NETWORK SECURITY PRACTICES – ATTACK AND DEFENSE

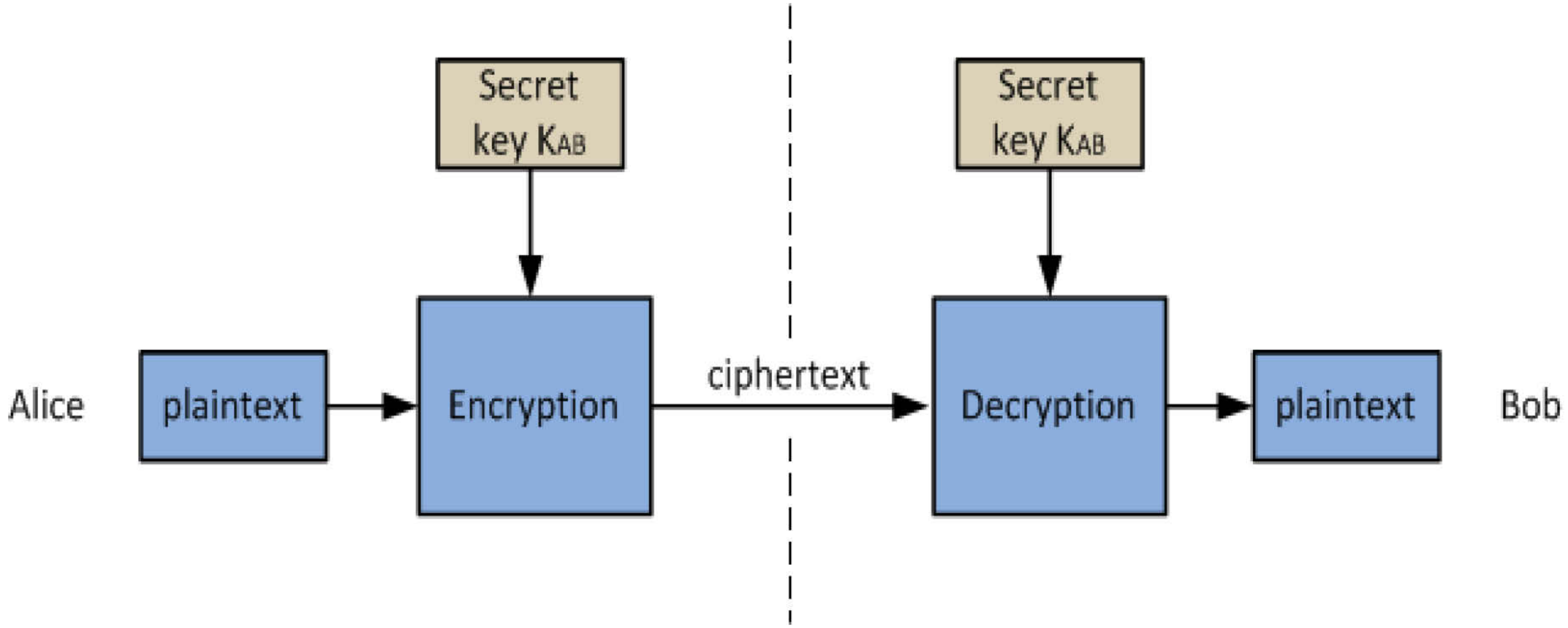
Cryptographic Primitives

Cryptography

- Cryptosystem components
  - Plaintext
  - Ciphertext
  - Encryption
  - Decryption
  - Key

Symmetric Key Cryptosystem

- Use the same secret key
- DES, RC4, AES...etc
- Block / Stream



What is cipher text ?

