3 : Symmetric Key Encryption

IT5306 - Principles of Information Security

Level III - Semester 5

List of sub topics

3.1. The Data Encryption Standard (DES)

3.2. Triple DES

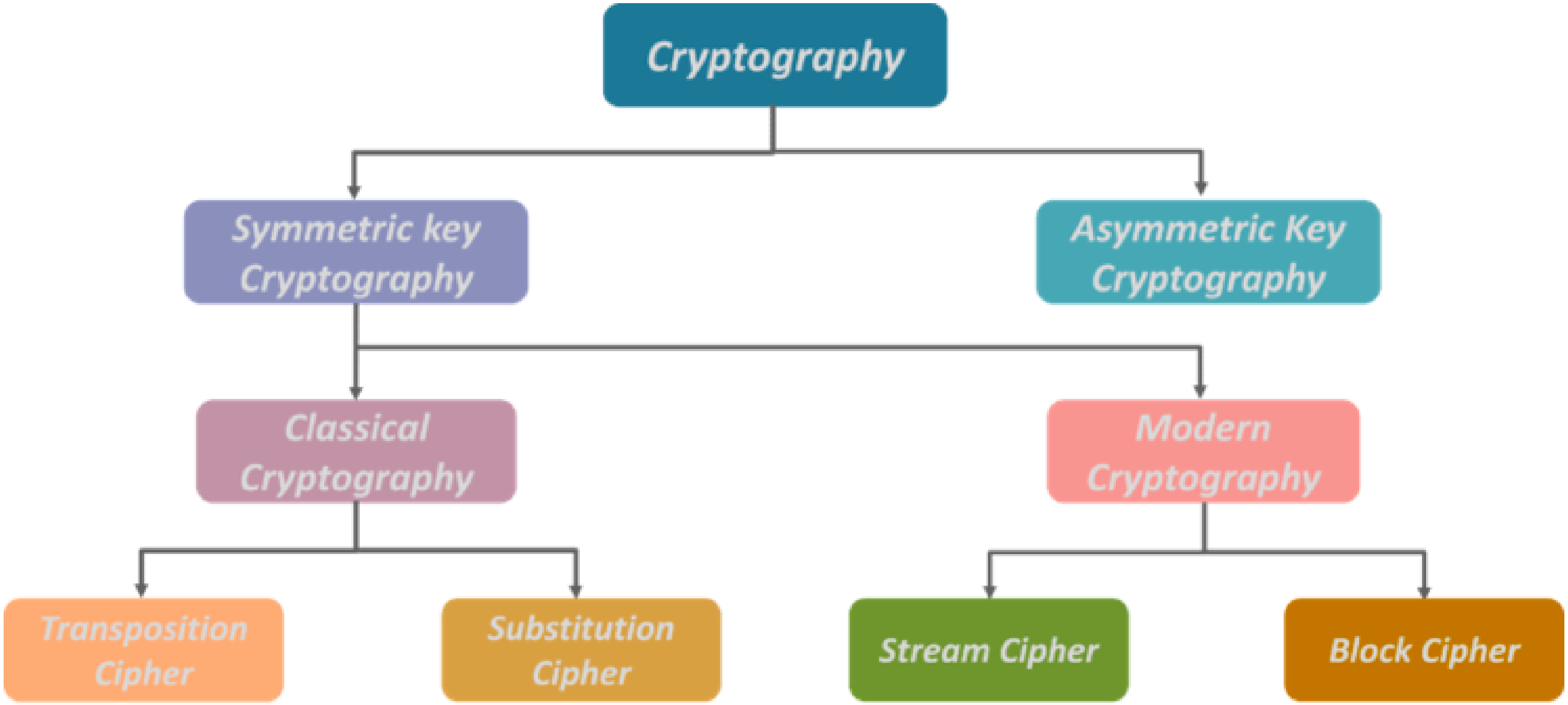
3.3. Advanced Encryption Standard (AES)

3.4. Block Cipher Modes

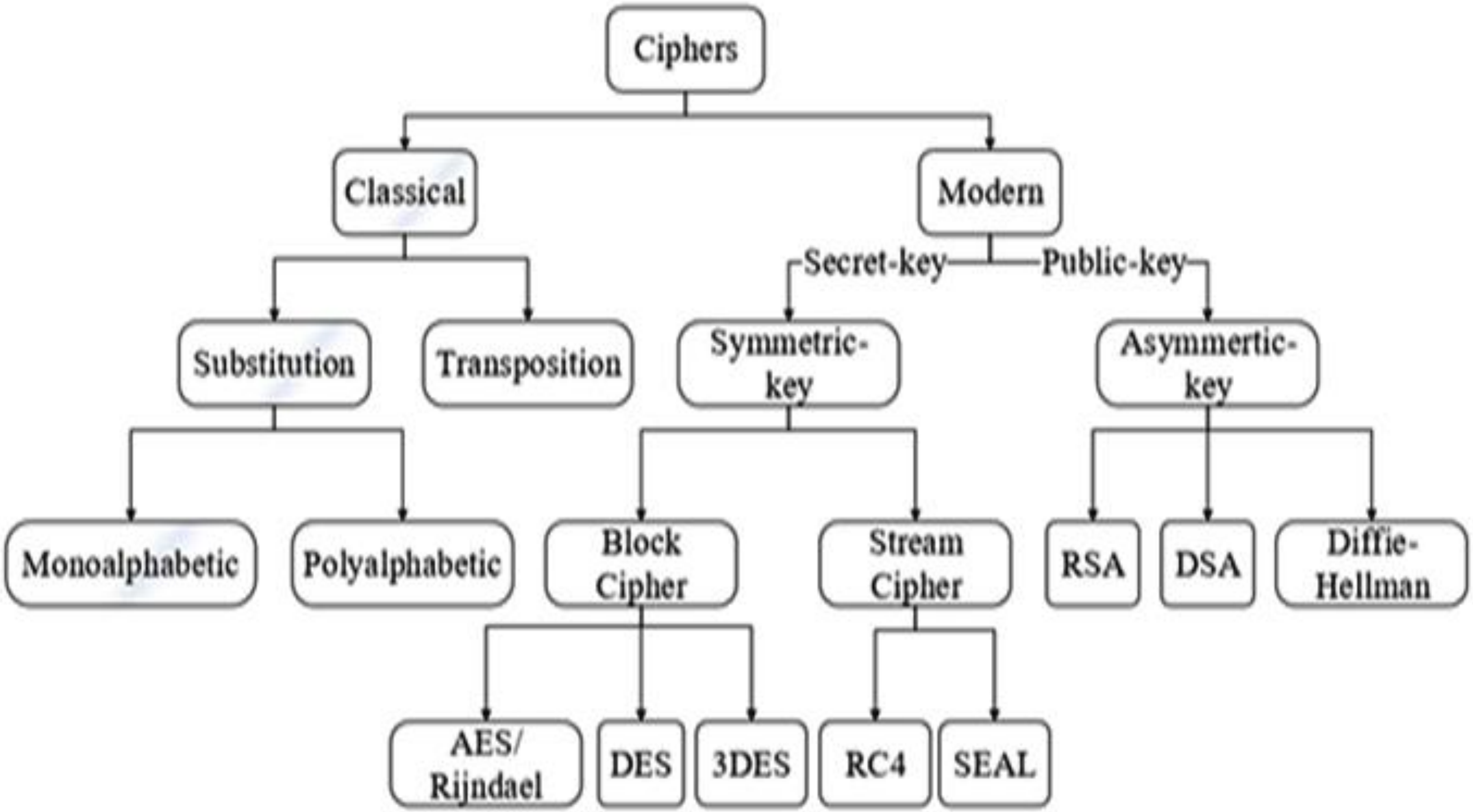
3.5. Applications of Symmetric Key Encryption

3.6. Advantages and Disadvantages

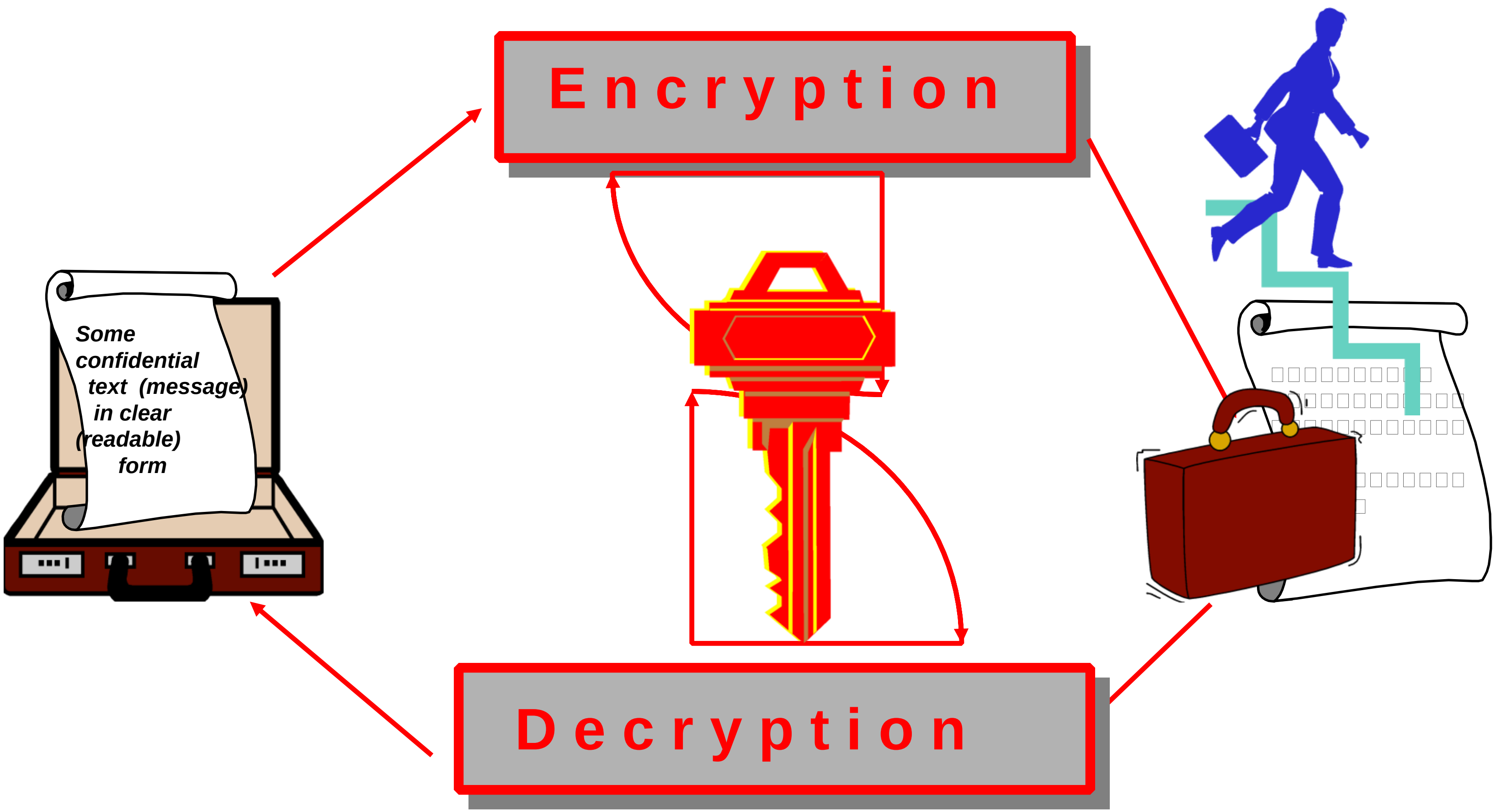
Cryptographic Algorithms



Cryptographic Algorithms



Symmetric key Cryptograms



Symmetric Encryption

There are two types of symmetric encryption algorithms:

Block algorithms. Set lengths of bits are encrypted in blocks of electronic data with the use of a specific secret key. As the data is being encrypted, the system holds the data in its memory as it waits for complete blocks.

Stream algorithms. Data is encrypted as it streams instead of being retained in the system's memory.

