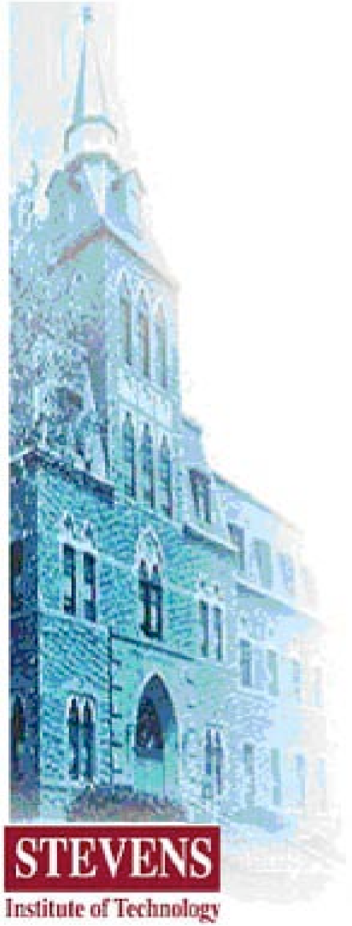


Data Encryption

Data Encryption Terminology

- Plaintext: an original message to be transformed
- Ciphertext: the resulting message after the transformation
- Encryption, Decryption: conversion of plaintext to cyphertext and vice versa
- Cryptosystem: system for encryption and decryption of information. It can be:
  - Simmetric: same key used for encryption and decryption
  - Asymmetric: different keys
- Cryptography, Criptoanalysis: the study of systems to keepconfidentiality, of breaking cryptosystems respectively

Basic Encryption Examples

- 1. A ciphertext alphabet can be defined which is the plaintext alphabet simple shifted by n places where n is the key. If n = 3, the resulting alphabet is as follows:
  - Plaintext alphabet: a b c d e f g …
  - Ciphertext alphabet: d e f g h i j …
- 2. A ciphertext alphabet that is a random mix of the plaintext alphabet.
  - Plaintext alphabet: a b c d e f g …
  - Ciphertext alphabet: n z q a i y m …

Modern Cryptography

- Modern cryptosystems are can be divided in:
  - Private key systems: P=Dk(Ek(P)) (symmetric)
  - Public key systems: P=Dk1(Ek2(P)) (asymmetric)

- These system have the following features:
  - They are specialized for binary data
  - They are based on a open design
  - They rely on algorithms for which an exhaustive search is impractical because computationally too intensive

Private Key

- Pro:
  - Relatively fast
  - Widespread use thanks to standardization
  - Public domain (i.e. not patented)

- Con:
  - Requires key to be known to both parties (*key distribution problem*)
  - Cannot be used for other applications such as digital signatures, etc.

- Example: Data Encryption Standard (DES)

Public Key

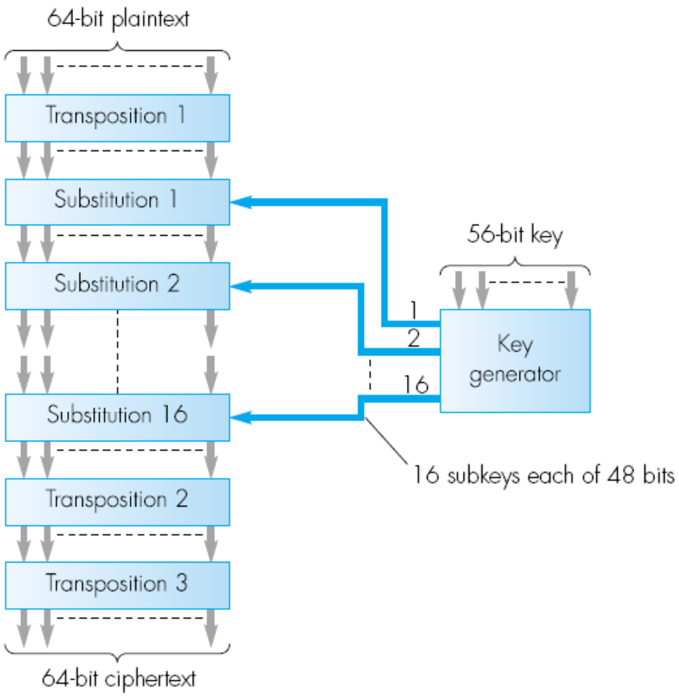
- Basic scheme suggested by Diffie and Hellman is based on one-way functions
  - one-way function:
    - given x, computing f(x) is easy
    - given y, finding x such that y=f(x) is very hard
  - therefore f^{-1} is hard to derive even if f is known