- Has to conceal information of plaintext
  - => confidentiality
- Indistinguishability under chosen-plaintext attack (IND-CPA)
  - An adversary presents two plaintext M_0 and M_1
  - The challenger picks either M_0 and M_1 at random and encrypt it into ciphertext C
  - The adversary performs a polynomial time algorithm (including encryption through an oracle) on C, M_0, and M_1 to decide if
    - C = Encrypt(M_0)
  - If any such adversary can only decide with probability ½ + e(k) , the cryptosystem is said to be IND-CPA

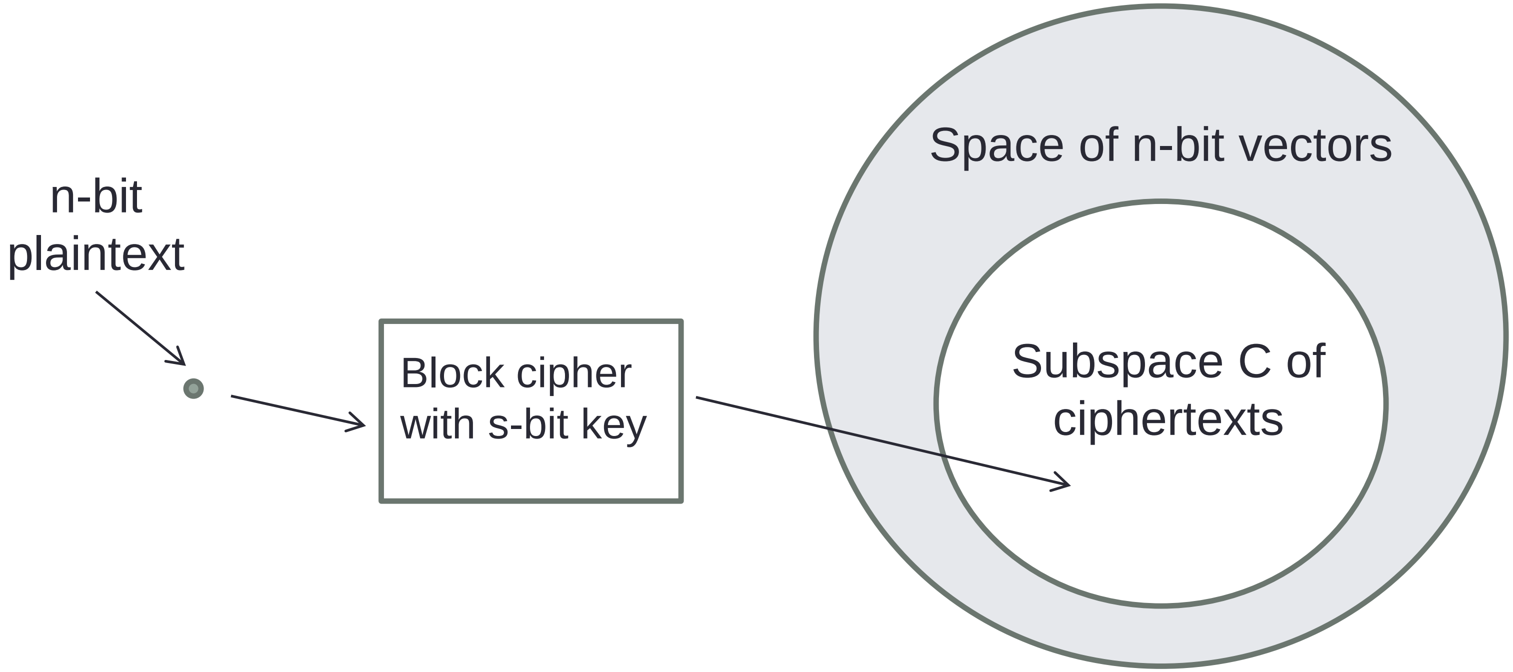
What is cipher text?

- Indistinguishability under chosen ciphertext attack (IND-CCA1)
  - Similar to IND-CPA, but the adversary is given a decryption oracle
  - Advasary can query the decryption oracle until challenger sends ciphertext *C*
- Indistinguishability under adaptive chosen ciphertext attack (IND-CCA2)
  - Similar to IND-CCA1, but the adversary can keep query the decryption oracle with any ciphertext other than *C*

