Chapter 9

Public Key Cryptography and RSA

1

Public-Key Encryption

- Also referred to as non-secret, asymmetric or two-key encryption
- Most significant revolution in cryptography in modern times
  - Became widely used in the late 70s
- Rely on mathematical equations for transformations
  - Symmetric schemes rely on permutations and substitutions
- Require two keys: one public and one private
  - This allows public-key ciphers to be used in other tasks e.g. authentication and signature generation

2

Table 9.1 Terminology Related to Asymmetric Encryption

Asymmetric Keys
Two related keys, a public key and a private key that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

Public Key Certificate
A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key. The certificate indicates that the subscriber identified in the certificate has sole control and access to the corresponding private key.

Public Key (Asymmetric) Cryptographic Algorithm
A cryptographic algorithm that uses two related keys, a public key and a private key. The two keys have the property that deriving the private key from the public key is computationally infeasible.

Public Key Infrastructure (PKI)
A set of policies, processes, server platforms, software and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, and revoke public key certificates.

Source: *Glossary of Key Information Security Terms*, NIST IR 7298

3

Misconceptions Concerning Public-Key Encryption

- Public-key encryption is more secure from cryptanalysis than symmetric encryption
- Public-key encryption is a general-purpose technique that has made symmetric encryption obsolete
- There is a feeling that key distribution is trivial when using public-key encryption, compared to the cumbersome handshaking involved with key distribution centers for symmetric encryption

4

Principles of Public-Key Cryptosystems

- The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption:

Key distribution
- How to have secure communications in general without having to trust a KDC with your key

Digital signatures
- How to verify that a message comes intact from the claimed sender

- Whitfield Diffie and Martin Hellman from Stanford University achieved a breakthrough in 1976 by coming up with a method that addressed both problems and was radically different from all previous approaches to cryptography

5

Public-Key Cryptosystems

- A public-key encryption scheme has six ingredients:

Plaintext	Encryption algorithm	Public key	Private key	Ciphertext	Decryption algorithm
The readable message or data that is fed into the algorithm as input	Performs various transformations on the plaintext	Used for encryption or decryption	Used for encryption or decryption	The scrambled message produced as output	Accepts the ciphertext and the matching key and produces the original plaintext

6

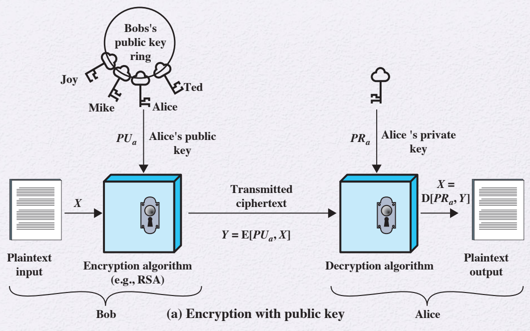


Figure 9.1 Public-Key Cryptography

7

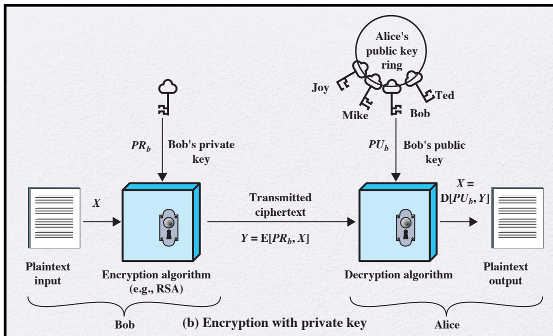


Figure 9.1 Public-Key Cryptography

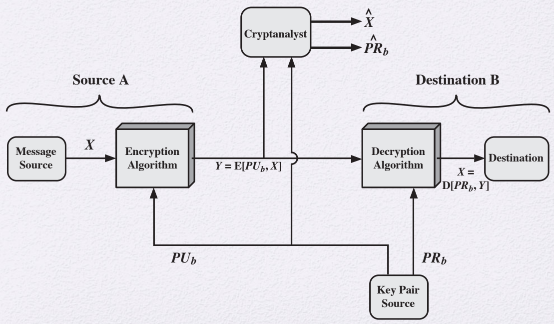
8

Table 9.2 CONVENTIONAL AND PUBLIC-KEY ENCRYPTION

Conventional 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 a related algorithm for 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 the key is kept secret.	2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret.
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.