- trapdoor one-way function: a one-way function for which f^{-1} is easy to compute, provided a certain additional piece of information is provided

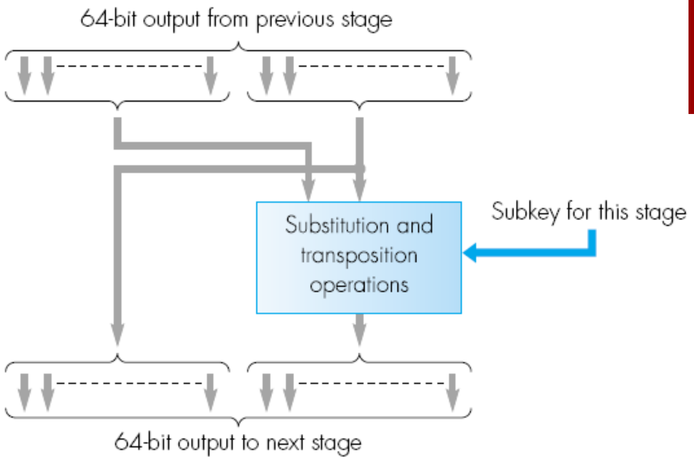
The Data Encryption Standard(DES)

- The DES algorithm is a block cipher, which means that it works on fixed-sized blocks of data.
    - A complete message is first split (segmented) into blocks or plaintext, each comprising 64 bits.
    - A unique 56-bit key is used to encrypt each block of plaintext into a 64-bit data block of ciphertext, which is subsequently transmitted through the network.
    - The receiver used the same key to perform the inverse (decryption) operation on each 64-bit data block it receives, thereby reassembling the blocks into complete messages.
- DES is a private key cryptography standard
- The crucial aspect is that the key must be sufficiently long

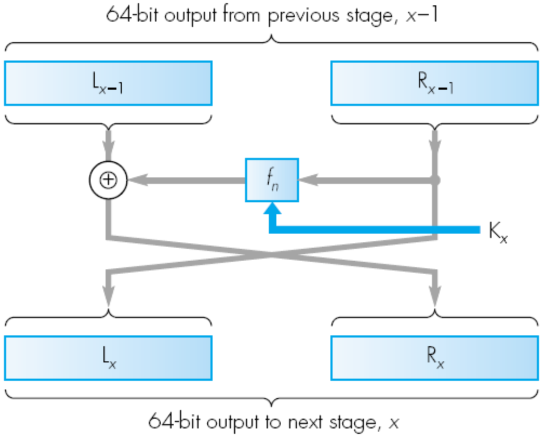
DES Algorithm Principles



Overall schematic


64-bit output from previous stage
Substitution and transposition operations
Subkey for this stage
64-bit output to next stage

Substitution schematic


64-bit output from previous stage, $x-1$
L_{x-1}
R_{x-1}
$\oplus$
f_n
K_x
L_x
R_x
64-bit output to next stage, x