Block Cipher

- An n-bit plaintext is encrypted to an n-bit ciphertext
  - $\mathcal{P}$: $\{0,1\}^n$
  - $\mathcal{C}$: $\{0,1\}^n$
  - $\mathcal{K}$: $\{0,1\}^s$
  - **E**: $\mathcal{K} \times \mathcal{P} \rightarrow \mathcal{C}$: E_k: a permutation on $\{0,1\}^n$
  - **D**: $\mathcal{K} \times \mathcal{C} \rightarrow \mathcal{P}$: D_k is E_k^{-1}
  - Block size: n
  - Key size: s

Block Cipher



$$\mathrm{Dim}(C) \le s$$

Ideal Block Cipher

- An ideal block cipher is a substitution cipher from $\{0,1\}^n$ to $\{0,1\}^n$
- Total number of keys: 2^n
  - insecure when n is small
  - impractical when n is large
- Solution: approximation of the ideal block cipher for large n
  - Use a subset of the 2^n possible mappings

Ideal Block Cipher – One-Time Padding

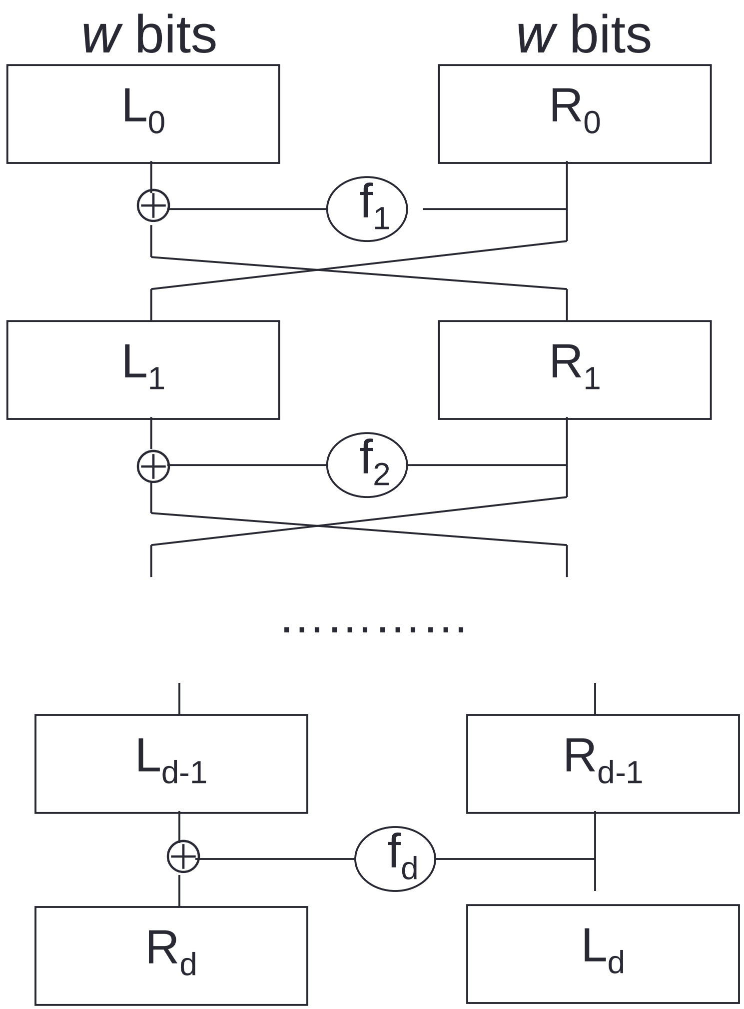
- Use key size at least as long as the plaintext and ciphertext (s>=n)
  - P : $\{0,1\}^n$, C : $\{0,1\}^n$, K : $\{0,1\}^n$

$$C = P \oplus K$$

$$P = C \oplus K$$

- Attain all 2^n possible mappings
- K has to be unpredictable
- Shannon security $H(P) = H(P \mid C)$

Feistel Network



Encryption:

$L_1=R_0$ $\quad$ $R_1=L_0 \oplus f_1(R_0)$

$L_2=R_1$ $\quad$ $R_2=L_1 \oplus f_2(R_1)$

...

$L_d=R_{d-1}$ $\quad$ $R_d=L_{d-1} \oplus f_d(R_{d-1})$

Decryption:

$R_{d-1}=L_d$ $\quad$ $L_{d-1}=R_d \oplus f_d(L_d)$

...