Symmetric Algorithms

- AES (Advanced Encryption Standard)
- DES (Data Encryption Standard)
- IDEA (International Data Encryption Algorithm)
- Blowfish (Drop-in replacement for DES or IDEA)
- RC4 (Rivest Cipher 4)
- RC5 (Rivest Cipher 5)
- RC6 (Rivest Cipher 6)

Symmetric Key Cryptosystem

- Uses a single Private Key shared between users

- Strengths
  - Speed/ Efficient Algorithms – much quicker than Asymmetric
  - Hard to break when using a large Key Size
  - Ideal for bulk encryption / decryption

- Weaknesses
  - Poor Key Distribution (must be done out of band – ie phone, mail, etc)
  - Poor Key Management / Scalability (each user needs a unique key)
  - Cannot provide authenticity or non-repudiation – only confidentiality

Requirements for Symmetric Key Cryptography

Two requirements for secure use of symmetric encryption:

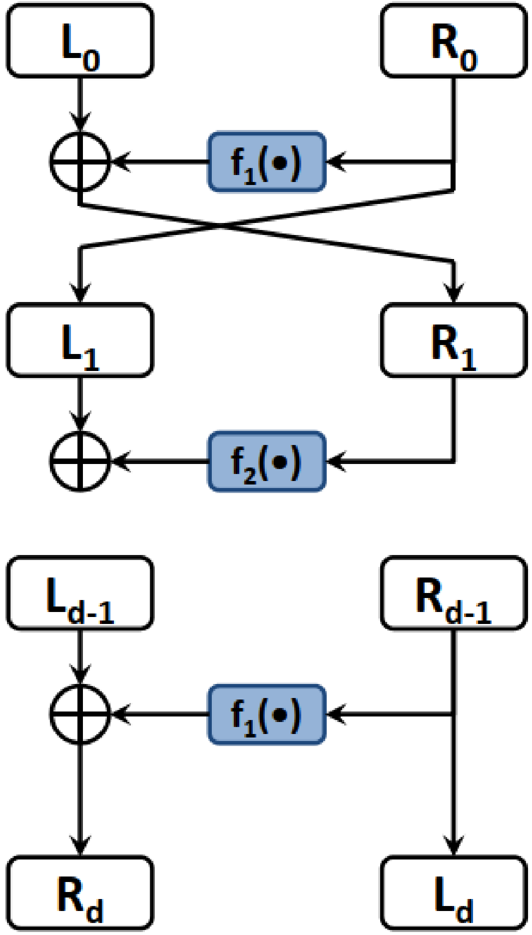
- a strong encryption algorithm
- a secret key, *K*, known only to sender / receiver

$Y = \mathrm{E}K(X)$

$X = \mathrm{D}K(Y)$

- Assume encryption algorithm is known
- Implies a secure channel to distribute key

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

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

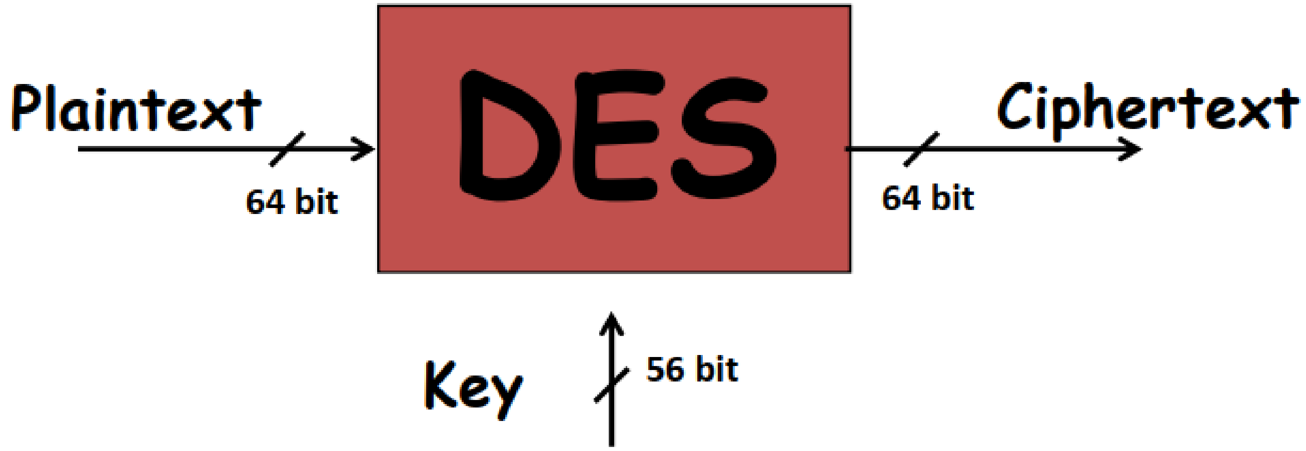
5

Data Encryption Standard (DES)

- Most widely used block cipher in world
- Adopted in 1977 by NBS (now NIST) as FIPS PUB 46
- Encrypts 64-bit data using 56-bit key
- Has widespread use
- Has been the subject of considerable controversy over its security

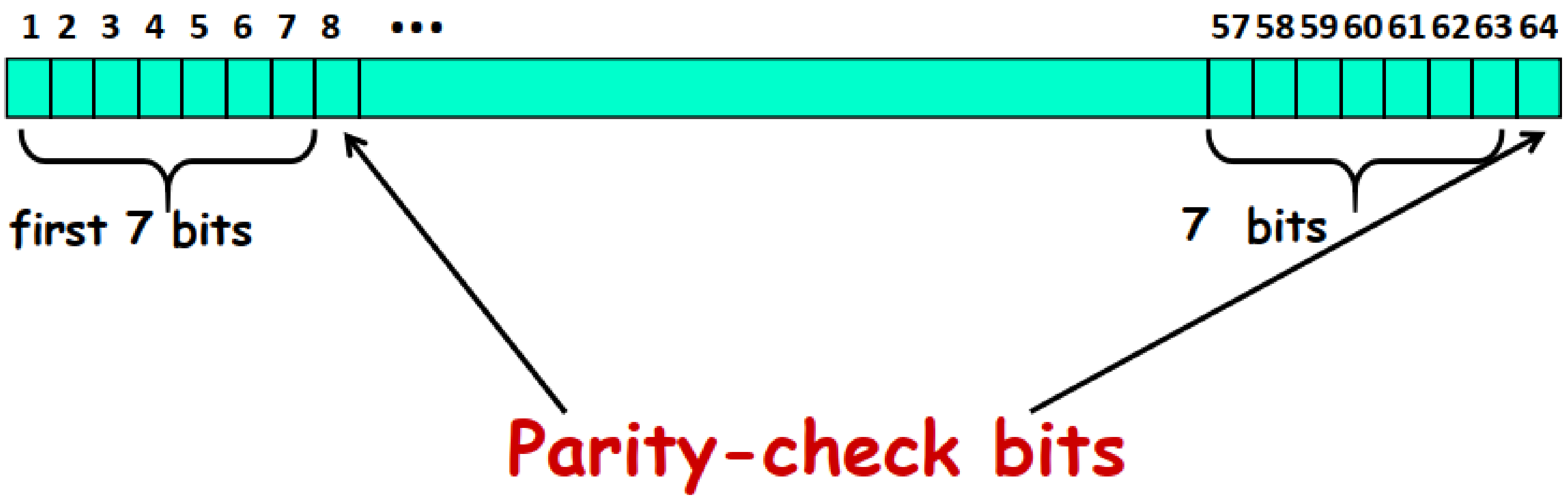
DES Features

- Features:
  - Block size = 64 bits
  - Key size = 56 bits (in reality, 64 bits, but 8 are used as parity-check bits for error control, see next slide)
  - Number of rounds = 16
  - 16 intermediary keys, each 48 bits



Key length in DES

- In the DES specification, the key length is 64 bit:
- 8 bytes; in each byte, the 8th bit is a parity-check bit



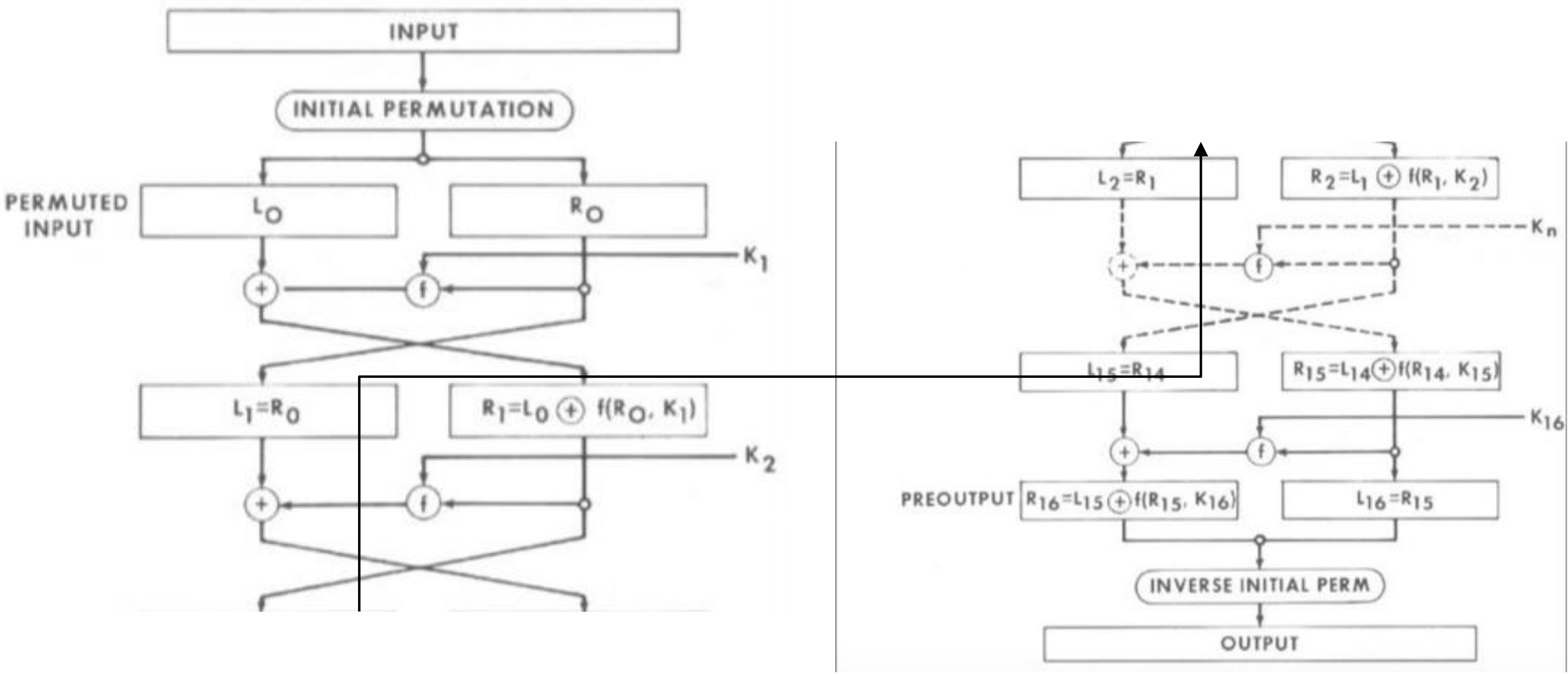
Each parity-check bit is the XOR of the previous 7 bits

DES – Key Size

- 56-bit keys have 2^{56} = 7.2 x 1016 values
- Brute force search looks hard
- Recent advances have shown that this is possible
  - in 1997 on Internet in a few months
  - in 1998 on DES Cracker dedicated h/w (EFF) in a less than 3 days (cost: $250,000)
  - in 1999 on Internet in a few hours
  - in 2010 above on Internet in a few minutes

