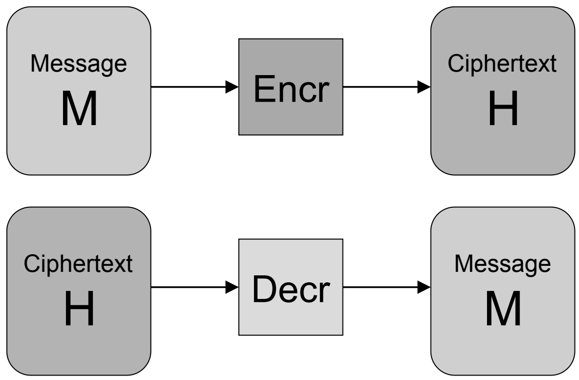
Applications 9A

Cryptography

Cryptography

- Cryptography is the process of encoding and decoding messages so that only intended recipients can read the messages.
- Security is extremely important in the age of the Internet.
  - Tampering
  - Eavesdropping
  - Theft
  - Impersonation

Process of Encryption



15-105 Principles of Computation, Carnegie Mellon University - CORTINA 3

Properties of Encryption

- Let M = the original message.
- H = Encr(M)
- M = Decr(H)
- M = Decr(Encr(M))
- H = Encr(Decr(H))
- Encr is the inverse function of Decr

15-105 Principles of Computation, Carnegie Mellon University - CORTINA 4

Example: Caesar cipher

- Encr: Take each letter in the message and replace it with the letter i positions ahead in the alphabet (wrapping around to 'A' if necessary).
- Decr: Take each letter in the ciphertext and replace it with the letter i positions before in the alphabet (wrapping around to 'Z' if necessary)

`ABCDEFGHIJKLMNOPQRSTUVWXYZ`

- Message: **`COMPUTATION`** **`i = 10`**
- Ciphertext: **`MYWZEDKDSYX`**

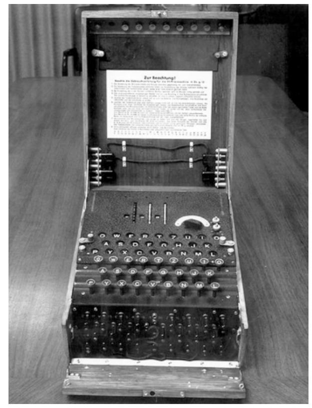
What if you don't know `i` to decode the message?

Secure Encryption

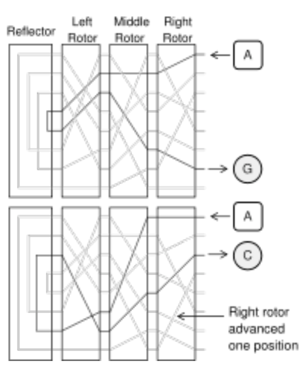
- The Caesar cipher is very easy to break.
  - Why?
- We need an encryption function (Encr) that is easy and fast to compute.
- We need a decryption function (Decr) that is very difficult to compute without knowing what it is.
  - Another way to look at it: Decr should be a function that would take a very, very long time to figure out by brute force.

Enigma Machine

- Used by the Germans in World War II to encode messages.
- Consisted of 3 rotors and a reflector.
- After each letter is encoded, the first rotor is rotated one position.
- If the first rotor rotates a full round, the second rotates one position also, etc.
- The same letter encoded twice won't yield the same result.



Enigma Machine



images from Wikipedia

Public-key Systems

- Each person P has his or her own $Encr_P$ function and his or her own $Decr_P$ function.
- For each person P, the $Encr_P$ function is made <u>public</u> for all to use. Anyone who wants to send a message to P uses the $Encr_P$ function to encode it.
- Once the encoded message is sent, person P uses the $Decr_P$ function to decode it. $Decr_P$ is kept <u>private</u> and only person P knows it.
- It is very important that no one else can determine how the private $Decr_P$ works given the public $Encr_P$.
  - Deducing $Decr_P$ should be computationally infeasible.

