Bachelor of Computer Science

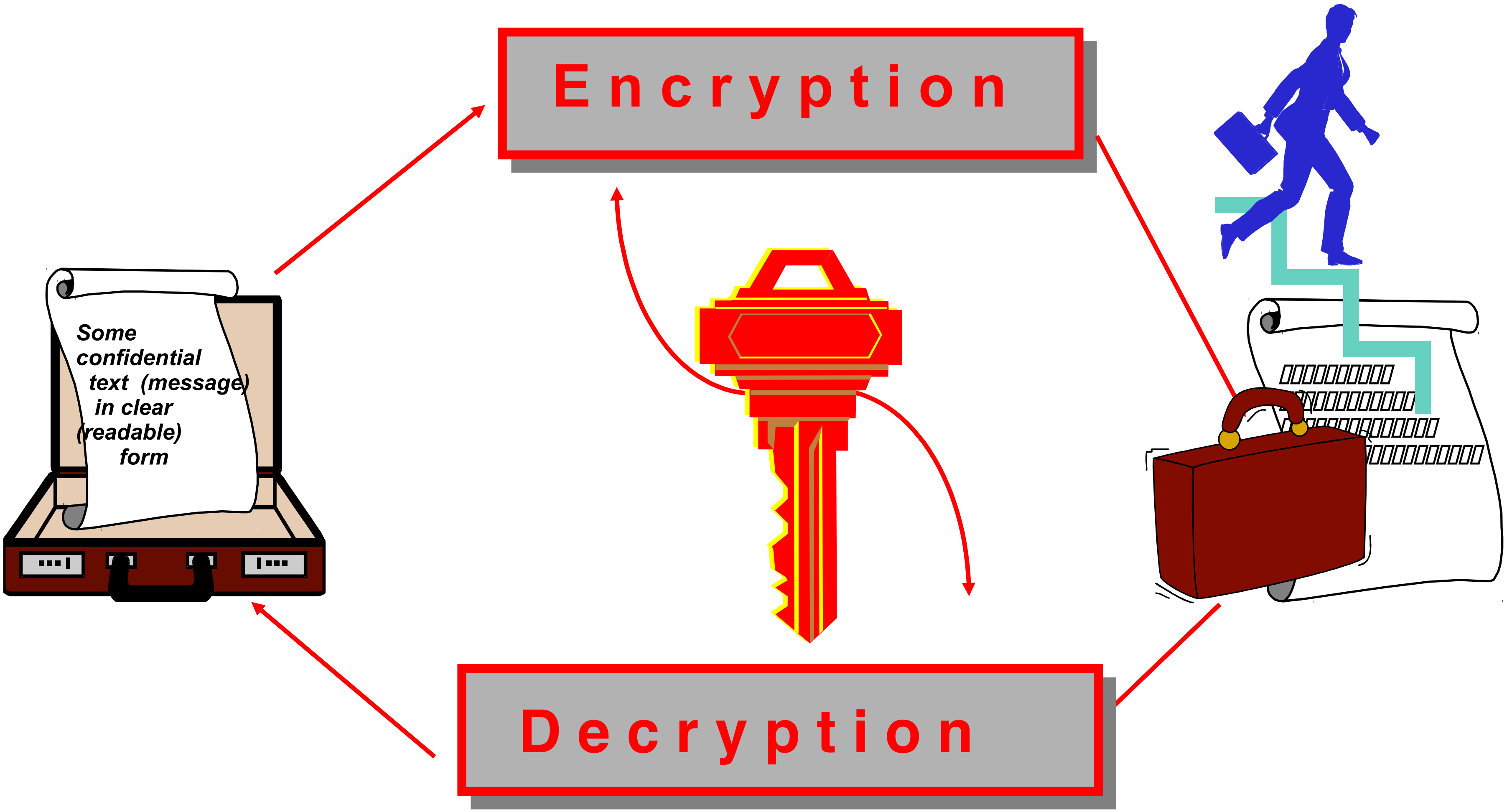
SCS2214 - Information System Security

Handout 3 - Symmetric Key Encryption

Kasun de Zoysa
kasun@ucsc.cmb.ac.lk

UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING

Symmetric key Cryptograms



The classic cryptography

- **Encryption algorithm and related key are kept secret.**
- **Breaking the system is hard due to large numbers of possible keys.**
- **For example: for a key 128 bits long**
- **there are $2^{128} \approx 10^{38}$ keys to check using brute force.**

The fundamental difficulty is key distribution to parties who want to exchange messages.

Symmetric Key / Private 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

Data Encryption Standard (DES)

