03. Public Key Encryption

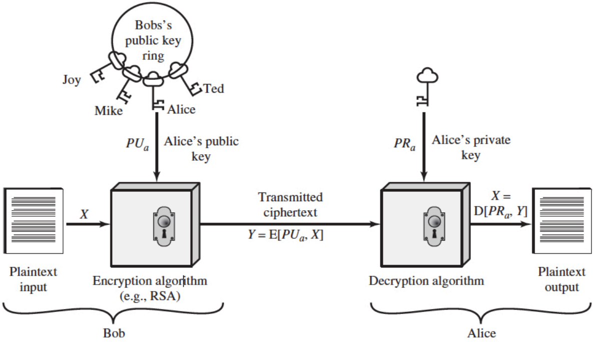
이형태

2019학년도 2학기

Overview of Public Key Encryption

Public Key Encryption

- A key for encryption is different from a key for decryption



(a) Encryption with public key

- RSA, ElGamal, NTRU
- Pros: Easy for key sharing, small number of keys
- Cons: Slower than symmetric (private key) encryption

Picture from [SB15]

Definition of Public Key Encryption

Definition

A public key encryption **PKE** consists of the following (polynomial-time) algorithms:

- KeyGen(λ): It takes a security parameter λ as an input and returns a public key pk and a secret key sk.
- Enc(pk, M): It takes a public key pk and a plaintext M as inputs and returns a ciphertext C.
- Dec(sk, C): It takes a secret key sk and a ciphertext C as inputs and returns a plaintext M.

Correctness

A public key encryption **PKE** is *correct* if the following holds: For any security parameter λ and any plaintext M,

$$\mathrm{Dec}(sk, \mathrm{Enc}(pk, M)) = M$$

where (pk, sk) is an output of KeyGen(λ).

Security Model for Public Key Encryption

- Consider the following game between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$:

1. **Setup**: $\mathcal{C}$ runs $\mathrm{KeyGen}(\lambda)$ to obtain (pk, sk) and passes a public key pk to $\mathcal{A}$.
2. **Phase 1**: $\mathcal{A}$ asks decryption queries on C_i's to a decryption oracle $\mathcal{D}_1$ and receives corresponded plaintexts.
3. **Challenge**: $\mathcal{A}$ submits two plaintexts M_0, M_1 to $\mathcal{C}$. $\mathcal{C}$ tosses a coin $b \in \{0, 1\}$, runs $\mathrm{Enc}(pk, M_b) \rightarrow C_b^*$, and passes C_b^* to $\mathcal{A}$ as the challenge ciphertext.
4. **Phase 2**: $\mathcal{A}$ asks decryption queries on C_i's to a decryption oracle $\mathcal{D}_2$ and receives corresponded plaintexts. The constraint is that C_b^* cannot be queried.
5. **Guess**: $\mathcal{A}$ outputs b'.

$\mathcal{A}$ wins the game if $b = b'$. The advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}(\lambda) := \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

Security Model for Public Key Encryption (Cont.)

Definition

A public key encryption scheme is IND-XXX secure if there is no adversary whose advantage is non-negligible in the security parameter.

- XXX is determined by decryption oracles allowed to the adversary
  - XXX = CPA (chosen plaintext attacks): Neither $\mathcal{D}_1$ nor $\mathcal{D}_2$ are allowed.
  - XXX = CCA (non-adaptive chosen ciphertext attacks): Only $\mathcal{D}_1$ is allowed.
  - XXX = CCA2 (adaptive chosen ciphertext attacks): Both $\mathcal{D}_1$ and $\mathcal{D}_2$ are allowed.
- Consider "computational security" and reduction from security of public key encryption schemes to cryptographic hard problems:
  - e.g.) If a factoring problem is hard, RSA encryption is secure.

    $\Longleftrightarrow$ If RSA encryption is broken, then we can factor a hard-to-factor integer.

Essential Number Theory

RSA: Key Generation

Key Generation

1. Select two large primes p and q
2. Compute $N = pq$
3. Compute $\phi(N) = (p-1)(q-1)$
4. Select the public exponent $e \in \mathbb{Z}^*_{\phi(N)}$. (Note that $\gcd(e, \phi(N)) = 1$.)
5. Compute the private key d such that

$$d \cdot e \equiv 1 \pmod{\phi(N)}$$

   (Use Extended Euclidean Algorithm!)

6. Output a pair of the public key and private key

$$pk = (N, e), \quad sk = d$$

RSA: Encryption & Decryption

Encryption

Given the public key $pk = (N, e)$ and a plaintext $M \in \mathbb{Z}_N$, compute

$$C := \mathsf{RSA.Enc}(pk, M) = M^e \pmod{N}$$

and output C.

