Unit III. Asymmetric Ciphers
Public Key Cryptosystems, RSA

Er. Kobid Karkee
Himalaya College of Engineering

Background

- Public-key cryptography is also known as asymmetric cryptography
- Motivation behind Public-Key cryptography are two difficult problems in Symmetric Key cryptography:

1. Key distribution:
    - Some sort of mechanism or protocol is needed to provide for the secure distribution of keys between two parties.
    - Frequent key changes are usually desirable in case an attacker knows the key.
    - The strength of any cryptographic system rests with the key distribution technique.

Background

2. Digital signatures:

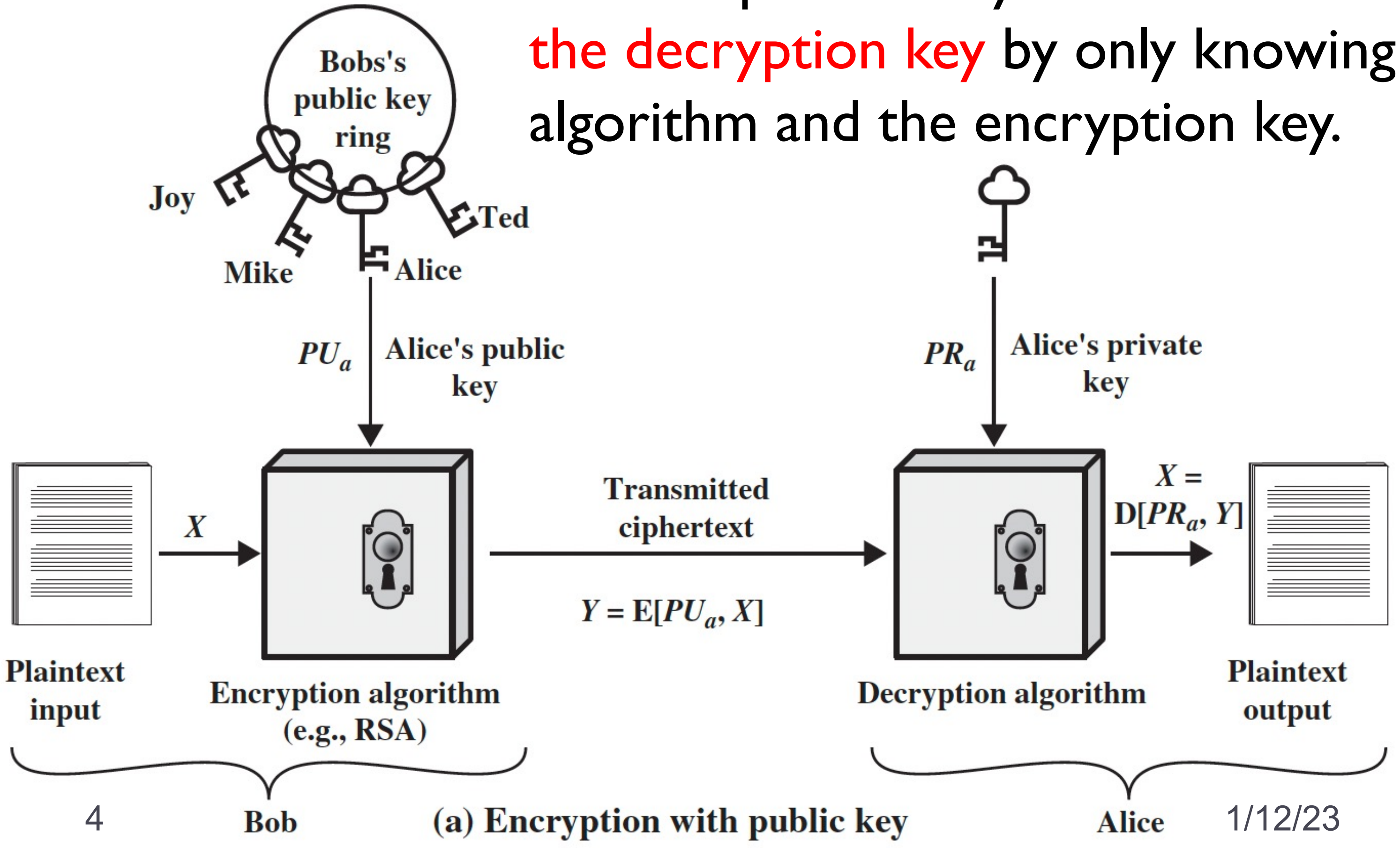
  - What-if electronic messages and documents would need the equivalent of signatures used in paper documents?
  - Could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person?