9

Public-Key Cryptosystem: Confidentiality



10

Public-Key Cryptosystem: Authentication

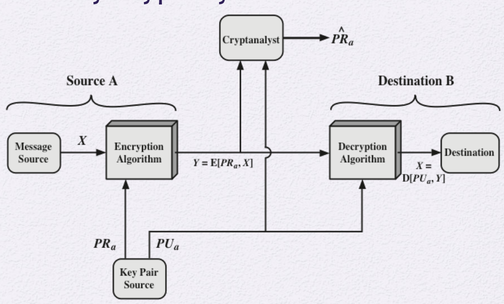


Figure 9.3 Public-Key Cryptosystem: Authentication

11

Applications for Public-Key Cryptosystems

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

Encryption/decryption	• The sender encrypts a message with the recipient's public key
Digital signature	• The sender "signs" a message with its private key
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

12

Table 9.3

Applications for Public-Key Cryptosystems

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Table 9.3 Applications for Public-Key Cryptosystems

- RSA – Rivest-Shamir-Adleman Cipher
- DSS – Digital Signature Standard Cipher

13

Public-Key Requirements

- Conditions that these algorithms must fulfill:
  1. It is computationally **easy** for a party B to generate a pair (public-key PU_b, private key PR_b)
  2. It is computationally **easy** for a sender A, knowing the public key and the message to be encrypted, to generate the corresponding ciphertext
  3. It is computationally **easy** for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message
  4. It is computationally **infeasible** for an adversary, knowing the public key, to determine the private key
  5. It is computationally **infeasible** for an adversary, knowing the public key and a ciphertext, to recover the original message
  6. The two keys can be applied in **either** order

14

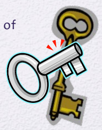
Public-Key Requirements

- Need a trap-door one-way function
  - A one-way function is one that maps a domain into a range such that every function value has a unique inverse, with the condition that the calculation of the function is easy, whereas the calculation of the inverse is infeasible
    - $Y = f(X)$ easy
    - $X = f^{-1}(Y)$ infeasible
- A trap-door one-way function is a family of invertible functions f_k, such that
  - $Y = f_k(X)$ easy, if k and X are known
  - $X = f_k^{-1}(Y)$ easy, if k and Y are known
  - $X = f_k^{-1}(Y)$ infeasible, if Y known but k not known
- A practical public-key scheme depends on a suitable trap-door one-way function

15

Public-Key Cryptanalysis

- A public-key encryption scheme is vulnerable to a brute-force attack
  - Countermeasure: use large keys
  - Key size must be small enough for practical encryption and decryption
  - Key sizes that have been proposed result in encryption/decryption speeds that are too slow for general-purpose use
  - Public-key encryption is currently confined to key management and signature applications
- 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
- Finally, there is a probable-message attack
  - This attack can be thwarted by appending some random bits to simple messages



16

Rivest-Shamir-Adleman (RSA) Algorithm

- Developed in 1977 at MIT by Ron Rivest, Adi Shamir & Len Adleman
- Most widely used general-purpose approach to public-key encryption
- Is a 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

17

RSA Algorithm

- RSA makes use of an expression with exponentials
- Plaintext is encrypted in blocks with each block having a binary value less than some number n
- Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C

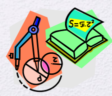
  $C = M^e \bmod n$

  $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$
- Both sender and receiver must know the value of n
- The sender knows the value of e, and only the receiver knows the value of d
- This is a public-key encryption algorithm with a public key of $PU=\{e,n\}$ and a private key of $PR=\{d,n\}$

18

Algorithm Requirements

- For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:
  1. It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$
  2. It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$
  3. It is infeasible to determine d given e and n



19

Key Generation by Alice

Select p, q	p and q both prime, $p \neq q$
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	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Figure: RSA Key Generation

20

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Private Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure: RSA Encryption/Decryption

21

RSA Implementation

RSA Step	Problem	Number Theory Method
Select p , q	Finding large prime numbers	Miller-Rabin Algorithm
Calculate n	Calculations involving large integers	Chinese Remainder Theorem
Calculate Φ (n)	Calculations involving large integers	Chinese Remainder Theorem
Select integer e	Find GCD	Euclidean Algorithm
Calculate d	Find Modular Multiplicative Inverse	Extended Euclidean Algorithm
$C=M^e \bmod n$, $M= C^d \bmod n$	Exponentiation to a large power	Euler's Theorem

