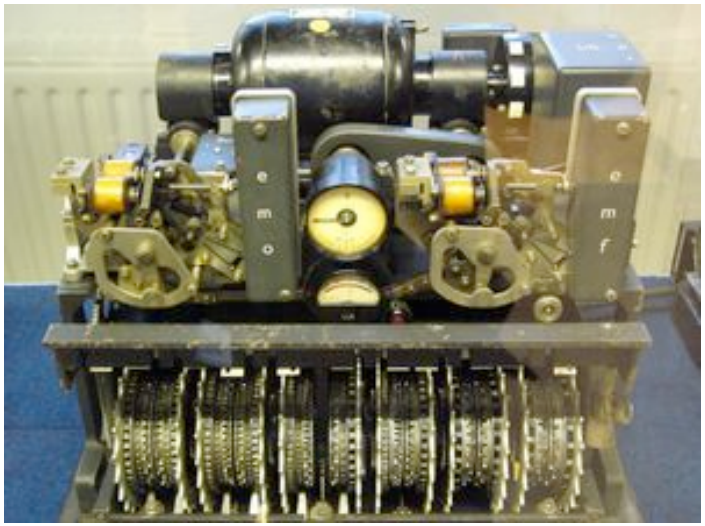


Network and Communication Security

Section 8.1 - 8.3

COS 461: Computer Networks

Spring 2011

Mike Freedman

http://www.cs.princeton.edu/courses/archive/spring11/cos461/

Overview

- Network security and definitions
- Brief introduction to cryptography
  - Cryptographic hash hunctions
  - Symmetric-key crypto
  - Public-key crypto

Internet's Design: Insecure

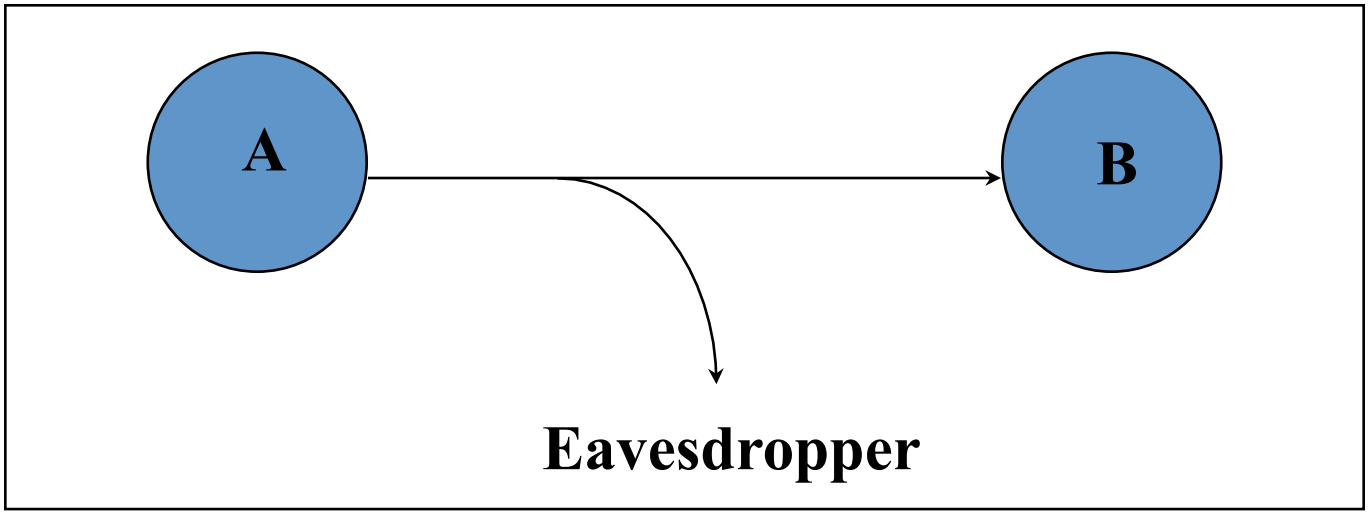
- Designed for simplicity
- "On by default" design
- Readily available zombie machines
- Attacks look like normal traffic
- Internet's federated operation obstructs cooperation for diagnosis/mitigation