- Whitfield Diffie and Martin Hellman , in 1976, proposed a radically different method (Diffie-Hellman key exchange) that defined public-key cryptography.

"What good would it do after all to develop impenetrable cryptosystems, if their users were forced to share their keys with a KDC that could be compromised by either burglary or subpoena?" - Diffie

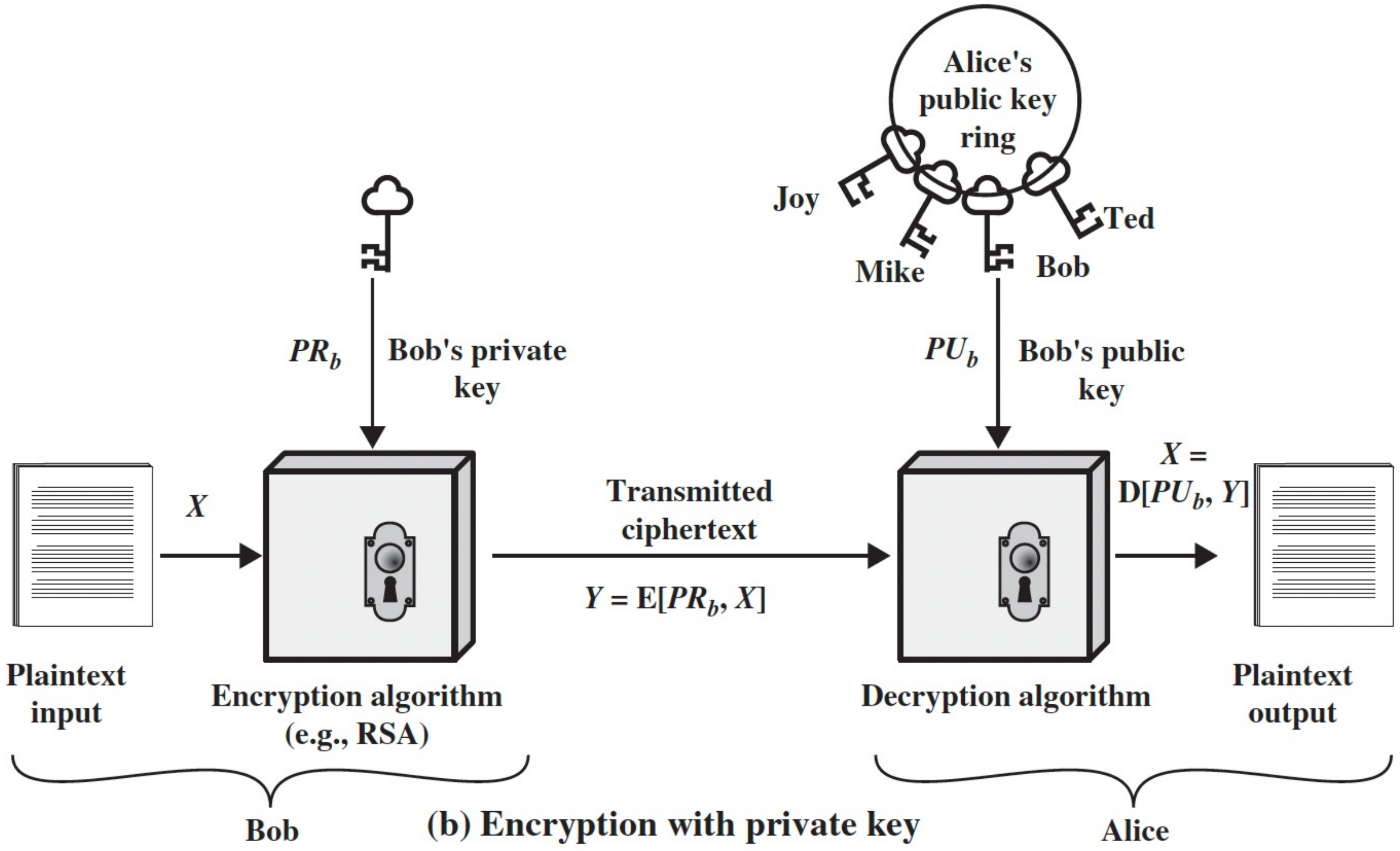
Public-Key Cryptosystems

- Asymmetric algorithms rely on one key for encryption and a different but related key for decryption.
  - It is computationally infeasible to determine the decryption key by only knowing the algorithm and the encryption key.



Public-Key Cryptosystems

- Either of the two related keys can be used for encryption, with the other used for decryption, e.g., RSA.



(b) Encryption with private key

Public-Key Encryption Scheme

Has six ingredients: -

1. Plaintext:

   readable message that is fed into the algorithm as input.
2. Encryption algorithm:

   performs various transformations on the plaintext.
3. Public and private keys:

   a pair of keys : one is used for encryption; the other is used for decryption..
4. Ciphertext:

   the scrambled message produced as output.
5. Decryption algorithm:

   accepts the ciphertext and the matching key and produces the original plaintext.

Essential steps: -

1. Each user generates a pair of keys.
2. Each user places one of the two keys in a public register or other accessible file. The companion key is kept private.
3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4. When Alice receives the message, she decrypts it using her private key.

Note: At any time, a system can change its private key and publish the companion public key to replace its old public key.

Conventional vs Public-Key Encryption

Symmetric Encryption	**Public-Key Encryption**
Needed to Work:	Needed to Work:
1. The same algorithm with the same key is used for encryption and decryption.	1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption.
2. The sender and receiver must share the algorithm and the key.	2. The sender and receiver must each have one of the matched pair of keys (not the same one).
Needed for Security:	Needed for Security:
1. The key must be kept secret.	1. One of the two keys must be kept secret.
2. It must be impossible or at least impractical to decipher a message if no other information is available.	2. It must be impossible or at least impractical to decipher a message if no other information is available.
3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

MISCONCEPTIONS CONCERNING PUBLIC-KEY ENCRYPTION

- Public-key encryption is more secure from cryptanalysis than symmetric encryption
  - Not true – they depend on different principles, but can be equally secure
- Public-key encryption has made symmetric encryption obsolete
  - Not true – symmetric encryption is still used in several areas, quite successfully.

Public-Key Cryptosystem: Secrecy

Plaintext $X = [X_1, X_2, \ldots, X_M]$

Pair of keys:

a public key (Pu_b),

a private key (Pr_b)

