

# 1. Introduction

- Basic encryption process:
  - A has a message (**plaintext**) which is **encrypted** using an **encryption key** to produce the **ciphertext**, which is sent to *B*.
  - B uses a **decryption key** (which depends on the encryption key) to **decrypt** the ciphertext and recover the original plaintext.
  - It should be computationally infeasible to determine the plaintext without knowing the decryption key.
- **Caesar cipher**:
  - Add a constant to each letter in the plaintext to produce the ciphertext:

$$\text{ciphertext letter} = \text{plaintext letter} + k \pmod{26}$$

- To decrypt,
$$\text{plaintext letter} = \text{ciphertext letter} - k \pmod{26}$$
  - The key is  $k \pmod{26}$ .
- Cryptosystem objectives:
  - **Secrecy**: the intercepted message should be not able to be decrypted
  - **Integrity**: a message should not allowed to be altered without the receiver knowing
  - **Authenticity**: the receiver should be certain of the identity of the sender
  - **Non-repudiation**: the sender should not be able to claim they sent a message; the receiver should be able to prove they did.
- **Kerckhoff's principle**: a cryptographic system should be secure even if the details of the system are known to an attacker.
- Types of attack:
  - **Ciphertext-only**: the plaintext is deduced from the ciphertext.
  - **Known-plaintext**: intercepted ciphertext and associated stolen plaintext are used to determine the key.
  - **Chosen-plaintext**: an attacker tricks a sender into encrypting various chosen plaintexts and observes the ciphertext, then uses this information to determine the key.
  - **Chosen-ciphertext**: an attacker tricks the receiver into decrypting various chosen ciphertexts and observes the resulting plaintext, then uses this information to determine the key.

## 2. Symmetric key ciphers

- **Converting letters to numbers**: treat letters as integers modulo 26, with  $A = 1, Z = 0 \equiv 26 \pmod{26}$ . Treat a string of text as a vector of integers modulo 26.
- **Symmetric key cipher**: one in which encryption and decryption keys are equal.
- **Key size**:  $\log_2(\text{number of possible keys})$ .

- **Substitution cipher:** key is permutation of  $\{a, \dots, z\}$ . Key size is  $\log_2(26!)$ . It is vulnerable to plaintext attacks and ciphertext-only attacks, since different letters (and letter pairs) occur with different frequencies in English.
- **Stirling's formula:**

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

- **One-time pad:** key is uniformly, independently random sequence of integers mod 26,  $(k_1, k_2, \dots)$ , it is known to the sender and receiver. If message is  $(m_1, m_2, \dots, m_r)$  then ciphertext is  $(c_1, c_2, \dots, c_r) = (k_1 + m_1, k_2 + m_2, \dots, k_r + m_r)$ . To decrypt the ciphertext,  $m_i = c_i - k_i$ . Once  $(k_1, \dots, k_r)$  have been used, they must never be used again.
  - One-time pad is information-theoretically secure against ciphertext-only attack:  $\mathbb{P}(M = m \mid C = c) = \mathbb{P}(M = m)$ .
  - Disadvantage is keys must never be reused, so must be as long as message.
  - Keys must be truly random.
- **Chinese remainder theorem:** let  $m, n \in \mathbb{N}$  coprime,  $a, b \in \mathbb{Z}$ . Then exists unique solution  $x \bmod mn$  to the congruences

$$\begin{aligned} x &\equiv a \pmod{m} \\ x &\equiv b \pmod{n} \end{aligned}$$

- **Block cipher:** group characters in plaintext into blocks of  $n$  (the **block length**) and encrypt each block with a key. So plaintext  $p = (p_1, p_2, \dots)$  is divided into blocks  $P_1, P_2, \dots$  where  $P_1 = (p_1, \dots, p_n)$ ,  $P_2 = (p_{n+1}, \dots, p_{2n})$ . Then ciphertext blocks are given by  $C_i = f(\text{key}, P_i)$  for some encryption function  $f$ .
- **Hill cipher:**
  - Plaintext divided into blocks  $P_1, \dots, P_r$  of length  $n$ .
  - Each block represented as vector  $P_i \in (\mathbb{Z}/26\mathbb{Z})^n$
  - Key is invertible  $n \times n$  matrix  $M$  with elements in  $\mathbb{Z}/26\mathbb{Z}$ .
  - Ciphertext for block  $P_i$  is

$$C_i = MP_i$$

It can be decrypted with  $P_i = M^{-1}C_i$ .