Substitution operation

Public Key Encryption and Digital Signatures

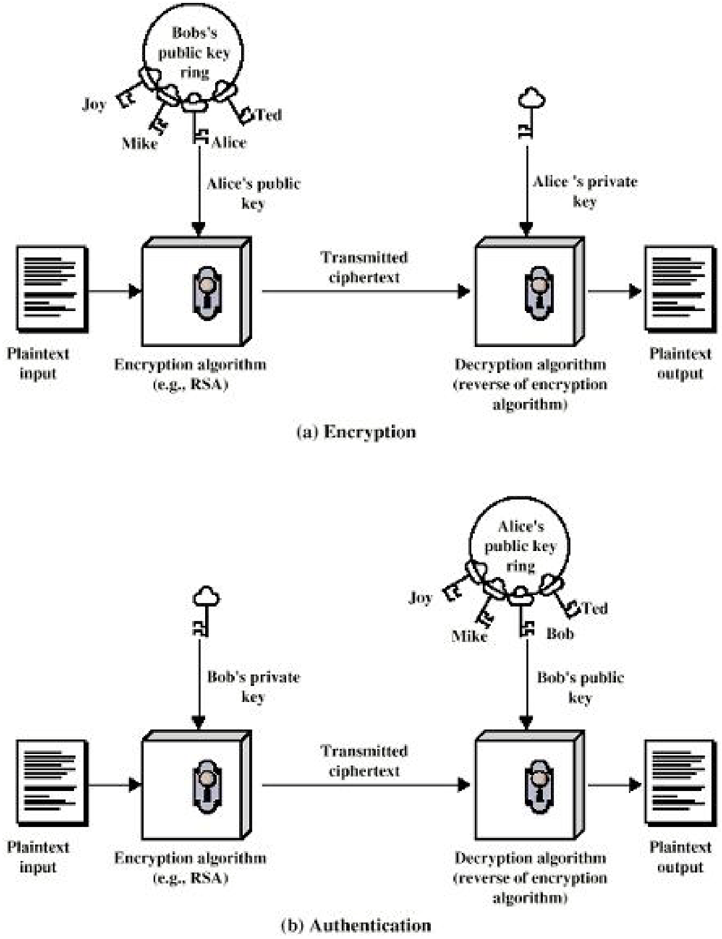
- Public-key cryptography is asymmetric, involving the use of two separate keys, in contrast to the symmetric conventional encryption, which uses only one key.

- The security of any encryption scheme depends on (1) the length of the key and (2) the computational work involved in breaking a cipher. There is nothing in principle about either conventional or public-key encryption that makes one superior to another from the viewpoint of resisting cryptanalysis. Because of the computational overhead of current public-key encryption schemes, there seems no foreseeable likelihood that conventional encryption will be abandoned.

- A public-key encryption scheme has five ingredients:
  - Plaintext: This is the readable message or data that is fed into the algorithm as input.
  - Encryption algorithm: The encryption algorithm performs various transformations on the plaintext.
  - Public and private key: This is a pair of keys that have been selected so that if one is used for encryption the other is used for decryption. The exact transformations performed by the encryption algorithm depend on the public or private key that is provided as input.
  - Ciphertext: This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
  - Decryption algorithm: This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

Public Key Encryption

- The public key of the pair is made public for others to use, while the private key is known only to its owner. These algorithms have the following important characteristics:
- It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.
- For most public-key schemes, either the two related keys can be used for encryption, with the other used for decryption.



The Essential Steps of Public Key Encryption

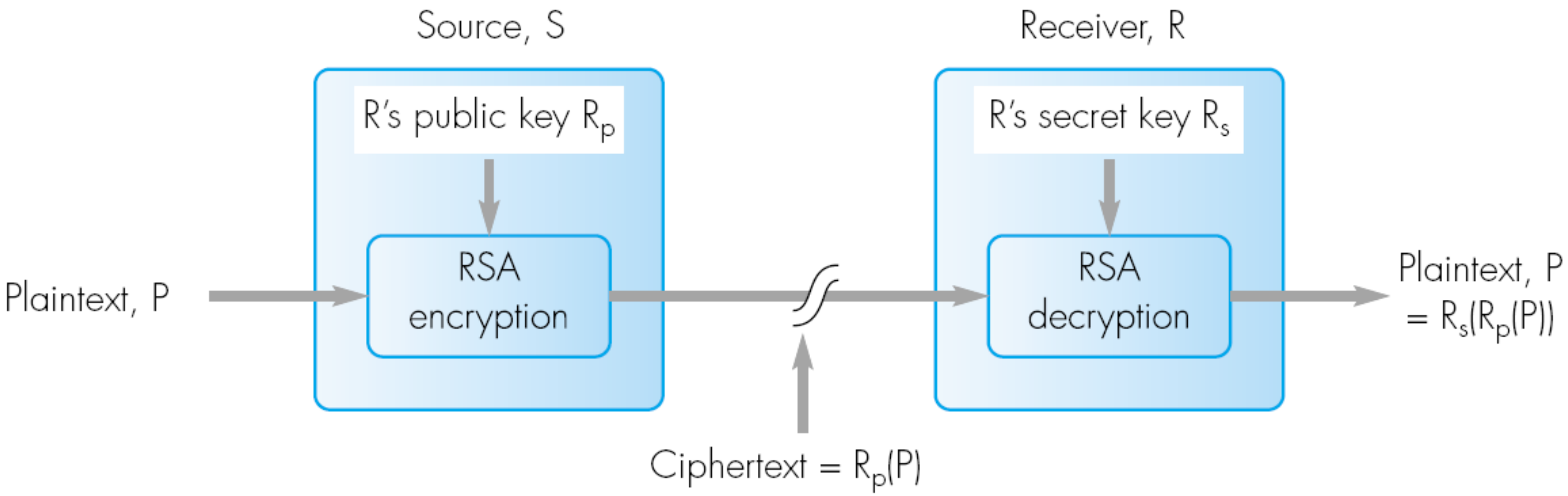
- Each other generates a pair of keys to be used for the encryption and decryption messages.
- Each users places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.
- As the above figure suggests, each user maintains a collection of public keys obtained from others.
- If Bob wishes to send a private message to Alice, Bob encrypts the message using Alice's public key.
- When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.
- With this approach, all participants have access to public keys, and private keys are generated locally by each participant and therefore need never be distributed.