Now we have alternatives to DES

DES

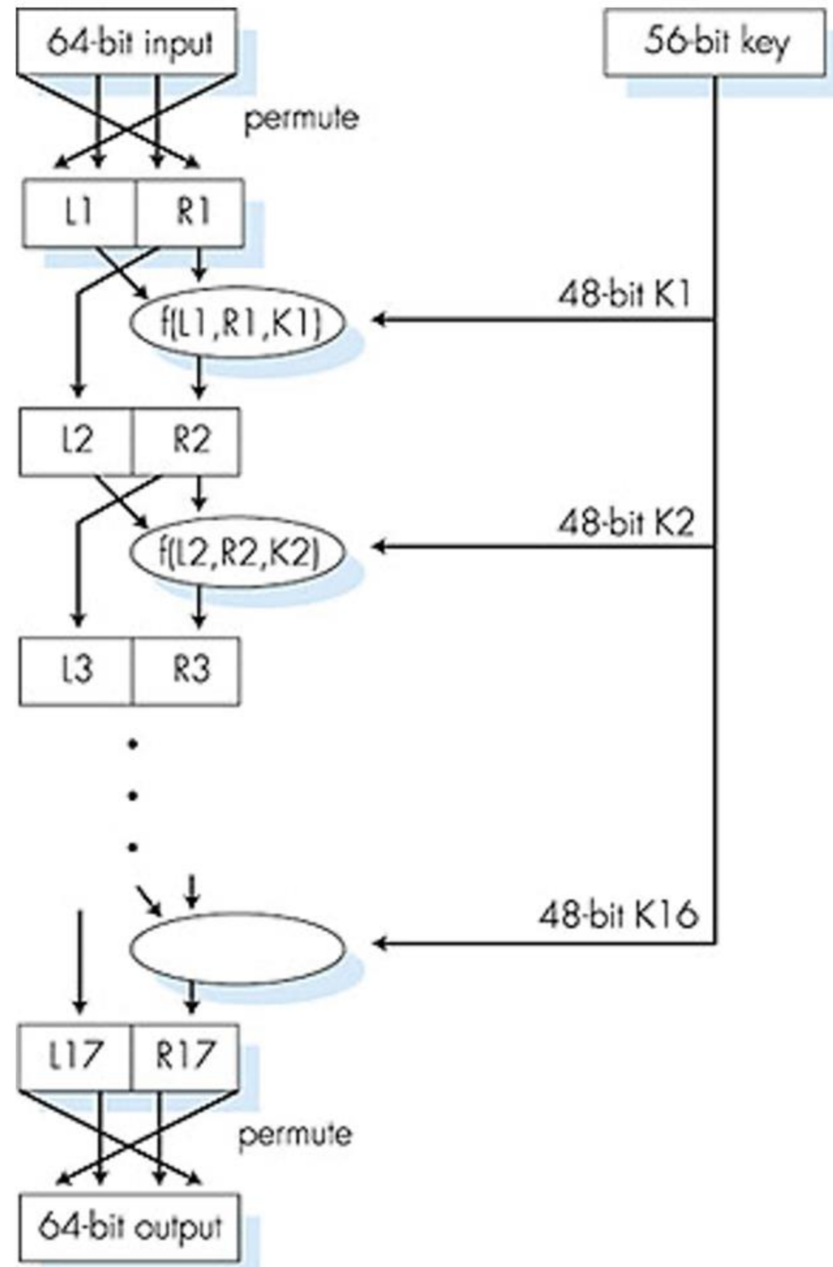


INPUT
INITIAL PERMUTATION
PERMUTED INPUT
L_0
R_0
K_1
$L_1=R_0$
$R_1=L_0 \oplus f(R_0, K_1)$
K_2
$L_2=R_1$
$R_2=L_1 \oplus f(R_1, K_2)$
K_n
$L_{15}=R_{14}$
$R_{15}=L_{14} \oplus f(R_{14}, K_{15})$
K_{16}
PREOUTPUT
$R_{16}=L_{15} \oplus f(R_{15}, K_{16})$
$L_{16}=R_{15}$
INVERSE INITIAL PERM
OUTPUT

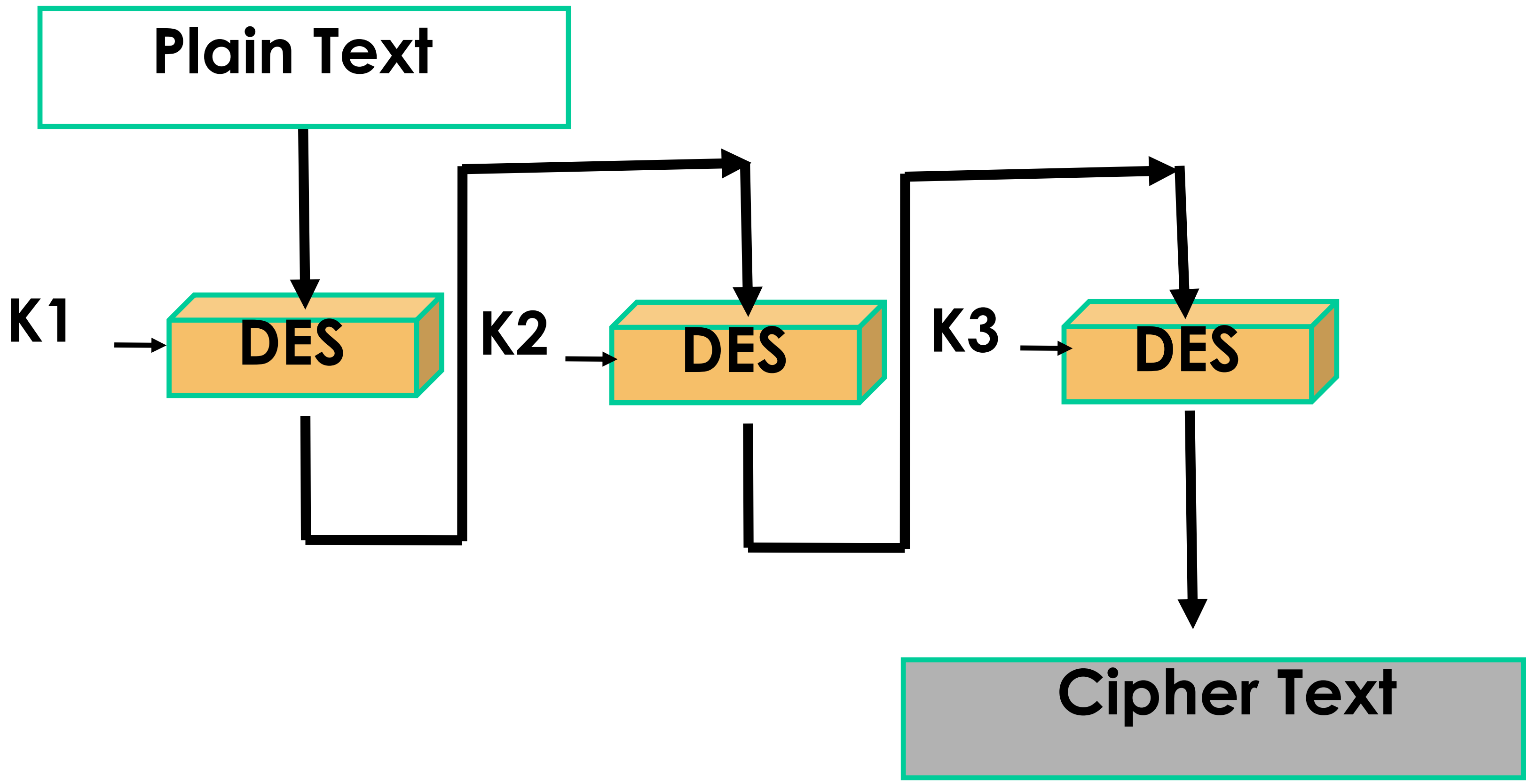
Symmetric key crypto: DES

initial permutation

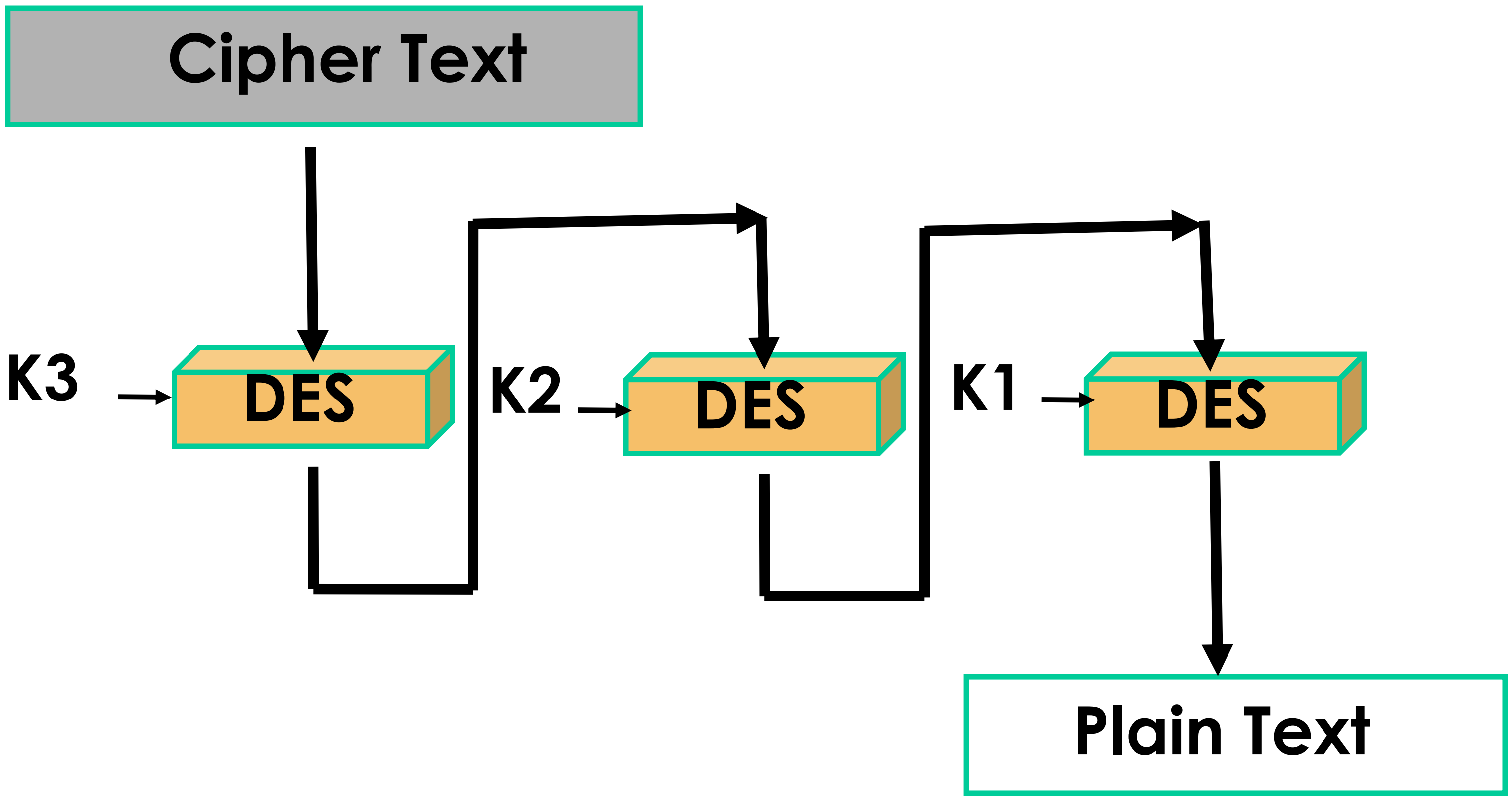
16 identical "rounds" of function application, each using different 48 bits of key final permutation