Decryption

Given the private key $sk = d$ and a ciphertext $C \in \mathbb{Z}_N$, compute

$$M' := \mathsf{RSA.Dec}(sk, C) = C^d \pmod{N}$$

and output M'.

Modular Exponentiation

- $g^a \pmod p$: a remainder when g^a is divided by p

$$\begin{aligned} & 5^4 \pmod{31} \\ &= 5 \cdot 5 \cdot 5 \cdot 5 \pmod{31} \\ &= 25 \cdot 25 \pmod{31} \\ &= 625 \pmod{31} \\ &= 5 \ (\because 625 = 20 \cdot 31 + 5) \end{aligned}$$

$$\begin{aligned} & 3^9 \pmod{31} \\ &= 3 \cdot 3 \cdots 3 \pmod{31} \\ &= 9^4 \cdot 3 \pmod{31} \\ &= 19,683 \pmod{31} \\ &= 29 \ (\because 19,683 = 634 \cdot 31 + 29) \end{aligned}$$

- Naive way to compute $g^a \pmod p$: Need $(a-1)$ modular multiplications

Left-to-Right Algorithm for Exponentiation

Algorithm 1 Left-to-Right Algorithm for Modular Exponentiation

Input: three positive integers $g, a = (a_{\ell-1}, a_{\ell-2}, \ldots, a_1, a_0)_2$ and p
Output: $g^a \pmod p$

$R \leftarrow 1$
for i from $\ell - 1$ to 0 **do**
    $R \leftarrow R \cdot R \pmod p$
    if $a_i = 1$ **then**
        $R \leftarrow R \cdot g \pmod p$
    end if
end for
return R

- Need $\lfloor \log_2 a \rfloor + \mathrm{HW}(a) - 1$ multiplications where $\mathrm{HW}(a)$ is the number of ones in a_i's for $0 \leq i \leq \ell - 1$

Right-to-Left Algorithm for Modular Exponentiation

Algorithm 2 Right-to-Left Algorithm

Input: three positive integers $g, a = (a_{\ell-1}, a_{\ell-2}, \ldots, a_1, a_0)_2$ and p
Output: $g^a \pmod{p}$

```
R ← 1, T ← g
for i from 0 to ℓ − 1 do
    if a_i = 1 then
        R ← R · T (mod p)
    end if
    T ← T · T (mod p)
end for
return R
```

- As the left-to-right algorithm, need $\lfloor \log_2 a \rfloor + \mathrm{HW}(a) - 1$ multiplications where $\mathrm{HW}(a)$ is the number of ones in a_i's for $0 \le i \le \ell - 1$

Euclidean Algorithm over the Integers

Fact

If a and b are positive integers with $a > b$, then $\gcd(a, b) = \gcd(b, a \bmod b)$.

Algorithm 3 Euclidean Algorithm over the Integers

Input: two non-negative integers a and b, with $a \geq b$

Output: $\gcd(a, b)$

```
r ← a, r′ ← b
while r′ ≠ 0 do
    r″ ← r mod r′
    (r, r′) ← (r′, r″)
end while
d ← r
return d
```

Extended Euclidean Algorithm over the Integers

Theorem

Let a, b, r be integers and $d = \gcd(a, b)$. Then, there exist $s, t \in \mathbb{Z}$ such that $as + bt = r$ if and only if $d | r$.

Algorithm 4 Extended Euclidean Algorithm over the Integers

Input: two non-negative integers a and b, with $a \geq b$

Output: d, s, t such that $d = \gcd(a, b)$ and $as + bt = d$

```
r ← a, r′ ← b
s ← 1, s′ ← 0
t ← 0, t′ ← 1
while r′ ≠ 0 do
    q ← ⌊r/r′⌋, r′′ ← r mod r′
    (r, s, t, r′, s′, t′) ← (r′, s′, t′, r′′, s − s′q, t − t′q)
end while
d ← r
return d, s, t
```

Euler Phi Function: ϕ

Definition (Euler Phi Function)

$$\begin{aligned}\phi(m) &= \text{the number of integers in } \mathbb{Z}_m \text{ relatively prime to } m \\ &= \left|\mathbb{Z}_m^* := \{a \in \mathbb{Z}_m \,|\, \gcd(a,m) = 1\}\right|\end{aligned}$$