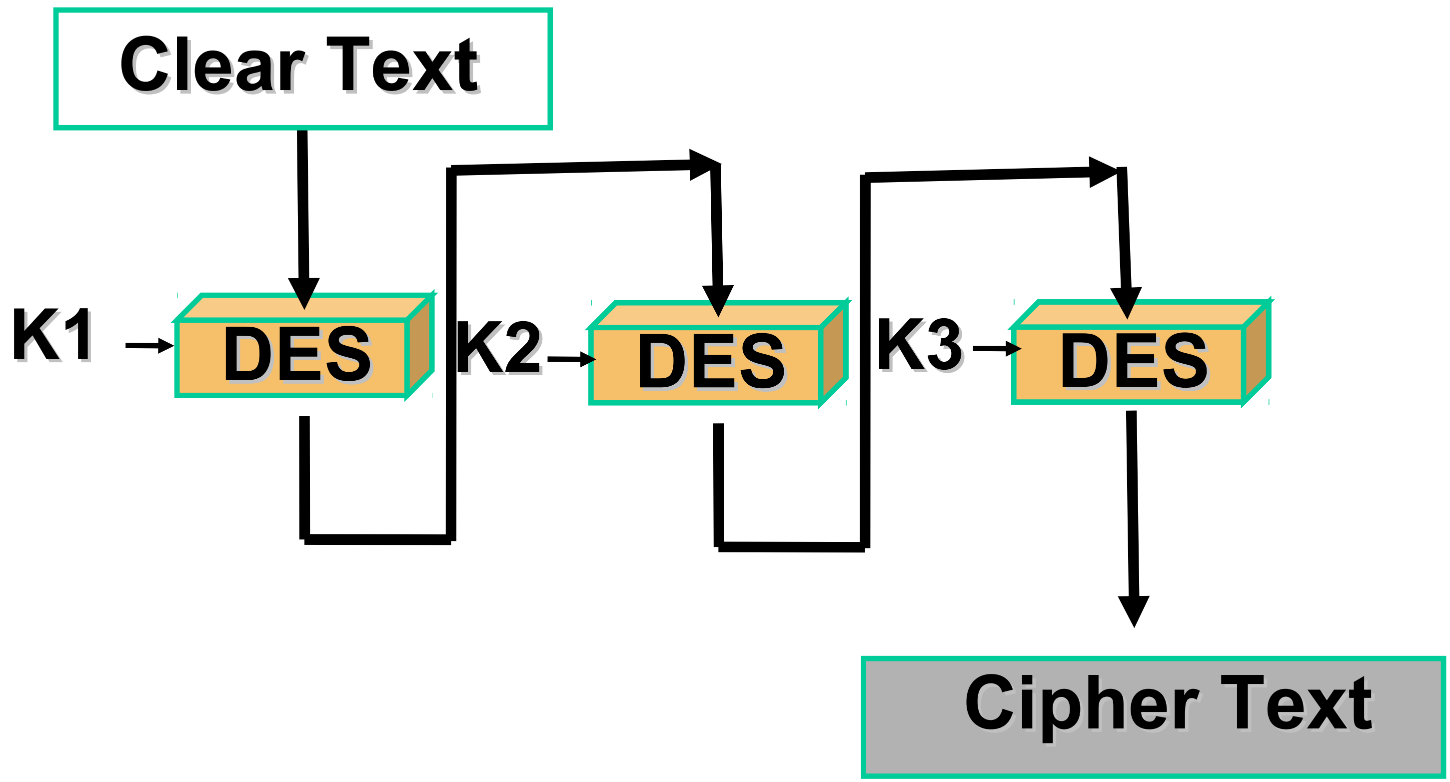
- Most widely used block cipher in world
- Adopted in 1977 by NBS (now NIST) as FIPS PUB 46
- Encrypts 64-bit data using 56-bit key
- Has widespread use
- Has been the subject of considerable controversy over its security

DES – 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

Triple DES



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

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

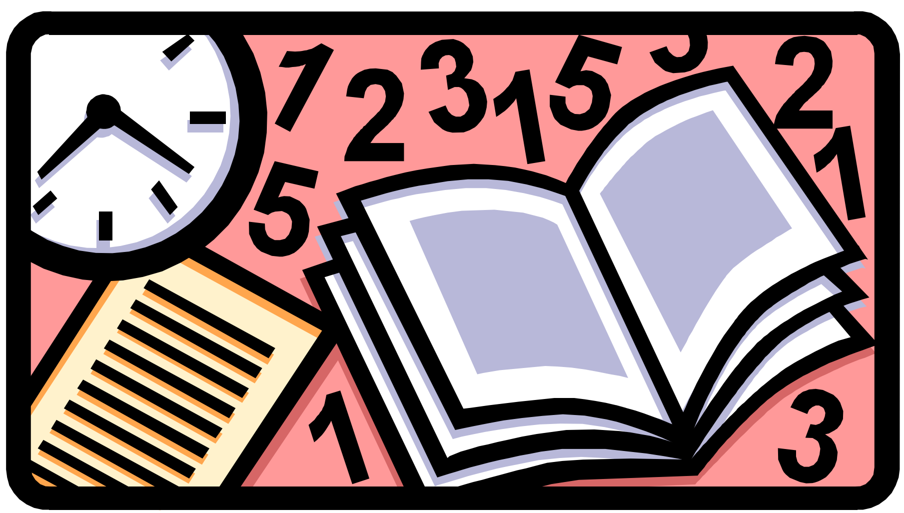
- Private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger & faster than Triple-DES
- Active life of 20-30 years (+ archival use)
- Provide full specification & design details
- Both C & Java implementations
- NIST has released all submissions & unclassified analyses

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
- Then subject to further analysis & comment
- Saw contrast between algorithms with
  - few complex rounds verses many simple rounds
  - which refined existing ciphers verses new proposals