Encryption function

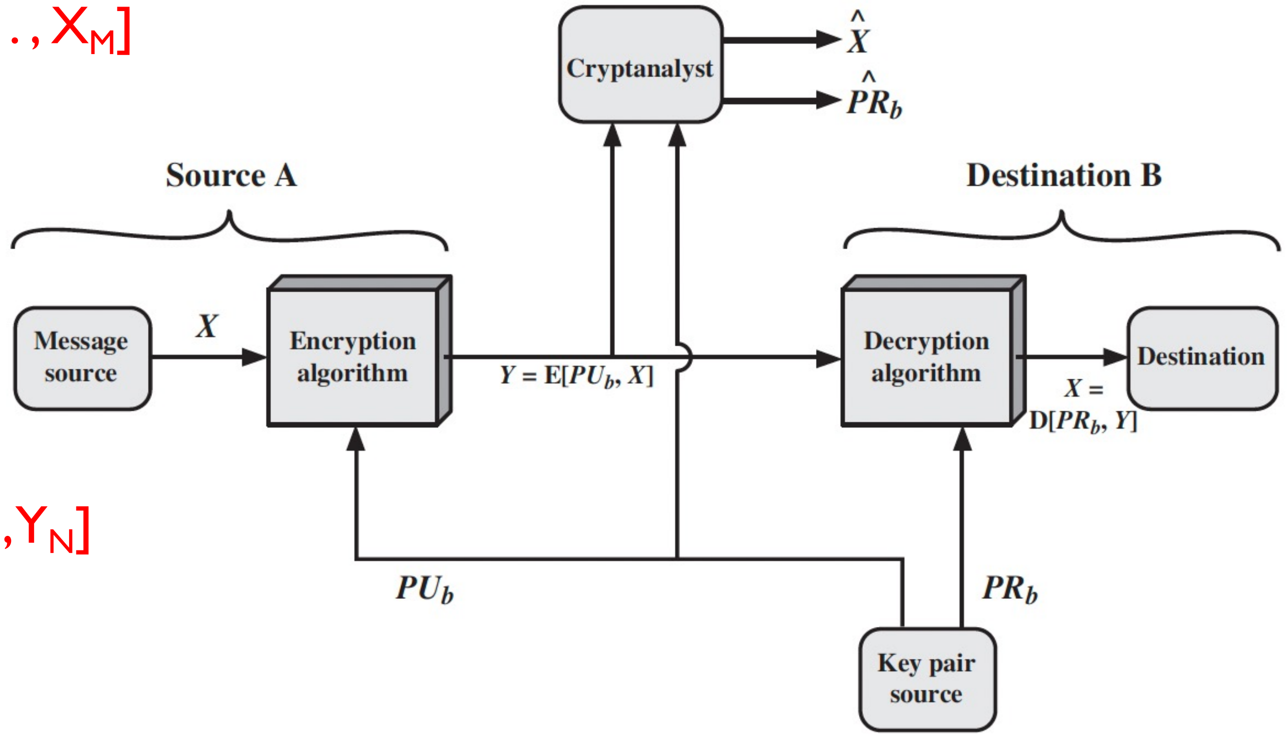
$Y = E(Pu_b, X)$

Ciphertext

$Y = [Y_1, Y_2, \ldots, Y_N]$

Decryption function

$X = D(Pr_b, Y)$



In a public-key encryption system, anyone with a public key can encrypt a message, yielding a ciphertext, but only those who know the corresponding private key can decrypt the ciphertext to obtain the original message.

Public-Key Cryptosystem: Authentication

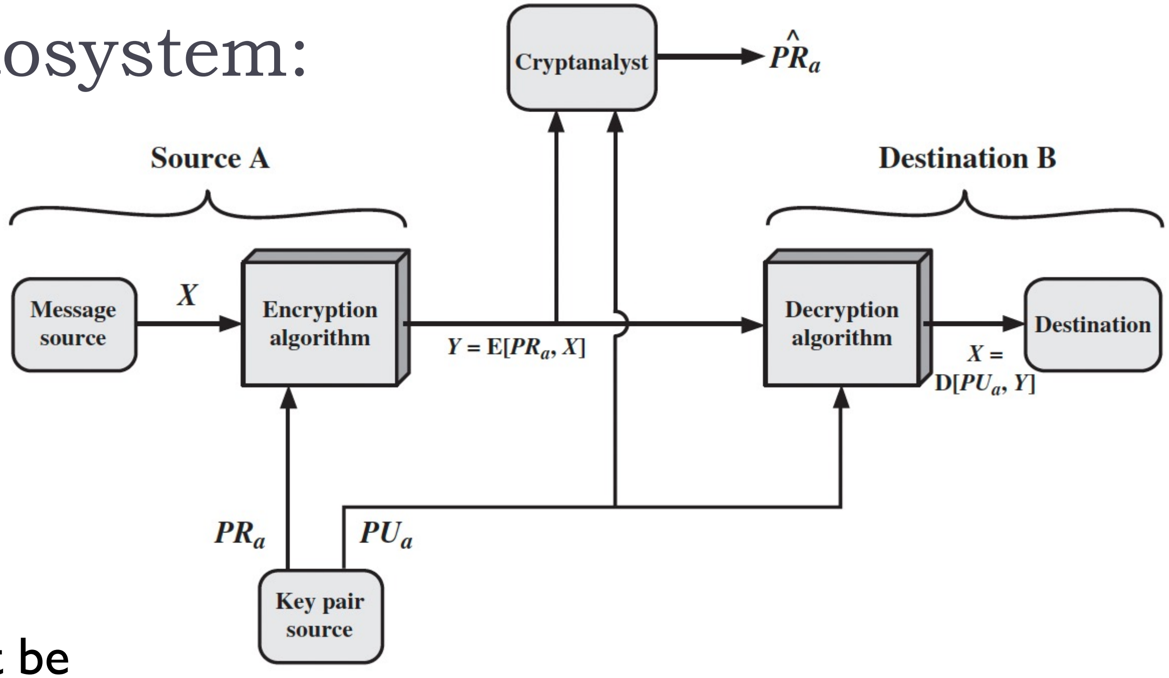
Encryption $Y = E(PR_a, X)$
Decryption $X = D(PU_a , Y)$