Theorem

Let $m = p_1^{e_1} p_2^{e_2} \cdots p_n^{e_n}$ where p_i's are distinct primes and e_i's are positive integers. Then,

$$\phi(n) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1})$$

Example

1. $\phi(6) = 2$ ($\because \cancel{0}, 1, \cancel{2}, \cancel{3}, \cancel{4}, 5$), $\phi(5) = 4$ ($\because \cancel{0}, 1, 2, 3, 4$)
2. Let $m = 240 = 2^4 \cdot 3 \cdot 5$. Then

$$\phi(240) = (2^4 - 2^3)(3^1 - 3^0)(5^1 - 5^0) = (16 - 8)(3 - 1)(5 - 1) = 64.$$

Fermat Little Theorem

Theorem (Fermat Little Theorem)

Let a be an integer and p be a prime. Then,

$$a^p \equiv a \pmod{p}.$$

Example

Let $p = 7$ and $a = 2$. Then,

$$2^7 = 128 = (18 \cdot 7 + 2) \pmod{7}.$$

Corollary

Let a be an integer and p be a prime with $\gcd(a, p) = 1$. Then,

1. $a^{p-1} \equiv 1 \pmod{p}$.
2. $a^{-1} = a^{p-2} \pmod{p}$.

$$2^5 = 32 = (4 \cdot 7 + 4) = 4 \pmod{7} \Rightarrow 2 \cdot 4 = 8 = (1 \cdot 7 + 1) = 1 \pmod{7}$$

Euler Theorem

Theorem (Euler Theorem)

Let a and m be integers with $\gcd(a, m) = 1$. *Then,*

$$a^{\phi(m)} = 1 \pmod{m}$$

Example

Let $a = 5$ and $m = 12$. Then

$$\begin{aligned} \phi(12) &= \phi(2^2 \cdot 3) = (2^2 - 2^1)(3^1 - 3^0) = (4-2)(3-1) = 4 \\ & \quad (\because \cancel{0}, \ 1, \ \cancel{2},, \ \cancel{3},, \ \cancel{4},, \ 5, \ \cancel{6},, \ 7, \ \cancel{8},, \ \cancel{9},, \ \cancel{10},, \ 11) \end{aligned}$$

Thus,

$$5^{\phi(12)} = 5^4 = 625 = (52 \cdot 12 + 1) = 1 \pmod{12}.$$

Chinese Remainder Theorem

Theorem ((Simplified) Chinese Remainder Theorem)

Suppose p and q are relatively prime. Then, the system of equations

$$
\begin{aligned}
x &\equiv a \pmod{p} \\
x &\equiv b \pmod{q}
\end{aligned}
$$

has a unique solution for x modulo pq.

Proof.

1. Existence:

   $$x = a \cdot q \cdot M_q + b \cdot p \cdot M_p \pmod{pq}$$

   where $M_q = q^{-1} \pmod{p}$ and $M_p = p^{-1} \pmod{q}$.

2. Uniqueness: If $x = y \pmod{p}$ and $x = y \pmod{q}$, then $x - y$ is a multiple of both p and q. Thus, $x - y$ is a multiple of pq and $x = y \pmod{pq}$.

□

Example of Chinese Remainder Theorem

Example

Find x in $\mathbb{Z}_{105}$ such that $x \equiv 3 \pmod 7$ and $x \equiv 5 \pmod{15}$.

- $105 = 15 \cdot 7$, $\gcd(7, 15) = 1$
- Let $p = 7$ and $q = 15$.
- $M_q = q^{-1} \pmod p = 15^{-1} \pmod 7 = 1$ ($\because 15 \pmod 7 = 1$ & $1 \cdot 1 = 1$)
- $M_p = p^{-1} \pmod q = 7^{-1} \pmod{15} = 13$ ($\because 7 \cdot 13 = 91 = 1 \pmod{15}$)
- $x = a \cdot q \cdot M_q + b \cdot p \cdot M_p = 3 \cdot 15 \cdot 1 + 5 \cdot 7 \cdot 13 = 500 = 80 \pmod{105}$

(Probabilistic) Primality Test I: Fermat Test

Theorem (Fermat Little Theorem)

Let a be an integer and p be a prime. Then,

$$a^p \equiv a \pmod{p} \; (\Longleftrightarrow a^{p-1} \equiv 1 \pmod{p})$$

Algorithm 5 Fermat Primality Test

Input: candidate $\overline{p}$ and security parameter λ
Output: " $\overline{p}$ is prime" or " $\overline{p}$ is composite"

for i from 1 to λ **do**
  select a random element a from $\{2, 3, \ldots, \overline{p} - 2\}$
  if $a^{\overline{p}-1} \neq 1 \pmod{\overline{p}}$ **then**
    return (" $\overline{p}$ is composite")
  end if
end for
return (" $\overline{p}$ is prime")

Counterexample: Carmichael Number

Definition (Carmichael Number)

A composite integer C that holds

$$a^{C-1} \equiv 1 \pmod{C}$$

for all integers a such that $\gcd(a, C) = 1$.