Advance 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

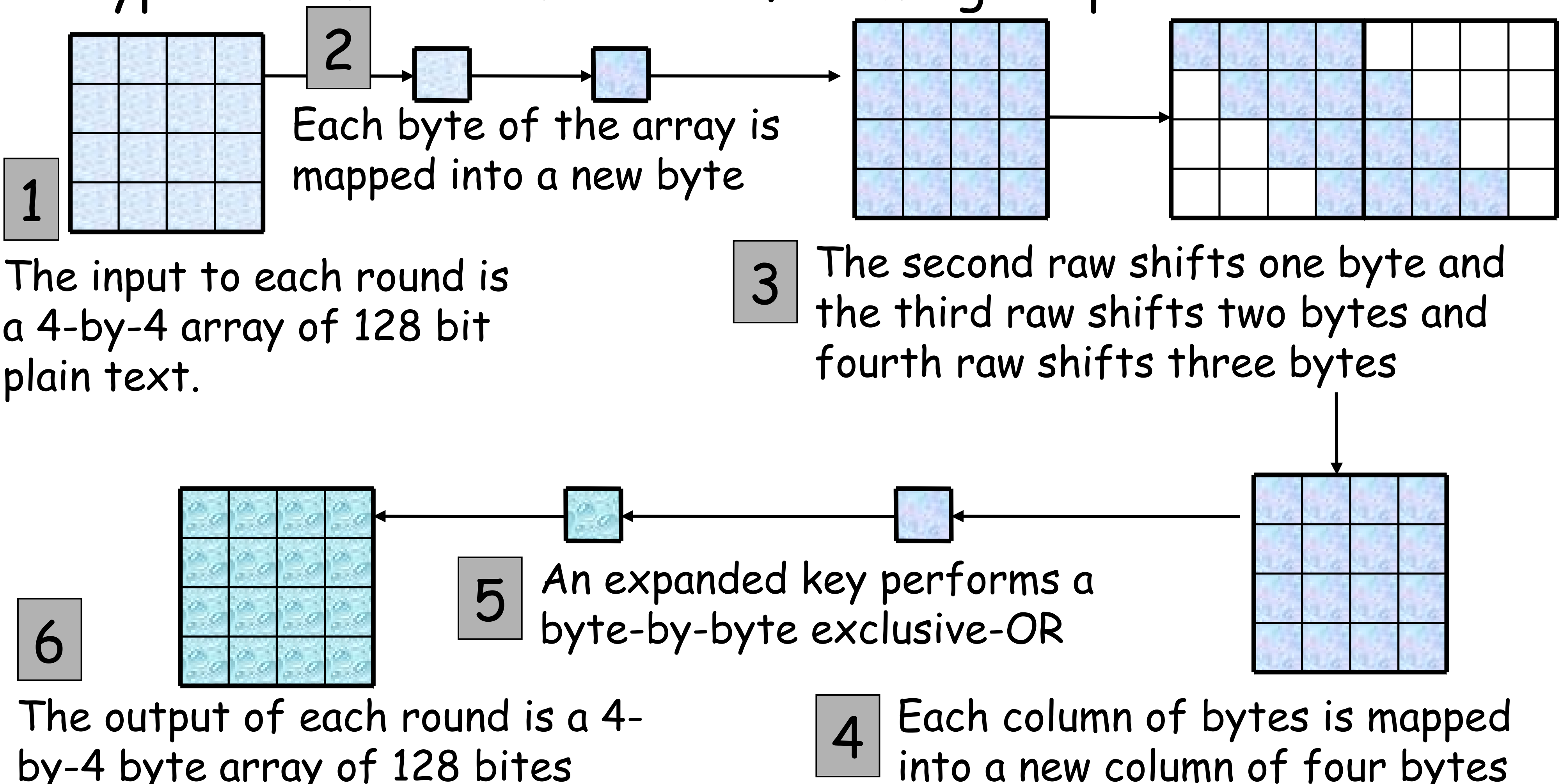


Advance Encryption Standard (AES)

- 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

Advance Encryption Standard (AES)

A Typical round includes the following steps



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

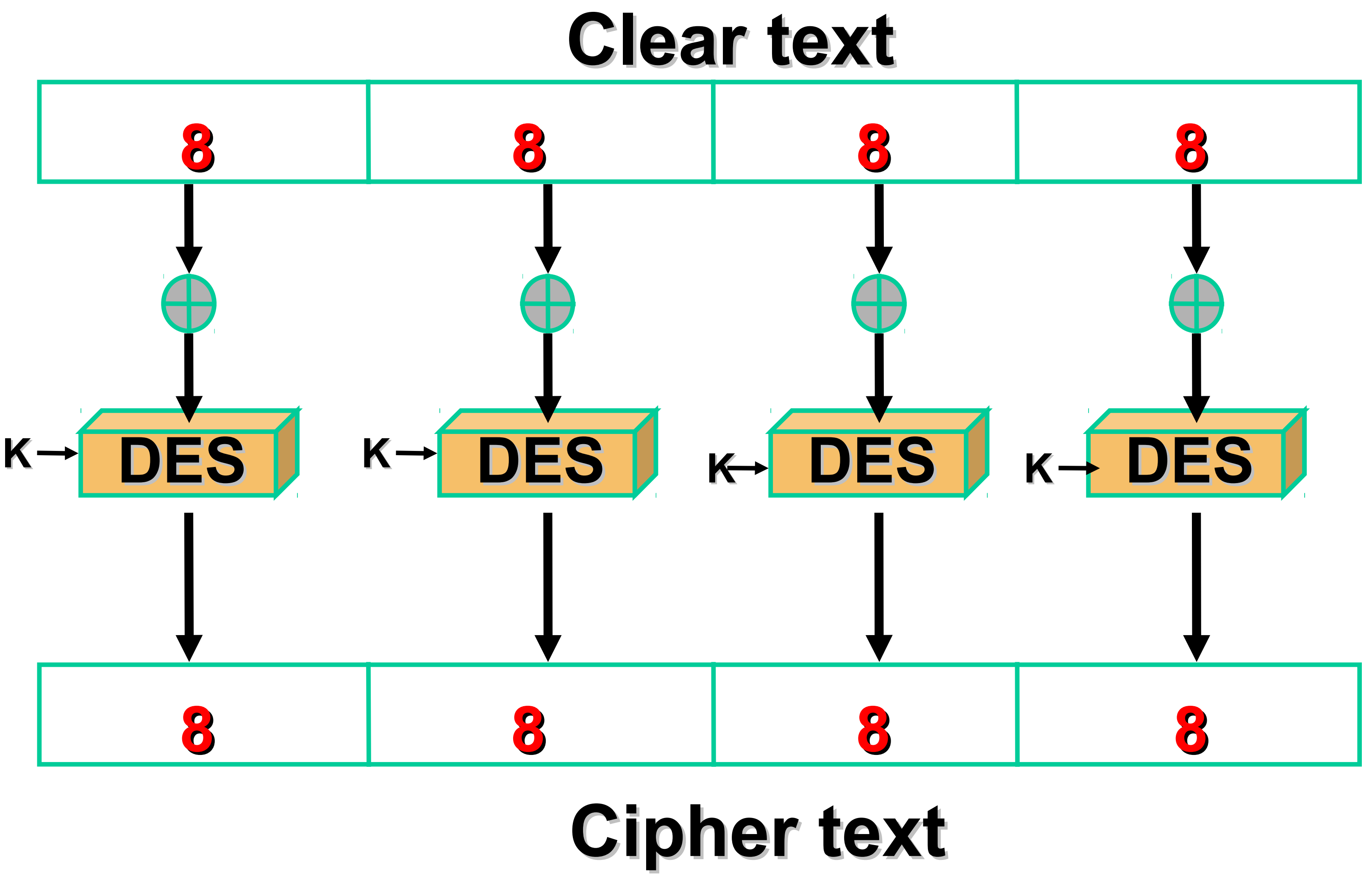
Electronic Codebook Book (ECB)

- 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

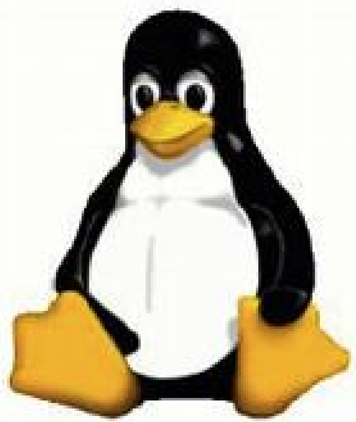
Electronic Code Book Mode (ECB)