Digital Signature

- Public-key encryption can be used in another way, as illustrated in the above figure.
- Suppose that Bob wants to send a message to Alice and, although it is not important that the message be kept secret, he wants Alice to be certain that the message is indeed from him.
- In this case, Bob uses his own private key to encrypt the message. When Alice receives the ciphertext, she finds that she can decrypt it with Bob's public key, thus proving that the message must have been encrypted by Bob. No one else has Bob's private key and therefore noone else could have created a cipher-text that could be decrypted with Bob's public key.
- Therefore, the entire encrypted message serves as a digital signature.

Rivest-Shamir-Adleman (RSA) Method

- The RSA algorithm (named after its inventors, Riverst, Shamir, and Adleman) is a widely accepted scheme for public key cryptography. It utilizes modular arithmetic and factorization of larger numbers.



RSA Encryption

- 1. Choose two large prime number ***p*** and ***q*** such that ***n = p*q***. The plaintext ***P*** that is represented by a number must be less than ***n***. In practice, ***n*** is a few hundred bits long.

- 2. Find a number ***e*** that is relatively prime to ***(p-1)*(q-1)***.Two numbers are said to be relatively prime if they have no common factors except 1. The public key consists of ***{e, n}***

- 3. Find a number ***d*** such that ***[modulo(d*e ,((p-1)*(q-1)))]= 1***. In other words, ***d*** and ***e*** are multiplicative inverses of each other modulo ***((p-1)*(q-1))***. The private key consists of ***{d, n}***.

- 4. The ciphertext ***C*** generated by using public key ***{e, n}*** to encrypt ***P*** is:

$$C = P^{e} \ (mod\ n)$$

RSA Decryption

- To decrypt the ciphertext ***C***, the private key ***{d, n}*** is needed. The decryption process is as following:

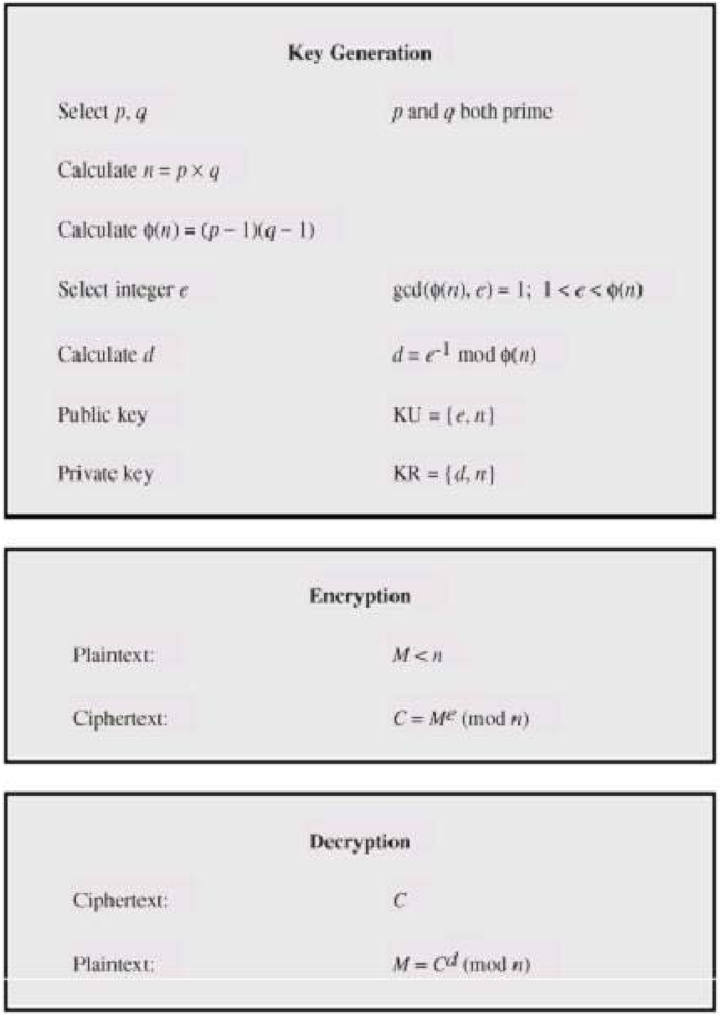
$$C^d \pmod{n} = (P^e)^d \pmod{n} = P^{de} \pmod{n}$$