- Let  $P = (P_1, \dots, P_r)$ ,  $C = (C_1, \dots, C_r)$ , then  $C = MP$ .
- **Confusion:** each character of ciphertext depends on many characters of key.
- **Diffusion:** each character of ciphertext depends on many characters of plaintext. Ideal diffusion changes a proportion of  $(S - 1)/S$  of the characters of the ciphertext, where  $S$  is the number of possible symbols.
- For Hill cipher,  $i$ th character of ciphertext depends on  $i$ th row of key - this is medium confusion. If  $j$ th character of plaintext changes and  $M_{ij} \neq 0$  then  $i$ th character of ciphertext changes.  $M_{ij}$  is non-zero with probability roughly  $25/26$  so good diffusion.
- Hill cipher is susceptible to known plaintext attack:

- If  $P = (P_1, \dots, P_n)$  are  $n$  blocks of plaintext with length  $n$  such that  $P$  is invertible and we know  $P$  and the corresponding  $C$ , then we can recover  $M$ , since  $C = MP \implies M = CP^{-1}$ .
- If enough blocks of ciphertext are intercepted, it is very likely that  $n$  of them will produce an invertible matrix  $P$ .

### 3. Public key cryptography and the RSA algorithm

- **Euler  $\varphi$  function:**

$$\varphi : \mathbb{N} \rightarrow \mathbb{N}, \varphi(n) = |\{1 \leq a \leq n : \gcd(a, n) = 1\}| = |(\mathbb{Z}/n\mathbb{Z})^\times|$$

- $\varphi(p^r) = p^r - p^{r-1}$ ,  $\varphi(mn) = \varphi(m)\varphi(n)$  for  $\gcd(m, n) = 1$ .
- **Euler's theorem:** if  $\gcd(a, n) = 1$ ,  $a^{\varphi(n)} \equiv 1 \pmod{n}$ .
- **Public key cryptography:**
  - Create two keys,  $k_D$  and  $k_E$ .  $k_E$  is public,  $k_D$  is private.
  - Plaintext  $m$  is encrypted as  $c = f(m, k_E)$ .
  - Ciphertext decrypted by  $m = g(c, k_D)$ .
- **RSA:**
  - $k_E$  is pair  $(n, e)$  where  $n = pq$  is product of two distinct primes and  $e \in \mathbb{Z}$  is coprime to  $\varphi(n)$ .
  - $k_D$  is integer  $d$  such that  $de \equiv 1 \pmod{\varphi(n)}$ .
  - $m$  is an integer modulo  $n$ ,  $m$  and  $n$  are coprime.
  - Encryption:  $c = m^e \pmod{n}$ .
  - Decryption:  $m = c^d \pmod{n}$ .
- **RSA problem:** given  $n = pq$  a product of two unknown primes,  $e$  and  $m^e \pmod{n}$ , recover  $m$ . If  $n$  can be factored, the RSA is solved.
- It is recommended that  $n$  have at least 2048 bits. A typical choice of  $e$  is  $2^{16} + 1$ .
- **Attacks on RSA:**
  - If you can factor  $n$ , you can compute  $d$ , so can break RSA (as then you know  $\varphi(n)$  so can compute  $e^{-1} \pmod{\varphi(n)}$ ).
  - If  $\varphi(n)$  is known, then we have  $pq = n$  and  $(p-1)(q-1) = \varphi(n)$  so  $p + q = n - \varphi(n) + 1$ . Hence  $p$  and  $q$  are roots of  $x^2 - (n - \varphi(n) + 1)x + n = 0$ .
  - **Known  $d$ :** we have  $de - 1$  is multiple of  $\varphi(n)$ . Look for a factor  $A$  of  $de - 1$  such that  $(p-1) \mid A$ ,  $(q-1) \nmid A$ . Then try  $x^A - 1$  for random  $x$ , this satisfies  $x^A - 1$  is divisible by  $p$ , hence  $\gcd(x^A - 1, n) = p$ .
- **RSA signatures:**
  - Public key is  $(n, e)$  and private key is  $d$ .
  - When sending a message  $m$ , message is **signed** by also sending  $s = m^d \pmod{n}$ .
  - $(m, s)$  is received, **verified** by checking if  $m = s^e \pmod{n}$ .
  - Forging a signature on a message  $m$  would require finding  $s$  with  $m = s^e \pmod{n}$ . This is the RSA problem.
  - However, choosing signature  $s$  first then taking  $m = s^e \pmod{n}$ .
  - To solve this,  $(m, s)$  is sent where  $s = h(m)^d$ ,  $h$  is **hash function**. Then the message receiver verifies  $h(m) = s^e \pmod{n}$ .