Basic Components

- **Confidentiality:** Concealment of information or resources
- **Authenticity:** Identification and assurance of origin of info
- **Integrity:** Trustworthiness of data or resources in terms of preventing improper and unauthorized changes
- **Availability:** Ability to use desired info or resource
- **Non-repudiation:** Offer of evidence that a party indeed is sender or a receiver of certain information
- **Access control:** Facilities to determine and enforce who is allowed access to what resources (host, software, network, …)

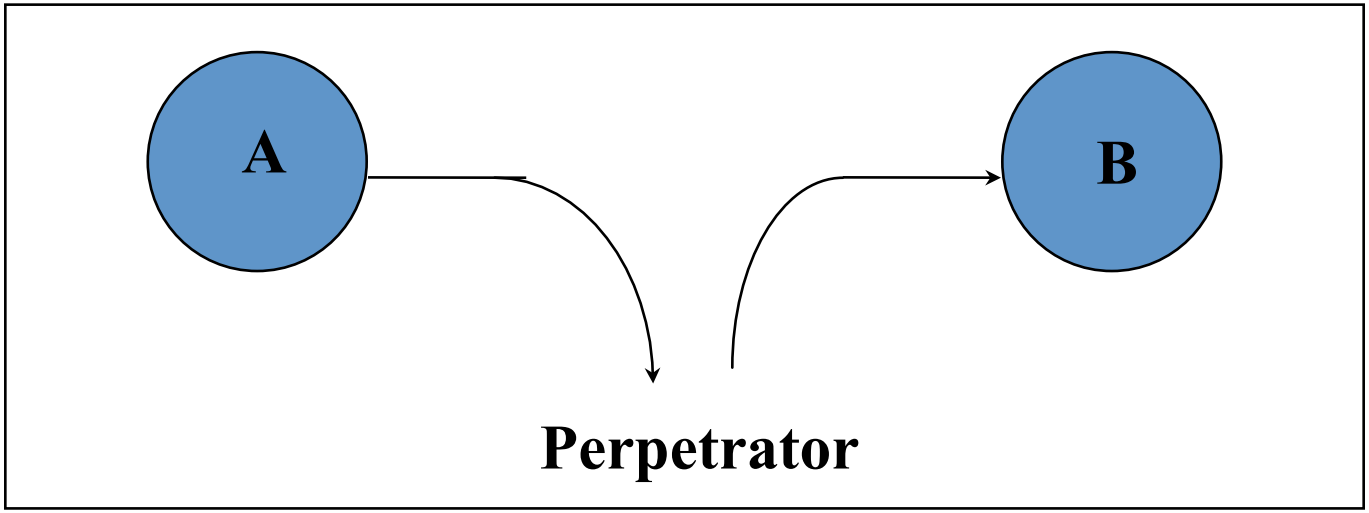
Eavesdropping - Message Interception (Attack on Confidentiality)

- Unauthorized access to information
- Packet sniffers and wiretappers (e.g. tcpdump)
- Illicit copying of files and programs