Advantages and Limitations of ECB

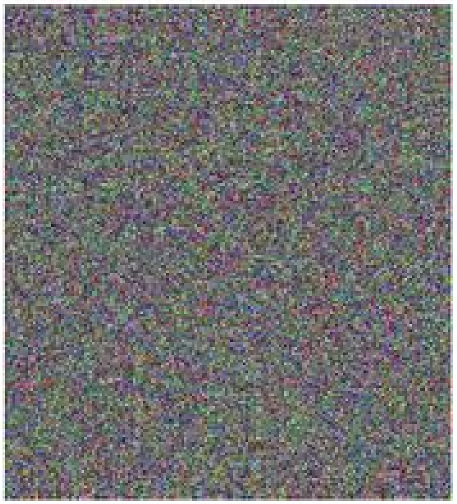
- Repetitions in message may show in ciphertext if aligned with message block particularly with data such graphics or with
- Messages that change very little
- Weakness due to encrypted message blocks being independent
- Main use is sending a few blocks of data

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)

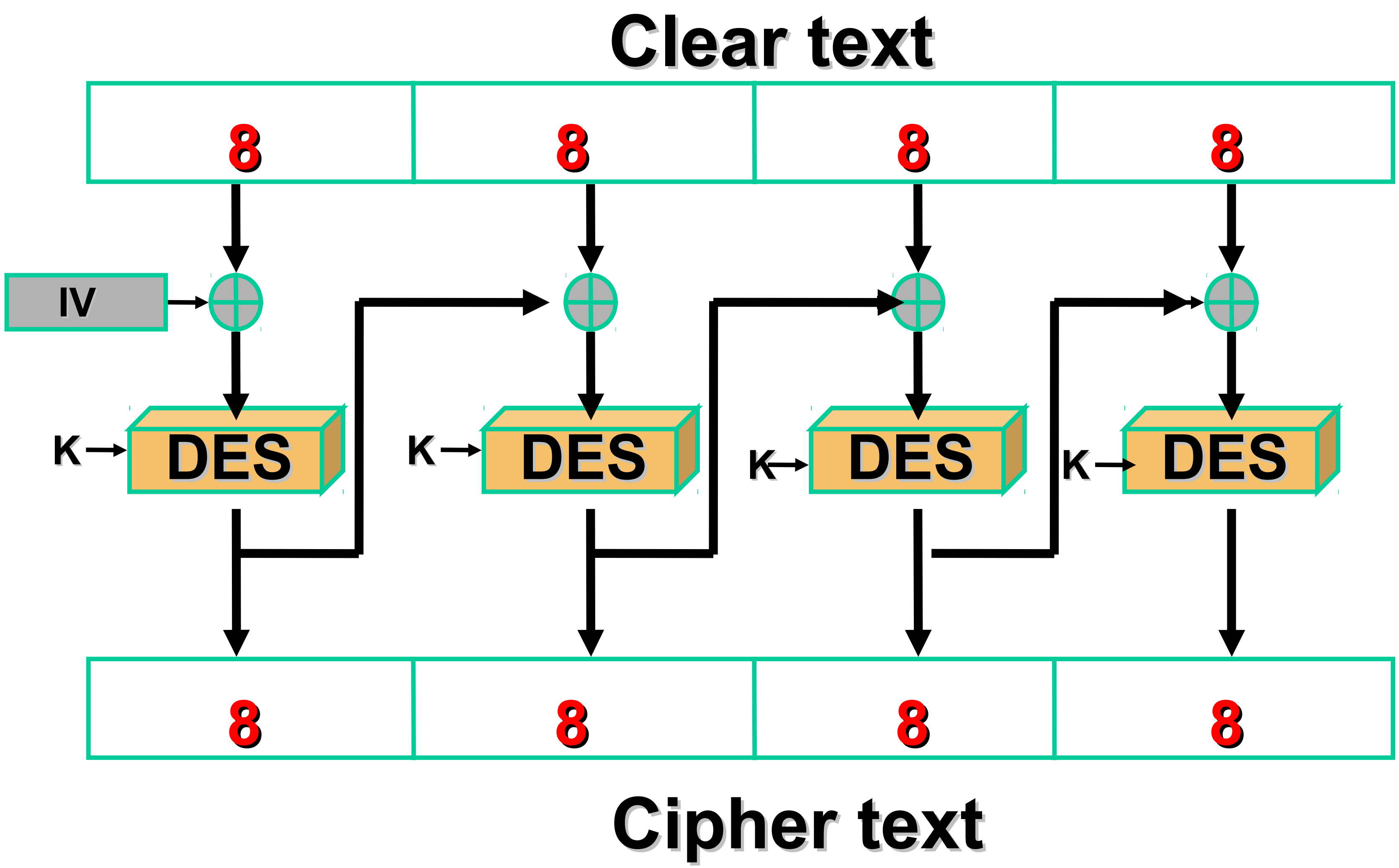
- Message is broken into blocks
- But these are linked together in the encryption operation
- Each previous cipher blocks is chained with current plaintext block, hence name
- Use Initial Vector (IV) to start process