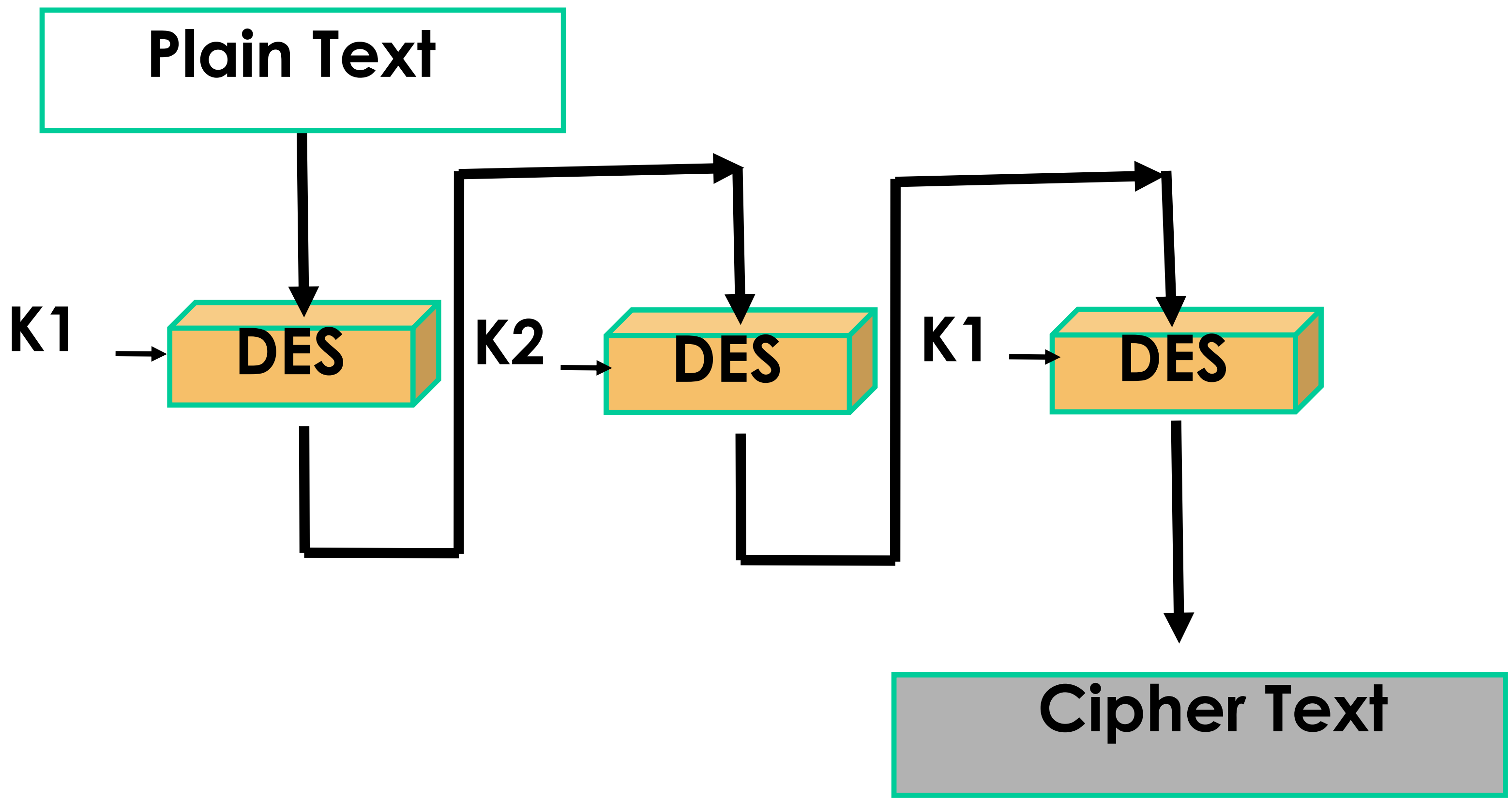
Triple DES - Encryption



Triple DES - Decryption



Triple DES with Two Keys



Triple-DES with Two-Keys

- Use 3 encryptions

  would seem to need 3 distinct keys

  But can use 2 keys with E-D-E sequence

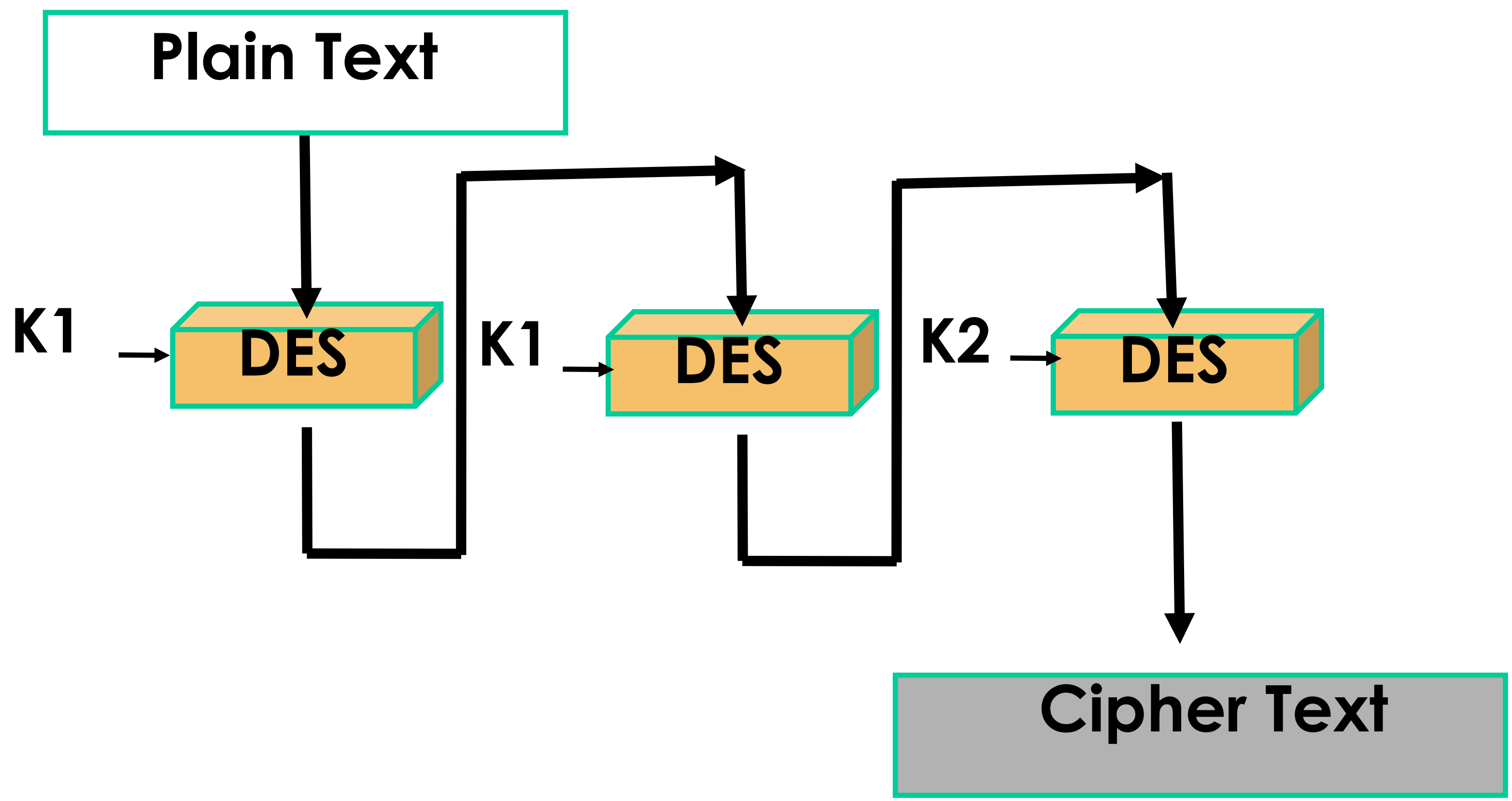
C = EK1[DK2[EK1[P]]]

  Note: encrypt & decrypt equivalent in security

  if K1=K2 then can work with single DES

- Standardized in ANSI X9.17 & ISO8732
- No current known practical attacks

Triple DES Backward Compatibility



DES- AES

- Clearly, a replacement for DES was needed
  - have theoretical attacks that can break it
  - have demonstrated exhaustive key search attacks
- Can use Triple-DES – but slow with small blocks
- NIST issued a call for ciphers in 1997
- 15 candidates accepted in June 1998
- 5 were short listed in August 1999
- Rijndael was selected as the AES in October 2000
- Issued as FIPS PUB 197 standard in November 2001

AES Requirements

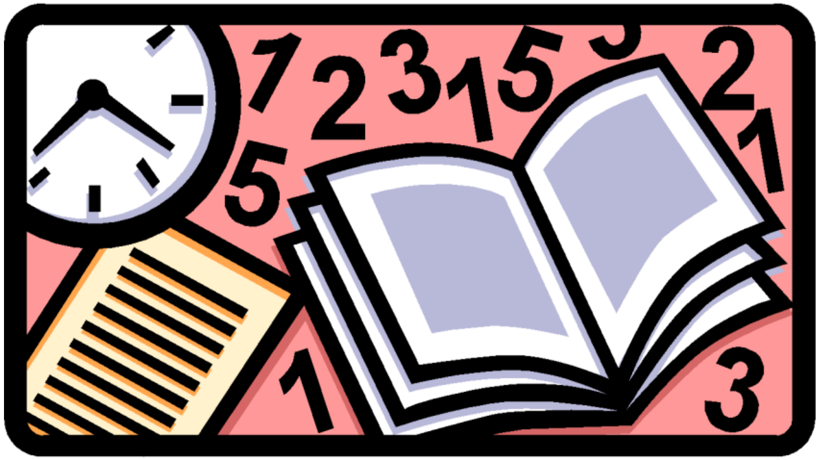
- Private key symmetric block cipher.
- 128-bit data, 128/192/256-bit keys.
- Stronger & faster than Triple-DES.
- Active life of 20-30 years. (+ archival use)
- Provide full specification & design details.
- NIST has released all submissions & unclassified analyses.

AES Shortlist

- After testing and evaluation, shortlist in August 1999:
  - MARS (IBM) - complex, fast, high security margin
  - RC6 (USA) - v. simple, v. fast, low security margin
  - Rijndael (Belgium) - clean, fast, good security margin
  - Serpent (Euro) - slow, clean, v. high security margin
  - Twofish (USA) - complex, v. fast, high security margin

Advanced Encryption Standard (AES)

- In 2001, National Institute of Standards and Technology (NIST) issued AES known as FIPS 197
- AES is based on Rijndael proposed by Joan Daemen, Vincent Rijmen from Belgium

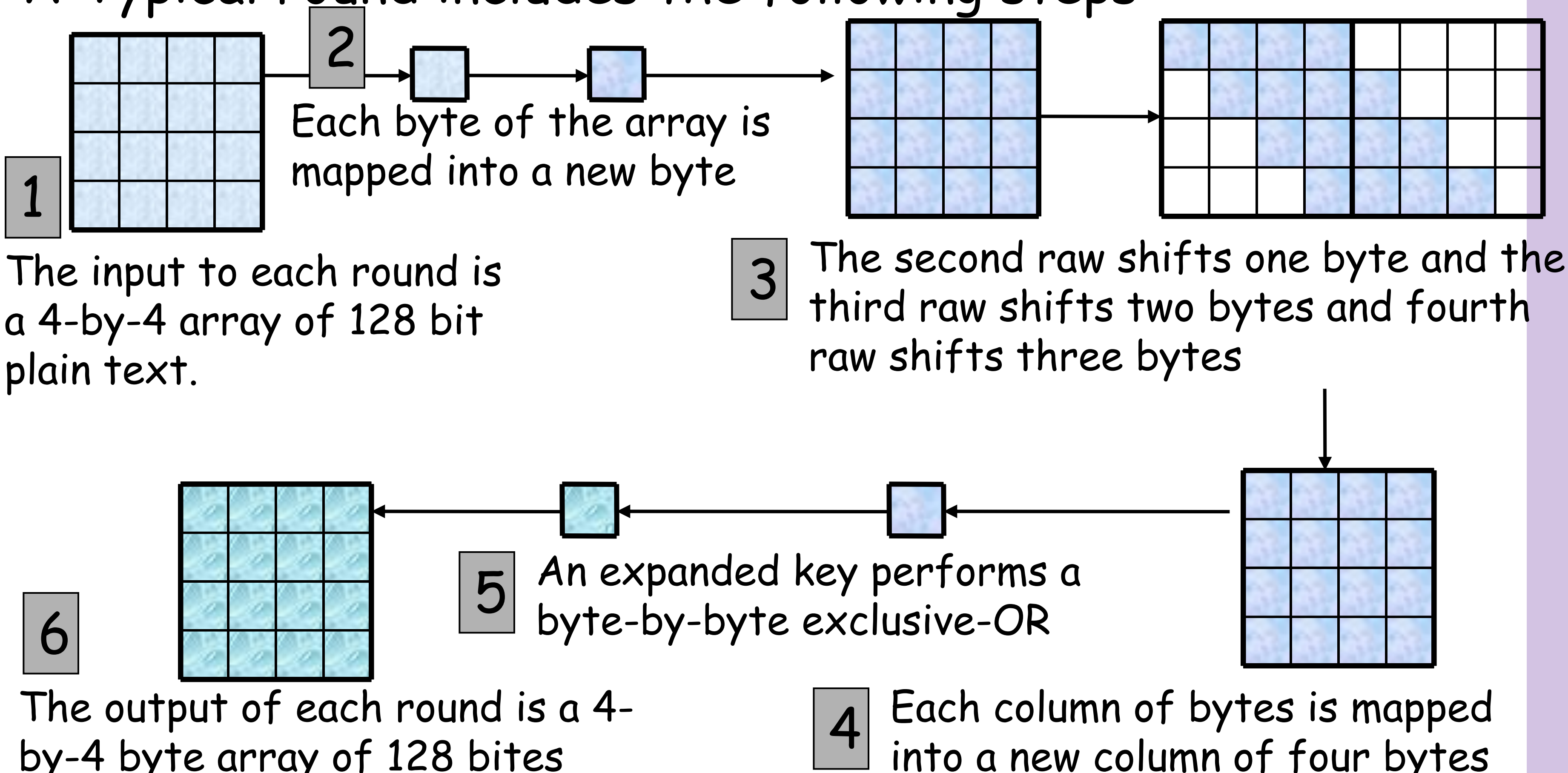


Advanced Encryption Standard (AES)