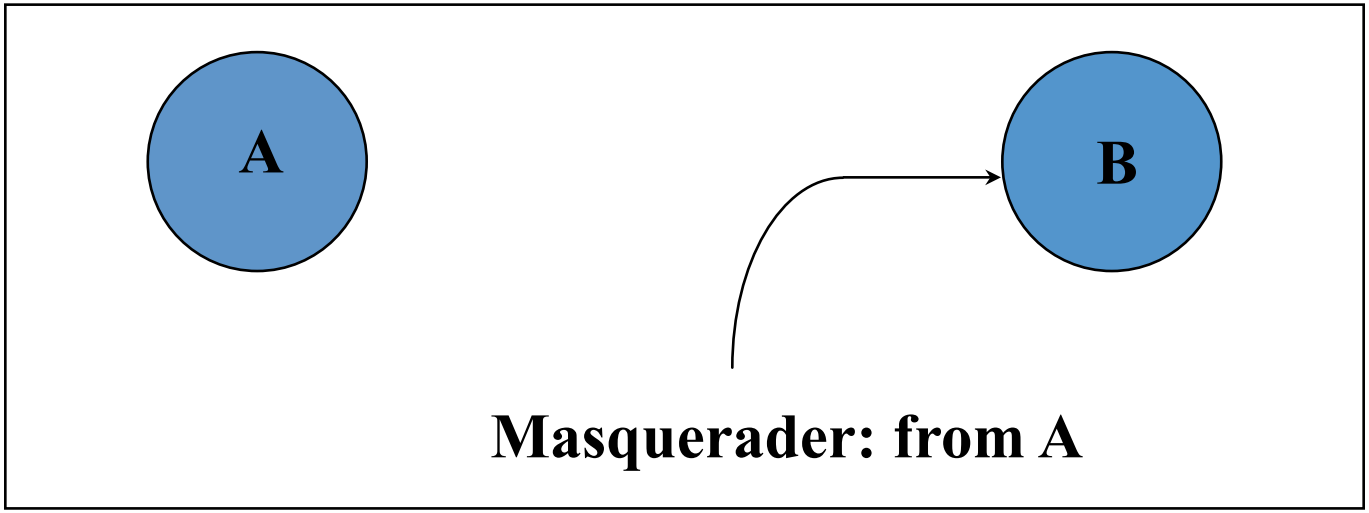
Integrity Attack - Tampering

- Stop the flow of the message
- Delay and optionally modify the message
- Release the message again



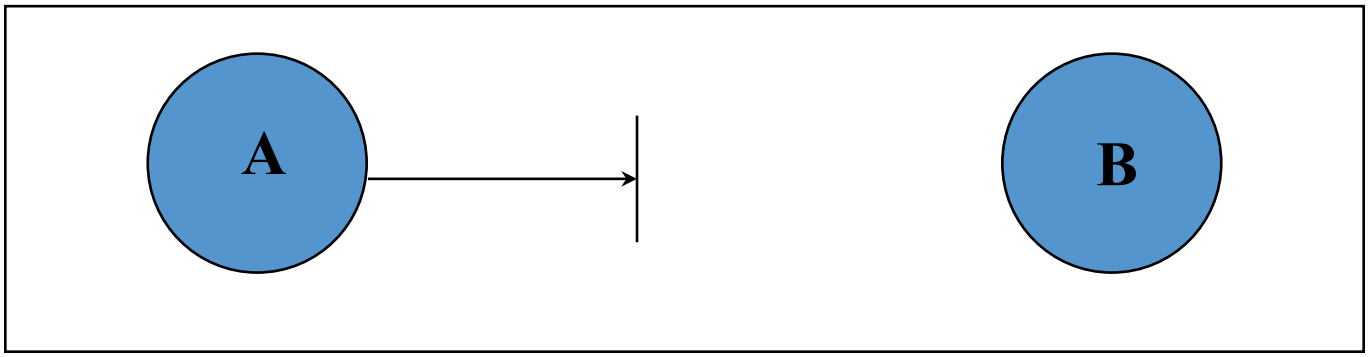
Authenticity Attack - Fabrication

- Unauthorized assumption of other's identity
- Generate and distribute objects under identity



Attack on Availability

- Destroy hardware (cutting fiber) or software
- Modify software in a subtle way
- Corrupt packets in transit



- Blatant *denial of service* (DoS):
  - Crashing the server
  - Overwhelm the server (use up its resource)

Impact of Attacks

- Theft of confidential information
- Unauthorized use of
  - Network bandwidth
  - Computing resource
- Spread of false information
- Disruption of legitimate services

Introduction to Cryptography

What is Cryptography?