Example

- $N = 561 = 3 \cdot 11 \cdot 17$
- For all a such that $\gcd(a, 561) = 1$,

$$a^{560} \equiv 1 \pmod{561}$$

- Fermat Test says that Camichael numbers are prime.
- There are approximately only 10^6 Camichael numbers below 10^{15}.

(Probabilistic) Primality Test II: Miller-Rabin Test - Idea

Theorem

Let $\overline{p} - 1 = 2^r \cdot s$ where s is odd.If there exists an integer a such that

$$a^s \neq 1 \pmod{\overline{p}} \text{ and } a^{s \cdot 2^j} \neq \overline{p} - 1 \pmod{\overline{p}}$$

for all $j \in \{0, 1, \ldots, r-1\}$, then $\overline{p}$ is composite. Otherwise, it is probably a prime.

Example

- $\overline{p} = 561 = 3 \cdot 11 \cdot 17$
- $\overline{p} - 1 = 560 = 2^4 \cdot 35$

$$\begin{aligned} 5^{35} &= 23 \pmod{561} \\ 5^{35 \cdot 2} &= 529 \pmod{561} \\ 5^{35 \cdot 2^2} &= 529^2 = 463 \pmod{561} \\ 5^{35 \cdot 2^3} &= 463^2 = 67 \pmod{561} \end{aligned}$$

$\Rightarrow$ 561 is composite!

Description of Miller-Rabin Primality Test

Algorithm 6 Miller-Rabin Primality Test

Input: candidate $\overline{p}$ with $\overline{p} - 1 = 2^r s$ for odd s and security parameter λ
Output: " $\overline{p}$ is prime" or " $\overline{p}$ is composite"

for i from 1 to λ **do**
    select a random element a from $\{2, 3, \ldots, \overline{p} - 2\}$
    $z \leftarrow a^s \pmod{\overline{p}}$
    if $z \neq 1$ and $z \neq \overline{p} - 1$ **then**
        for $j = 1$ to $r - 1$ **do**
            $z \leftarrow z^2 \pmod{\overline{p}}$
            if $z = 1$ **then**
                return ("$\overline{p}$ is composite")
            end if
        end for
        if $z \neq \overline{p} - 1$ **then**
                return ("$\overline{p}$ is composite")
        end if
    end if
end for
return (" $\overline{p}$ is prime")

RSA Encryption

Overview of RSA Encryption

- Designed by Rivest, Shamir and Adleman in 1978
- The most popular and widely utilized public key encryption scheme
- Based on the hardness of factoring problems
- Encryption and decryption algorithms consist of modular exponentiations
- Deterministic encryption $\Rightarrow$ CANNOT achieve IND-XXX security
- Use variants of RSA encryption (e.g., RSA-OAEP) in practice
- Insecure against quantum attacks

RSA: Key Generation

Key Generation

1. Select two large primes p and q
2. Compute $N = pq$
3. Compute $\phi(N) = (p-1)(q-1)$
4. Select the public exponent $e \in \mathbb{Z}^*_{\phi(N)}$. (Note that $\gcd(e, \phi(N)) = 1$.)
5. Compute the private key d such that

   $$d \cdot e \equiv 1 \pmod{\phi(N)}$$

   (Use Extended Euclidean Algorithm!)
6. Output a pair of the public key and private key

   $$pk = (N, e), \quad sk = d$$

RSA: Encryption & Decryption

Encryption

Given the public key $pk = (N, e)$ and a plaintext $M \in \mathbb{Z}_N$, compute

$$C := \mathsf{RSA.Enc}(pk, M) = M^e \pmod{N}$$

and output C.

Decryption

Given the private key $sk = d$ and a ciphertext $C \in \mathbb{Z}_N$, compute

$$M' := \mathsf{RSA.Dec}(sk, C) = C^d \pmod{N}$$

and output M'.

RSA: Correctness

Recall: Correctness of Public Key Encryption

A public key encryption **PKE** is *correct* if the following holds: For any security parameter λ and any plaintext M,

$$\mathsf{Dec}(sk, \mathsf{Enc}(pk, M)) = M$$

where (pk, sk) is an output of $\mathsf{KeyGen}(\lambda)$.

