

# Network Security

Week 4

1

- Most slides are taken from
  - Cryptography and Network Security, by William Stallings, Lecture slides by Lawrie Brown

2

## Message Padding

- at end of message must handle a possible last short block
  - which is not as large as block size of cipher
  - pad either with known non-data value (eg nulls)
  - or pad last block along with count of pad size
    - eg. [ b1 b2 b3 0 0 0 0 5]
    - means have 3 data bytes, then 5 bytes pad+count
  - this may require an extra entire block over those in message
- there are other, more esoteric modes, which avoid the need for an extra block

3

## Modes of Operation

- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks with 56-bit key
- need some way to en/decrypt arbitrary amounts of data in practise
- **ANSI X3.106-1983 Modes of Use** (now FIPS 81) defines 4 possible modes
- subsequently 5 defined for AES & DES
- have **block** and **stream** modes

4

## Electronic Codebook Book (ECB)

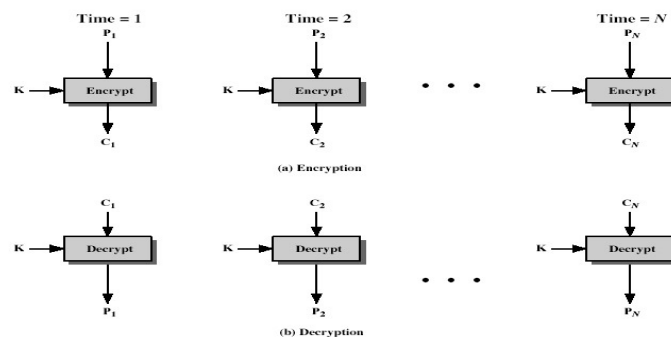
- message is broken into independent blocks which are encrypted
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks

$$C_i = \text{DES}_{K1}(P_i)$$

- uses: secure transmission of single values

5

## Electronic Code Book (ECB)



6

## Advantages and Limitations of ECB

- message repetitions may show in ciphertext
  - if aligned with message block
  - particularly with data such graphics
  - or with messages that change very little, which become a code-book analysis problem
- weakness is due to the encrypted message blocks being independent
- main use is sending a few blocks of data

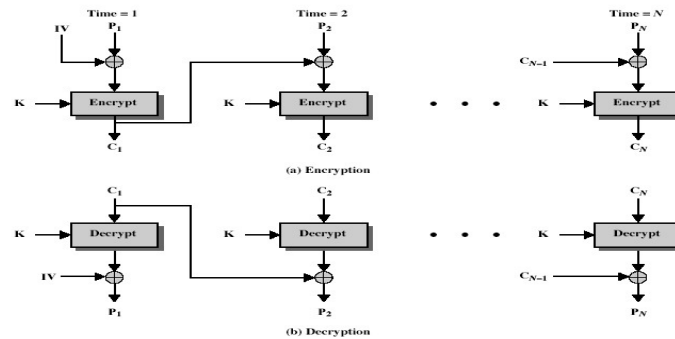
7

## Cipher Block Chaining (CBC)

- message is broken into blocks
- linked together in encryption operation
- each previous cipher blocks is chained with current plaintext block, hence name
- use Initial Vector (IV) to start process
$$C_i = \text{DES}_{K1}(P_i \text{ XOR } C_{i-1})$$
$$C_{-1} = \text{IV}$$
- uses: bulk data encryption, authentication

8

## Cipher Block Chaining (CBC)



9

## Advantages and Limitations of CBC

- a ciphertext block depends on **all** blocks before it
- any change to a block affects all following ciphertext blocks
- need **Initialization Vector (IV)**
  - which must be known to sender & receiver
  - if sent in clear, attacker can change bits of first block, and change IV to compensate
  - hence IV must either be a fixed value (as in EFTPOS)
  - or must be sent encrypted in ECB mode before rest of message