- AES has block length 128
- Supported key lengths are 128, 192 and 256
- AES requires 10 rounds of processing
- Key is expanded into 10 individual keys
- Decryption algorithm uses the expanded keys in reverse order
- Decryption algorithm is not identical to the encryption algorithm

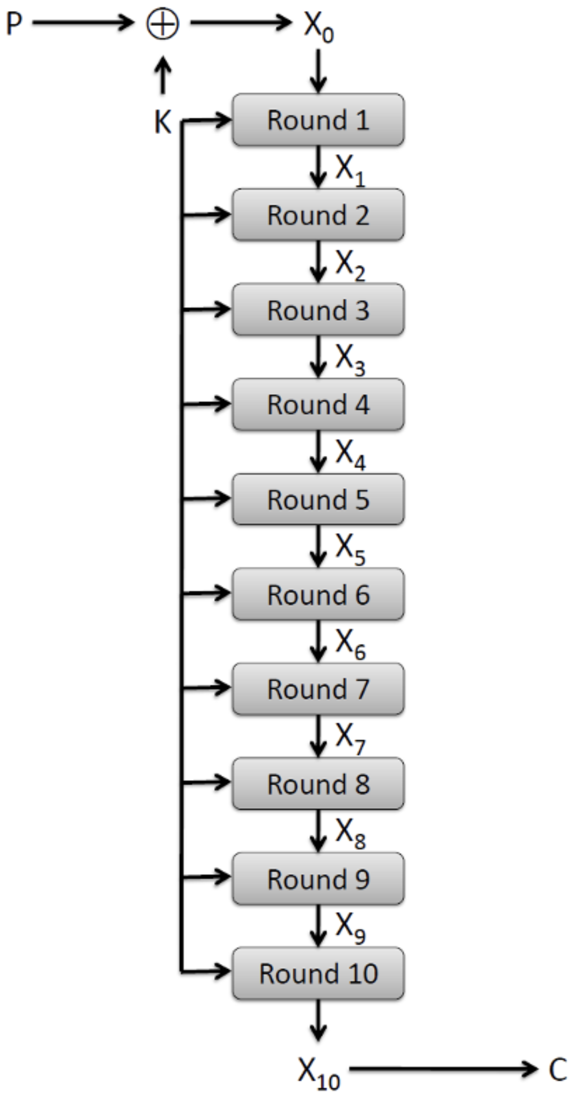
Advanced Encryption Standard (AES)

- A Typical round includes the following steps



AES Round Structure

- The 128-bit version of the AES encryption algorithm proceeds in ten rounds.
- Each round performs an invertible transformation on a 128-bit array, called **state**.
- The initial state X_0 is the XOR of the plaintext P with the key K:
- X_0 = P XOR K.
- Round i (i = 1, …, 10) receives state X_{i-1} as input and produces state X_i.
- The ciphertext C is the output of the final round: $C = X_{10}$.



AES Rounds

Each round is built from four basic steps:

SubBytes step: an S-box substitution step

ShiftRows step: a permutation step

MixColumns step: a matrix multiplication step

AddRoundKey step: an XOR step with a round key derived from the 128-bit encryption key

Block Ciphers - Modes of Operation

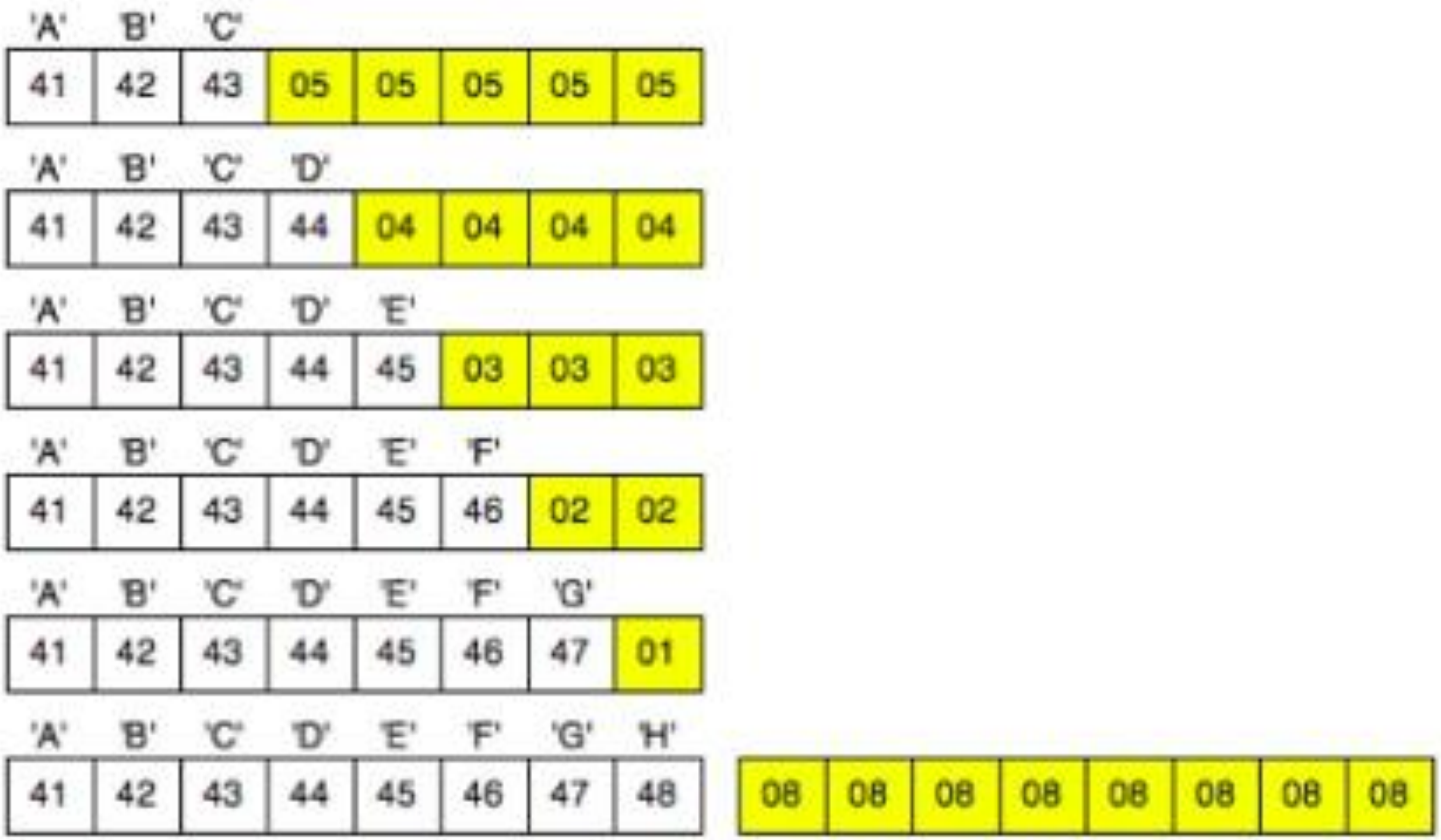
- Block ciphers encrypt fixed size blocks
  - E.g. DES encrypts 64-bit blocks, with 56-bit key

- Given that one needs to encrypt arbitrary amount of information, how do we use in practice,
  - Four modes were defined for DES in ANSI standard
  - **ANSI X3.106-1983 Modes of Use**
  - Subsequently now have 5 for DES and AES

PKCS5 Padding Scheme

- Assume block cipher is 64-bits
- Any message not a multiple of 8 bytes is padded
- Valid pad:
- 1 byte needed: 0x1
- 2 bytes needed: 0x2 0x2
- 3 bytes needed: 0x3 0x3 0x3
- ....
- No padding: 0x8 0x8 0x8 0x8 0x8 0x8 0x8 0x8
- (If the length of the original data is an integer multiple of the block size B, then an extra block of bytes with value B is added.)

PKCS5 Padding Scheme



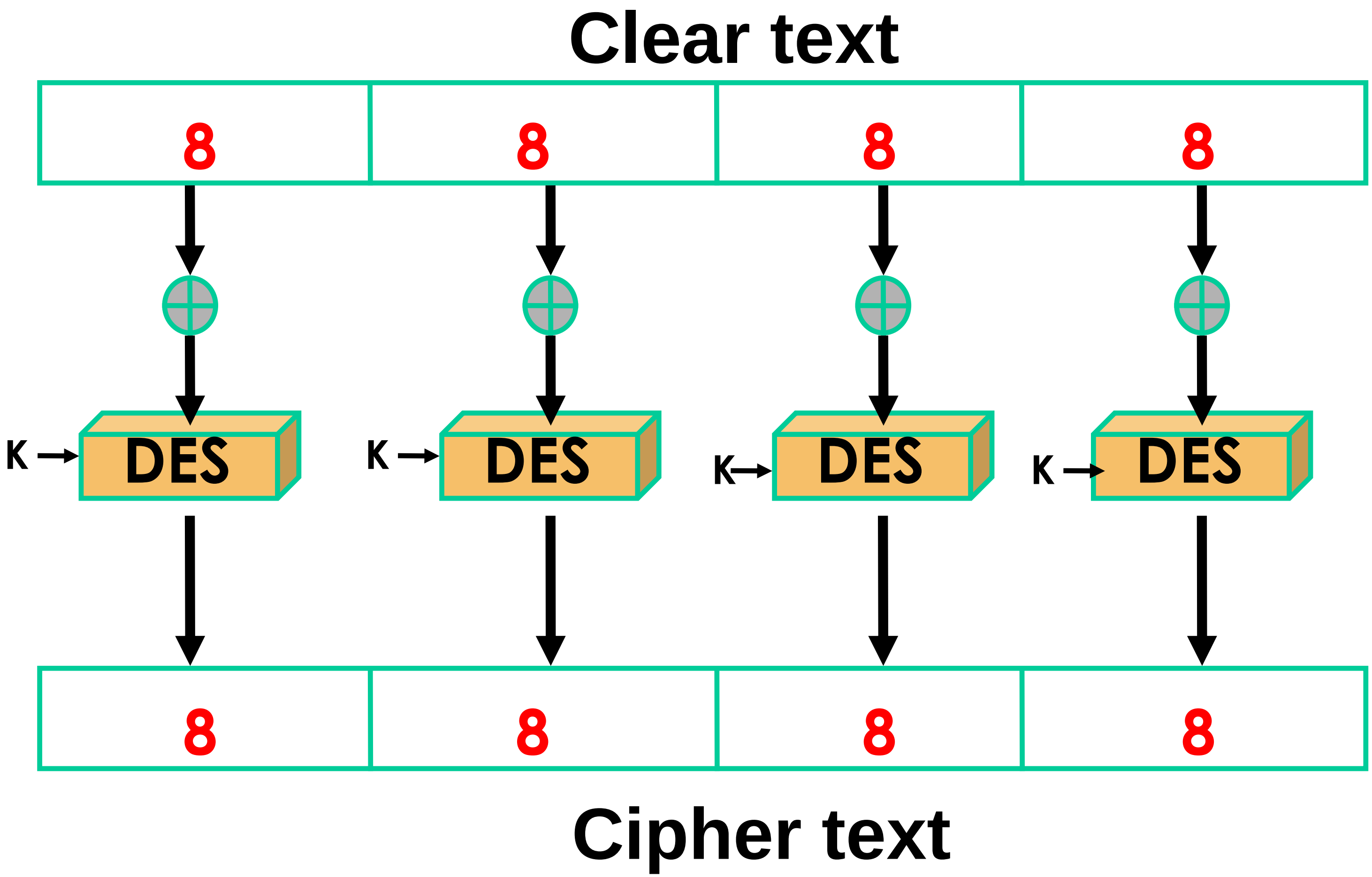
Electronic Codebook Book (ECB)

- Message is broken into independent blocks which are encrypted
- Each block is a value which is substituted, like a codebook, hence name
- Each block is encoded independently of the other blocks

$$C_i = DES_K(P_i)$$

- Uses: secure transmission of single values

Electronic Code Book Mode (ECB)



