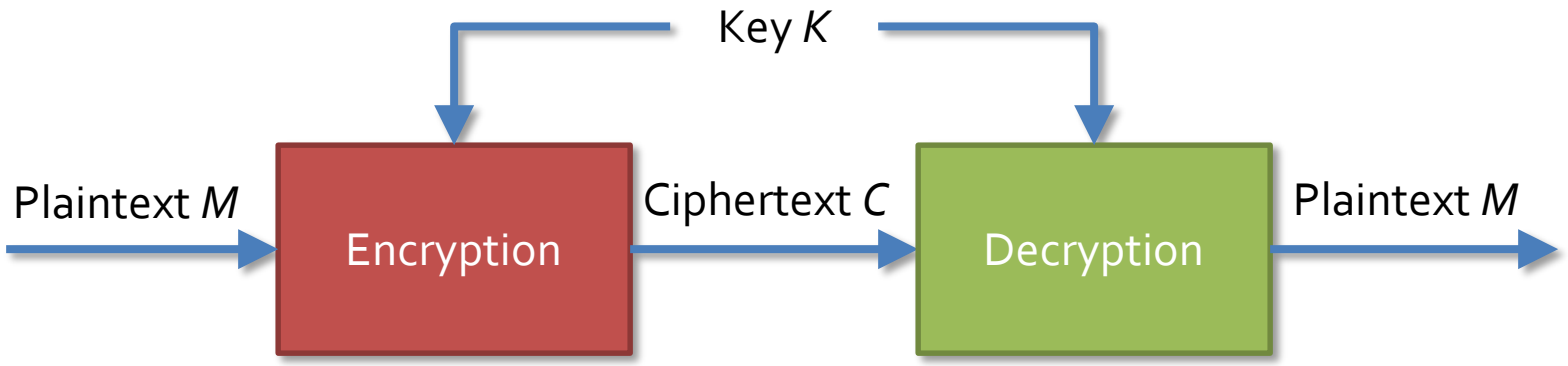
Public Key Cryptography

Public key cryptography

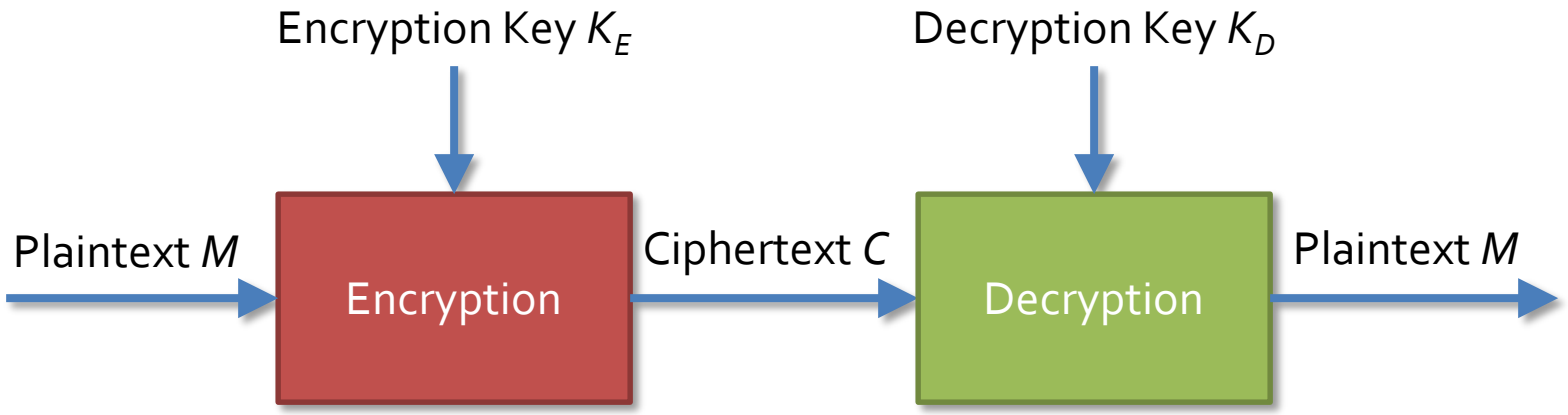
- Sometimes, we need something different
- We want a **public key** that *anyone* can use to encrypt a message to Alice
- Alice has a **private key** that can decrypt such a message
- The **public key** can only encrypt messages; it **cannot** be used to decrypt messages
- Public key cryptography is enormously useful, since companies can publish their public key far and wide
  - Anyone who wants to send them a secret message can do so
  - No secret needs to be shared ahead of time

Symmetric vs. public key

Symmetric Key Cryptography



Public Key Cryptography



RSA Algorithm

- RSA is the most commonly used public key cryptosystem
- 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:
  $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$

Why it's safe

- Crazy number theory
- For RSA, the modulus $n = p \cdot q$ where p and q are two large (hundreds of digits) primes
- It's easy to compute d, the decryption exponent, if you know p and q
- No one knows an efficient way to factor a large composite number
- However, quantum computers *could* make RSA much less safe

Cryptographic Hash Functions

Where do passwords go?

- What magic happens when you type your password into…
  - Windows or Unix to log on?
  - Amazon.com to make a purchase?
  - A Mandalorian fan site so that you can post on the forums?
- A genie from the 8th dimension travels back in time and checks to see what password you originally created

In reality…

- The password is checked against a file on a computer
- But, how safe is the whole process?
  - The Mandalorian fan site may not be safe at all
  - Amazon.com is complicated, much depends on the implementation of public key cryptography
  - What about your Windows or Unix computer?

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 (also called a cryptographic checksum) 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

Pigeonhole principle

- The **pigeonhole principle** says that if you try to put ***m*** items into ***n*** categories, where ***m*** > ***n***, then at least 2 things will be in the same category
- Imagine that you have a 40,000 byte message and a 256-bit hash digest
- How does the pigeonhole principle apply?