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

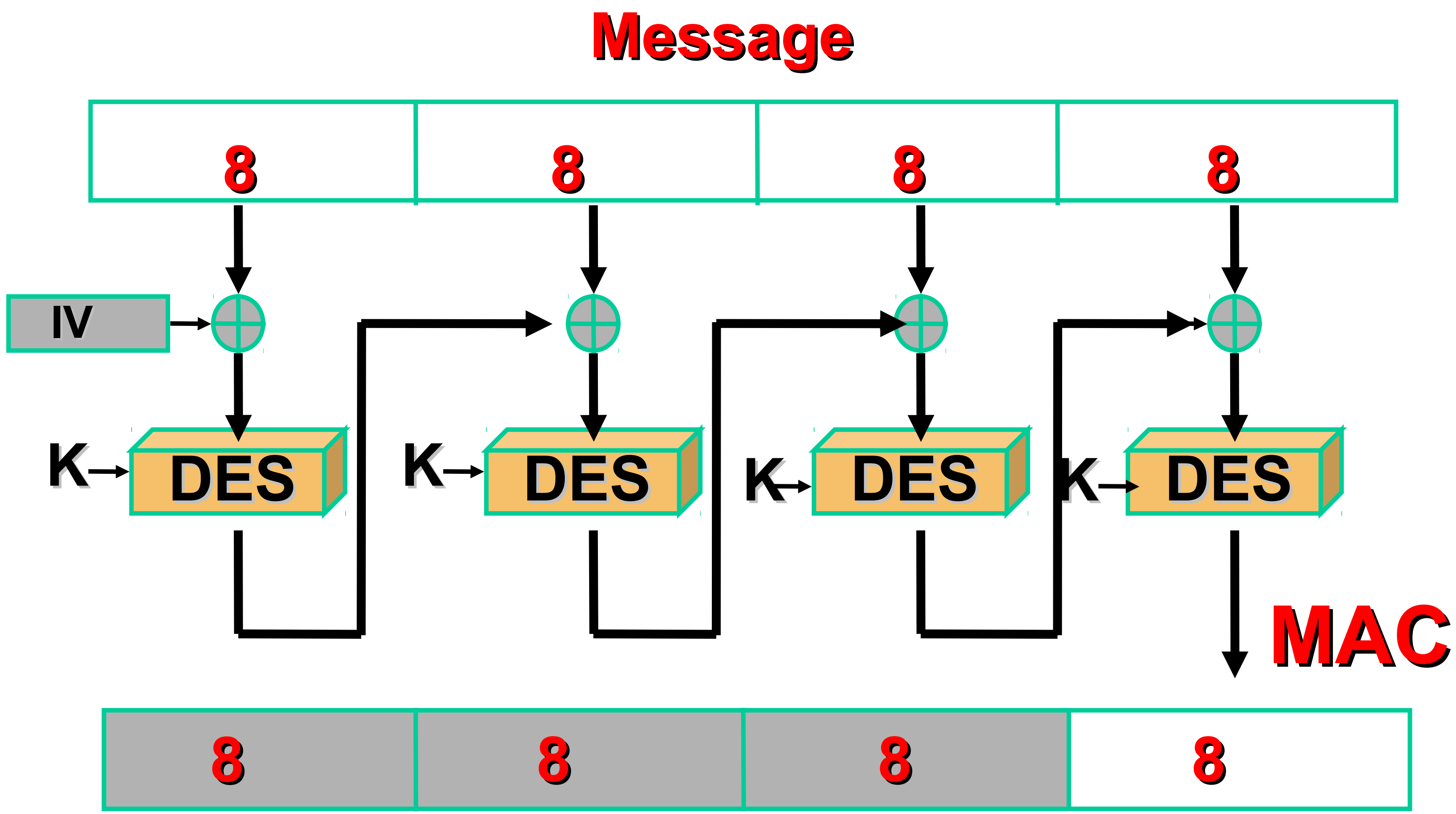
$$C_{-1} = IV$$

- 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

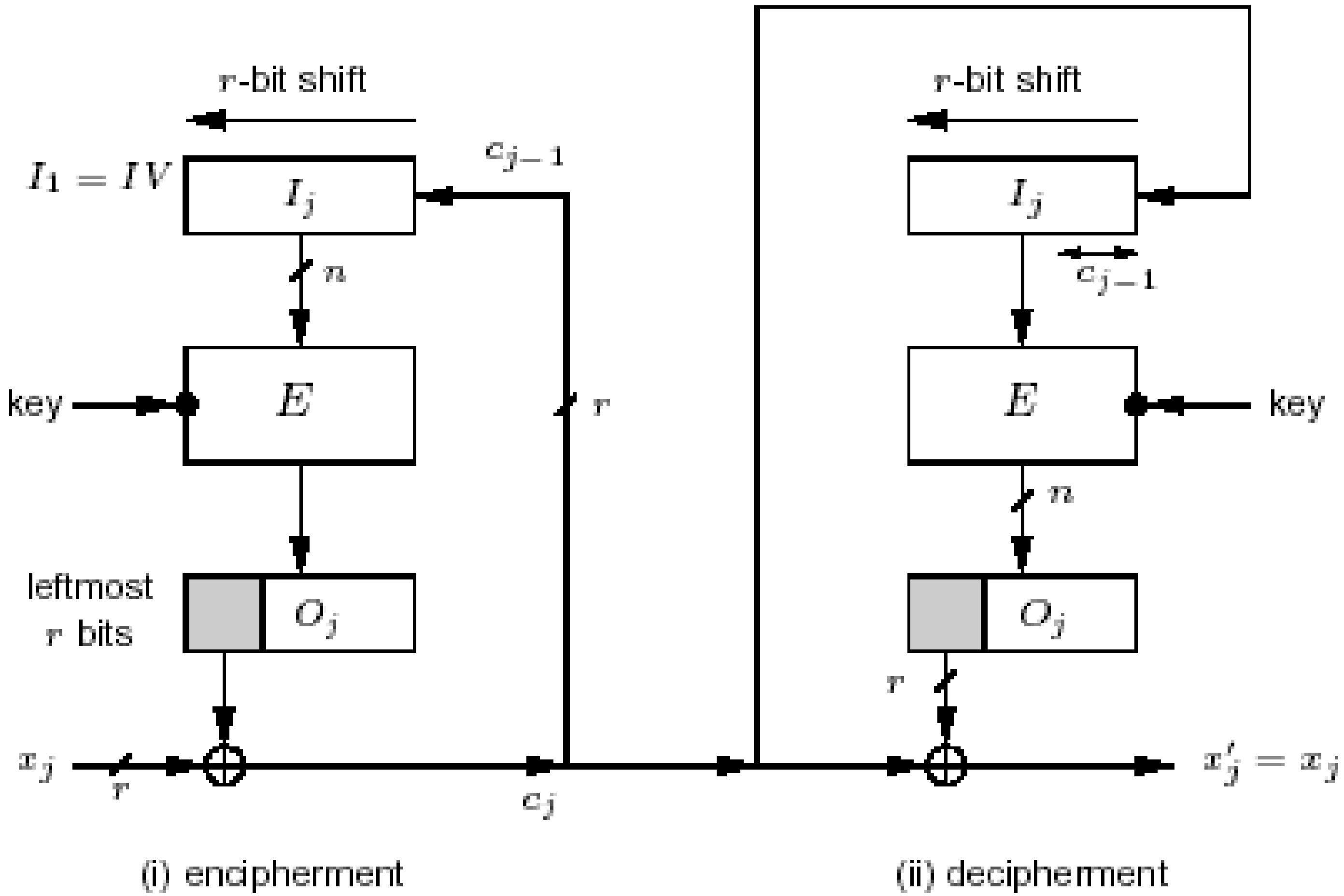
- Need **Initial Value** (IV) known to sender & receiver however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate hence either IV must be a fixed value or it must be sent encrypted in ECB mode before rest of message