$$= P \pmod{n} = P$$

- Calculating modular arithmetic involving large numbers can be simplified by using the following property:

$$(a*b) \bmod n = ((a \bmod n) * (b \bmod n)) \bmod n$$

RSA Schematic



Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d = e^{-1} \bmod \phi(n)$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption

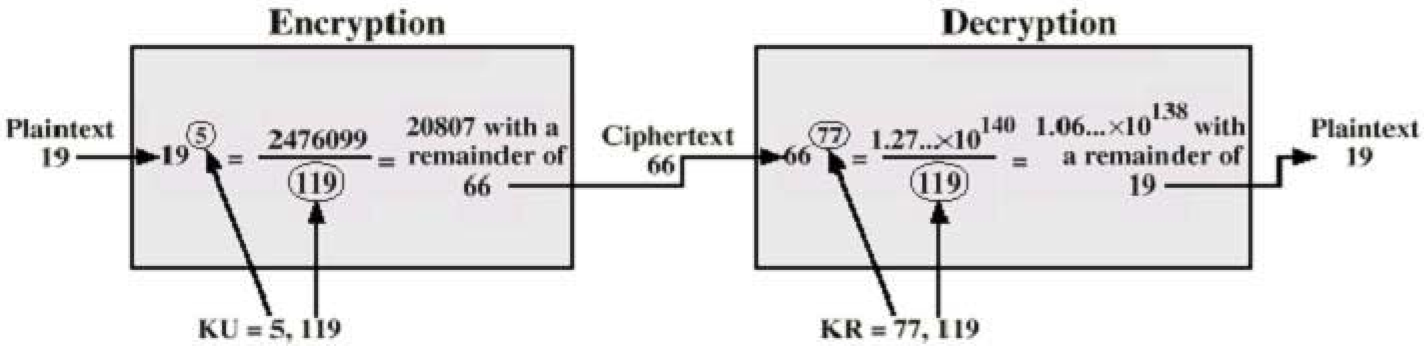
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

A RSA Example

- Select two prime numbers, p = 7 and q = 17.
- Calculate n = p*q = 7*17 = 119.
- Calculate O(n) = (p-1)(q-1) = 96.
- Select e such that e is relatively prime to O(n) = 96 and less than O(n); in this case, e = 5.
- Determine d such that de = 1 mod 96 and d<96. The correct value is d = 77, because 77*5 = 385 = 4 * 96 + 1.
- The resulting keys are public key KU = {5, 119} and private key KR = {77, 119}. The example shows the use of these keys for a plaintext input of M=19. For encryption, 19 is raised to the fifth power, yielding 2,476,099. Upon division by 119, the remainder is determined to be 66. Hence 195 = 66 mod 119, and the ciphertext is 66. For decryption, it is determined that 6677 = 19 mod 119.