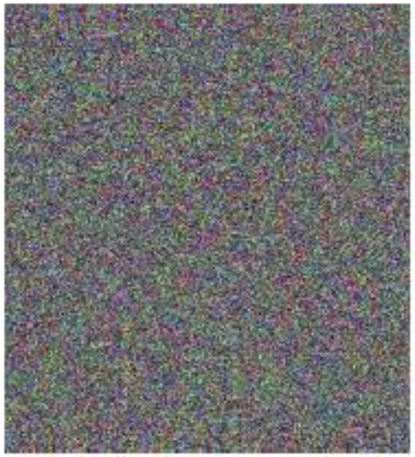
Advantages and Limitations of ECB

- Repetitions in message may show in ciphertext if aligned with message block particularly with data such graphics or with
- Messages that change very little
- Weakness due to encrypted message blocks being independent
- Main use is sending a few blocks of data

ECB vs CBC

Original

Encrypted using ECB mode



Encrypted using other modes

Electronic codebook (ECB), Cipher block chaining (CBC), Cipher feedback (CFB), Output feedback (OFB)

Cipher Block Chaining (CBC)

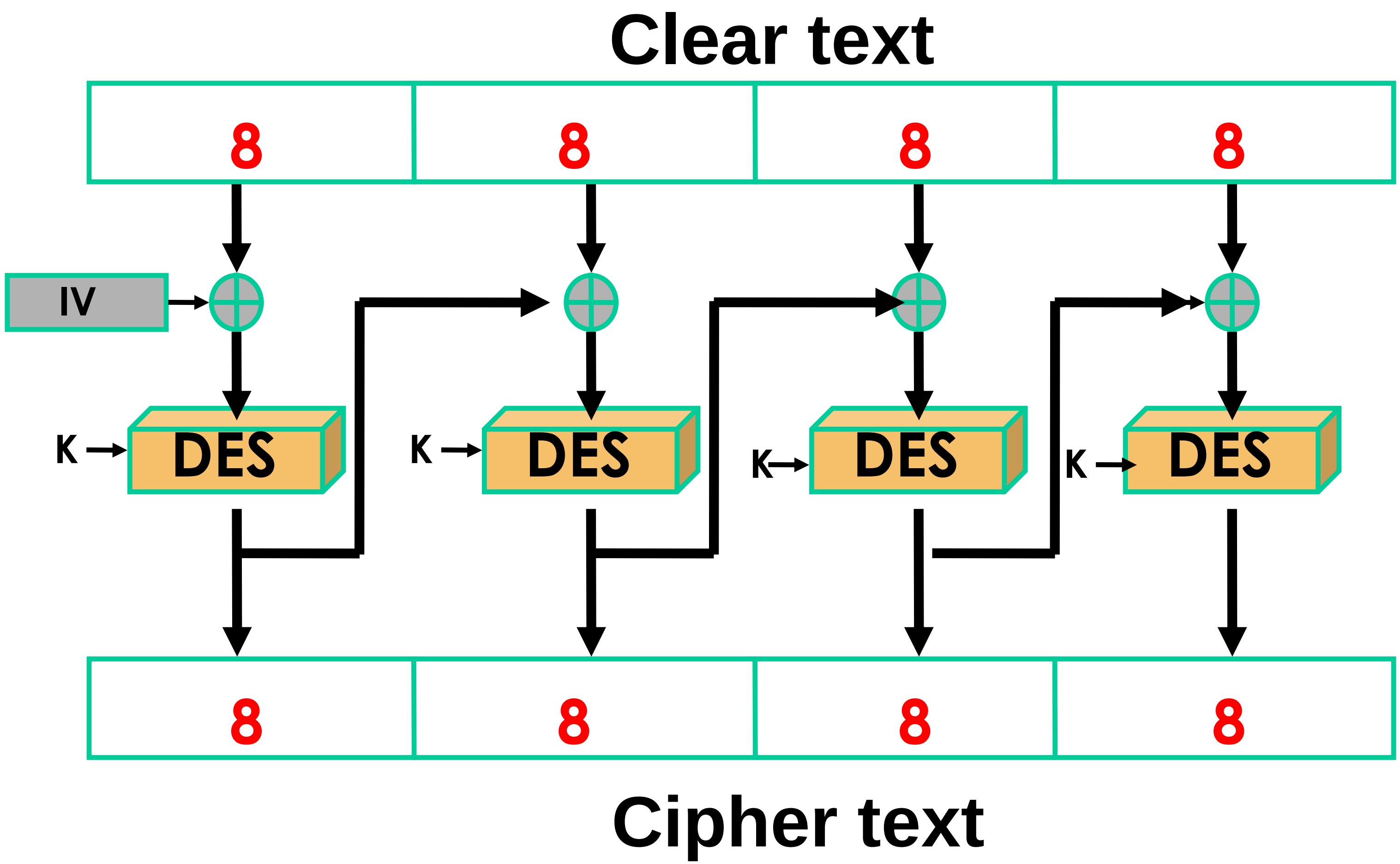
- Message is broken into blocks
- But these are linked together in the encryption operation
- Each previous cipher blocks is chained with current plaintext block, hence name
- Use Initial Vector (IV) to start process

$$C_i = DES_K(P_i \; XOR \; C_{i-1})$$

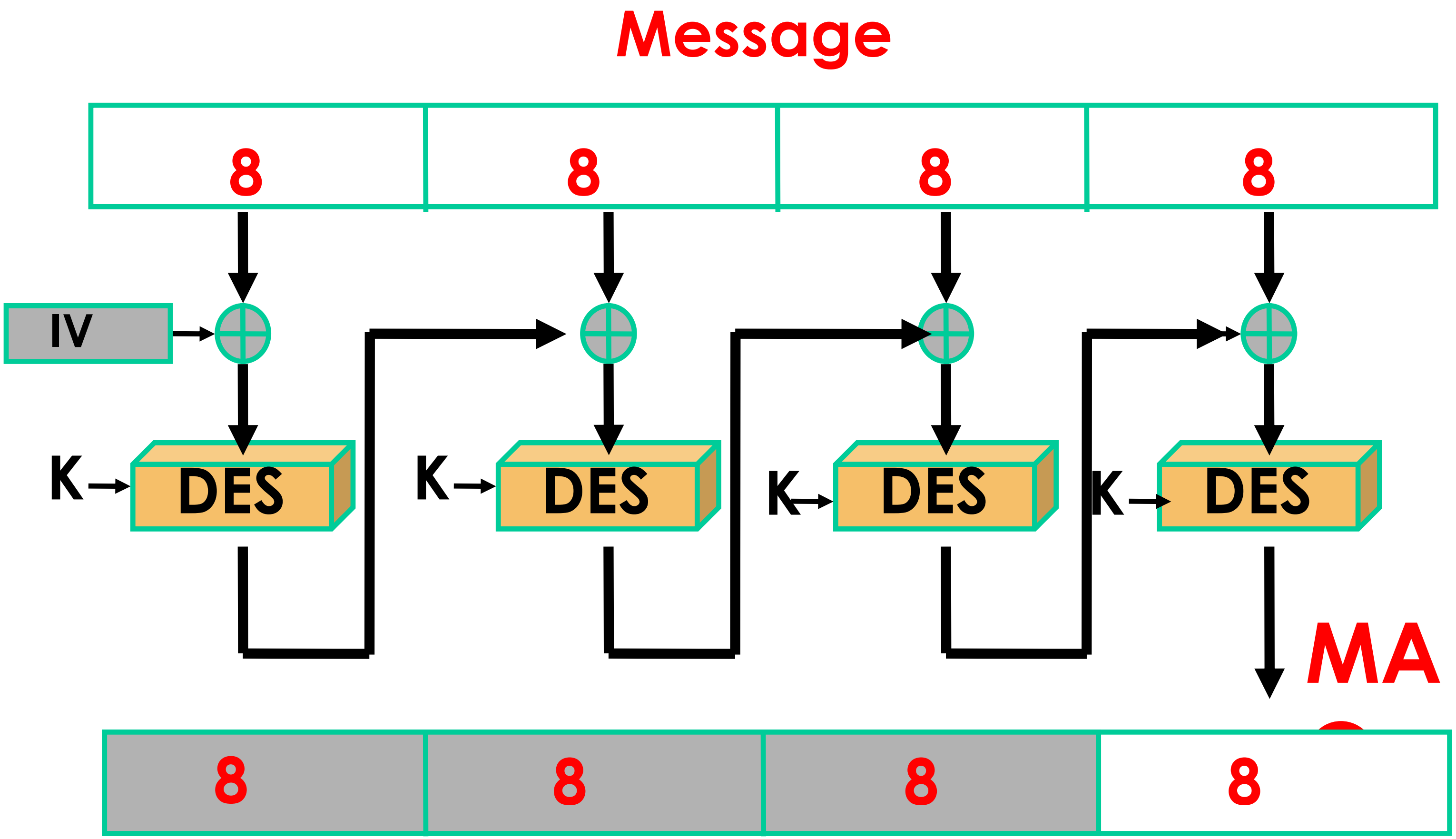
$$C_{-1} = IV$$

- Uses: bulk data encryption, authentication

Cipher Block Chaining Mode (CBC)



MAC based on CBC



Advantages and Limitations of CBC

- Each ciphertext block depends on **all** preceding message blocks thus a change in the message affects all ciphertext blocks after the change as well as the original block

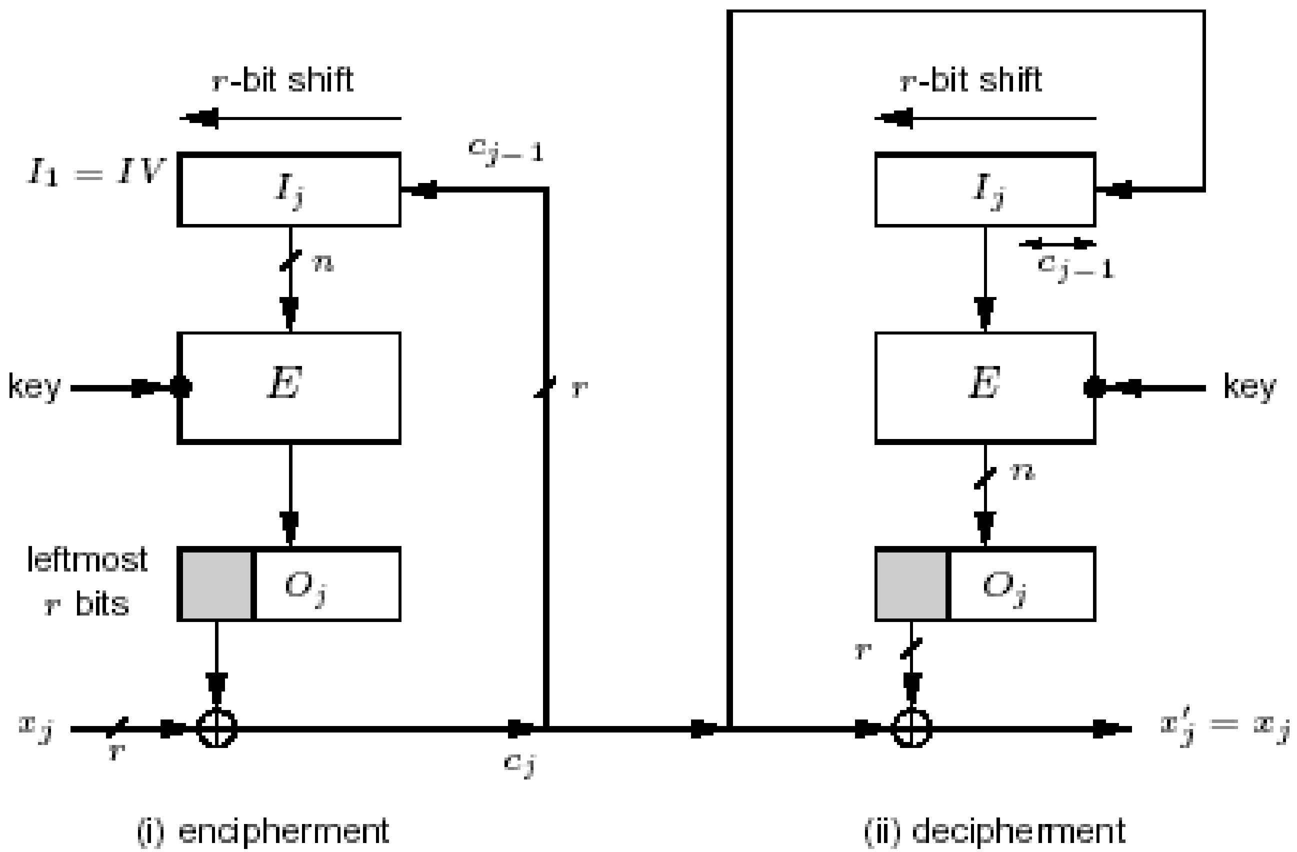
- Need **Initial Value** (IV) known to sender & receiver however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate hence either IV must be a fixed value or it must be sent encrypted in ECB mode before rest of message

