Asymmetric cryptography

Asymmetric (Block) Ciphers

Use key pairs

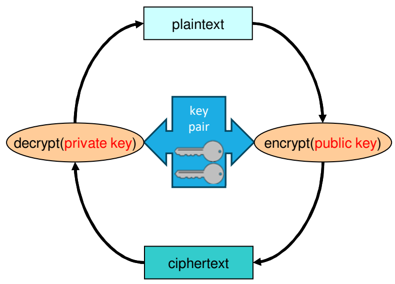
- One private key (personal, not transmittable)
- One public key, available to all

Allow

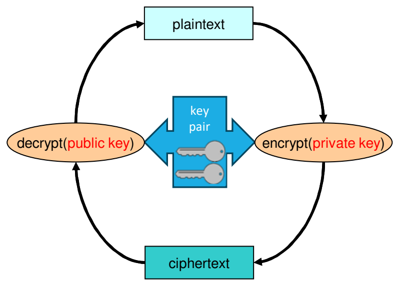
- Confidentiality without any previous exchange of secrets
- Authentication
  - Of contents (data integrity)
  - Of origin (source authentication, or digital signature)

Operations of an asymmetric cipher

Confidentiality



Authentication (signature)



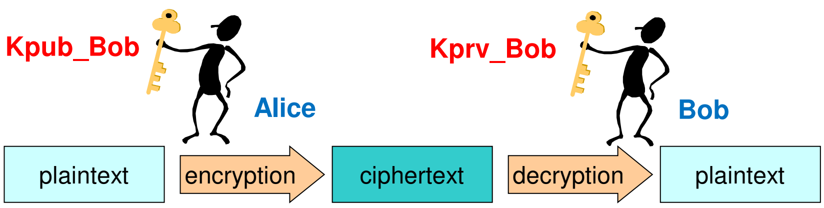
Use cases: secure communication

Secure communication with a target (Bob)

- Alice encrypts plaintext **P** with Bob's public key **Kpub_Bob**

  $\texttt{Alice: C = \{P\}}_{\mathbf{kpub\_Bob}}$
- Bob decrypts cyphertext **C** with his private key **Kprv_Bob**

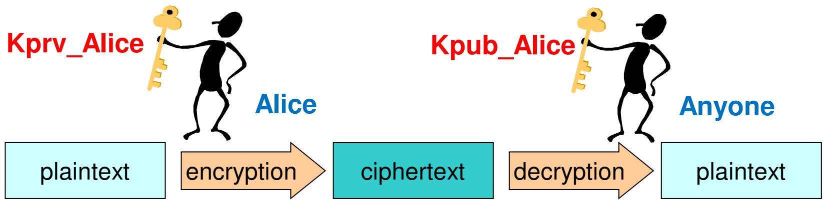
  $\texttt{Bob: P'= \{C\}}_{\mathbf{kprv\_Bob}}$
- **P'** should be equal to **P** (requires checking)
- **Kpub_Bob** needs to be known by Alice



Use cases: signature

Data signature by Alice

- Alice encrypts plaintext **P** with her private key **Kprv_Alice**
  - Alice: $C = \{P\}_{kprv\_Alice}$
- Anyone can decrypt cyphertext **C** with Alice's public key **Kpub_Alice**
  - Anyone: $P' = \{C\}_{kpub\_Bob}$
- If **P'** = **P**, then **C** is Alice's signature of **P**
- **Kpub_Alice** needs to be known by signature verifiers



Asymmetric ciphers

Advantages

- They are a fundamental authentication mechanism
- They allow to explore features that are not possible with asymmetric ciphers

Disadvantages

- Performance
- Usually are very inefficient and memory consuming

Problems

- Trustworthy distribution of public keys
- Lifetime of key pairs

Asymmetric ciphers

Approaches: complex mathematic problems

- Discrete logarithms of large numbers
- Integer factorization of large numbers

Most common algorithms

- RSA
- ElGamal
- Elliptic curves (ECC)

Other techniques with asymmetric key pairs

- Diffie-Hellman (key agreement)

RSA (Rivest, Shamir, Adelman, 1978)

Keys

- Private: (d, n)
- Public: (e, n)

Public key encryption (confidentiality)

- $C = P^e \bmod n$
- $P = C^d \bmod n$

P, C are numbers

$0 \leq P, C < n$

Private key encryption (signature)

- $C = P^d \bmod n$
- $P = C^e \bmod n$

RSA (Rivest, Shamir, Adelman, 1978)

Computational complexity

- Discrete logarithm
- Integer factoring

coprime → gcd(a, b) = 1
× → multiplication
mod → modulo operation
≡ → modular congruence

Key selection

- Large n (hundreds or thousands of bits)
- n = p × q with p and q being large (secret) prime numbers
- Chose an e co-prime with (p-1) × (q-1)
- Compute d such that e × d ≡ 1 (mod (p-1) × (q-1))
- Discard p and q
- The value of d cannot be computed out of e and n
  - Only from p and q