Encryption
Decryption
Plaintext 19
19 5 = 2476099 / 119 = 20807 with a remainder of 66
Ciphertext 66
66 77 = 1.27...×10^140 / 119 = 1.06...×10^138 with a remainder of 19
Plaintext 19
KU = 5, 119
KR = 77, 119

Defeating the RSA

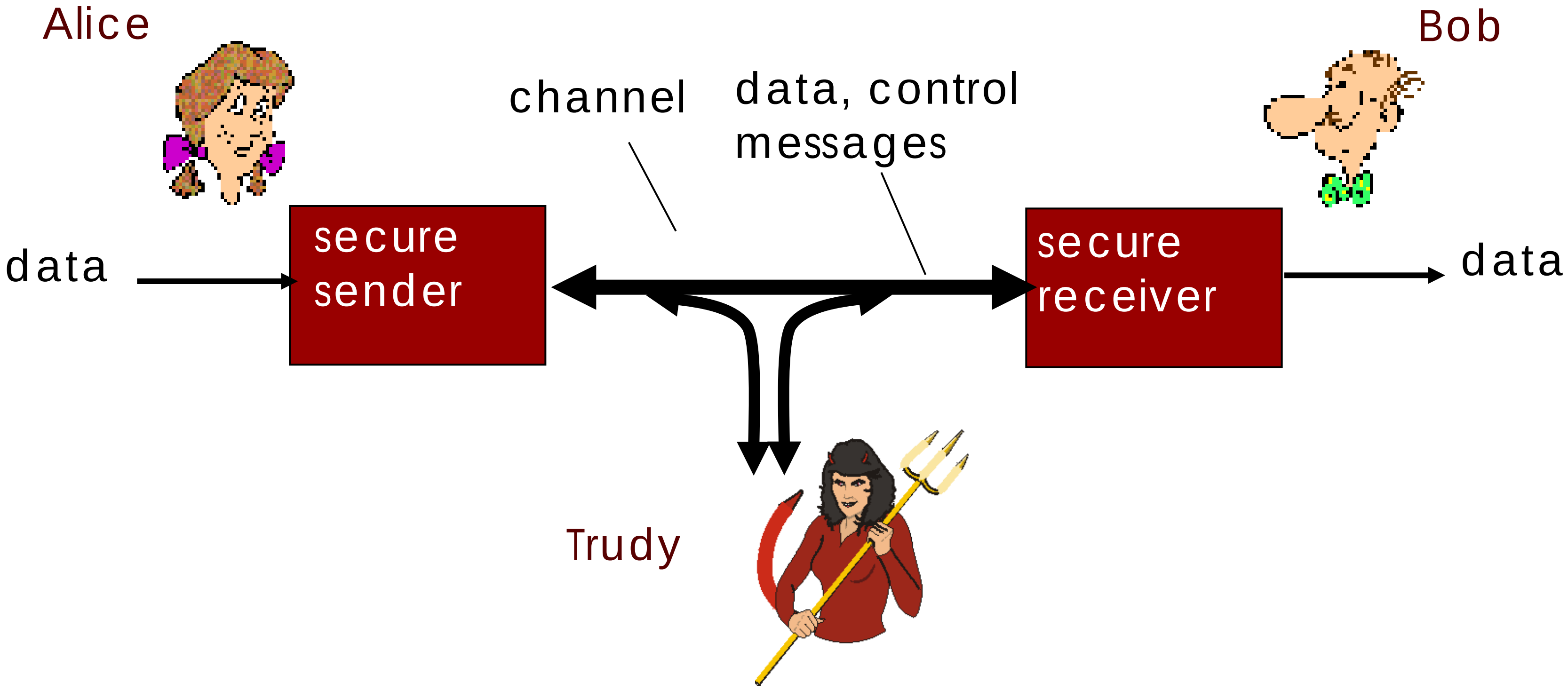
- There are two possible approaches to defeating the RSA algorithm.
- The first is the brute-force approach: try all possible private keys. Thus, the larger the number of bits in e and d, the more secure the algorithm. However, because the calculations involved, both in key generation and in encryption/decryption, are complex, the larger the size of the key, the slower the system will run.
- Most discussions of cryptanalysis of RSA have focused on the task of factoring n into its two prime factors. For a large n with large prime factors, factoring is a hard problem, but not as hard as it used to be. In April of 1994, a group working over the internet and using over 1600 computers claimed the prize after only l8 months of work [LEUT94]. This challenge used a public-key size (length of n) of 129 decimal digits, or around 428 bits. Currently, a 1024-bit key size (about 300 decimal digits) is considered strong enough for virtually all applications.