- At end of message, handle possible last short block by padding either with known non-data value (e.g. nulls) or pad last block with count of pad size

Cipher Feedback (CFB) mode

- A Stream Cipher where the Ciphertext is used as feedback into the Key generation source to develop the next Key Stream
- The Ciphertext generated by performing an XOR on the Plaintext with the Key Stream the same number of bits as the Plaintext
- Errors will propagate in this mode

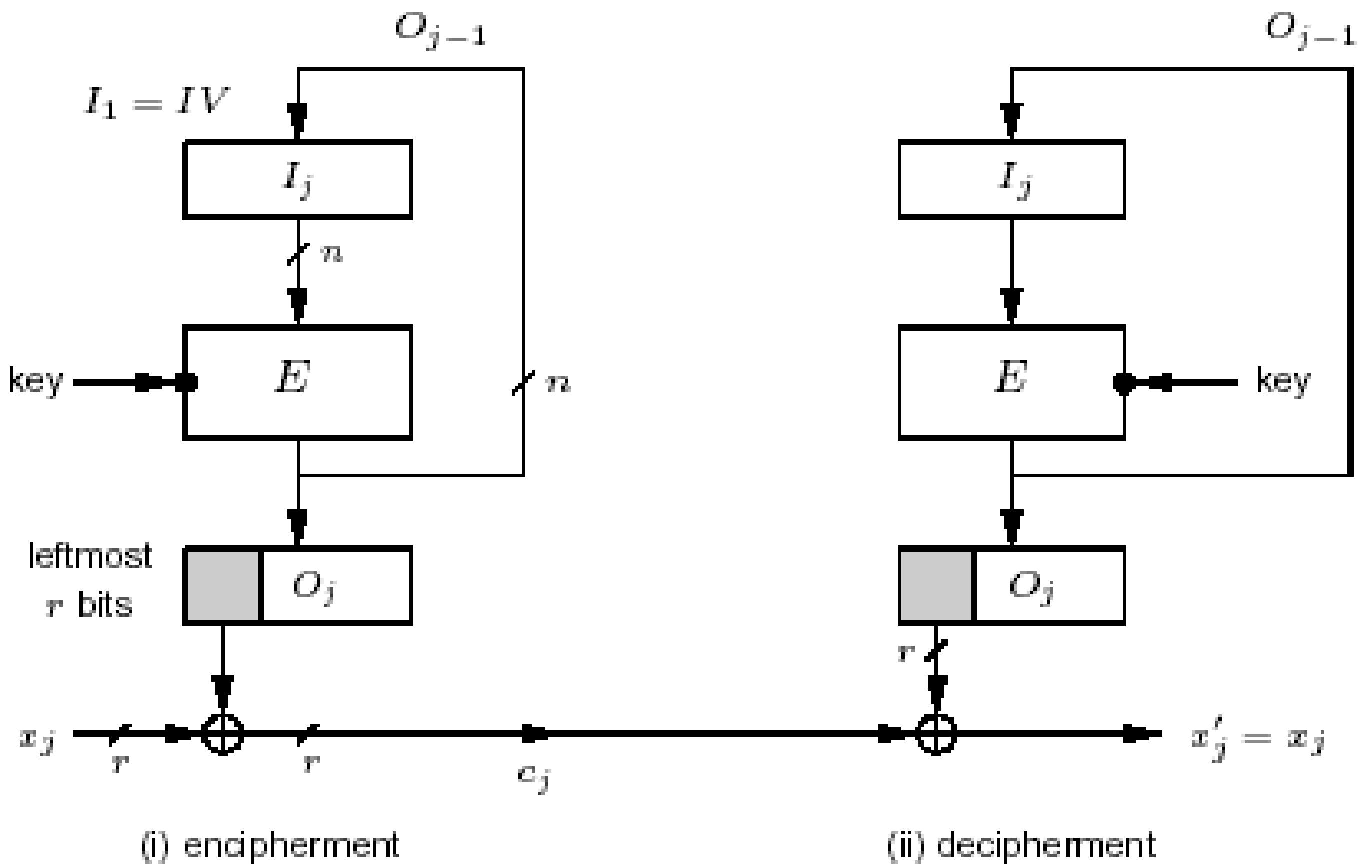
Cipher Feedback Mode (CFB)



Output Feedback(OFB) mode

- A Stream Cipher that generates the Ciphertext Key by XORing the Plaintext with a Key Stream.
- Requires an Initialization Vector
- Feedback is used to generate the Key Stream – therefore the Key Stream will vary
- Errors will not propagate in this mode

Output Feedback Mode (OFB)



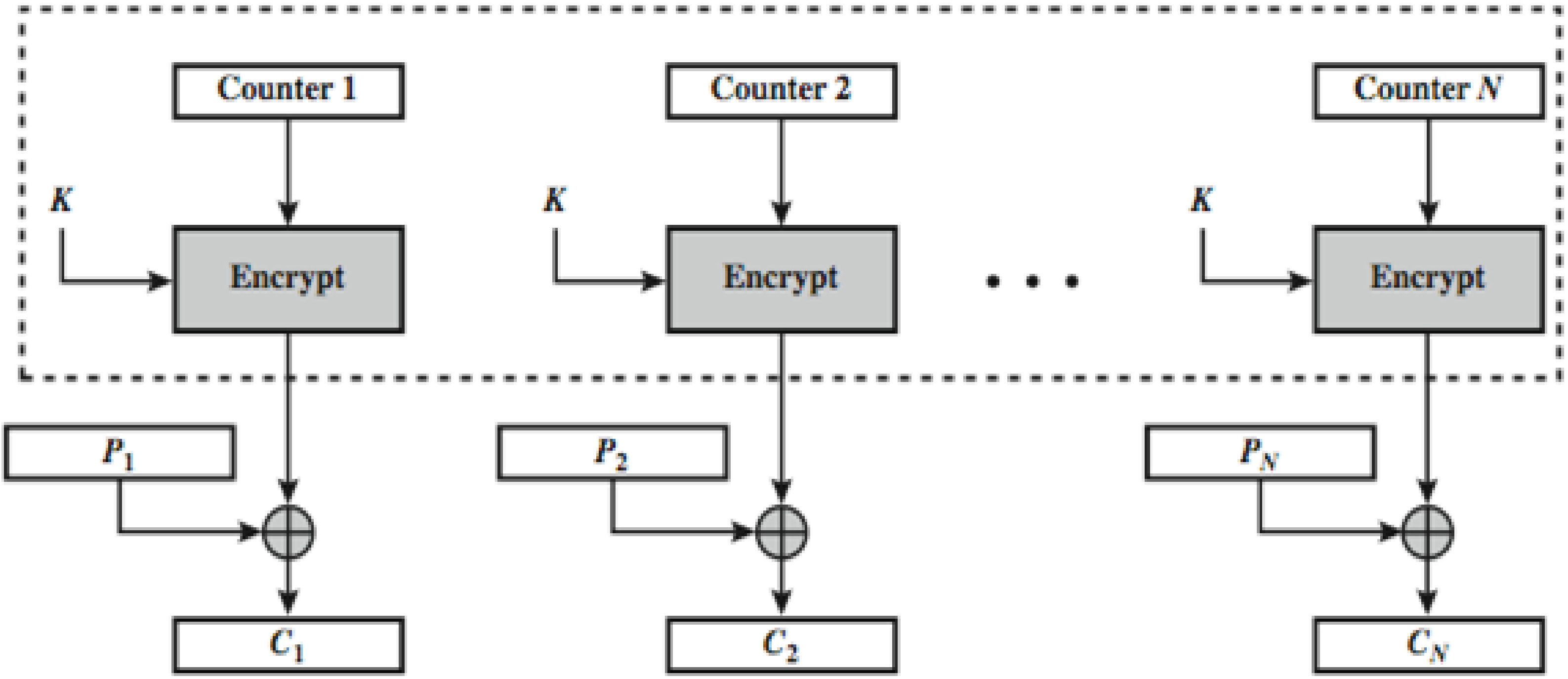
Counter (CTR)

a “new” mode, though proposed early on similar to OFB but encrypts counter value rather than any feedback value

$O_i = E_K(i)$
$C_i = P_i \text{ XOR } O_i$

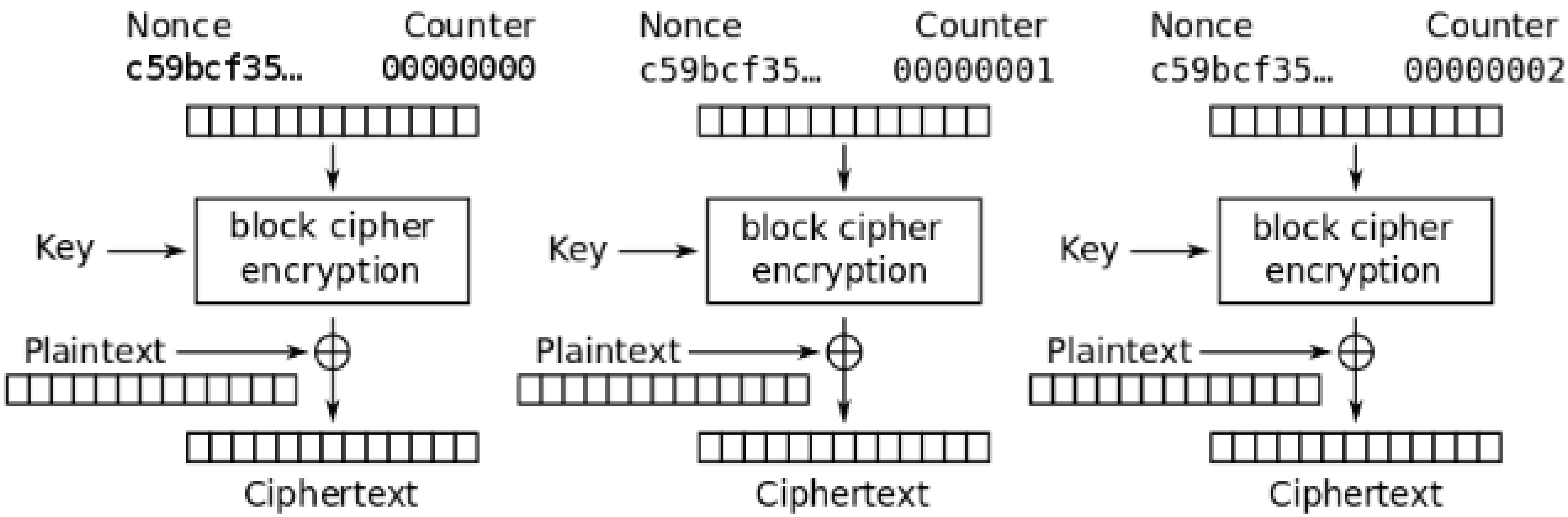
must have a different key & counter value for every plaintext block (never reused) again
uses: high-speed network encryptions

CTR



(a) Encryption

CTR



Nonce
c59bcf35…
Counter
00000000
Key
block cipher encryption
Plaintext
Ciphertext
Nonce
c59bcf35…
Counter
00000001
Key
block cipher encryption
Plaintext
Ciphertext
Nonce
c59bcf35…
Counter
00000002
Key
block cipher encryption
Plaintext
Ciphertext