Correctness of RSA

$$\begin{aligned} M' &= \mathsf{RSA.Dec}(sk, \mathsf{RSA.Enc}(pk, M)) \\ &= C^d = (M^e)^d = M^{ed} \\ &= M^{s\phi(N)+1} \pmod N \; (\because\; ed = s\phi(N) + 1 \text{ for some } s) \\ &= (M^{\phi(N)})^s \cdot M = M \; (\because \text{Euler Theorem}) \end{aligned}$$

RSA: Example

Key Generation

1. $p = 13,\ q = 17$
2. $N = 13 \cdot 17 = 221$
3. $\phi(N) = (13-1)(17-1) = 12 \cdot 16 = 192$
4. $e = 5$
5. $d = 77\ (\because 5 \cdot 77 = 385 \equiv 1 \pmod{192})$

Encryption

- $M = 3$

$$\begin{aligned} \Rightarrow C &= M^e = 3^5 \\ &= 243 \equiv 22 \pmod{221} \end{aligned}$$

Decryption

- $C = 22$

$$\begin{aligned} \Rightarrow M' &= C^d = 22^{77} \\ &= (22^7)^{11} = 61^{11} \\ &= 3 \pmod{221} \end{aligned}$$

RSA: Speed Up - Fast Encryption

Encryption

$$C := \mathrm{RSA.Enc}(pk, M) = M^e \pmod{N}$$

- Generally, the public key exponent e is approximately $(\log_2 N)$-bit.
  $\Rightarrow$ $1.5(\log_2 N)$ multiplications are required on average.
- Choose a very small e that has low Hamming weight, e.g., $e = 3, 17, 2^{16}+1$

e	e as binary string	Num of Mul
3	11_2	2
17	10001_2	5
$2^{16}+1$	$1\,0000\,0000\,0000\,0001_2$	17

- The private exponent d has almost full bit length, though e is (extremely) small.

RSA: Speed Up - Fast Decryption

- The private exponent d should be larger than $N^{0.292}$ by Coppersmith attack
  $\Rightarrow$ We cannot use a very small exponent d
- Use Chinese Remainder Theorem
  1. Compute
  $$\begin{aligned} d_p &= d \pmod{p-1} \\ d_q &= d \pmod{q-1} \end{aligned}$$
  2. Compute
  $$\begin{aligned} C_{d_p} &= C^{d_p} \pmod{p} \\ C_{d_q} &= C^{d_q} \pmod{q} \end{aligned}$$
  3. Compute
  $$C = C_{d_p} \cdot q \cdot M_q + C_{d_q} \cdot p \cdot M_p$$
  using Chinese Remainder Theorem.

RSA: Example of Decryption using CRT

Decryption in the Previous Example

- $N = 221,\ p = 13,\ q = 17,\ d = 77,\ C = 22$

$$\Rightarrow M' = C^d = 22^{77} = (22^7)^{11} = 61^{11} = 3 \pmod{221}$$

1. $\begin{cases} d_p = d \pmod{p-1} \\ d_q = d \pmod{q-1} \end{cases} \Rightarrow \begin{cases} 77 \pmod{12} = 5 \\ 77 \pmod{16} = 13 \end{cases}$

2. $\begin{cases} C_{d_p} = C^{d_p} \pmod{p} \\ C_{d_q} = C^{d_q} \pmod{q} \end{cases} \Rightarrow \begin{cases} 22^5 \pmod{13} = 9^5 \pmod{13} = 3 \\ 22^{13} \pmod{17} = 5^{13} \pmod{17} = 3 \end{cases}$

3. Using Chinese Remainder Theorem,

$$\begin{aligned} C &= C_{d_p} \cdot q \cdot M_q + C_{d_q} \cdot p \cdot M_p \\ &= 3 \cdot 13 \cdot 5 + 3 \cdot 17 \cdot 10 = 666 \\ &= 3 \pmod{221} \end{aligned}$$

$$(\because M_p = 17^{-1} = 10 \pmod{13}, M_p = 13^{-1} = 4 \pmod{17})$$

RSA: Security I

- Hard to recover the private key if factoring N is hard

Table: Main Results of RSA Factoring Challenge

RSA Number	Decimal Digits	Binary Digits	Digit	Factored by
RSA-100	100	330	April 1991	A. K. Lenstra
RSA-576	174	576	December 2003	J. Franke et al.
RSA-640	193	640	November 2005	J. Franke et al.
RSA-704	212	704	July 2012	S. Bai et al.
RSA-768	232	768	December 2009	T. Kleinjung et al.

The RSA Factoring Challenge is no longer active.