RSA example

p = 5 q = 11 (prime numbers)

- n = p x q = 55
- (p-1) x (q-1) = 40

e = 3 (public key = e, n)

- Coprime of 40

d = 27 (private key = d, n)

- e x d ≡ 1 (mod 40) → d x e mod 40 = 1, (27 x 3) mod 40 = 1

For P = 26 (notice that P, C∈ [0, n-1])

- $C = P^e \bmod n = 26^3 \bmod 55 = 31$
- $P = C^d \bmod n = 31^{27} \bmod 55 = 26$

Hybrid encryption

Combines symmetric with asymmetric cryptography

- Use the best of both worlds, while avoiding problems
- Asymmetric cipher: Uses public keys (but it is slow)
- Symmetric cipher: Fast (but with weak key exchange methods)

Method:

- Obtain K_{pub} from the receiver
- Generate a random K_{sym}
- Calculate $C1 = E_{sym}(K_{sym}, P)$
- Calculate $C2 = E_{asym}(K_{pub}, K_{sym})$
- Send C1 + C2
  - C1 = Text encrypted with symmetric key
  - C2 = Symmetric key encrypted with the receiver public key
    - May also contain the IV

Randomization of asymmetric encryptions

Non-deterministic (unpredictable) result of asymmetric encryptions

- **N** encryptions of the same value, with the same key, should yield **N** different results
- **Goal:** prevent the trial & error discovery of encrypted values

Approaches

- Concatenation of value to encrypt with two values
  - A fixed one (for integrity control)
  - A random one (for randomization)

Randomization of asymmetric encryptions: OAEP (Optimal Asymmetric Encryption Padding)

lHash: digest over **Label**

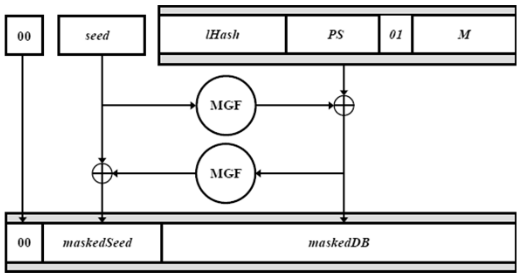
seed: random

PS: zeros

M: plaintext

MGF: Mask Generation Function

- Similar to Hash, but with variable size



Diffie-Hellman Key Agreement (1976)

q (large prime)
α (primitive root mod q)

a = random		**b = random**
$Y_a = \alpha^a \bmod q$	$Y_a \rightarrow$	
	$\leftarrow Y_b$	$Y_b = \alpha^b \bmod q$
$K_{ab} = Y_b^{\,a} \bmod q$		$K_{ba} = Y_a^{\,b} \bmod q$

$K_{ab} = K_{ba}$

DH Key Agreement: MitM attack

Alice		Attacker		Bob
a = random		c = random		b = random
$Y_a = \alpha^a \bmod q$	Y_a →	$Y_c = \alpha^c \bmod q$	← Y_b	$Y_b = \alpha^b \bmod q$
	← Y_c		Y_c →	
$K_{ac} = Y_c^{\,a} \bmod q$		$K_{ca} = Y_a^{\,c} \bmod q$ $K_{cb} = Y_b^{\,c} \bmod q$		$K_{bc} = Y_c^{\,b} \bmod q$

Elliptic Curve Cryptography (ECC)

Elliptic curves are specific functions

- They have a generator (G)
- A private key K_{prv} is an integer with a maximum of bits allowed by the curve
- A public key K_{pub} is a point $(x,y) = K_{prv} \times G$
- Given K_{pub}, it should be hard to guess K_{prv}

Curves

- NIST curves (15)
  - P-192, P-224, P-256, P-384, P-521
  - B-163, B-233, B-283, B-409, B-571
  - K-163, K-233, K-283, K-409, K-571

Other curves

- Curve25519 (256 bits)
- Curve448 (448 bits)

ECDH: DH with ECC

ECC curve → G

a = **random**		b = **random**
$Y_a = a\,G$	Y_a →	
	← Y_b	$Y_b = b\,G$
$K_{ab} = a\,Y_b$	$K_{ab} = K_{ba}$	$K_{ba} = b\,Y_a$

ECC public key encryption

Combines hybrid encryption with ECDH

Method:

- Obtain K_{pub_recv} from the receiver
- Generate a random K_{prv_send} and the corresponding K_{pub_send}
- Calculate $K_{sym} = K_{prv\_send}\,K_{pub\_recv}$
- $C = E(P, K_{sym})$
- Send $C + K_{pub\_send}$

- Receiver calculates $K_{sym} = K_{pub\_send}\,K_{prv\_recv}$
- $P = D(C, K_{sym})$