- Now, for a signature to be forged, an attacker would have to find  $m$  with  $h(m) = s^e \bmod n$ .
- **Hash function** is function  $h : \{\text{messages}\} \rightarrow \mathcal{H}$  that:
  - Can be computed efficiently
  - Is preimage-resistant: can't quickly find  $m$  with given  $h(m)$ .
  - Is collision-resistant: can't quickly find  $m, m'$  with  $h(m) = h(m')$ .

Example is SHA-256.

- **Theorem:** it is no easier to find  $\varphi(n)$  than to factorise  $n$ .
- **Theorem:** it is no easier to find  $d$  than to factor  $n$ .
- **Miller-Rabin algorithm:**
  1. Choose random  $x \bmod n$ .
  2. Let  $n - 1 = 2^r s$ ,  $y = x^s$ .
  3. Compute  $y, y^2, \dots, y^{2^r} \bmod n$ .
  4. If 1 isn't in this list,  $n$  is **composite** (with witness  $x$ ).
  5. If 1 is in list preceded by number other than  $\pm 1$ ,  $n$  is **composite** (with witness  $a$ ).
  6. Other,  $n$  is **possible prime** (to base  $x$ ).

### 3.1. Factorisation

- **Trial division algorithm:** for  $p = 2, 3, 5, \dots$  test whether  $p \mid n$ .
- **Fermat's method:**
  - Let  $a = \lceil \sqrt{n} \rceil$ . Compute  $a^2 \bmod n$ ,  $(a + 1)^2 \bmod n$  until a square  $x^2 \equiv (a + i)^2 \bmod n$  appears. Then compute  $\gcd(a + i - x, n)$ .
  - Works well under special conditions on the factors: if  $|p - q| \leq 2\sqrt{2}\sqrt[4]{n}$  then Fermat's method takes one step:  $x = \lceil \sqrt{n} \rceil$  works.
- An integer is  $B$ -smooth if all its prime factors are  $\leq B$ .
- **Quadratic sieve:**
  - Choose  $B$  and let  $m$  be number of primes  $\leq B$ .
  - Look at integers  $x = \lceil \sqrt{n} \rceil + k$ ,  $k = 1, 2, \dots$  and check whether  $y = x^2 - n$  is  $B$ -smooth.
  - Once  $y_1 = x_1^2 - n, \dots, y_t = x_t^2 - n$  are all  $B$ -smooth with  $t > m$ , find some product of them that is a square.
  - Deduce a congruence between the squares.
- **Other factorisation algorithms:**
  - Pollard's  $\rho$  algorithm.
  - Pollard's  $p - 1$  algorithm.
  - Lenstra's algorithm using elliptic curves.
  - General number field sieve
  - Shor's algorithm:  $\ln(N)^2 \ln(\ln(N))$ .

### 3.2. Primitive roots

- Let  $p$  prime,  $g \in \mathbb{F}_p^\times$ . **Order** of  $g$  is smallest  $a \in \mathbb{N}_0$  such that  $g^a = 1$ .  $g$  is **primitive root** if its order is  $p - 1$ .