- NIST Recommendation

Security (bits)	Bit Length of RSA Modulus
80	1024
112	2048
128	3072
192	7680
256	15360

RSA: Security II

- The textbook RSA is deterministic encryption

  $\Rightarrow$ CANNOT achieve IND-CPA security

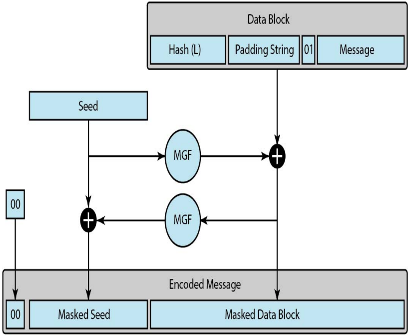
- The textbook RSA is homomorphic

$$C \cdot C' = M^e \cdot M'^e = (MM')^e$$

  $\Rightarrow$ CANNOT achieve IND-CCA2 security

Optimal Asymmetric Encryption Padding (OAEP)

1. Generate a string PS of length $k - |M| - 2|H| - 2$
2. Concatenate
$$DB = \text{Hash}(L) \| \text{PS} \| 0\text{x}01 \| M$$
3. Generate a random byte string *Seed* of length $|H|$
4. Compute $dbMask = \text{MGF}(Seed, k - |H| - 1)$
5. Compute $maskedDB = DB \oplus dbMask$
6. Compute $SeedMask = \text{MGF}(maskedDB, |H|)$
7. Compute $maskedSeed = Seed \oplus SeedMask$
8. Concatenate
$$EM = 0\text{x}00 \| maskedSeed \| maskedDB$$



Picture from https://commons.wikimedia.org/wiki/File:EME-OAEP.jpg

Discrete Logarithm Problem

Group I

Definition (Group)

A set $\mathbb{G}$ with a binary operator $\circ$ is a *group* if it satisfies the following conditions:

1. The operation $\circ$ is *closed*, i.e., for any $a, b \in \mathbb{G}$, $a \circ b \in \mathbb{G}$.
2. The operation $\circ$ is *associative*, i.e., for any $a, b, c \in \mathbb{G}$, $(a \circ b) \circ c = a \circ (b \circ c)$.
3. There exists an element $id \in \mathbb{G}$ such that $a \circ id = id \circ a = a$ for all $a \in \mathbb{G}$.
4. For any element $a \in \mathbb{G}$, there exists an element $a^{-1} \in \mathbb{G}$ such that $a \circ a^{-1} = a^{-1} \circ a = id$.

We say that a group $\mathbb{G}$ with a binary operator $\circ$ is *abelian (commutative)* if it additionally holds that

$$a \circ b = b \circ a$$

for any $a, b \in \mathbb{G}$.

Definition (Subgroup)

A group $(\mathbb{H}, \circ)$ is a *subgroup* of $(\mathbb{G}, \circ)$ if $\mathbb{H}$ is a subset of $\mathbb{G}$ and a group itself.

Group II

Example

- $(\mathbb{Z}, +)$, $(\mathbb{Q}, +)$, $(\mathbb{R}, +)$, $(\mathbb{C}, +)$: abelian groups
  $\Rightarrow$ $(\mathbb{Z}, +)$ is a subgroup of $(\mathbb{Q}, +)$
- $(\mathbb{Q} \setminus \{0\}, \times)$, $(\mathbb{R} \setminus \{0\}, \times)$, $(\mathbb{C} \setminus \{0\}, \times)$: abelian groups
- $(\mathbb{Z} \setminus \{0\}, \times)$: not a group ($\because$ 3^{-1}?)
- (The set of invertible 2×2 matrices, $\times$): a group, but not an abelian group ($\because$ $AB \neq BA$)

Definition (Finite Group)

A group $(\mathbb{G}, \circ)$ is *finite* if it has a finite number of elements. ($|\mathbb{G}|$ denotes the cardinality of $\mathbb{G}$.)

Example

- $(\mathbb{Z}_n, +)$: an abelian group for positive integer n ($|\mathbb{Z}_n| = n$)
- $(\mathbb{Z}_n^*, \times)$: an abelian group for positive integer n ($|\mathbb{Z}_n^*| = \phi(n)$)

Order of an Element

Definition (Order of an Element)

The order of an element a of a group $(\mathbb{G}, \circ)$, denoted by $ord(a)$, is the smallest positive integer k such that

$$\underbrace{a \circ a \circ \cdots \circ a}_{k \text{ times}} = 1$$

where 1 is the identity element of $\mathbb{G}$.

Example

- The order of 3 in $(\mathbb{Z}_{11}^*, \times)$
  - $3^1 = 3$
  - $3^2 = 9$
  - $3^3 = 27 = 5 \pmod{11}$
  - $3^4 = 5 \cdot 3 = 4 \pmod{11}$
  - $3^5 = 4 \cdot 3 = 1 \pmod{11}$

$\Rightarrow ord(3) = 5$ in $(\mathbb{Z}_{11}^*, \times)$

Cyclic Group

Definition (Cyclic Group)

A group $\mathbb{G}$ which contains an element g with the maximum order $ord(g) = |\mathbb{G}|$ is said to be *cyclic*. We call g a *generator* or a *primitive element*.