- Comes from Greek word meaning "secret"
  - Primitives also can provide integrity, authentication
- Cryptographers invent secret codes to attempt to hide messages from unauthorized observers

- Modern encryption:
  - *Algorithm* public, *key* secret and provides security
  - May be symmetric (secret) or asymmetric (public)

Cryptographic Algorithms: Goal

- Given key, relatively easy to compute
- Without key, hard to compute (invert)
- “Level” of security often based on “length” of key

Three Types of Functions

- Cryptographic hash Functions
  - Zero keys
- Secret-key functions
  - One key
- Public-key functions
  - Two keys

Cryptographic hash functions

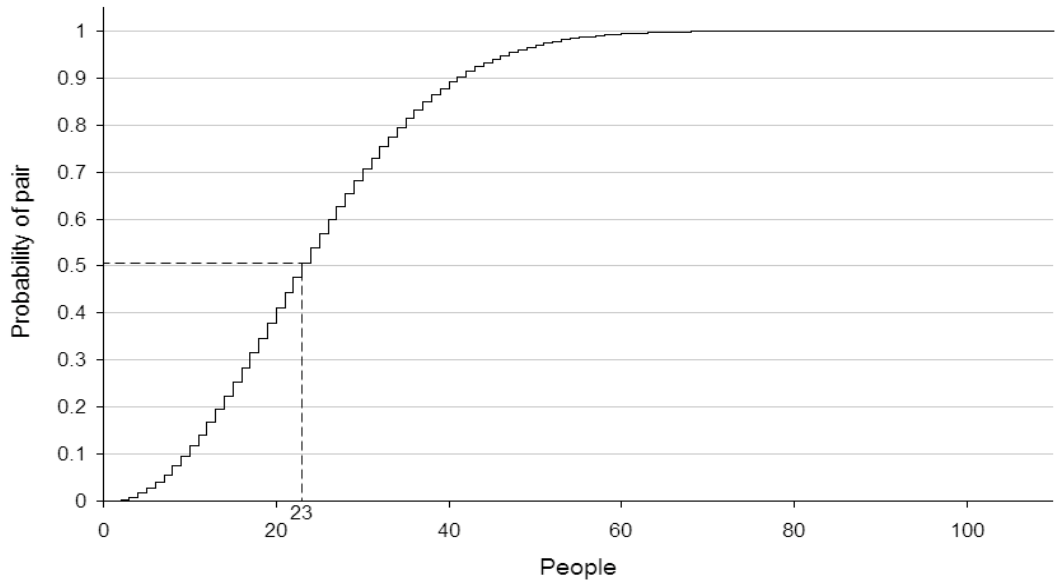
Cryptography Hash Functions

- Take message, *m*, of arbitrary length and produces a smaller (short) number, *h(m)*

- Properties
  - Easy to compute *h(m)*
  - Pre-image resistance: Hard to find an *m*, given *h(m)*
    - *"One-way function"*
  - Second pre-image resistance: Hard to find two values that hash to the same *h(m)*
    - *E.g.* discover collision: *h(m) == h(m') for m != m'*
  - Often assumed: output of hash fn's "looks" random

How hard to find collisions? Birthday Paradox

- Compute probability of *different* birthdays
- Random sample of *n* people taken from *k*=365 days
- Probability of no repetition:
  - $P = 1 – (1) (1 - 1/365) (1 – 2/365) (1 – 3/365) … (1 – (n-1)/365)$
  - $P \approx 1 – e^{-(n(n-1)/2k}$
  - Let $k=n$, $P \approx 2^{(N/2)}$



How Many Bits for Hash?

- If *m* bits, takes $2^{m/2}$ to find weak collision
  - Still takes 2^m to find strong (pre-image) collision
- 64 bits, takes 2^{32} messages to search (easy!)
- Now, MD5 (128 bits) considered too little
- SHA-1 (160 bits) getting old

Example use #1: Passwords

- Password hashing
  - Can't store passwords in a file that could be read
    - Concerned with insider attacks!
  - Must compare typed passwords to stored passwords
    - Does hash (typed) == hash (password) ?
  - Actually, a "salt" is often used: hash (input || salt)
    - Avoids precomputation of all possible hashes in "rainbow tables" (available for download from file-sharing systems)

Example use #2: Self-certifying naming

- File-sharing software (LimeWire, BitTorrent)
  - File named by F_{name} = hash (data)
  - Participants verify that hash (downloaded) == F_{name}
    - If check fails, reject data
- Recursively applied…
  - BitTorrent file has many chunks
  - Control file downloaded from tracker includes:
    - \forall chunks, $F_{chunk\ name}$ = hash (chunk)
  - BitTorrent client verifies each individual chunk