The entire encrypted message serves as a **digital signature.**

Benefit: the message cannot be altered without access to A's private key and hence maintains data integrity.

Problem: requires more storage.

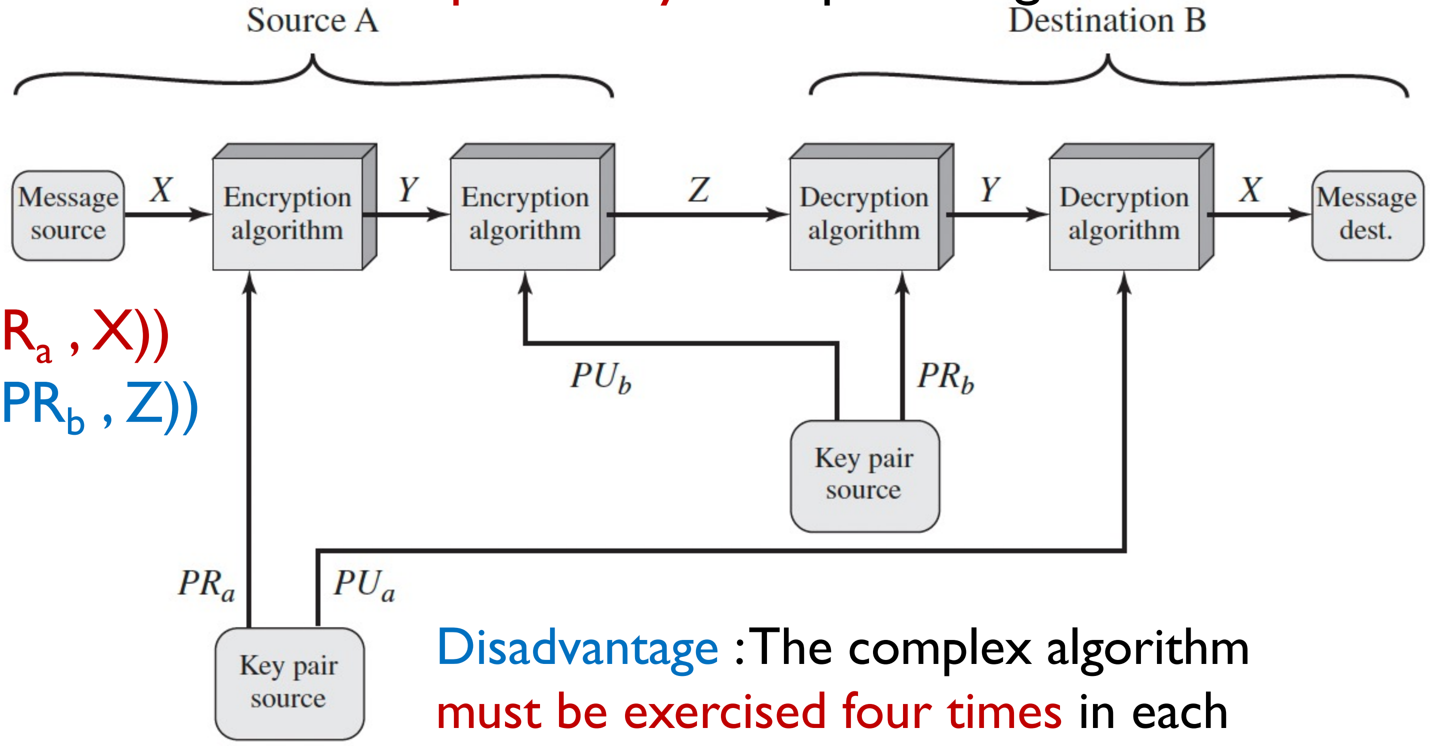
A more efficient way of achieving the same results is to encrypt a small block of bits (called authenticator) of the document.