- 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 feed back (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

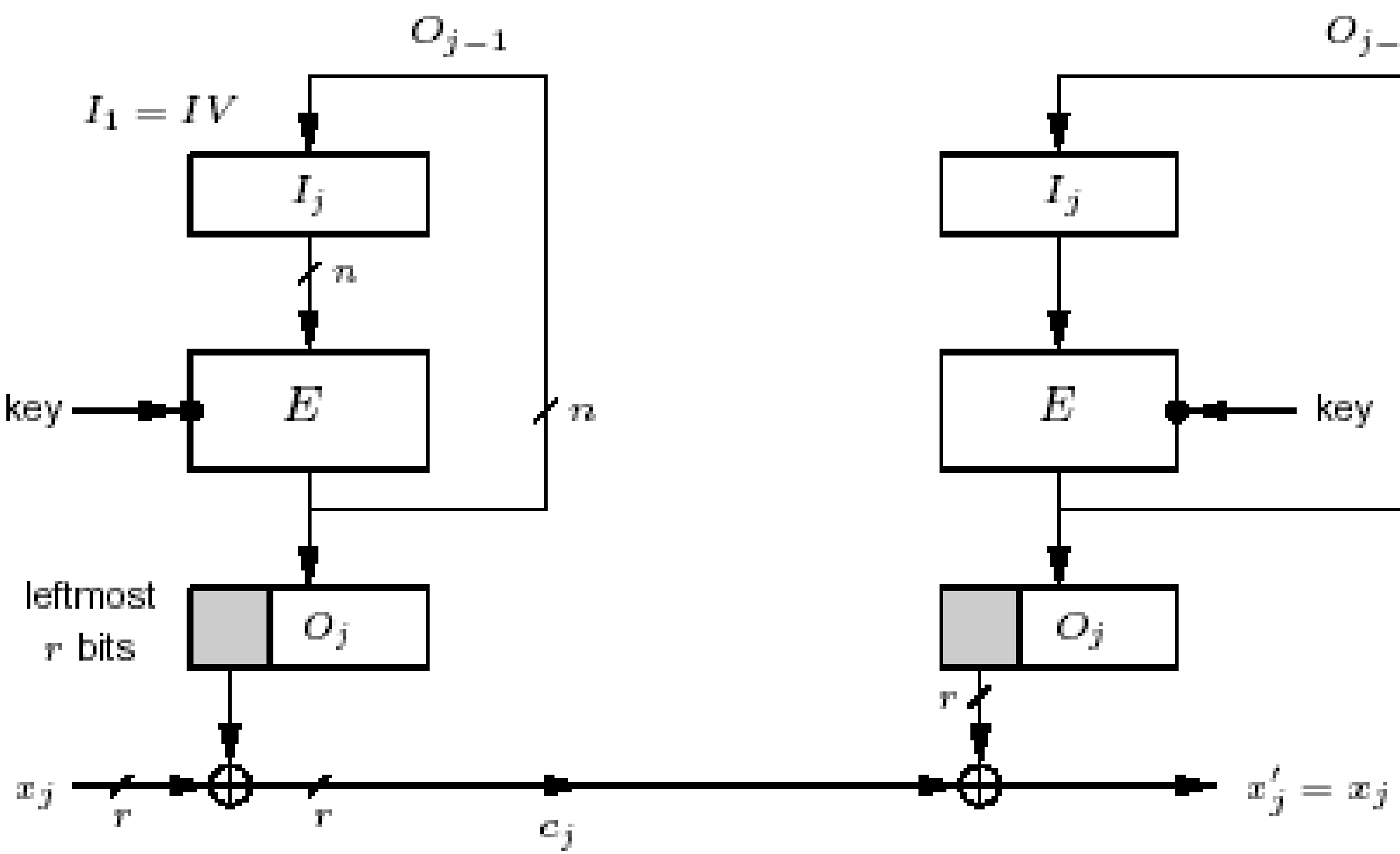
Cipher Feedback Mode (CFB)



Output Feed Back(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)



(i) encipherment

(ii) decipherment

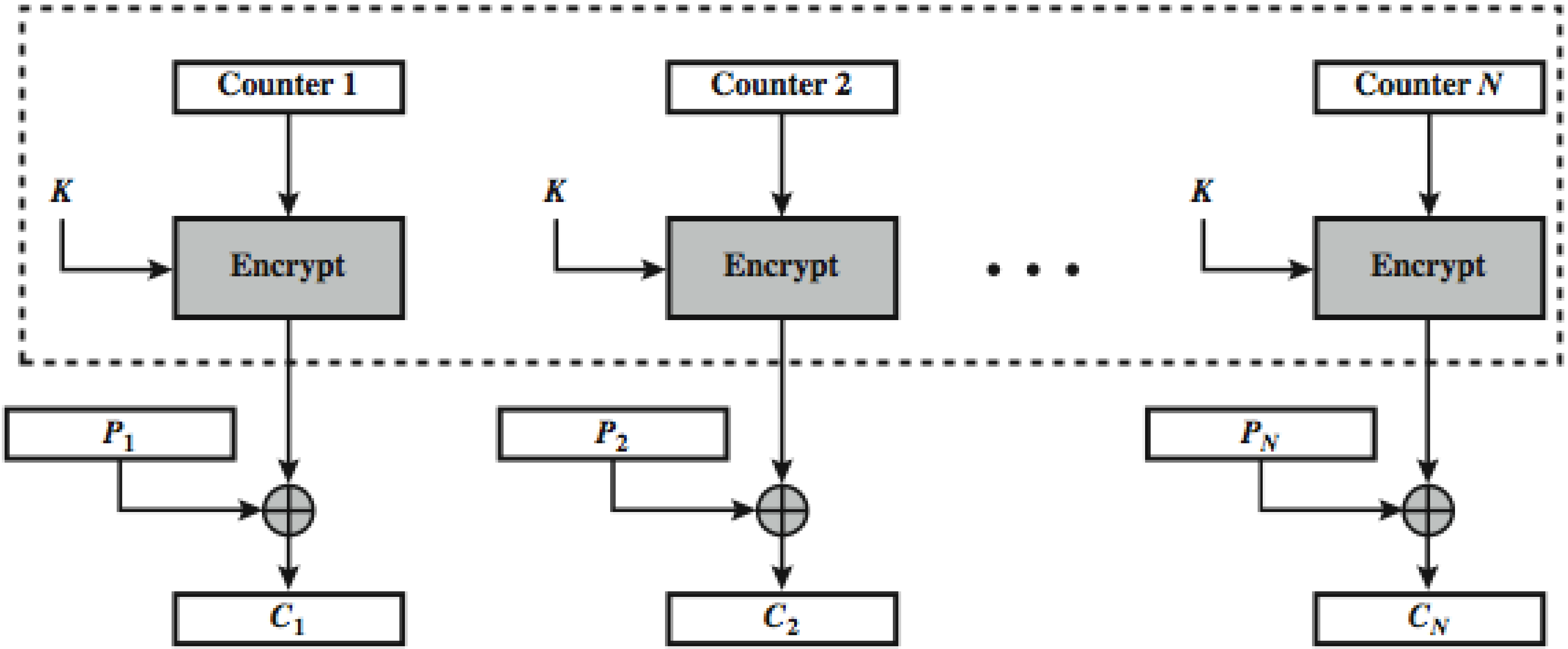
Counter (CTR)

