Lecture 6

Asymmetric Cryptography

R. Rhouma

UTAS
Sultanate of Oman
September 2022

CSSY2201 : Introduction to Cryptography

Plan

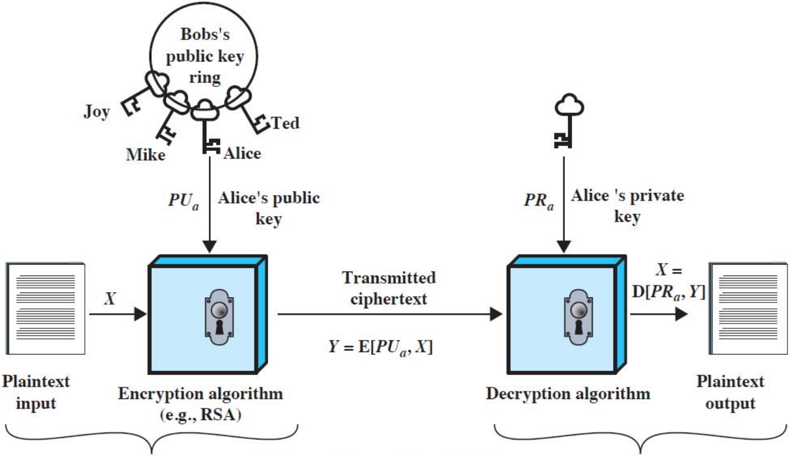
Principles of asymmetric cryptography

- asymmetric cryptography solve two main problems :
  - Distribution of keys : How to establish a secret communication without the intervention of a third trusted party (KDC : Key distribution Center)
  - Digital Signature : A way to authenticate the message and the message origin (the sender authentication)
- Whitfield Diffie and Martin Hellman from Stanford University have proposed in 1976 the first new concept of asymmetric cryptography.

Terminology of Asymmetric cryptography

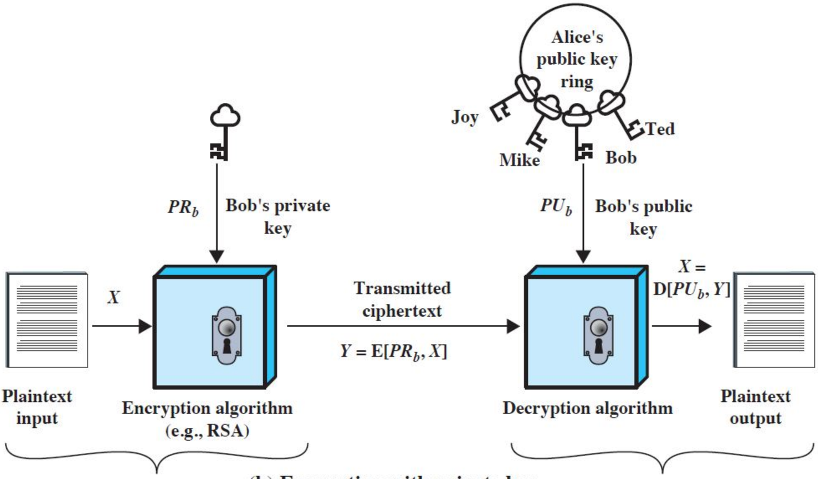
- Plaintext : original text
- Ciphertext : encrypted text
- Encryption : The process of conversion from plaintext to ciphertext
- Decryption : The process of conversion from ciphertext to plaintext
- Public key : used in the Encryption algorithm (by the sender for confidentiality)
- Private key : used in the Decryption algorithm (by the receiver for confidentiality)