$R_0=L_1$ $\quad$ $L_0=R_1 \oplus f_1(L_1)$

Data Encryption Standard (DES)

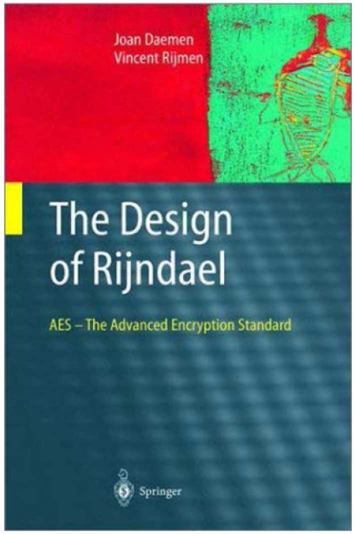
- Designed by IBM, with modification proposed by NSA
- US national standard from 1977 to 2001
- Block size 64 bits;
- Key size 56 bits
- 16-round Feistel network
- Designed mostly for hardware implementations
- Insecure to use now because the key space is too small
  - Takes 6.4 days with <USD$10,000 machine in 2008

Attacking Block Ciphers

- Types of attacks to consider
  - known plaintext: given several pairs of plaintexts and ciphertexts, recover the key (or decrypt another block encrypted under the same key)
  - how would chosen plaintext and chosen ciphertext work?
- Standard attacks
  - exhaustive key search
  - dictionary attack
  - differential cryptanalysis, linear cryptanalysis
- Side-channel attacks

Advanced Encryption Standard (AES)

- AES, also known Rijndael, is adopted as an encryption standard by the U.S. government in the year 2000 and now used widely.
- Designed to be efficient in both hardware and software across a variety of platforms.
- Not a Feistel Network
- Block size: 128 bits
- Variable key size: **128, 192, or 256 bits.**
- Variable number of rounds (10, 12, 14):
  - 10 if K = 128 bits
  - 12 if K = 192 bits
  - 14 if K = 256 bits
- No known weaknesses

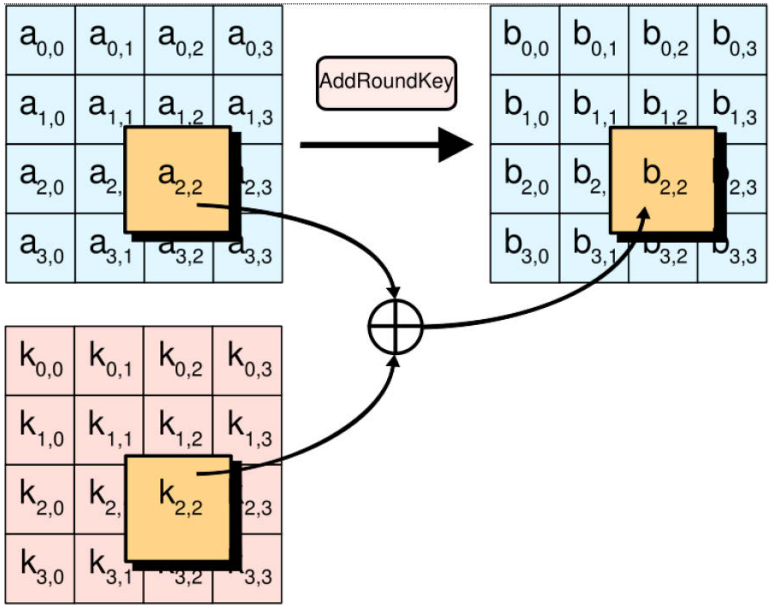


AES

- AES operates on a 4 by 4 byte array, which is called the “state”, in several rounds(10, 12, or 14)
- In each round, 4 steps are taken
  - Adding round keys
  - Bytes Substitution
  - Row Shifting
  - Column Mixing