Steganography/Watermarking versus Cryptography

- Steganography/watermarking: the purpose of both is to provide secret communication.
- Cryptography hides the contents of the message from an attacker, but not the existence of the message.
- Steganography/watermarking even hide the very existence of the message in the communicating data.
- Consequently, the concept of breaking the system is different for cryptosystems and stegosystems (watermarking systems).
- A cryptographic system is broken when the attacker can read the secrete message.
- Breaking of a steganographic/watermarking system has two stages:
  - The attacker can detect that steganography/watermarking has been used;
  - The attacker is able to read, modify or remove the hidden message.
- A steganography/watermarking system is considered as insecure already if the detection of steganography/watermarking is possible.

Friends and enemies: Alice, Bob, Trudy

- well-known in network security world
- Bob and Alice want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



Who might Bob, Alice be?

- Web browser/server for electronic transactions (e.g., on-line purchases)
- on-line banking client/server
- DNS servers
- routers exchanging routing table updates

Who might Trudy be?

- *eavesdrop:* intercept messages
- actively *insert* messages into connection
- *impersonation:* can fake (spoof) source address in packet (or any field in packet)
- *hijacking:* “take over” ongoing connection by removing sender or receiver, inserting himself in place
- *denial of service*: prevent service from being used by others (e.g., by overloading resources)

Computer Forensics and Security

70s. System Admins directly monitor user activities

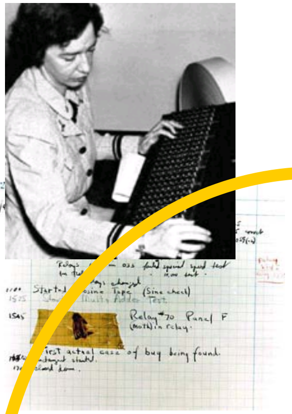
90s. Real time IDS

Late 70 - early 80s. System Admins review audit logs for evidence of unusual behavior.

1990s - First Commercial Antivirus

Programs analyze audit log, usually at night.

1991 – Norton Antivirus released by Symantec



Grace Hopper. MIT - First Computer Bug

Penrose: Self-reproducing machines

Covert attack

Computer viruses on ARPANET

Trojan Horse

Morris worm

Melissa virus, damage = $80 M

Phishing Attacks

Misinformation

Sober

Why not use the concept of Covert Channel for information security against Hackers?

1940 **1945** **1951** **1959** **~1960** **1970** **1982** **1988** **1990** **1999** **2001** **2005** **2008**

Von Neumann studied self reproducing mathematical automata

Von Neumann demonstrated how to create self-reproducing automata

Stahl reproduces Penrose idea in machine code on an IBM 650

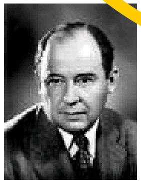
First virus in the wild

Malicious programs exploit vulnerabilities in applications and operating systems

Code Red worm, damage = $2 B

Defending Covert Channel

FOCUS OF MOST SECURITY WORK



THEORETICAL WORK

Emerging

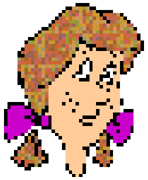
Intrusion Detection System (IDS) are becoming popular that are mainly based on signature matching and anomaly detection

What is Covert Channel?

- A covert channel is a “parasitic communication channel” that is neither designed nor intended to transfer information at all [Lampson 1973]
- A covert channel refers to the mechanism of stealth information transfer using a legitimate communication channel visible to the rest of the world
- The main focus is to hide secret, valuable information through the usage of some other “normal, harmless” information
- Particularly applicable in Multimedia Networking

A simple illustration: “Harmless” Communication

Alice



(Transmitter)

Hello

Bob

I

am

Alice

Are

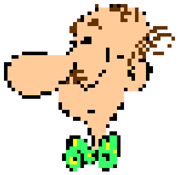
you

There

Listening?

Ed
(Eavesdropper)

Bob



(Receiver)

Covert Channel

Alice

(Transmitter)

Hello

Bob

I

am

Alice

Are

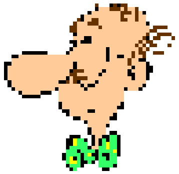
you

There

Listening?