Example use #3: TCP SYN cookies

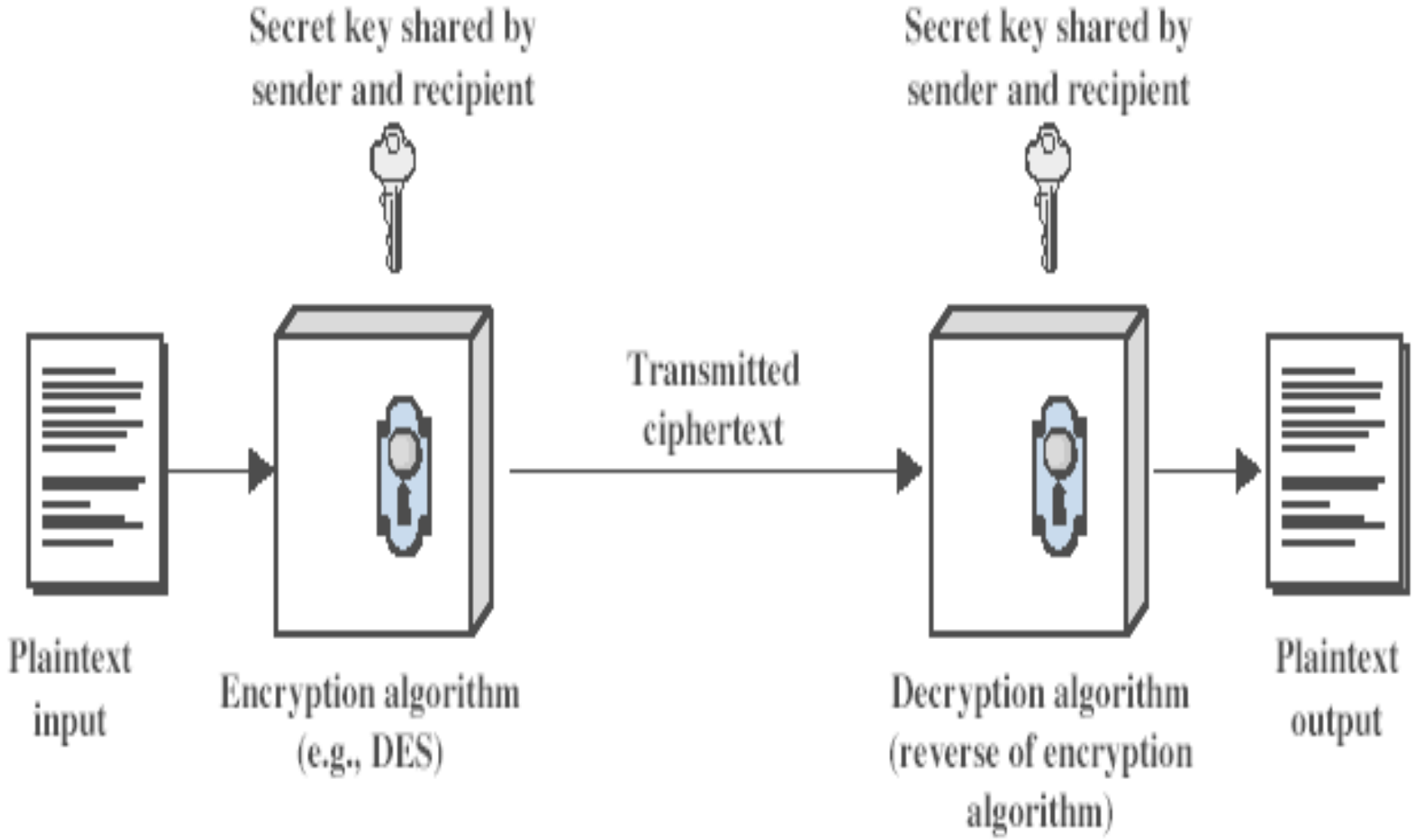
- What state is established during TCP conn setup?
  - Server: Initial Sequence Number (ISN), kernel bufs, MSS info
  - Attack: Setup state at server with SYN, never follow-up
    - Particularly bad: client spoofs IP, never needs response
- General idea of SYN cookies:
  - Server responds to Client SYN with SYN-ACK cookie
    - timestamp = time() mod 32
    - seqno = f (timestamp, src ip, src port, dest ip, dest port)
    - Server's ISN = timestamp [5b] || mss[3b] || seqno [24b]
    - Keeps no state. Delays memory buffer allocation.
  - Honest client responds with ACK (of server ISN)
  - Server checks response. If seqno valid, establishes conn

