Crypto Formulas

Cryptography Algorithms - Complete Formulations

RSA (Rivest-Shamir-Adleman)

Type: Public Key Cryptography, Asymmetric Encryption

Key Generation:

1. Choose two large distinct prime numbers p and q
2. Compute modulus n = p × q
3. Compute Euler's totient function phi, φ(n) = (p-1)(q-1)
4. Choose public exponent e such that 1 < e < φ(n) and gcd(e, φ(n)) = 1
5. Compute private exponent d = e^(-1) mod φ(n)

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

Encryption: c = m^e mod n
Decryption: m = c^d mod n

Components:

- p, q: Large prime numbers
- n: Modulus (product of p and q)
- φ(n): Euler's totient function
- e: Public exponent (usually 65537)
- d: Private exponent
- m: Plaintext message (0 ≤ m < n)
- c: Ciphertext

DES (Data Encryption Standard) - 16 Rounds

Type: Symmetric Key Block Cipher

Key Generation:

1. Start with 64-bit key (56 bits + 8 parity bits)
2. Generate 16 subkeys (K_1 to K_{16}) of 48 bits each using:
   - Permuted Choice 1 (PC-1): 64-bit → 56-bit
   - Left circular shifts
   - Permuted Choice 2 (PC-2): 56-bit → 48-bit

Encryption:
Initial Permutation (IP) on 64-bit plaintext → (L_0, R_0)
For rounds i = 0 to 15:
$L_{i+1} = R_i$

$R_{i+1} = L_i \oplus F(R_i, K_i)$
Final Permutation (IP^{-1}) on (L_{16}, R_{16}) → 64-bit ciphertext

F-Function:
F(R, K):

1. Expansion: 32-bit R → 48-bit using expansion table E
2. XOR: Expanded_R ⊕ K (48 bits)
3. S-box substitution: 48-bit → 32-bit using 8 S-boxes (6-to-4 bits each)
4. Permutation: 32-bit permutation using P-table

Decryption: Same algorithm with reversed key order (K_{16} to K_1)

Components:

- Plaintext: 64-bit input block
- Ciphertext: 64-bit output block
- Key: 64-bit (56 effective bits + 8 parity)
- L_i, R_i: Left/Right 32-bit halves
- K_i: 48-bit round subkey
- F(): Round function
- S-boxes: Non-linear substitution tables

AES (Advanced Encryption Standard)

Type: Symmetric Key Block Cipher

Key Expansion:
Takes 128/192/256-bit key → generates round keys

Encryption:
State = Plaintext (16 bytes arranged in 4×4 matrix)
AddRoundKey(state, w[0,3])

For round = 1 to N_r-1:
SubBytes(state) # S-box substitution
ShiftRows(state) # Row shifting
MixColumns(state) # Column mixing
AddRoundKey(state, w[4×round, 4×round+3])

Final round:
SubBytes(state)
ShiftRows(state)
AddRoundKey(state, w[4×N_r, 4×N_r+3])

Decryption: Inverse operations in reverse order

Components:

- State: 4×4 byte matrix (16 bytes = 128 bits)
- N_r: Number of rounds (10 for 128-bit, 12 for 192-bit, 14 for 256-bit key)
- SubBytes: Non-linear byte substitution using S-box
- ShiftRows: Cyclic shift of rows

- MixColumns: Matrix multiplication in $GF(2^8)$
- AddRoundKey: XOR with round key
- Key Schedule: Generates round keys from main key

Diffie-Hellman Key Exchange

Type: Key Exchange Protocol

Setup:
Public parameters: Large prime p, Generator g of multiplicative group $\mathbb{Z}_p$*

Key Exchange Process:
User A: User B:
Choose private key a Choose private key b
(1 < a < p-1) (1 < b < p-1)

Compute public key: Compute public key:
A = g^a mod p B = g^b mod p

Exchange public keys A and B

Compute shared secret: Compute shared secret:
K = B^a mod p K = A^b mod p

Result: Both compute same K = g^(ab) mod p

Components:

- p: Large prime number
- g: Generator (primitive root modulo p)
- a, b: Private keys (random integers)
- A, B: Public keys
- K: Shared secret key

Stream Cipher

Type: Symmetric Key Stream Cipher

General Structure:
Keystream Generator: Key + IV → Keystream (k_1, k_2, k_3, ...)
Encryption: $c_i = m_i \oplus k_i$ for each bit/byte
Decryption: $m_i = c_i \oplus k_i$ for each bit/byte

Components:

- Key: Secret symmetric key
- IV: Initialization Vector (for some stream ciphers)
- Keystream: Sequence of pseudo-random bits/bytes
- m_i: Plaintext bit/byte at position i
- c_i: Ciphertext bit/byte at position i
- $\oplus$: XOR operation

- S[]: Internal state array (in RC4)