In a digital signature system, a sender can use a private key together with a message to create a signature.
Anyone with the corresponding public key can verify whether the signature matches the message, but a forger who does not know the private key cannot find any message/signature pair that will pass verification with the public key.

Public-Key Cryptosystem: Authentication and Secrecy

Begin by encrypting a message, using the sender's private key, which provides the *digital signature*.
Next, encrypt with the receiver's public key, thus providing the *confidentiality*.



$Z = E(PU_b, E(PR_a, X))$

$X = D(PU_a, D(PR_b, Z))$

Disadvantage : The complex algorithm must be exercised four times in each communication.

Public-Key Cryptosystem Application

- Public-key cryptosystems can be classified into three categories:

1. Encryption/decryption: The sender encrypts a message with the recipient's public key
2. Digital Signatures: The sender "signs" a message with its private key
3. Key Exchange: Two sides cooperate to exchange a session key

- Some algorithms are suitable for all three applications, whereas others can be used only for one or two

Public-Key Cryptosystem Application

Algorithm	Encryption/ Decryption	Digital Signature	Key Exchange
RSA	YES	YES	YES
Elliptic Curves	YES	YES	YES
Diffie-Hellman	NO	NO	YES
DSS	NO	YES	NO

Public-Key Requirements

Computationally easy

- for party B to generate a pair (public-key PU_b , private key PR_b)
- for sender A, knowing the public key and the message, to generate the corresponding ciphertext
- for receiver B to decrypt the resulting ciphertext using the private key to recover the original message

Computationally infeasible for an adversary

- knowing the public key, to determine the private key
- knowing the public key and a ciphertext, to recover the original message

Public-Key Requirements

- Need a trap-door one-way function
- $f : \{0,1\}^n \rightarrow \{0,1\}^n$ is a one-way function if
  - $Y = f(X)$ can easily be computed for X in $\{0,1\}^n$
  - $X = f^{-1}(Y)$ infeasible for Y in $\{0,1\}^n$
- A trap-door one-way function is a family of invertible functions f_k, such that computing
  - $Y = f_k(X)$ is easy, if k and X are is known
  - $X = f_k^{-1}(Y)$ is easy, if k and Y are known
  - $X = f_k^{-1}(Y)$ infeasible, if Y is known but k not known
- A practical public-key scheme depends on a suitable trapdoor one-way function

Security of Public-Key Schemes

- A public-key encryption scheme also is vulnerable to a brute-force attack. So, the key size must be large enough to make brute-force attack impractical but small enough for practical encryption and decryption.
- Another form of attack is to find some way to compute the private key given the public key. To date, it has not been mathematically proven that this form of attack is infeasible for a particular public-key algorithm. Therefore, RSA algorithm has been a constant target of cryptanalysts.
- A probable-message attack is peculiar for public-key cryptosystems. An adversary could encrypt all possible keys using the public key and could discover the encrypted key by matching the transmitted ciphertext.