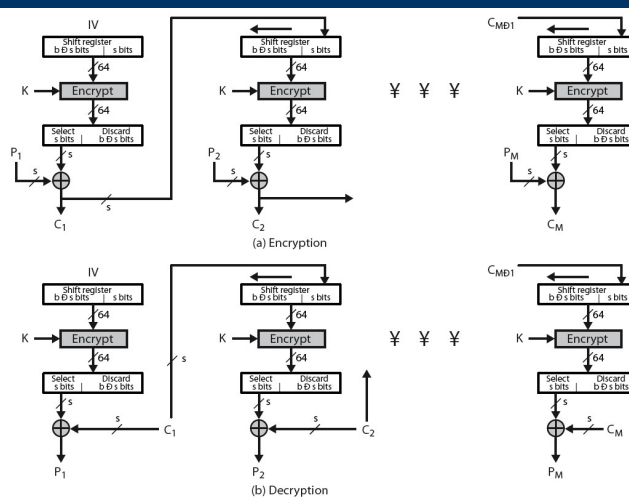
10

## Cipher FeedBack (CFB)

- message is treated as a stream of bits
- added to the output of the block cipher
- result is feed back for next stage (hence name)
- standard allows any number of bit (1, 8, 64 or 128 etc) to be feed back
  - denoted CFB-1, CFB-8, CFB-64, CFB-128 etc
- most efficient to use all bits in block (64 or 128)
  - $C_i = P_i \text{ XOR } \text{DES}_{K1}(C_{i-1})$
  - $C_{-1} = \text{IV}$
- uses: stream data encryption, authentication

11

## Cipher FeedBack (CFB)



12

## Advantages and Limitations of CFB

- appropriate when data arrives in bits/bytes
- most common stream mode
- limitation is need to stall while do block encryption after every n-bits
- note that the block cipher is used in **encryption** mode at **both** ends
- errors propagate for several blocks after the error

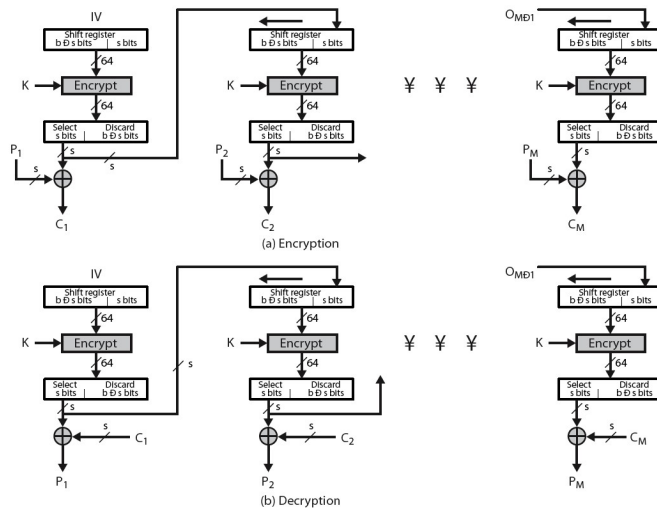
13

## Output FeedBack (OFB)

- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance
$$C_i = P_i \text{ XOR } O_i$$
$$O_i = \text{DES}_{K1}(O_{i-1})$$
$$O_{-1} = \text{IV}$$
- uses: stream encryption on noisy channels

14

## Output FeedBack (OFB)



15

## Advantages and Limitations of OFB

- bit errors do not propagate
- more vulnerable to message stream modification
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
- sender & receiver must remain in sync
- originally specified with m-bit feedback
- subsequent research has shown that only **full block feedback** (ie OFB-64 or OFB-128) should ever be used

16



## Counter (CTR)

- a “new” mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

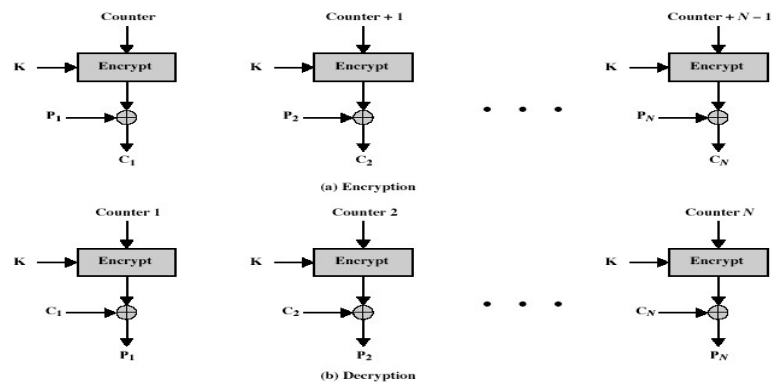
$$C_i = P_i \text{ XOR } O_i$$

$$O_i = \text{DES}_{K1}(i)$$

- uses: high-speed network encryptions

17

## Counter (CTR)



18

## Advantages and Limitations of CTR

- efficiency
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance of need (output of the encryption boxes)
  - good for bursty 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