Encryption with the public key



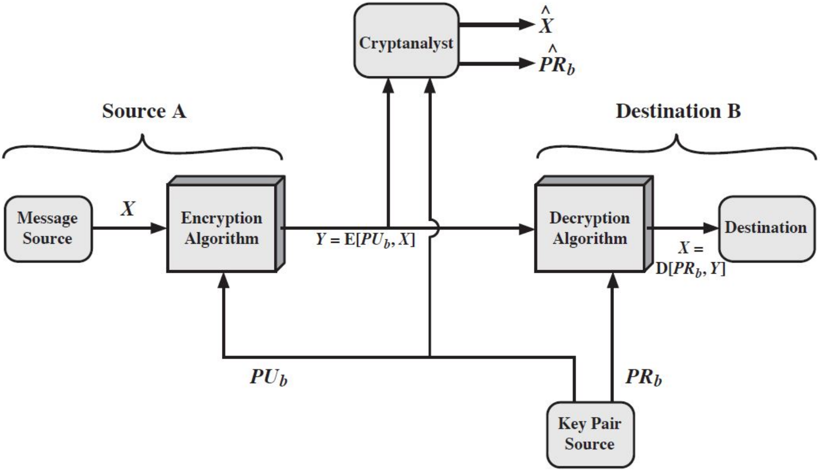
(a) Encryption with public key

Encryption with the private key

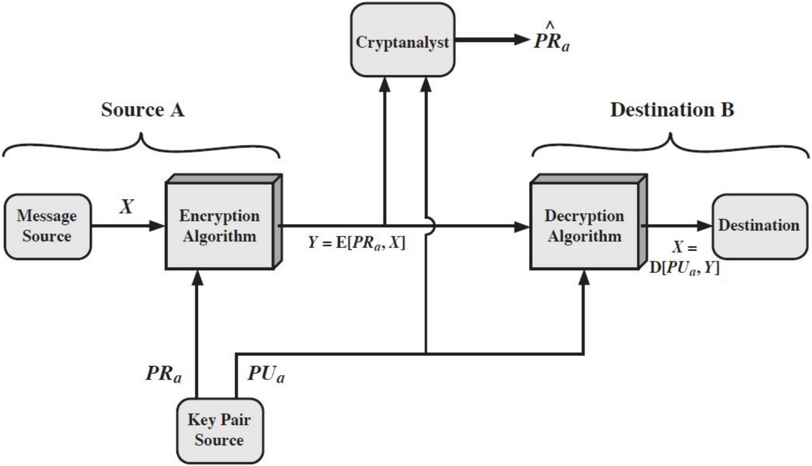


(b) Encryption with private key

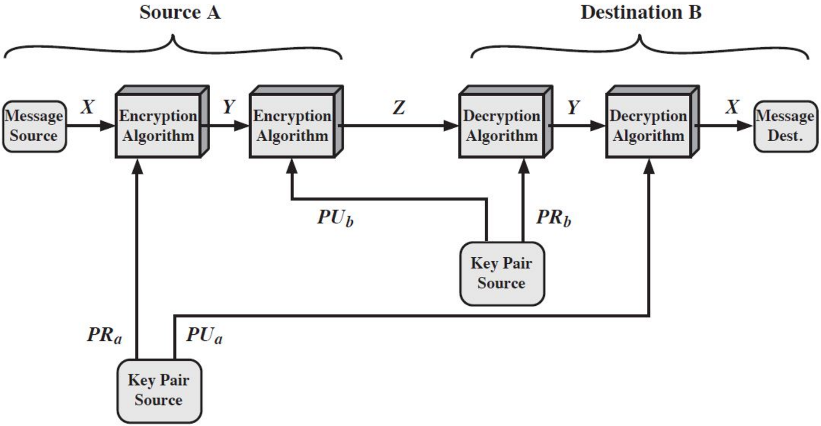
Asymmetric cryptography : confidentiality



Asymmetric cryptography : authentication



Asymmetric cryptography : confidentiality & authentication



Applications of asymmetric cryptography

- Asymetric cryptography are used in three main applications :
  - Encryption/Decryption : Sender encrypts a plaintext using the receiver's public key.
  - Digital signature : Sender signs a message using his private key.
  - Key Exchange : Sender and receiver negotiate to establish a common secret key.
- Some algorithms can be used in all those applications. Some others are appropriate for one or to applications.

Applications of asymmetric cryptography

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

Asymmetric Cryptography Requirements

These alg must have the following requirements

- need one way trap function
- a one-way function verifies the following :
  - $Y = f(X)$ is easy
  - $X = f^{-1}(Y)$ is not feasible
- a one-way trapdoor function is a f_k family of reversible functions satisfying :
  - $Y = f_k(X)$ is easy if k and X are known
  - $X = f_k^{-1}(Y)$ is easy if k and Y are known
  - $X = f_k^{-1}(Y)$ is not feasible if Y is known and k not known
- a public key alg is therefore based on a one-way trap function

RSA

- Developed in 1977 at MIT by Ron Rivest, Adi Shamir& Len Adleman
- the most used public key alg
- is an alg whose plaintext and ciphertext are integers between 0 and n-1.
- n is a number of size 1024 bits or 309 decimal digits

RSA Algorithm

- the plaintext is encrypted in blocks, each block has a value less than n
- Encryption of a plaintext block is as follows :

$$C = M^e \; mod \; n$$

- decryption is as follows :

$$M = C^d \; mod \; n = (M^e)^d \; mod \; n = M^{ed} \; mod \; n$$

- sender and receiver know the value of n
- The sender knows the value of e
- only the receiver knows the value of d
- the public key is the pair (e, n)
- the private key is the pair (d, n)

Key Generation

- each user generates their own keys (private and public) by :
- select two large primes p and q
- calculate $n = p \times q$
- calculate $\phi(n) = (p-1) \times (q-1)$
- randomly select a number e with : $1 < e < \phi(n)$ and $GCD(e, \phi(n)) = 1$
- solve this equation to find d with $0 \leq d \leq n$:

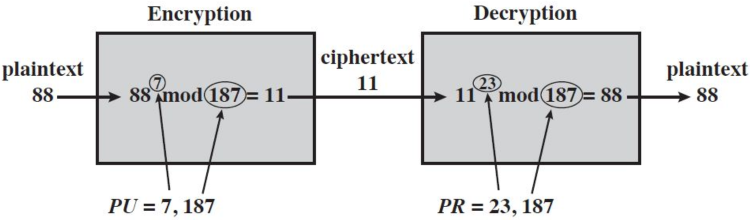
$$e \times d = 1 \; mod \; \phi(n)$$

  d is called the multiplicative inverse of $e \; mod \phi(n)$
- publish pair $PU = \{e, n\}$ as public key
- secretly keep $PR = \{d, n\}$ pair as private key

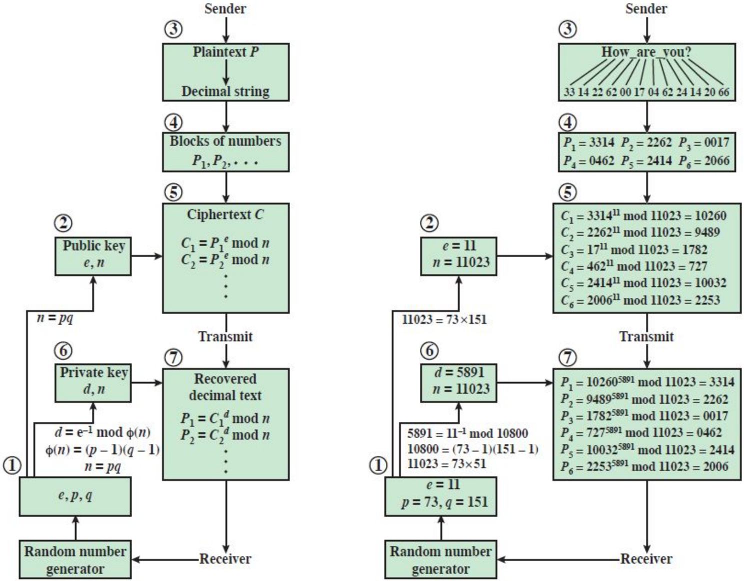
Toy example

1. Select prime numbers : $p = 17$ & $q = 11$
2. Calculate $n = p \times q = 17 \times 11 = 187$
3. Calculate $\phi(n) = (p - 1) \times (q - 1) = 16 \times 10 = 160$
4. Select e : $gcd(e, 160) = 1$; choose $e = 7$
5. Determine d such that $d \times e = 1\ mod\ 160$: the value is $d = 23$ since $23 \times 7 = 161 = 10 \times 160{+}1$
6. Publish public key $PU = \{7, 187\}$
7. Keep private key $PR = \{23, 187\}$ secret

Enryption/decryption



Example RSA on a long message



Exponentiation in RSA

- encryption and decryption manipulates exponentiations of large numbers modulo n
- we can use properties of modular arithmetic :

$$(a \, mod \, n) \times (b \, mod \, n) = (a \times b) \, mod \, n$$

- we must also try to do the exponentiation as quickly as possible
- we can render the exponentiation in $O(log_2 n)$ multiplications for a number n
  ex1 : $7^5 = 7^4 \times 7^1 = 3 \times 7 = 10 \, mod \, 11 \Longrightarrow log_2 5 = 3$ multiplications
  ex2 : $3^{129} = 3^{128} \times 3^1 = 5 \times 3 = 4 \, mod \, 11 \Longrightarrow log_2 129 = 8$ multiplications

Computing of a^b *mod n*

```
c ← 0; f ← 1
for i  ← k downto 0
      do    c ← 2 × c
            f ← (f × f) mod n
      if    b_i = 1
            then c ←  c + 1
                 f ← (f × a) mod n
return f
```

with b is an integer converted to binary in $b_k b_{k-1} \dots b_0$
Example of calculation of a^b *mod n*, for $a = 7$,
$b = 560 = (1000110000)_2$, and $n = 561$

i	9	8	7	6	5	4	3	2	1	0
b_i	1	0	0	0	1	1	0	0	0	0
c	1	2	4	8	17	35	70	140	280	560
f	7	49	157	526	160	241	298	166	67	1

Attacks on RSA ?

Three approaches to attack RSA :

- Factor n into two primes p and q. This will lead to find $\phi(n) = (p-1)(q-1)$ which leads to determine $d = e^{-1}\, mod\, \phi(n)$
- determine $\phi(n)$ directly without finding p and q, which leads to determine $d = e^{-1}\, mod\, \phi(n)$
- Determine d directly without determining $\phi(n)$