RSA Algorithm

- Developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and published in 1978.
- The Rivest-Shamir-Adleman (RSA) scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and n-1 for some n.
- A typical size for n is 1024 bits, or 309 decimal digits.
  - That is, n is less than 2^{1024}.
- It requires computation of key pair (K_{pub}, K_{pr}), unlike any symmetric algorithm (AES, 3DES etc.).
- Security is due to cost of factoring large numbers
  - Note: factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup

- each user generates a public/private key pair by:
- selecting **two large primes** at random - p, q
- computing their system modulus $N=p.q$
  - **note** $\varnothing(N)=(p-1)(q-1)$ (Euler Totient Function remember!!!)
- selecting at random the **encryption key** e
  - where $1<e<\varnothing(N)$, $\gcd(e,\varnothing(N))=1$
- solve following equation to find **decryption key** d
  - $e.d=1 \bmod \varnothing(N)$ and $0\le d\le N$
  - Here, $d\equiv e^{-1}$ (use Extended Euclidean Algorithm for getting gcd and d)
- publish their public encryption key: KU={e,N}
- keep secret private decryption key: KR={d,p,q}

RSA Use

- to encrypt a message M the sender:
  - obtains **public key** of recipient **`KU={e,N}`**
  - computes: $C=M^e \bmod N$, where $0 \leq M < N$
- to decrypt the ciphertext C the owner:
  - uses their private key **`KR={d,p,q}`**
  - computes: $M=C^d \bmod N$
- note that the message M must be smaller than the modulus N (block if needed)

Why RSA Works?

- because of Euler's Theorem:
- $a^{\varnothing(n)} \bmod N = 1$
  - where $\gcd(a,N)=1$
- in RSA have:
  - $N=p.q$
  - $\varnothing(N)=(p-1)(q-1)$
  - carefully chosen e & d to be inverses $\bmod \varnothing(N)$
  - hence $e.d=1+k.\varnothing(N)$ for some k
- hence :

  $C^d = (M^e)^d = M^{1+k.\varnothing(N)} = M^1.(M^{\varnothing(N)})^q = M^1.(1)^q = M^1 = M \bmod N$

RSA Example

1. Select primes: $p=17$ & $q=11$
2. Compute $n = pq = 17 \times 11 = 187$
3. Compute $\varnothing(n) = (p-1)(q-1) = 16 \times 10 = 160$
4. Select $e : \gcd(e,160)=1$; choose $e=7$
5. Determine d: $de=1 \bmod 160$ and $d < 160$

   Value is $d=23$ since $23 \times 7 = 161 = 10 \times 160 + 1$
6. Publish public key $KU=\{7,187\}$
7. Keep secret private key $KR=\{23,17,11\}$

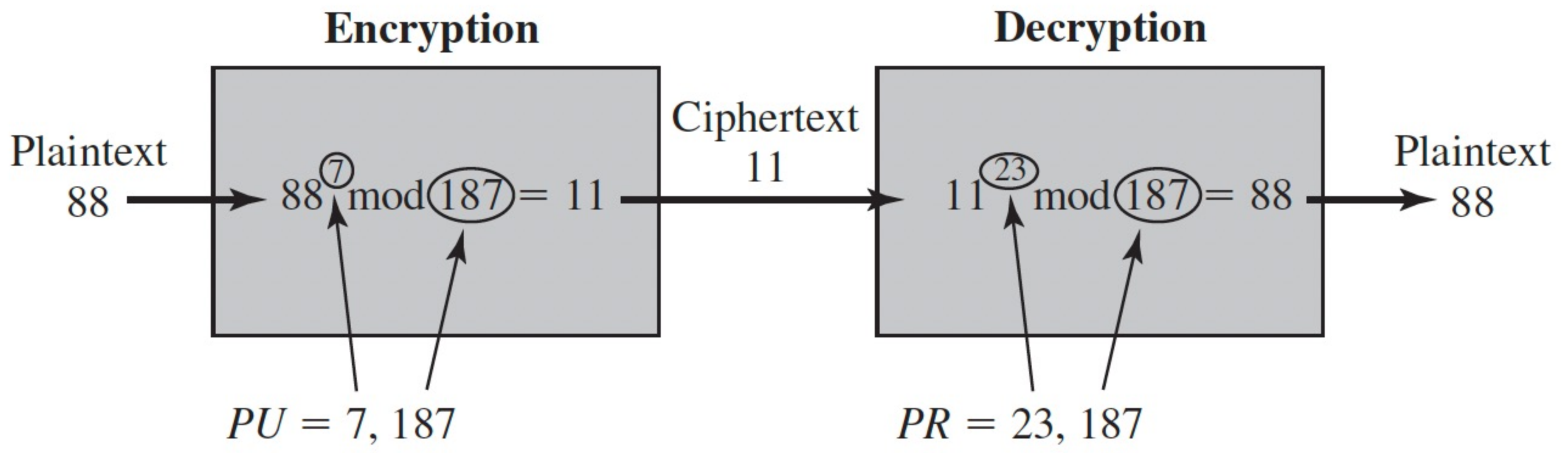
- Here walk-through example using "trivial" sized numbers. Selecting primes requires the use of primality tests. Finding d as inverse of e mod ø(n) requires use of Inverse algorithm