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

Oi = EK(i)
Ci = Pi XOR Oi

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

CTR



(a) Encryption

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

OpenSSL

 # encrypt file.txt to file.enc using 256-bit AES in CBC mode

>openssl enc -aes-256-cbc -in file.txt -out file.enc

 # decrypt binary file.enc

>openssl enc -d -aes-256-cbc -in file.enc

 # see the list under the 'Cipher commands' heading

>openssl -h

Random Number Generator (RNG)

The SecureRandom class is an engine class that provides the functionality of a Random Number Generator (RNG). It differs from the Random class in that it produces cryptographically strong random numbers.

Random Number Generator (RNG)

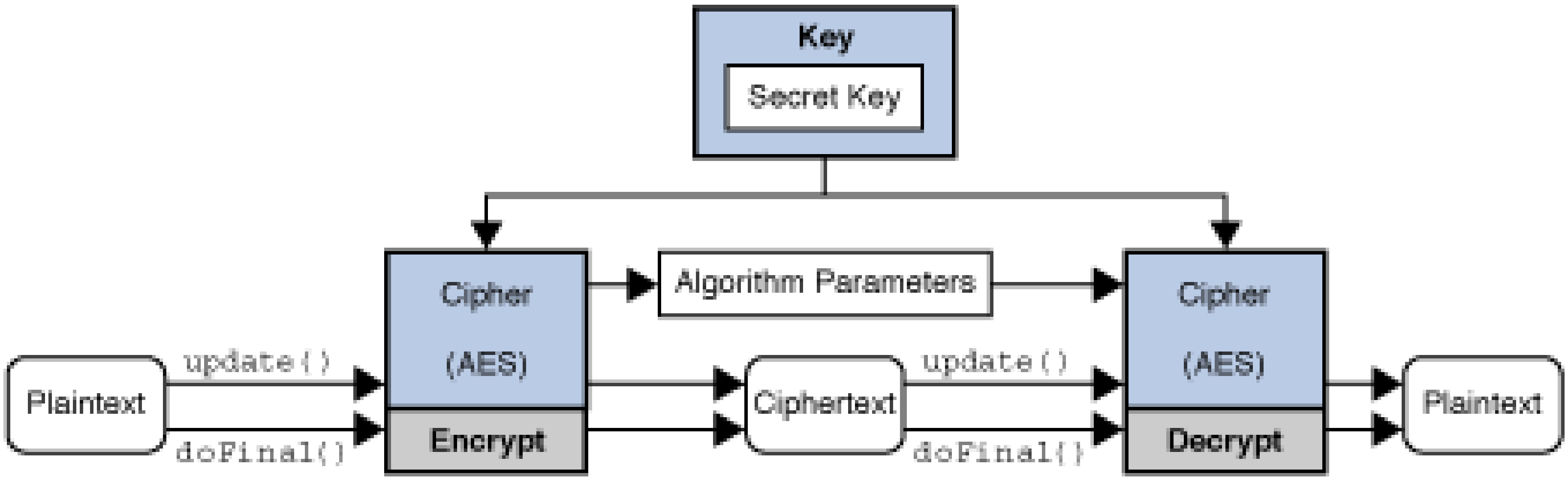
```
//Initialize secure random generator
 SecureRandom sr =   SecureRandom.getInstance("SHA1PRNG");

//Generate and set seed value
   int seedByteCount = 10;
   byte[] seed = sr.generateSeed(seedByteCount);
   sr.setSeed(seed);

// Get 256 random bits
   byte[] bytes = new byte[256/8];
   sr.nextBytes(bytes);

// Get next 256 random bits
   sr.nextBytes(bytes);
```

Encryption: AES ECB



AES-ECB Encryption

Encryption:

1. Key Generation

```
KeyGenerator    generator = KeyGenerator.getInstance("AES");
 generator.init(128);
 Key key = generator.generateKey();
```

2. Obtain the cipher engine

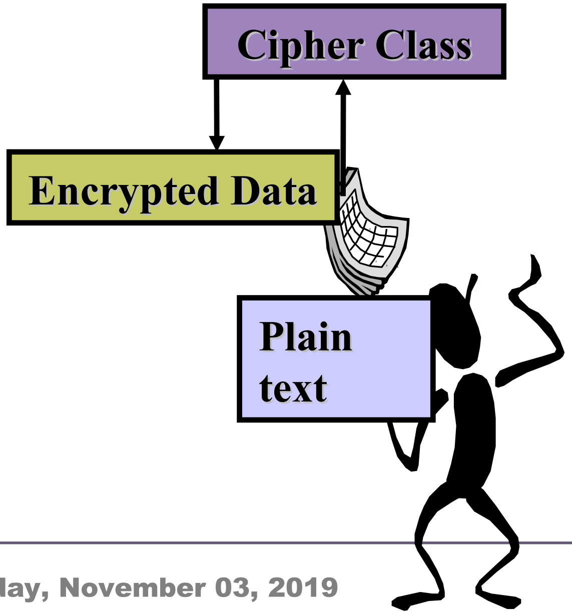
```
Cipher c = Cipher.getInstance(“AES/ECB/PKCS5Padding”)
```