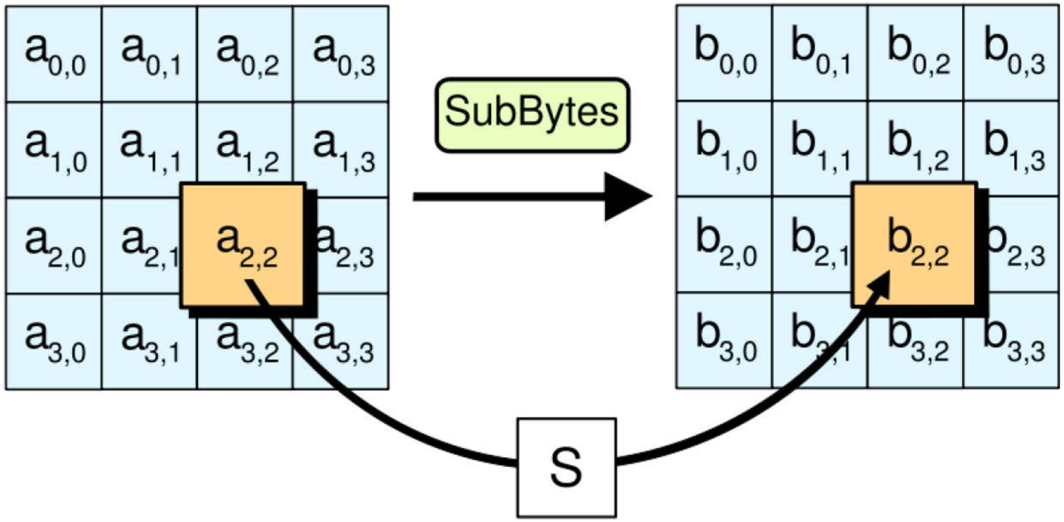
AES – Adding Round Key

- AES uses an elaborated key schedule algorithm to generate round keys from the secret key
- Each round key is also a 4 by 4 byte array
- The state will be XORed with the round key

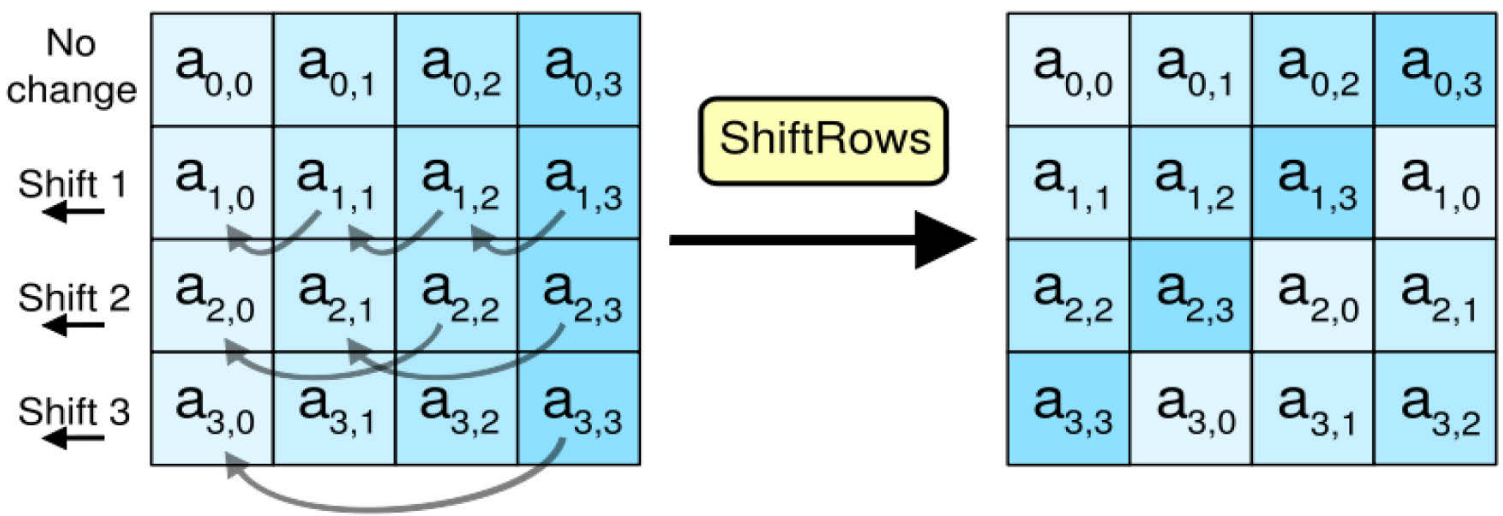


AES – Bytes Substitution

- Every bytes is then replaced by a S-box
- This step makes sure of the non-linear property in AES