Ed
(Eavesdropper)

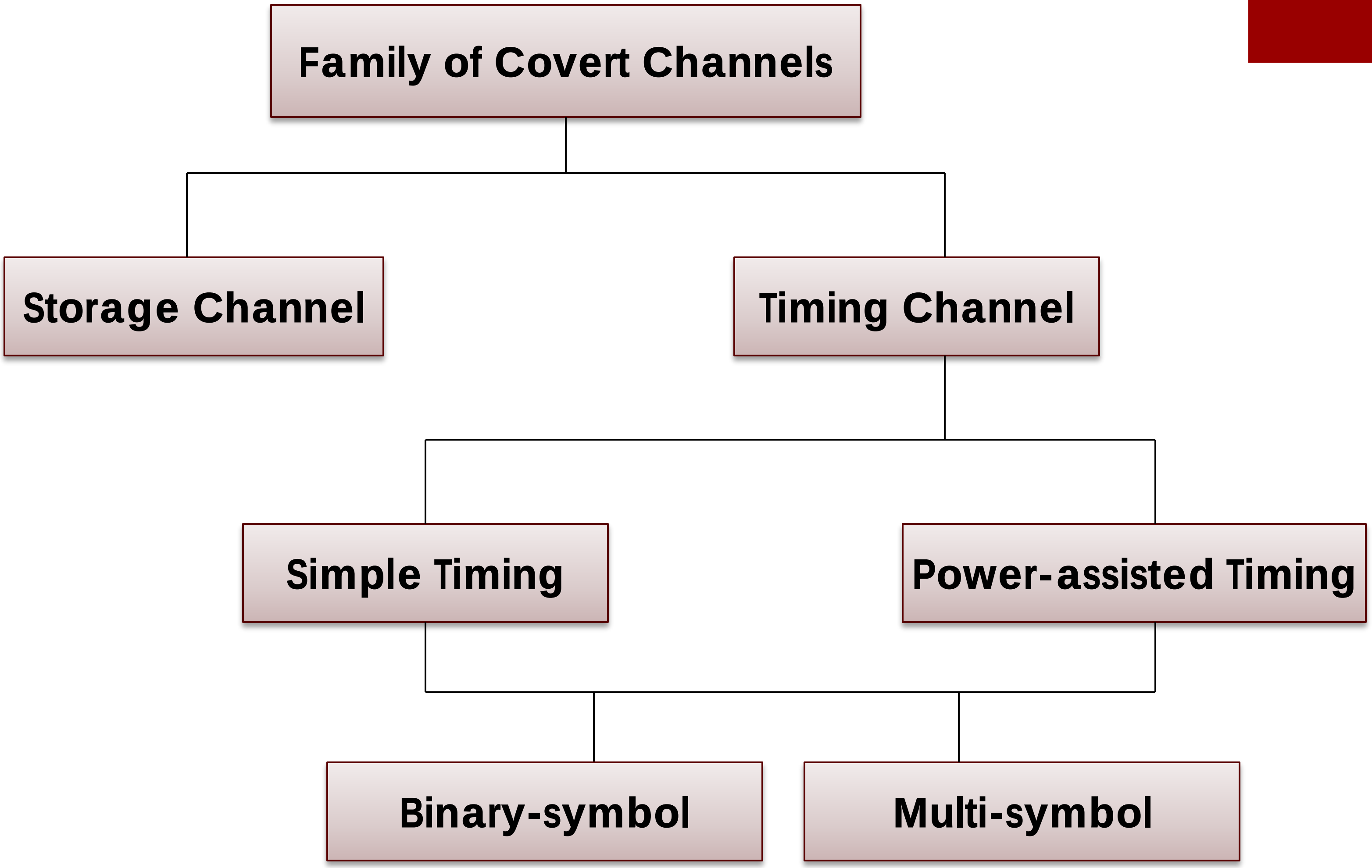
Bob



(Receiver)

Inter-arrival time	Covert Bit Sequence
1s	0
2s	1
2s	1
2s	1
1s	0
1s	0
2s	1
2s	1

Hierarchy of Covert Channels



Covert Channels Definitions

- Covert storage channels are based on the use of a shared data storage area and rely on locks or semaphores (Typical example, Trojan Horse)

- A potential covert channel is a timing channel if its scenario of use involves a process that modulates the interval between the packet transmission times

Challenges in Covert Timing Channels

- Good for wired medium but Unreliable wireless medium – challenge to implement
- Results in variable delay, which is a prime factor in designing timing channel
- For a binary timing channel, inter-packet delays for ‘0’ and ‘1’ are different
  - Different transmission rates
  - Asymmetric bit error rates (‘0’ received as ‘1’ and vice-versa)
- Higher error rates with shorter inter-packet delay transmissions