Electronic Signatures

- Alice sends a message to Bob using Bob's public encoding procedure.
  - "I think Carol is good. - Alice"
- Bob decodes the message using his private decoding procedure. He then adds an additional message to Alice's message.
  - "I think Carol is good for nothing. - Alice"
- He then sends this message (encoded) to Carol.
- Carol decodes it and calls up Alice to yell at her.

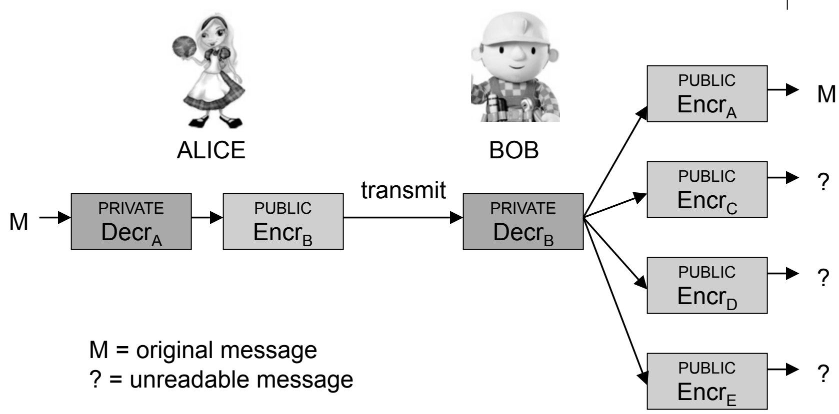
Commutative Functions

- We need to encode the signature as a function of the message.
- This way, when Bob alters the message, the signature won't match anymore.
- To do this, we must have an encryption and decryption scheme that is commutative.
  - Decr(Encr(M)) = M and Encr(Decr(M)) = M

Signing Securely

- Alice takes her message M and "signs" it by using her private decryption function to generate S = $Decr_A$(M).
- Alice then encrypts S using Bob's public function to get T = $Encr_B$(S) and sends T to Bob.
- Bob receives T and decodes it using his private function $Decr_B$(T) = $Decr_B$($Encr_B$(S)) = S.
  - Note: S is still unreadable by Bob.
- Bob then uses all of his friends' public encryption functions and finds that Alice's public encryption function yields a readable message: $Encr_A$(S) = $Encr_A$($Decr_A$(M)) = M.

Signing Securely



Signing Securely

- Bob tries to alter Alice's message to make M'.
- But he can't sign it as Alice since he would need Alice's private Decr$_A$ function.
- But Bob can send Alice's original message to Carol since he has S (the signed message before its decoded).
- Carol will then think that Alice, rather than Bob, sent her the message when she decodes it.

The RSA Cryptosystem

- Developed around 1977 for preventing outside parties from reading encrypted messages.
- Alice generates two extremely large prime numbers p and q. (Each number might be 1024 bits.) Let n = pq.
- Let r = (p-1)(q-1). Alice chooses e such that e and r are relatively prime (have no factors in common).
- She computes d such that de-1 is evenly divisible by r.

- $H = Encr_A (M) = M^e$ modulo n ⟵ **Alice gives out n and e as the public key.**
- $M = Decr_A (H) = H^d$ modulo n ⟵ **Alice does not give out d.**

RSA Example

http://en.wikipedia.org/wiki/RSA

Choose 2 prime numbers: p = 61, q = 53

Compute n: n = pq = 3233

Compute r: r = (p-1)(q-1) = 3120

Choose e > 1 such that
e and r are relatively prime: e = 17

Choose d such that
de - 1 is evenly divisible by r: d = 2753
(2753*17-1)/3120 = 15

PUBLIC KEY: $H = M^e$ modulo n

PRIVATE KEY: $M = H^d$ modulo n

Example: Encoding M = 123: $H = 123^{17}$ modulo 3233 = 855

Decoding H = 855: $M = 855^{2753}$ modulo 3233 = 123

Summary

- The RSA Algorithm has not been cracked.
  - There are no known ways to factor n into p and q in polynomial time.
  - If we knew a way to factor n into p and q quickly, we could compute d and then decode messages meant for Alice only.
- Security on the Internet is one of the big research areas in computer science.
  - Electronic commerce
  - National security

Look for `https://` on the web.