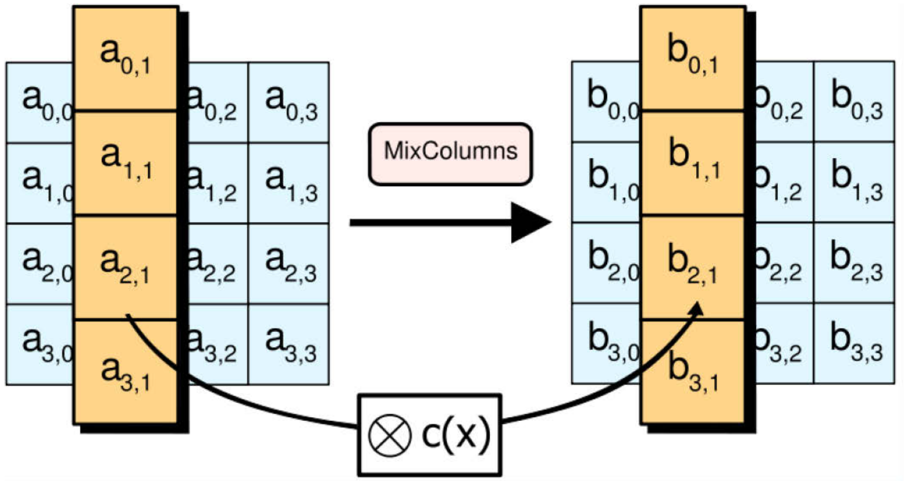


AES Row Shifting



AES Column Mixing

- Every bytes in the new column depends on every bytes in the original column.

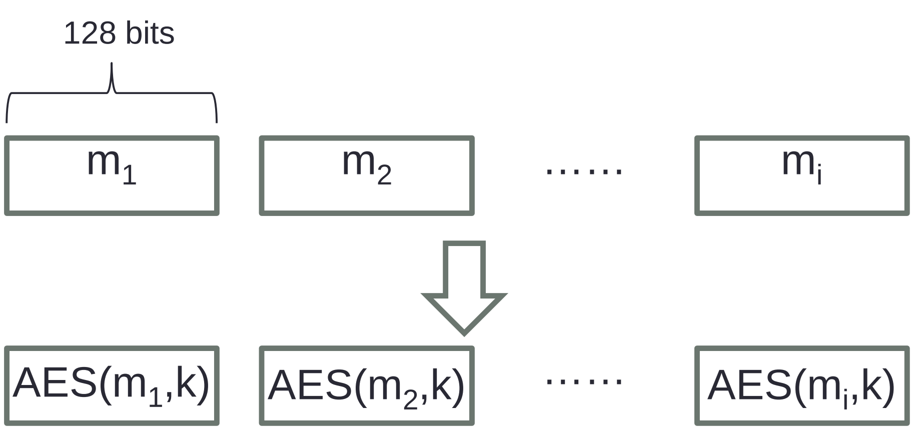


- Row shifting and column mixing provide diffusion in the ciphertext.

Each column is treated as a polynomial over **GF**(2^8) and is then multiplied modulo x^4+1 with a fixed polynomial $c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02$

Block Cipher Encryption Modes

- Block Cipher works on fixed n-bit length block
  - n=64 for DES; n=128 for AES
- What if we need to encrypt data of some arbitrary length?



Sounds like an idea?

