# Cryptography and security

Pierre Colson

## Contents



---

**Markdown** version on *github*
Compiled using *pandoc*

## General

- $b \in \mathbb{Z}_p^*$ has a quare root if and only if $b^{\frac{p-1}{2}} \mod p = 1$

## Diffie Helman

- We check that $X$ and $Y$ are in $\langle g \rangle$
- Use a KDF to fix bad distribution of $g^{xy}$
- We wheck the lower order $X \neq 1$, $X^2 \neq 1$
- If $n = pq$ then $\mathbb{Z}_n$ ring is isomorphic to $\mathbb{Z}_p \times \mathbb{Z}_q$ and $\mathbb{Z}_n^*$ ring is isomorphic to $\mathbb{Z}_p^* \times \mathbb{Z}_q^*$

## RSA

- Square and multiply algorithm to compute $x^e$ or $x^d$
- Primality test : Verify that a number os prime
- To check if a number is coprime is another one use euclid algorithm
- To compute the inverse of an elem use extended euclid algorithm
- $\varphi(p^\alpha) = (p-1)p^{\alpha-1}$
- We can compute square root of $n$ in $\mathcal{O}(\log n)^3$

## Elliptic Curve

- All finite fields have a cardinality of form $p^k$ where $p$ is a prime number This prime number $p$ is called the **characteristic** of the field.
- A **binary** field is a field with characteristic equal to 2
- Over a field $\mathbb{R}$, an elliptic curve with parameters $a$ and $b$ consists of a special point $\mathcal{O}$ called the *point at infinity* and the points $(x, y)$ which are the solutions of the equation $y^2 = x^3 + ax + by$

- Elliptic Curve over a **Prime Field**
  - The **discriminent** is $\Delta = -16(4a^3 + 27b^2)$
  - The curve is **non-singular** iff $\Delta \neq 0$
  - We define the **j-invariant** $j = 1728\frac{4a^3}{Aa^3+27b^2}$, two isomorphic curves have the same j-invariant
- Elliptic Curve over a **Binary Field**
  - **Ordinary** curves are defined by two fieds elements denoted $a_2$ and $a_6$

$$E_{a_2,a_6}(\mathbb{K}) = \{\mathcal{O}\} \cup \{(x, y) \in \mathbb{K}^2; y^2 + xy = x^3 + a_2x^2 + a_6\}$$

  - We define the **j-invariant** $j = \frac{1}{\Delta}$
- Simple factoring method : Pollard's (also called $p - 1$ algorithm)
- **Elliptic Curve Method** (ECM) is the best method to find $p$ when it is small
- **ECDH** key exchange protocol is tha variant of Diffie-Hellman protocol working over an elliptic curve group
  - We have two participant $U$ and $V$ using the same subgroup of order $n$ genrated by some point $G$ over an elliptic curve.
  - They both select their secret key $d_U, d_V \in \mathbb{Z}_n^*$
  - They compute their public key $Q_U = d_U.G$ and $Q_V = d_V.G$ which are point and exchange them.
  - Then, they both check that the received publoc key is actually a point of the curve which is generated by $G$, different from the point at infinity, and that its order is a factor of $n$.
  - They both compute the a point $P$, either by $P = d_U.Q_V$ or by $P = d_V.Q_U$
  - They take the first coordinate $x_p$ of $P$ and convert it into a byte string $Z$
  - Finally they compute $K = KDF(Z)$