Advantages and Limitations of CTR

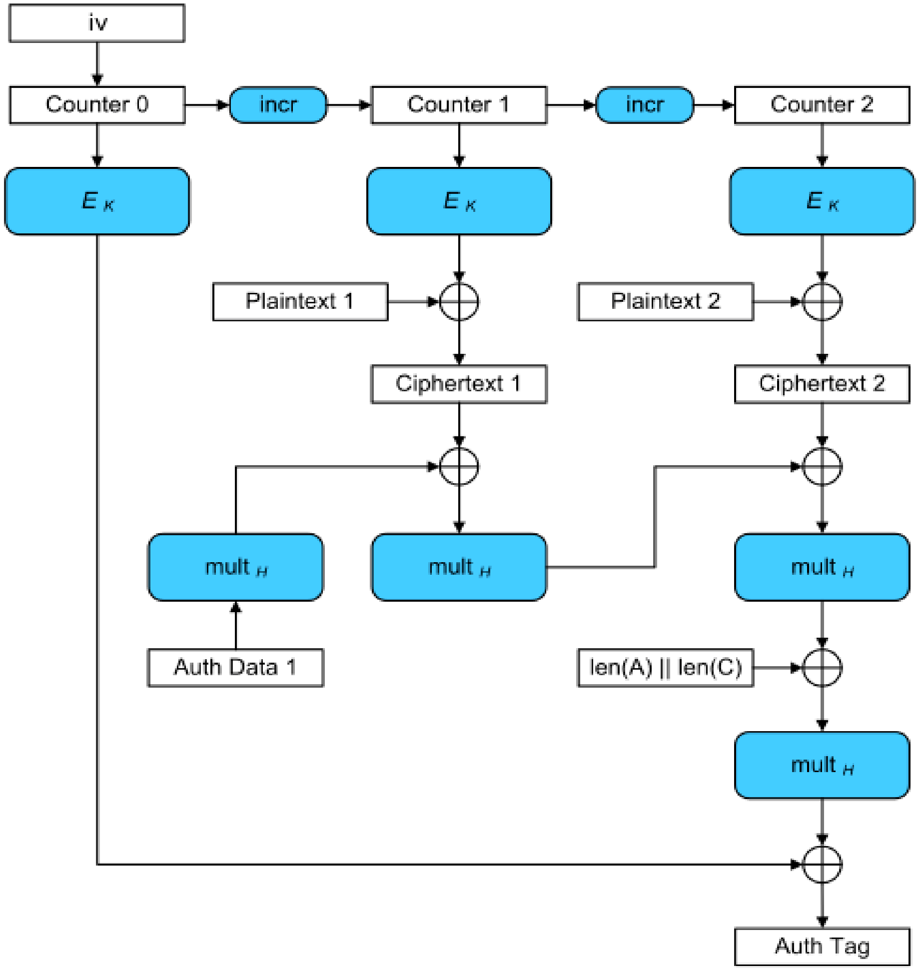
- can do parallel encryptions in h/w or s/w
- can preprocess in advance of need
- good for high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break

GCM (Galois/Counter) Block Mode

The GCM mode uses a counter, which is increased for each block and calculated a message authentication tag (MAC code) after each processed block.

The final authentication tag is calculated from the last block. Like all counter modes, GCM works as a stream cipher, and so it is essential that a different IV is used at the start for each stream that is encrypted.

The key-feature is the ease of parallel-computation of the Galois field multiplication used for authentication.



AES-GCM Authenticated Encryption

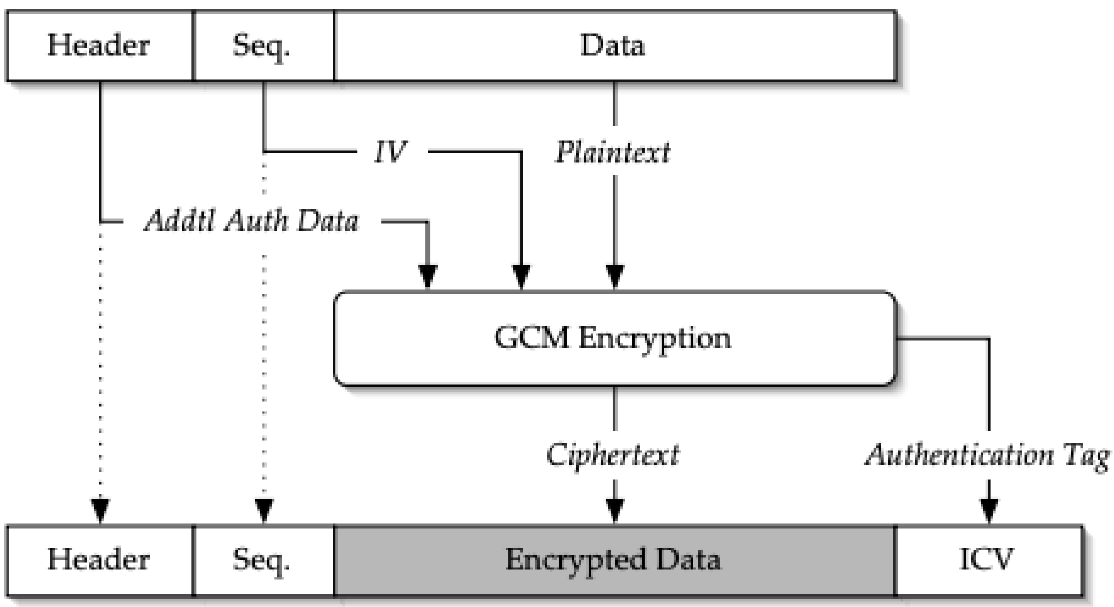
But not really the standard $GF(2^{128})$ arithmetic

- AES-GCM Authenticated Encryption (D. McGrew & J. Viega)
  - Designed for high performance (Mainly with a HW viewpoint)
  - A NIST standard FIPS 800-38D (since 2008)
    - Included in the NSA Suite B Cryptography.
- Also in:
  - IPsec (RFC 4106)
  - IEEE P1619 Security in Storage Working Group http://siswg.net/
  - TLS 1.2
- How it works:
  - Encryption is done with AES in CTR mode
  - Authentication tag computations - "Galois Hash" :
    - A Carter-Wegman-Shoup universal hash construction: polynomial evaluation over a binary field
    - Uses $GF(2^{128})$ defined by the "lowest" irreducible polynomial

$$g = g(x) = x^{128} + x^7 + x^2 + x + 1$$

  - Computations based on $GF(2^{128})$ arithmetic

AES- GCM

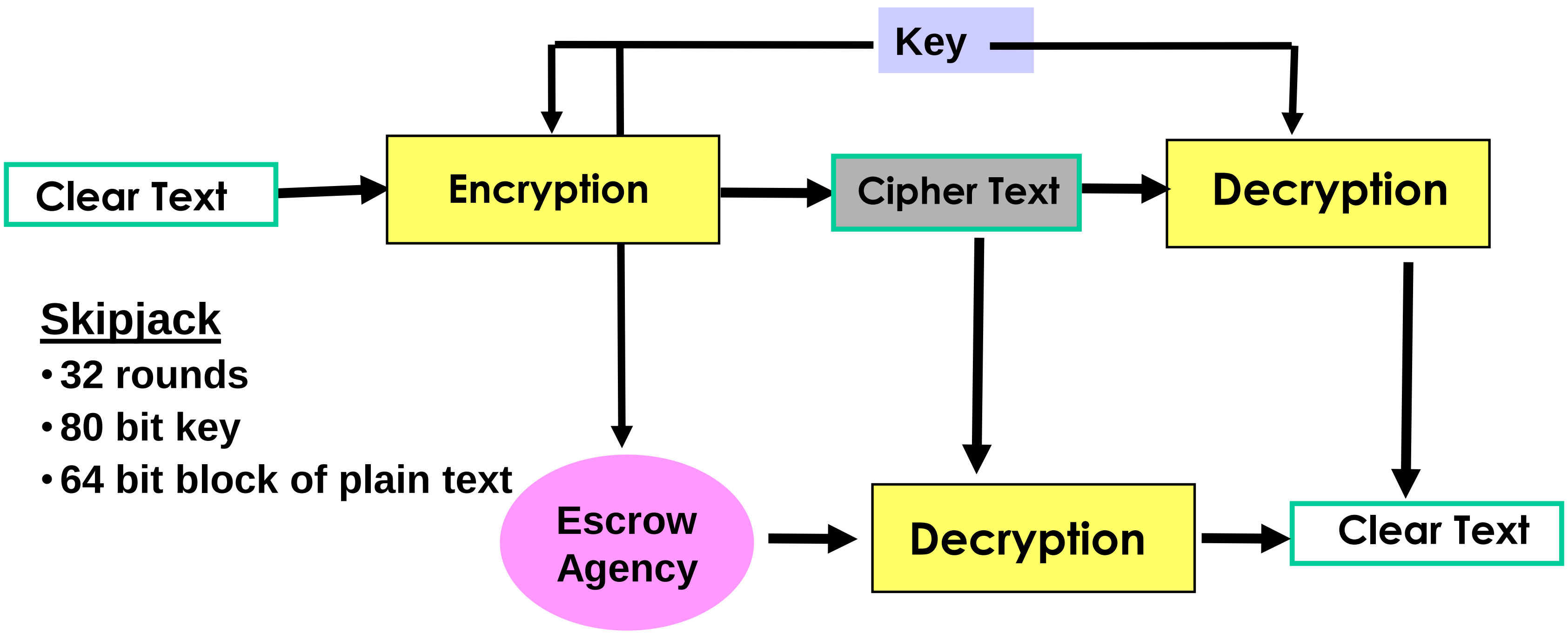


AES-GCM is the best performing Authenticated Encryption combination among the NIST standard options (esp. compared to using HMAC SHA-1)

Key Escrow

- Separate agencies maintain components of private key, which, when combined, can be used to decrypt ciphertext
- Stated reason is to decrypt drug related communications
- Clipper chip is an example
  - secret algorithm
  - Unpopular, unused
- Issues include key storage, Big Brother

Key Escrow Standard



<u>Skipjack</u>

- **32 rounds**
- **80 bit key**
- **64 bit block of plain text**

Other Symmetric Block Ciphers

- International Data Encryption Algorithm (IDEA)
  - 128-bit key
  - Used in PGP
- Blowfish
  - Easy to implement
  - High execution speed
  - Run in less than 5K of memory

Other Symmetric Block Ciphers

- RC5
  - Suitable for hardware and software
  - Fast, simple
  - Adaptable to processors of different word lengths
  - Variable number of rounds
  - Variable-length key
  - Low memory requirement
  - High security
  - Data-dependent rotations
- Cast-128
  - Key size from 40 to 128 bits
  - The round function differs from round to round

Stream Ciphers

- Process the message bit by bit (as a stream)
- Typically have a (pseudo) random **stream key**
- Combined (XOR) with plaintext bit by bit
- Randomness of **stream key** completely destroys any statistically properties in the message

  $C_i = M_i \text{ XOR } StreamKey_i$
- But must never reuse stream key

  otherwise can remove effect and recover messages

Stream Cipher Properties

Some design considerations are:

- long period with no repetitions
- statistically random
- depends on large enough key
- large linear complexity
- correlation immunity
- confusion
- diffusion
- use of highly non-linear Boolean functions

RC4

- A proprietary cipher owned by RSA DSI
- Another Ron Rivest design, simple but effective
- Variable key size, byte-oriented stream cipher
- Widely used (web SSL/TLS, wireless WEP)
- Key forms random permutation of all 8-bit values
- Uses that permutation to scramble input information processed a byte at a time

RC4 Security

- Claimed secure against known attacks
  - have some analyses, none practical
- Result is very non-linear
- Since RC4 is a stream cipher, must **never reuse a key**

Symmetric Key Applications

Communication Applications: Due to the better performance and faster speed of symmetric encryption, symmetric cryptography is typically used for encrypt the data transfer between two network endpoints. (Browser to Web Server)

Payment applications: Payment applications, such as card transactions where PII (Personal Identifying Information) needs to be protected to prevent identity theft or fraudulent charges without huge costs of resources.

Symmetric Key Applications

Protect Data at Rest

- Data at rest is data that is not actively moving from device to device or network-to-network such as data stored on a hard drive, laptop, flash drive, or archived/stored in some other way.

- Data protection at rest aims to secure inactive data stored on any device or network.

- While data at rest is sometimes considered to be less vulnerable than data in transit, attackers often find data at rest a more valuable target than data in motion.

- For protecting data at rest, enterprises can simply use symmetric key algorithms to encrypt sensitive files prior to storing them and/or choose to encrypt the storage drive itself.

Advantages and Disadvantages

Advantages

Algorithms are fast

- Encryption & decryption are handled by same key
- As long as the key remains secret, the system also provide authentication

Disadvantages

Key is revealed, the interceptors can decrypt all encrypted information

- Key distribution problem
- Number of keys increases with the square of the number of people exchanging secret information

Key Distribution Issue

Symmetric key cryptography: Alice and Bob share a common secret key.

Some means of distributing a copy of the secret key via the same network is an issue.

Physical distribution is not possible in huge open networks.

A better solution to the key distribution problem is obtained if we use symmetric key distribution protocols.

Scalability Issue

Assume that n users are connected in a network and any two of them may want to communicate

This would require each user to securely store $n - 1$ different

symmetric keys (one for each other user), resulting in a total of

$n(n - 1)/2$ keys.

If the network is connecting 2000 university students, then there will be roughly 2 million different keys.

A huge key management problem with questions like; How do you add add a new user to the system?

What if a user's key is compromised?

How long should a key be considered valid and how should we refresh them?

Thank You