Electronic Code Book (ECB)

- Message is broken into independent block;
- Electronic Code Book (ECB): each block encrypted separately.
- **Encryption: $c_i = E_k(m_i)$**
- **Decrytion: $m_i = D_k(c_i)$**

Properties of ECB

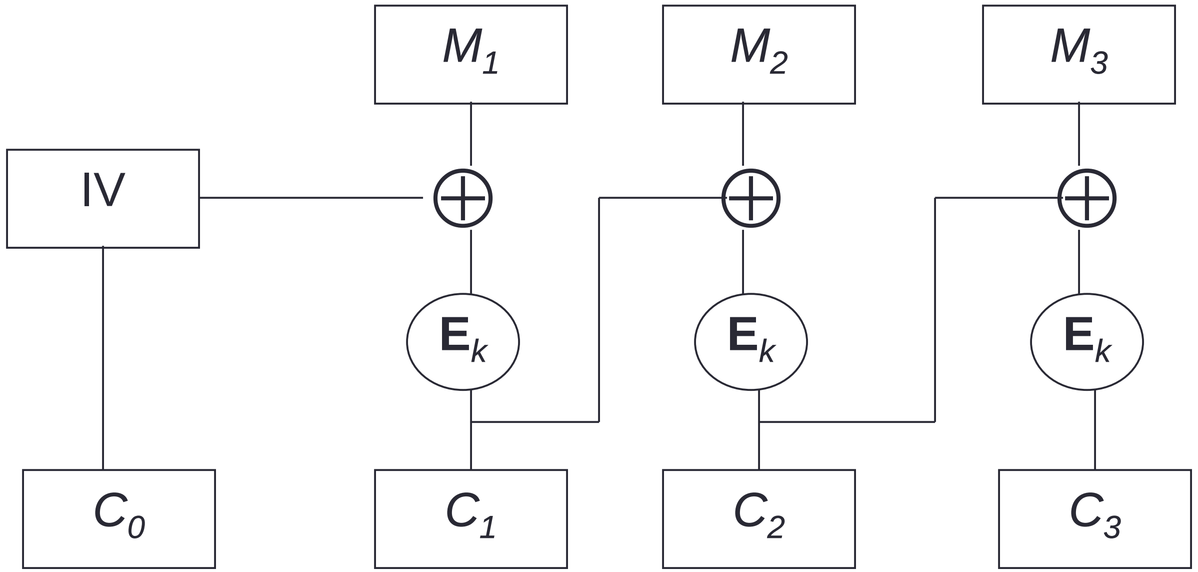
- Deterministic:
  - the same data block gets encrypted the same way,
    - reveals patterns of data when a data block repeats
  - when the same key is used, the same message is encrypted the same way
- Usage: not recommended to encrypt more than one block of data

Cipher Block Chaining (CBC)

- Cipher Block Chaining (CBC): next input depends upon previous output

  Encryption: $C_i = E_k\,(M_i \oplus C_{i-1})$, with $C_0 = IV$

  Decryption: $M_i = C_{i-1} \oplus D_k(C_i)$, with $C_0 = IV$



Properties of CBC

- Randomized encryption: repeated text gets mapped to different encrypted data.
  - can be proven to be “secure” assuming that the block cipher has desirable properties and that random IV’s are used
- A ciphertext block depends on all preceding plaintext blocks; reorder affects decryption
- Usage: chooses random IV and protects the integrity of IV

Counter Mode (CTR)

- Counter Mode (CTR): A way to construct PRNG using DES
  - $y_i = E_k[\text{counter}+i]$
  - Sender and receiver share: counter (does not need to be secret) and the secret key.

Properties of CTR

- Gives a stream cipher from a block cipher
  - subject to limitations of stream ciphers (what are they?)

- Randomized encryption:
  - when starting counter is chosen randomly

- Random Access: decryption of a block can be done in random order, very useful for hard-disk encryption.

Galois Counter Mode

- Provides authenticated encryption (AE)

Stream Cipher

- Both plaintext ($m_0, m_1, m_2, \ldots$) and ciphertext ($c_0, c_1, c_2, \ldots$) are viewed as bit streams
- Synchronized input / output bitstreams
  - Encryption: $m_p \rightarrow c_q$, $m_{p+1} \rightarrow c_{q+1}, \ldots$
  - Decryption: $c_i \rightarrow m_j$, $c_{i+1} \rightarrow m_{j+1}, \ldots$
- Typically very fast
- In practice, streams can be of arbitrary length
- Useful for real-time streaming