Collisions

- When two messages hash to the same value, this is called a **collision**
- Because of the pigeonhole principle, collisions are unavoidable
- The key feature we want from our hash functions is that collisions are difficult to predict

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

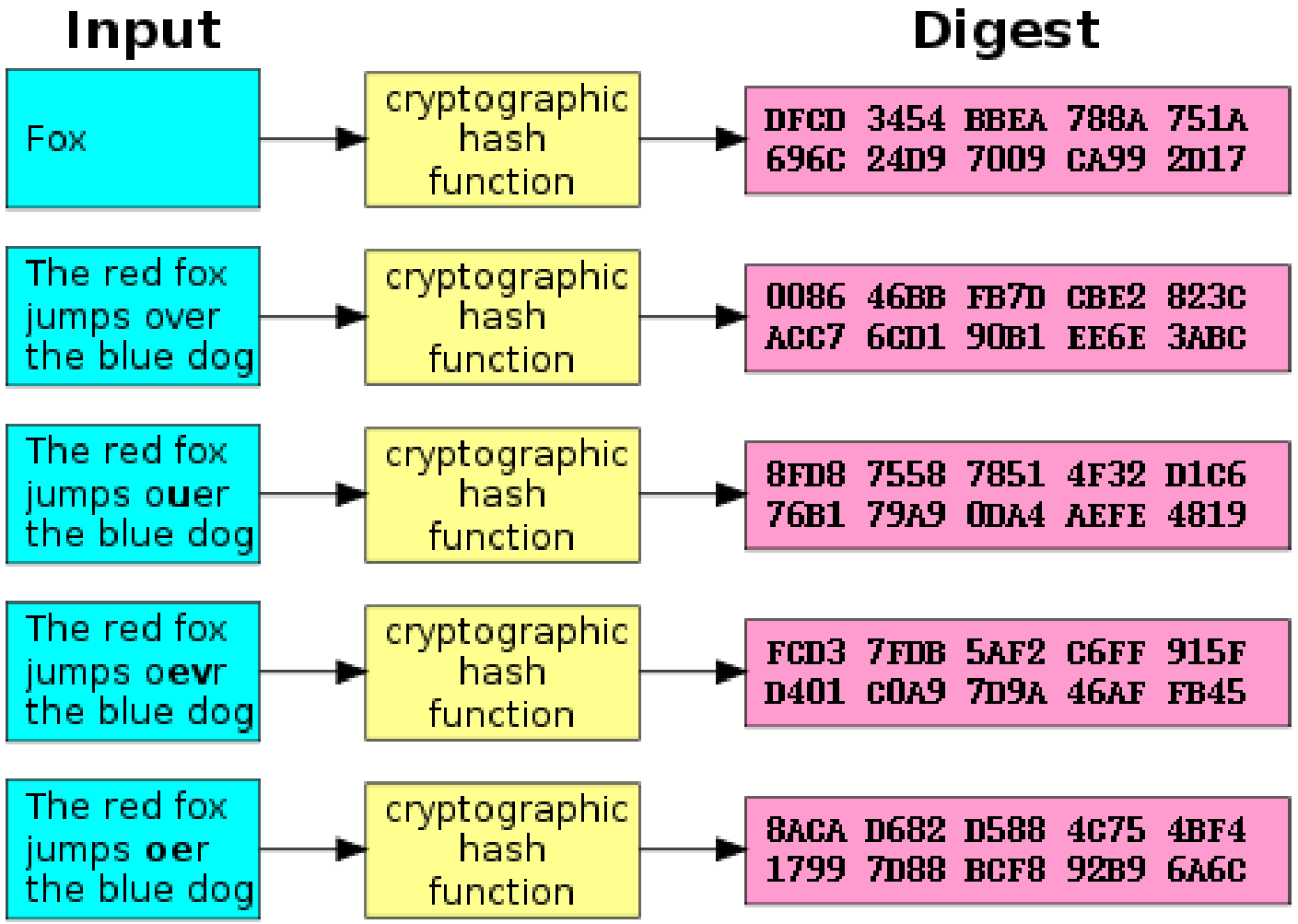
- Should be hard to find any two messages that hash to the same digest (collision)

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

Avalanching in action

- Example (from Wikipedia) of significant changes in hash output with small input changes
- SHA-1



Password dilemma resolved

- Instead of storing the actual passwords, Windows and Unix machines store the hash of the passwords
- When someone logs on, the operating system hashes the password and compares it to the stored version
- No one gets to see your original password!
- Hash functions are also used for digital signatures

MD5

- Message Digest Algorithm 5
- Popular but outdated 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
- SHA-1 security
  - Digest size: 160 bits
  - Considered unsafe
  - Theoretical attacks can run in 2^{63} SHA-1 evaluations
- SHA-2 is a successor family of hash functions
  - 224, 256, 384, 512 bit digests
  - Now the preferred hashing function
  - Designed by the NSA

SHA-3

- SHA-3 (Keccak) uses a completely different form of hashing than SHA-0, SHA-1, and SHA-2
- Although the attacks on SHA-1 are expensive and there are no real attacks on SHA-2, the attacks on SHA-0 made people nervous about hash functions following the same design
- SHA-3 also allows for variable size digests, for added security
  - 224, 256, 384, and 512 are standard
- Either SHA-2 or SHA-3 is considered secure (for now)

Upcoming

Next time…

- TLS
- Internet layer
- Link layer
- Wireless
- Start threads

Reminders

- Finish Project 2
- Start on Assignment 5
- Read sections 5.6, 5.7, 6.1, and 6.2
- **Have a great break!**