RSA Example cont.

- sample RSA encryption/decryption is:
- given message M = 88 (note: 88<187)
- encryption:

  $C = 88^7 \bmod 187 = 11$
- decryption:

  $M = 11^{23} \bmod 187 = 88$



RSA Encryption: Another Example

Alice

$X = 4$

$Y = X^e \bmod n$

$= 4^3 \bmod 33 = 31$

Bob

1. $p = 3, q = 11$
2. $n = 33$
3. $\phi(n) = 2 \times 10 = 20$
4. $e = 3$
5. $d \equiv e\text{-}1 \equiv 7 \pmod{20}$

$Y = 31$

$X = Y^d \bmod n$

$= 317\text{mod } 33 \equiv 4 \bmod 33$

Efficient Exponentiation

(Applicable to RSA encryption and decryption)

Naive way

$x^{1024} = x.x = x^2$

$= x^2.x = x^3$

…

…

$= x^{1023}.x$

{involved 1023 multiplications}

Better way

$x^{1024} = x.x = x^2$

$= x^2.x^2 = x^4$

…

…

$= x^{512}.x^{512}$

{involved 10 multiplications}

Efficient Exponentiation (Square and Multiply)

- Also called binary method or left-to-right exponentiation.
- Rules: -

1. Scan the exponent bit left-to-right
2. In every iteration, do SQUARE.
3. If current bit is 1, MULTIPLY by x.

Example (Decimal):	Example (Binary):
x^{26}	x^{11010}
$= (x^1)^2 = x^2$	$= (x^1)^2 = x^{10}$
$= x^2.x = x^3$	$= x^{10}.x = x^{11}$
$= (x^3)^2 = x^6$	$= (x^{11})^2 = x^{110}$
$= (x^6)^2 = x^{12}$	$= (x^{110})^2 = x^{1100}$
$= x^{12}.x = x^{13}$	$= x^{1100}.x = x^{1101}$
$= (x^{13})^2 = x^{26}$	$= (x^{1101})^2 = x^{11010}$

only takes $O(\log_2 n)$ multiples for number n

RSA Security

Four possible approaches to attacking the RSA algorithm :

- Brute force:
  - This involves trying all possible private keys.
- Mathematical attacks:
  - There are several approaches, all equivalent in effort to factoring the product of two primes.
- Timing attacks:
  - These depend on the running time of the decryption algorithm.
- Chosen ciphertext attacks:
  - This type of attack exploits properties of the RSA algorithm

Factoring Problem

- mathematical approach takes 3 forms:
  - factor N=p.q, hence find ø(N) and then d
  - determine ø(N) directly and find d
  - find d directly
- currently believe all equivalent to factoring
  - have seen slow improvements over the years
    - as of Aug-99 best is 130 decimal digits (512) bit with GNFS
  - biggest improvement comes from improved algorithm
    - cf "Quadratic Sieve" to "Generalized Number Field Sieve"
  - barring dramatic breakthrough 1024+ bit RSA secure
    - ensure p, q of similar size and matching other constraints

Timing Attacks

- developed in mid-1990's
- exploit timing variations in operations
  - eg. multiplying by small vs large number
  - or IF's varying which instructions executed
- infer operand size based on time taken
- RSA exploits time taken in exponentiation
- countermeasures
  - use constant exponentiation time
  - add random delays
  - blind values used in calculations