22

Example of RSA Algorithm

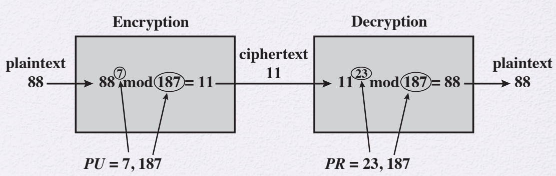
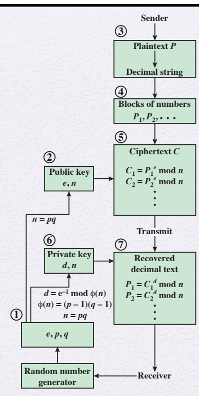


Figure 9.6 Example of RSA Algorithm

23

Figure 9.7

RSA Processing of Multiple Blocks

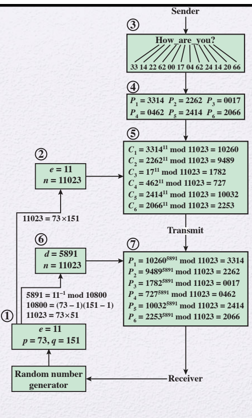


(a) General approach

24

Figure 9.7

RSA Processing of Multiple Blocks

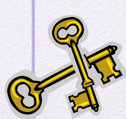


(b) Example

25

Key Generation

- Before the application of the public-key cryptosystem each participant must generate a pair of keys:
  - Determine two prime numbers *p* and *q*
  - Select either *e* or *d* and calculate the other
- Because the value of *n* = *pq* will be known to any potential adversary, primes must be chosen from a sufficiently large set
  - The method used for finding large primes must be reasonably efficient



26

Procedure for Picking a Prime Number

- Pick an odd integer *n* at random
- Pick an integer *a* < *n* at random
- Perform the probabilistic primality test with *a* as a parameter. If *n* fails the test, reject the value *n* and go to step 1
- If *n* has passed a sufficient number of tests, accept *n*; otherwise, go to step 2

27

The Security of RSA

28

Factoring Problem

- We can identify three approaches to attacking RSA mathematically:
  - Factor *n* into its two prime factors. This enables calculation of $ø(n) = (p - 1) \times (q - 1)$, which in turn enables determination of $d = e^{-1} \pmod{ø(n)}$
  - Determine $ø(n)$ directly without first determining *p* and *q*. Again this enables determination of $d = e^{-1} \pmod{ø(n)}$
  - Determine *d* directly without first determining $ø(n)$

29

Timing Attacks

- Paul Kocher, a cryptographic consultant, demonstrated that a snooper can determine a private key by keeping track of how long a computer takes to decipher messages
- Are applicable not just to RSA but to other public-key cryptography systems
- Are alarming for two reasons:
  - It comes from a completely unexpected direction
  - It is a ciphertext-only attack

30

Countermeasures

Constant exponentiation time
- Ensure that all exponentiations take the same amount of time before returning a result; this is a simple fix but does degrade performance

Random delay
- Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack

Blinding
- Multiply the ciphertext by a random number before performing exponentiation; this process prevents the attacker from knowing what ciphertext bits are being processed inside the computer and therefore prevents the bit-by-bit analysis essential to the timing attack

31

Fault-Based Attack

- An attack on a processor that is generating RSA digital signatures
  - Induces faults in the signature computation by reducing the power to the processor
  - The faults cause the software to produce invalid signatures which can then be analyzed by the attacker to recover the private key
- The attack algorithm involves inducing single-bit errors and observing the results
- While worthy of consideration, this attack does not appear to be a serious threat to RSA
  - It requires that the attacker have physical access to the target machine and is able to directly control the input power to the processor

32

Summary

- Present an overview of the basic principles of public-key cryptosystems
- Explain the two distinct uses of public-key cryptosystems
- List and explain the requirements for a public-key cryptosystem

- Present an overview of the RSA algorithm
- Understand the timing attack

33