Symmetric (Secret) Key Cryptography

Symmetric Encryption

- Also: "conventional / private-key / single-key"
  - Sender and recipient share a common key
  - All classical encryption algorithms are private-key
  - Dual use: confidentiality or authentication/integrity
    - Encryption vs. msg authentication code (MAC)
- Was only type of encryption prior to invention of public-key in 1970's
  - Most widely used
  - More computationally efficient than "public key"

Symmetric Cipher Model



Use and Requirements

- Two requirements
  - Strong encryption algorithm
  - Secret key known only to sender / receiver
- Goal: Given key, generate 1-to-1 mapping to ciphertext that looks random if key unknown
  - Assume *algorithm* is known (no security by obscurity)
  - Implies secure channel to distribute key

Confidentiality (Encryption)

Sender:

- Compute C = $AES_K(M)$
- Send C

Receiver:

- Recover M = $AES'_K(C)$

Auth/Integrity (MAC)

Sender:

- Compute H = $AES_K(SHA1\ (M))$
- Send <M, H>

Receiver:

- Computer H' = $AES_K(SHA1\ (M))$
- Check H' == H

Public-Key Cryptography

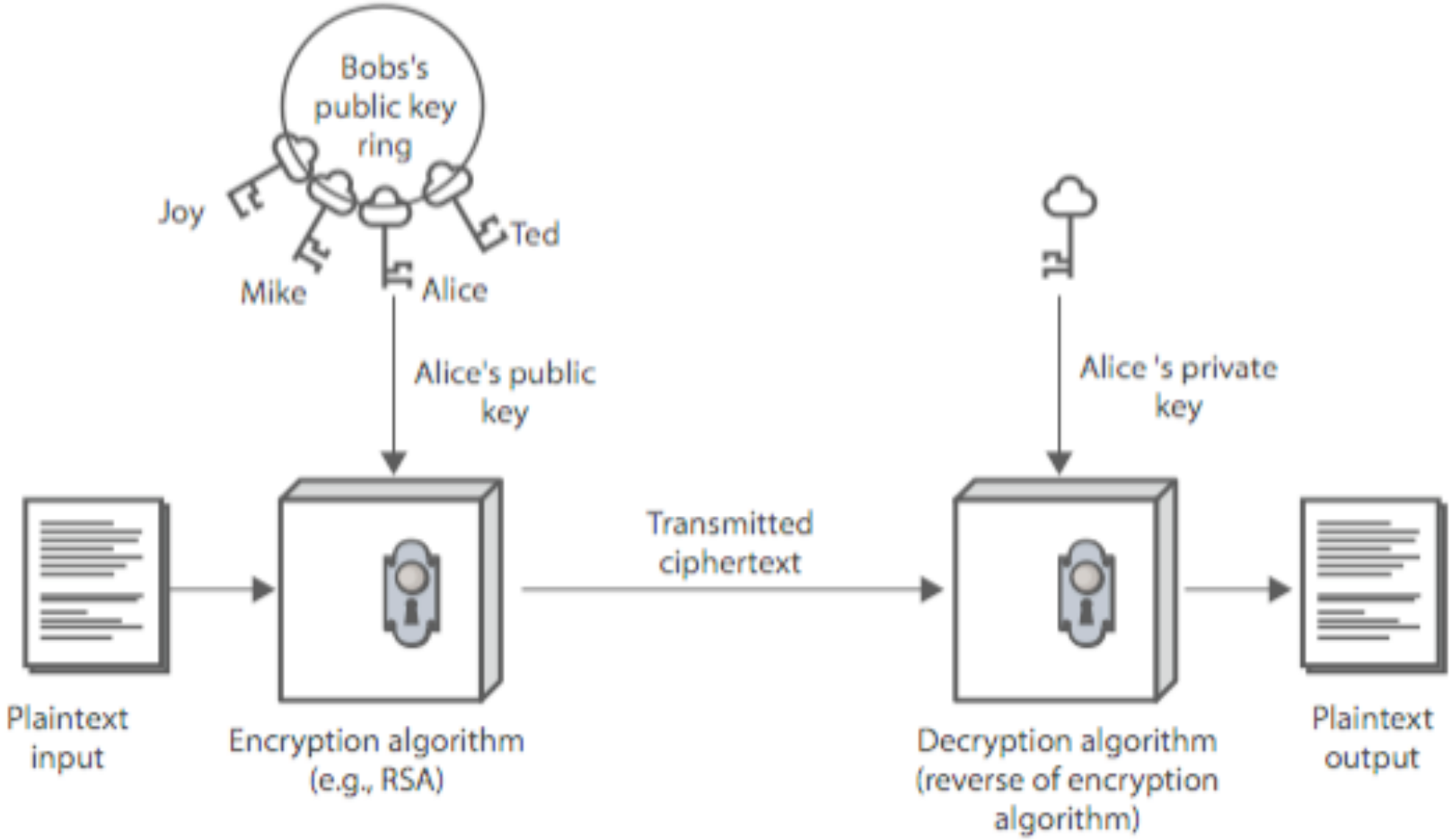
Why Public-Key Cryptography?