# Symmetric Encryption

## Block cypher

- **Block cyphers** encrypt/decrypt data by *blocks* of fixed length (typically 64 or 128 bits)
- **DES** : Blocks of 64 bits with a key of 56 effective bits (actually the key has 64 bits but on bit per byte is used for the checksum)
  - Internally the 56 bits key is expended into a number of 16 48 bits subkeys
  - The encryption goes through 16 rounds each of which uses on subkey as a round key
  - The round follows the **Feistel Scheme** :
    - The block is split into two halves
    - The right half goes through a round function with the round key
    - The output of this round function is XORed to the left half
    - The two halves are then exchanged before the next round starts
    - In the last round the exchange of halves if omitted
    - The *round function* is invertible
    - The inverse transfrom is actually a another Feistel scheme with the round key in reverse order
  - They are many know attack against DES
- Since 56 bits for a secret key are considered as too short, people considered triple encryption. This is **tripe-DES** standard
  - There are two variant :
    - Triple DES with two keys : $K_1 = K_2$
    - Triple DES with three keys

$$3DES_{K_1,K_2,K_3}(X) = DES_{K_3}(DES^{-1}_{K_2}(DES_{K_1}(X)))$$

  - A block cypher should be secure against **key recovery** and **decrytion attack**
- **AES** (Advanced Encryption Standard) it encrypts blocks of 128 bits using keys of 128, 192, 256 bits.
  - It structure consists of a keylenght-dependent number of rounds (10, 12, or 14 rounds) in which a round key is used
  - In AES, a message block and a *round key* are represented as a $4 \times 4$ matrix
  - Each byte actually represents an element of $GF(Z^8)$ with reference polynomial $P(X) = X^8 + X^4 + X^3 + X + 1$ I.e, a bitstring $a_7 \dots a_0$ represents the polynomial $a = a_7X^7 + \cdots + a_1X + a_0$ and additons and multiplications are done modulo 2 and modulo $P(X)$

  - The addition in the field corresponds to the XOR of the bitstrings
  - To multiply $a$ by $0x02$ we just shift the byte $a$ by one bit to the left and XOR the $0x1b$ if there is a carry bit
  - To multiply $a$ by $0x03$ we can multiply by $0x01$ and by $0x02$ and add (XOR) the twos results
  - In AES we only need to multiply by $0x01$, $0x02$ and $0x03$
  - Each round consists of four types of successice transform
    * *AddRoundKey* which adds (XOR) the round key to the block
    * *SubBytes* which substities every byte $a$ by the byte $S(a)$, following a table $S$ (called the *S-box*)
    * *ShiftRows* which consists of a circular shift of every row of the block by a variable number of positions
    * *MixColumns* which consists of mutlipying all columns of the block to the left by a prefined matrix $M$
  - To decrypt we just have to invert all subroutine processes
- If we want to encrypt a message which consists of several blocks, we need to plug the block cipher into a **mode of operation**
  - **Electric Codebook** (ECB) mode consists of encrypting each blokc separatly, using the block cipher
    * This is however insecure for most of applications : indeed in the messages that applications want to encrypt, it is very likely that some blocks of data repeat
  - **Cipher Block chaining** (CBC) mode, each block of plaintext is XORed to the previous ciphertext block before being encrypted. The first plaintext in XORed to an initial vetor IV. There is three ways to use IV:
    * Use a constant, publicly know IV
    * Use a secret IV (so the secret key becomes (IV, K))
    * Use a fresh random IV for every messages $x$ and add it as a part of the ciphertext
  - The **Output Feedback** (OFB) mode uses an IV. It consists of defining the sequence $k_i = ENC_K(k_{i-1}), i = 2, \ldots$ and $k_1 = ENC_K(IV)$. It requires the IV to be unique, due to the properties of the one-time-pad, we then call the IV a nonce
  - The **Cipher Feedback** (CFB) mode is defined by $y_i = x_i \oplus ENC_K(y_{i-1}), i = 2$, and $y_1 = x_1 \oplus ENC_K(IV)$ The nonce IV options are the same as for OFB mode. The CFB works even if the last plaintext block is incomplete
  - The **Counter** (CTR) mode ises a none $t_i$ for every block. The encryption of $x_i$ is $y_i = x_i \oplus ENC_K(t_i)$. The nonce is based on a counter. The CTR mode works even if the last plaintext is incomplete

## Stream Ciphers