Example

- $\mathbb{Z}_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$
- For $a = 2$,

$$\begin{array}{ll} a = 2 & a^2 = 4 \\ a^3 = 8 & a^4 = 16 \bmod 11 = 5 \\ a^5 = 5 \cdot 2 = 10 \bmod 11 & a^6 = 10 \cdot 2 = 9 \bmod 11 \\ a^7 = 9 \cdot 2 = 7 \bmod 11 & a^8 = 7 \cdot 2 = 3 \bmod 11 \\ a^9 = 3 \cdot 2 = 6 \bmod 11 & a^{10} = 6 \cdot 2 = 1 \bmod 11 \end{array}$$

  $\Rightarrow ord(2) = 10 = |\mathbb{Z}_{11}^*|$ & $\mathbb{Z}_{11}^*$ is a cyclic group.
- 3 is not a generator in $\mathbb{Z}_{11}^*$.
- $\langle 3 \rangle = \{3, 9, 5, 4, 1\}$ is a subgroup of $\mathbb{Z}_{11}^*$ and 3 is a generator $\mathbb{G}$ of $\langle 3 \rangle$.

Discrete Logarithm Problem

Discrete Logarithm Problem (DLP)

Given a group $(\mathbb{G}, \times)$, a generator g of $\mathbb{G}$, and an element $A \in \mathbb{G}$, find a such that $A = g^a$ in $(\mathbb{G}, \times)$.

- Easy over the real numbers (Compute $\log_g A$)
- Difficult over the discrete world
  - Given $\mathbb{Z}_{31}^*$, $g = 3$ and $A = 20$,
    $\Rightarrow 3 \bmod 31 = 31$, $3^2 \bmod 31 = 9$, $\ldots$, $3^7 \bmod 31 = 17$, $3^8 \bmod 31 = 20$
  - If a prime p is sufficiently large, the DLP over $\mathbb{Z}_p^*$ is hard to solve.

Algorithms for solving DLPs

- Solving DLPs defined over a subgroup $\mathbb{G}$ of $\mathbb{Z}_p^*$ where q is the order of $\mathbb{G}$
  - If p is not large, the best attack is (General/Special) Number Field Sieve.
  - If q is not large, the best attack is Pollard rho algorithm.
- Parameter sizes: NIST recommendation (2016)

Security	Discrete Logarithm	
	Key Size (q)	Modulus Size (p)
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

(Unit: bits)

Diffie-Hellman Key Exchange

Key Exchange

- Alice and Bob want to generate a shared secret key using data exchange through a public channel.

- Except Alice and Bob, NO one should get any information about the generated secret key.

Diffie-Hellman Key Exchange

- Proposed by W. Diffie and M. Hellman in 1976
  - First public key cryptosystem (W. Diffie and M. Hellman, “New directions in cryptography,” IEEE Transactions on Information Theory, 22(6): pp.644-654)
  - 2015 Turing Award (a.k.a. Nobel Prize of Computing)
- Pre-shared parameters: a prime p, a primitive element g in $\mathbb{Z}_p^*$

Alice

1. Generate a private key a
2. Compute $A = g^a \pmod p$
3. Send A to Bob
4. Compute $K_{AB} = B^a \pmod p$

Bob

1. Generate a private key b
2. Compute $B = g^b \pmod p$
3. Send B to Alice
4. Compute $K_{BA} = A^b \pmod p$

$$K_{AB} = B^a = (g^b)^a = g^{ba} = g^{ab} = (g^a)^b = A^b = K_{BA}$$

Example: $p = 31$, $g = 3$

Alice

1. Generate a private key $a = 6$
2. Compute $A = g^a \bmod p$

$$\begin{aligned} A &= 3^6 \bmod 31 \\ &= 729 \bmod 31 = 16 \end{aligned}$$

3. Send $A = 16$ to Bob
4. Compute $K_{AB} = B^a \bmod p$

$$\begin{aligned} K_{AB} &= 20^6 \bmod 31 \\ &= 64,000,000 \bmod 31 \end{aligned}$$