3. Initializing the cipher engine for encryption

```
c.int(Cipher.ENCRYPT_MODE, key)
```

4. Do the padding and finish the encryption

```
byte[] cipherText = c.doFinal(input);
```



AES-ECB Decryption

Decryption:

1. Generation the same key

```
KeyGenerator    generator = KeyGenerator.getInstance("AES");
 generator.init(128);
 Key key = generator.generateKey();
```

2. Obtain the cipher engine

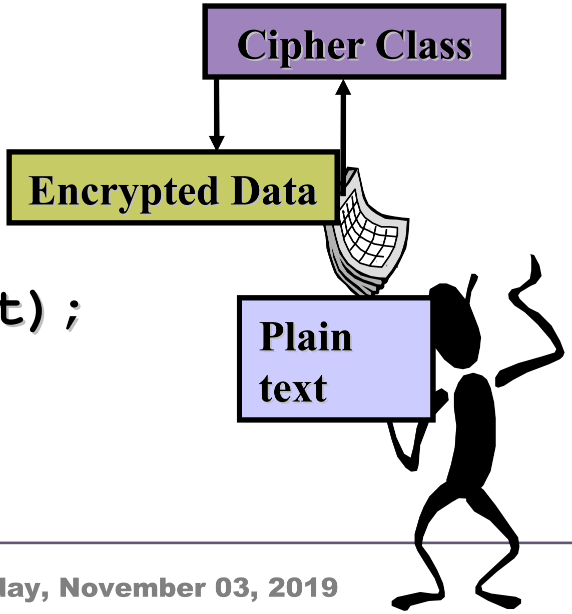
```
Cipher c = Cipher.getInstance(“AES/ECB/PKCS5Padding”)
```

3. Initializing the cipher engine for decryption

```
c.int(Cipher.DECRYPT_MODE, key)
```

4. Remove the padding and finish the decryption

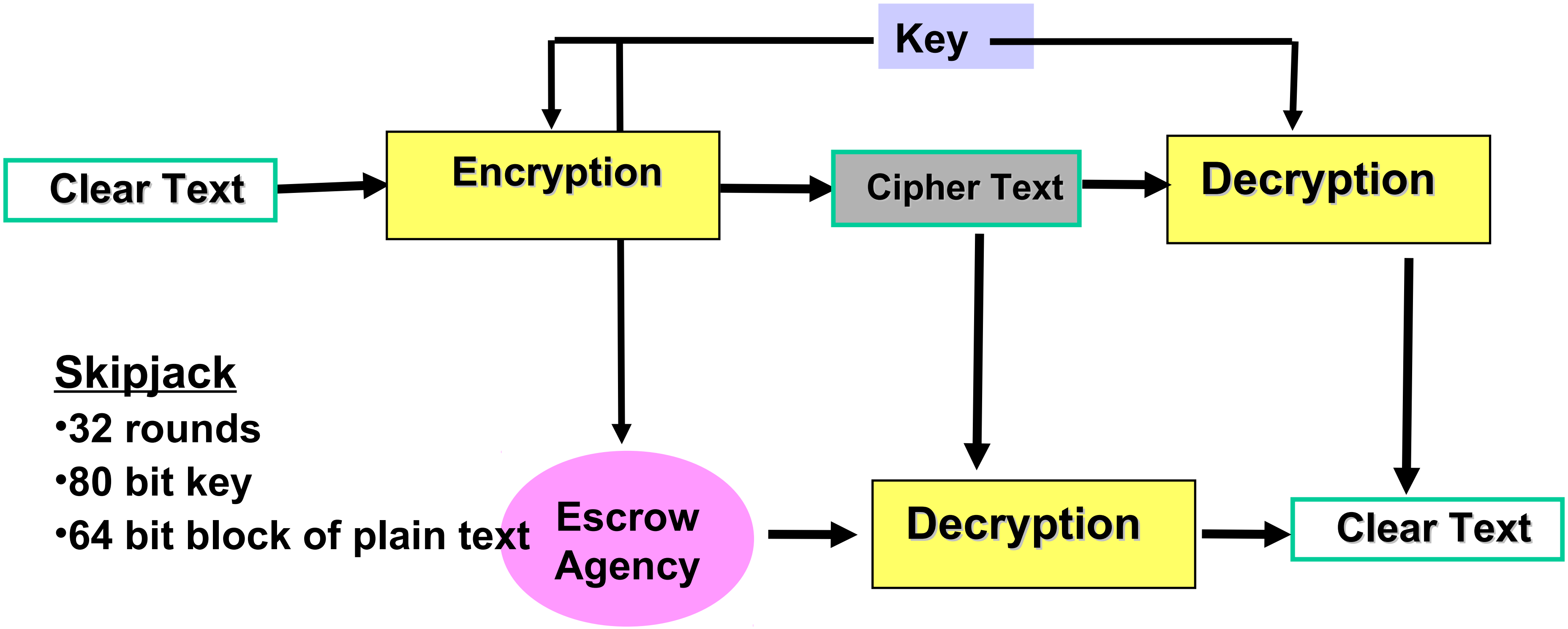
```
byte[] plainText = c.doFinal(cipherTest);
```



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

  $C_i = M_i$ XOR $StreamKey_i$

- But must never reuse stream key

  otherwise can remove effect and recover messages

Stream Cipher Properties

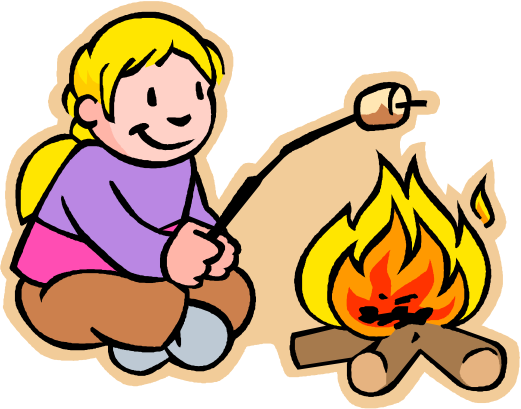
- Some design considerations are:
  - long period with no repetitions
  - statistically random
  - depends on large enough key
  - large linear complexity
  - correlation immunity
  - confusion
  - diffusion
  - use of highly non-linear Boolean functions

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**



Advantages & Disadvantages

<u>Advantages</u>

Algorithms are fast

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

<u>Disadvantages</u>

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***

Discussion