Stream Cipher

- Intuition from One-Time Padding

$$\begin{tikzcd} & (k_0,k_1,k_2,\ldots) \arrow[d, "k_i"] & \\ (m_0,m_1,m_2,\ldots) \arrow[r, "m_i"'] & \oplus \arrow[r, "c_i"'] & (c_0,c_1,c_2,\ldots) \end{tikzcd}$$

Use a pseudo random number generator (PRNG) seeded by secret key to generate the key stream

Stream Cipher – RC4

- Key Scheduling (Seeding the PRNG)

```
for i from 0 to 255
    S[i] := i
endfor
j := 0
for i from 0 to 255
    j := (j + S[i] + key[i mod keylength]) mod 256
    swap(&S[i],&S[j])
endfor
```

1 ≤ keylength ≤ 256, typically between 5 and 16, corresponding to a key length of 40 – 128 bits

Stream Cipher – RC4

- PRNG Generation and XORing

```
i  := 0
j  := 0
while GeneratingOutput:
    i  := (i + 1) mod 256
    j  := (j + S[i]) mod 256
    swap(&S[i],&S[j])
    byte_cipher := S[(S[i] + S[j]) mod 256]
    result_ciphered := byte_cipher XOR byte_message
endwhile
```

Stream Cipher – RC4

- By Ron Rivest in 1987
- Used in SSL, TLS, WEP, WPA, BitTorrent PE, MS RDP, PDF,…
  - Due to its simplicity and impressive speed
- Weakness in key scheduling
  - Need to pick keys carefully
  - Discard the first few bytes of the keystream ("RC4-drop[n]“)

Stream Cipher vs. Block Cipher

- One can use block cipher in (EBC/CBC/CTR) modes to simulate stream cipher
  - Use ciphertext stealing to fill the remaining space (padding is not a good idea)
- Stream cipher can be applied to a block of data
- The distinction between the two are not always clear due to more powerful hardware
  - TLS and WPA2 support AES (counter mode)

Diffie-Hellman Key Agreement

- g = public (prime) base, known to Alice, Bob, and Eve. $g = 5$
- p = public (prime) number, known to Alice, Bob, and Eve. $p = 23$
- a = Alice's private key, known only to Alice. $a = 6$
- b = Bob's private key known only to Bob. $b = 15$
- A = Alice's public key, known to Alice, Bob, and Eve. $A = g^a \bmod p = 8$
- B = Bob's public key, known to Alice, Bob, and Eve. $B = g^b \bmod p = 19$

Alice		Bob		Eve	
knows	doesn't know	knows	doesn't know	knows	doesn't know
$p = 23$	$b = ?$	$p = 23$	$a = ?$	$p = 23$	$a = ?$
base $g = 5$		base $g = 5$		base $g = 5$	$b = ?$
$a = 6$		$b = 15$			$s = ?$
$A = 5^a \bmod 23$		$B = 5^b \bmod 23$		$A = 8$	
$A = 5^6 \bmod 23 = 8$		$B = 5^{15} \bmod 23 = 19$		$B = 19$	
$B = 19$		$A = 8$		$s = 19^a \bmod 23 = 8^b \bmod 23$	
$s = B^a \bmod 23$		$s = A^b \bmod 23$			
$s = 19^6 \bmod 23 = 2$		$s = 8^{15} \bmod 23 = 2$			
$s = 2$		$s = 2$			

Blue:non-secret
Red:secret

- Now s = the shared secret key and it is known to both Alice and Bob, but *not* to Eve. $s = 2$

http://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange

Diffie-Hellman Key Agreement

- Based on hardness of the discrete logarithm problem
- Can be used to establish a secret encryption key ***s*** over a **authenticated** but not **private** channel
  - Subject to man-in-the-middle attacks
- Can be used to establish a public key cryptosystem
  - Alice's private key as ***a***
  - Alice's public key as (g^a mod p, g, p)
  - Bob chooses a random ***b*** and sends (g^b mod p) to Alice
  - Bob encrypts message to Alice with symmetric key $(g^a)^b$ mod p

  - A preshared public key can prevent man-in-the-middle attacks