- Developed to address two key issues:
  - Key distribution: Secure communication w/o having to trust a key distribution center with your key
  - Digital signatures: Verify msg comes intact from claimed sender (w/o prior establishment)
- Public invention due to Whitfield Diffie & Martin Hellman in 1976
  - Known earlier in classified community

Public-Key Cryptography

- **Public-key/asymmetric** crypto involves use of two keys
  - **Public-key:** Known by anybody, and can be used to encrypt messages and verify signatures
  - **Private-key:** Known only to recipient, used to decrypt messages and sign (create) signatures
- **Asymmetric** because
  - Can encrypt messages or verify signatures w/o ability to decrypt messages or create signatures
  - If "one-way function" goes $c \leftarrow F(m)$, then public-key encryption is a "trap-door" function:
    - Easy to compute $c \leftarrow F(m)$
    - Hard to compute $m \leftarrow F^{-1}(m)$ without knowing k
    - Easy to compute $m \leftarrow F^{-1}(m,k)$ by knowing k

Public-Key Cryptography



Security of Public Key Schemes

- Like private key schemes, brute force search possible
  - But keys used are too large (e.g., >= 1024bits)
- Security relies on a difference in computational difficulty b/w easy and hard problems
  - RSA: exponentiation in composite group vs. factoring
  - ElGamal/DH: exponentiation vs. discrete logarithm in prime group
  - Hard problem is known, but computationally expensive
- Requires use of very large numbers
  - Hence is slow compared to private key schemes
  - RSA-1024: 80 us / encryption; 1460 us / decryption [cryptopp.com]
  - AES-128: 109 MB / sec = 1.2us / 1024 bits

(Simple) RSA Algorithm

- **Security** due to cost of factoring large numbers
  - Factorization takes $O(e^{\log n \log\log n})$ operations (hard)
  - Exponentiation takes $O((\log n)^3)$ operations (easy)
- To encrypt a message M the sender:
  - Obtain public key `{e,n};` compute $C = M^e \bmod n$
- To decrypt the ciphertext C the owner:
  - Use private key `{d,n};` computes $M = C^d \bmod n$
- Note that msg M must be smaller than the modulus n
  - Otherwise, hybrid encryption:
    - Generate random symmetric key *r*
    - Use public key encryption to encrypt *r*
    - Use symmetric key encryption under *r* to encrypt *M*

Summary

- Network security and definitions
- Introduction to cryptography
  - Cryptographic hash functions
    - Zero keys, hard to invert, hard to find collisions
  - Symmetric-key crypto
    - One key, hard to invert, requires key distribution
  - Public-key crypto
    - Two keys, hard to invert, more expensive

- Wed: IPSec, HTTPS, DNSSEC, other security problems