Bob

1. Generate a private key $b = 8$
2. Compute $B = g^b \bmod p$

$$\begin{aligned} B &= 3^8 \bmod 31 \\ &= 6,561 \bmod 31 = 20 \end{aligned}$$

3. Send $B = 20$ to Alice
4. Compute $K_{BA} = A^b \bmod p$

$$\begin{aligned} K_{BA} &= 16^8 \bmod 31 \\ &= 4,294,967,296 \bmod 31 \end{aligned}$$

$$K_{AB} = 64,000,000 \bmod 31 = 4 = 4,294,967,296 \bmod 31 = K_{AB}$$

Security of Diffie-Hellman Key Exchange

- Eve wants to know the private key between Alice and Bob, $K_{AB} = K_{BA}$
- Eve has (p, g), $(A = g^a \bmod p, B = g^b \bmod p)$
- If Eve has Alice's private key a or Bob's private key b

  $\Rightarrow$ She can compute $K_{AB} = B^a = A^b = K_{BA}$, as Alice and Bob

  $\Rightarrow$ DLP should be infeasible

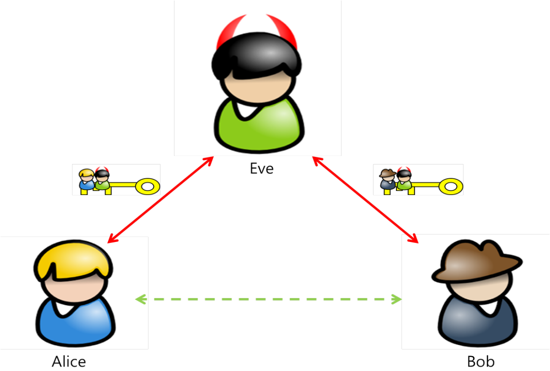
Computational Diffie-Hellman (CDH) Problem

Given (g, g^a, g^b), compute g^{ab}.

- Informally, the hardness of the CDH problem is equivalent to the hardness of DLP if the order of the underlying group is prime.

Man-in-the-Middle Attack: Attack Scenario

- If Eve intercepts Alice and Bob's public data and acts like Bob and Alice to Alice and Bob, respectively,...?



Man-in-the-Middle Attack: Description & Prevention

- Description
  1. Eve generates a private key e and computes $E = g^e \bmod p$.
  2. When Alice and Bob transmit A and B to each other, respectively, Eve intercepts and sends E to both.
  3. Then, Alice and Bob finally have

$$\begin{aligned} K_{AE} &= E^a = g^{ea} \bmod p \text{ and} \\ K_{BE} &= E^b = g^{eb} \bmod p, \end{aligned}$$

     respectively.

- Later, once Alice sends $C = \mathrm{Enc}(K_{AE}, M)$ to Bob, Bob cannot decrypt it whereas Eve can obtain M by decrypting it using $K_{EA} = A^e = (g^a)^e \bmod p$.

- Prevention: Use authentication
  - e.g., send public keys together with signatures when Alice and Bob send their public keys to each other.

ElGamal Encryption

Overview of ElGamal Encryption

- Proposed by T. El Gamal in 1985 (T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Transactions on Information Theory, 31(4): pp.469-472)

- Probabilistic encryption

- IND-CPA secure if the decisional Diffie-Hellman problem is infeasible

Decisional Diffie-Hellman (DDH) Problem

Given (g, g^a, g^b) and X, distinguish if $X = g^{ab}$ or g^c for a randomly chosen c.

- Believe that the DDH problem is hard if the CDH problem with the same instance is hard.

- The original ElGamal encryption is multiplicative homomorphic

  $\Rightarrow \mathsf{Enc}(M) \cdot \mathsf{Enc}(M') = \mathsf{Enc}(M \cdot M')$

- Can achieve IND-CCA2 security using the generic transformation

Description of ElGamal Encryption

Key Generation

1. Generate a prime p
2. Choose a generator g of a subgroup $\mathbb{G}$ of $\mathbb{Z}_p^*$ whose order is q
3. Select a random element x in $\mathbb{Z}_q$ and compute $X = g^x$
4. Output a public key $pk = (p, q, g, X)$ and a secret key $sk = x$

Encryption

To encrypt a message M with a public key $pk = (p, q, g, X)$

1. Select a random element $r \in \mathbb{Z}_q$
2. Compute $C_1 = g^r$ and $C_2 = M \cdot X^r$ and output $CT = (C_1, C_2)$

Decryption

Given the secret key $sk = x$ and a ciphertext $CT = (C_1, C_2)$, compute and output

$$C_2/(C_1)^x (= M \cdot X^r/(g^r)^x = M).$$

References

[MOV96] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996. (Chapter 2)

[PP10] C. Paar and J. Pelzl, Understanding Cryptography, Springer, 2010

[SB15] W. Stallings and L. Brown, Computer Security: Principles and Practice, 3rd edition, Pearson Prentice Hall, 2015

[Sho08] V. Shoup, A Computational Introduction to Number Theory and Algebra, 2nd ed., Cambridge University Press, 2008. (Chapter 4)