- Let  $p$  prime,  $g \in \mathbb{F}_p^\times$  primitive root. If  $x \in \mathbb{F}_p^\times$  then  $x = g^L$  for some  $0 \leq L < p - 1$ . Then  $L$  is **discrete logarithm** of  $x$  to base  $g$ . Write  $L = L_g(x)$ . It satisfies:
  - $g^{L_g(x)} \equiv x \pmod{p}$  and  $g^a \equiv x \pmod{p} \iff a \equiv L_g(x) \pmod{p-1}$ .
  - $L_g(1) = 0, L_g(g) = 1$ .
  - $L_g(xy) \equiv L_g(x) + L_g(y) \pmod{p-1}$ .
  - $h$  is primitive root mod  $p$  iff  $L_g(h)$  coprime to  $p-1$ . So number of primitive roots mod  $p$  is  $\varphi(p-1)$ .
- **Discrete logarithm problem:** given  $p, g, x$ , compute  $L_g(x)$ .
- **Diffie-Hellman key exchange:**
  - Two parties agree on prime  $p$  and primitive root  $g \bmod p$ .
  - Alice chooses secret  $\alpha \bmod(p-1)$  and sends  $g^\alpha \bmod p$  to Bob.
  - Bob chooses secret  $\beta \bmod(p-1)$  and sends  $g^\beta \bmod p$  to Alice.
  - Alice and Bob both compute  $\kappa = g^{\alpha\beta} = (g^\alpha)^\beta = (g^\beta)^\alpha \bmod p$ .
- **Diffie-Hellman problem:** given  $p, g, g^\alpha, g^\beta$ , compute  $g^{\alpha\beta}$ .
- If discrete logarithm problem can be solved, so can Diffie-Hellman problem (since could compute  $\alpha = L_g(g^\alpha)$  or  $\beta = L_g(g^\beta)$ ).
- **Elgamal public key encryption:**
  - Alice chooses prime  $p$ , primitive root  $g$ , private key  $\alpha \bmod(p-1)$ .
  - Her public key is  $y = g^\alpha$ .
  - Bob chooses random  $k \bmod(p-1)$
  - To send message  $m$  (integer mod  $p$ ), he sends the pair  $(r, m') = (g^k, my^k)$ .
  - To decrypt the message, Alice computes  $r^\alpha = g^{\alpha k} = y^k$  and then  $m = m'y^{-k} = m'r^{-\alpha}$ .
  - If Diffie-Hellman problem is hard, then Elgamal encryption is secure against known plaintext attack.
  - Key  $k$  must be random and different each time.
- **Decision Diffie-Hellman problem:** given  $g^a, g^b, c$  in  $\mathbb{F}_p^\times$ , decide whether  $c = g^{ab}$ .
  - This problem is not always hard, as can tell if  $g^{ab}$  is square or not. Can fix this by taking  $g$  to have large prime order  $q \mid (p-1)$ .  $p = 2q + 1$  is a good choice.
- **Elgamal signatures:**
  - Public key is  $(p, g)$ ,  $y = g^\alpha$  for private key  $\alpha$ .
  - **Valid Elgamal signature** on  $m$  is pair  $(r, s)$ ,  $r \geq 0, s < p-1$  such that
 
$$y^r r^s = g^m \pmod{p}$$
    - Alice computes  $r = g^k, k \in (\mathbb{Z}/(p-1))^\times$  random.
    - Then  $g^{\alpha r} g^{ks} \equiv g^m \pmod{p}$  so  $\alpha r + ks \equiv m \pmod{p-1}$  so  $s = k^{-1}(m - \alpha r) \pmod{p-1}$ .
- **Elgamal signature problem:** given  $p, g, y, m$ , find  $r, s$  such that  $y^r r^s = m$ .
- **Discrete logarithm problem:** given prime  $p$ , primitive root  $g \bmod p$ ,  $x \in \mathbb{F}_p^\times$ , calculate  $L_g(x)$ .

- **Baby-step giant-step algorithm** for solving DLP:
  - Let  $N = \lceil \sqrt{p-1} \rceil$ .
  - Baby-steps: compute  $g^j \bmod p$  for  $0 \leq j < N$ .
  - Giant-steps: compute  $xg^{-Nk} \bmod p$  for  $0 \leq k < N$ .
  - Look for a match between baby-steps and giant-steps lists:  
 $g^j = xg^{-Nk} \implies x = g^{j+Nk}$ .
  - Always works since if  $x = g^L$  for  $0 \leq L < p-1 \leq N^2$ , so  $L$  can be written as  $j + Nk$  with  $j, k \in \{0, \dots, N-1\}$ .