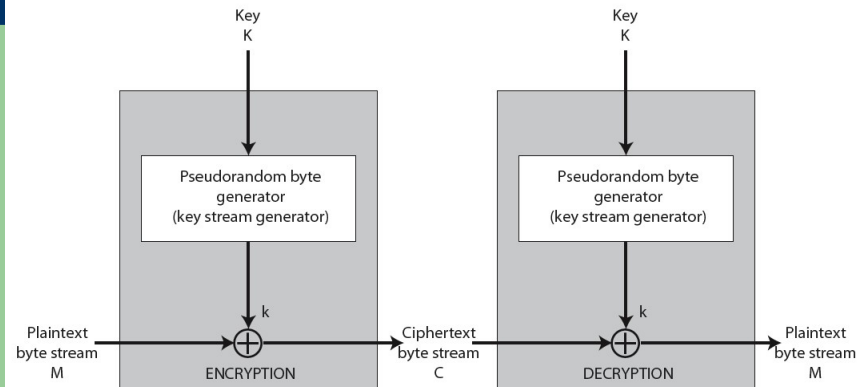
19

## Stream Ciphers

- process message bit by bit (as a stream)
- have a pseudo random **keystream**
- combined (XOR) with plaintext bit by bit
- randomness of **stream key** completely destroys statistically properties in message
  - $C_i = M_i \text{ XOR } \text{StreamKey}_i$
- but must never reuse stream key
  - otherwise can recover messages (cf book cipher)

20

## Stream Cipher Structure



21

## Stream Cipher Properties

- some design considerations are:
  - long period with no repetitions
  - statistically random
  - depends on large enough key
  - large linear complexity
- properly designed, can be as secure as a block cipher with same size key
- but usually simpler & faster

22

## 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 info processed a byte at a time

23

## RC4 Key Schedule

- starts with an array S of numbers: 0..255
- use key to well and truly shuffle
- S forms **internal state** of the cipher

```
for i = 0 to 255 do
  S[i] = i
  T[i] = K[i mod keylen])
j = 0
for i = 0 to 255 do
  j = (j + S[i] + T[i]) (mod 256)
  swap (S[i], S[j])
```

24

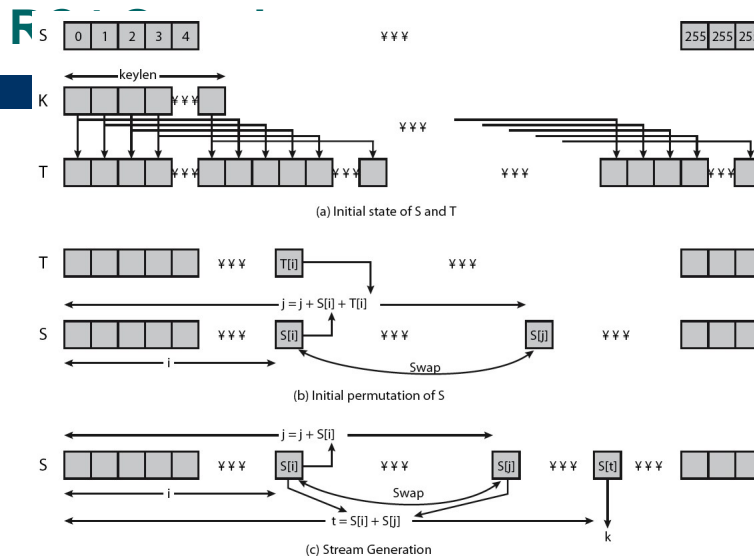
## RC4 Encryption

- encryption continues shuffling array values
- sum of shuffled pair selects "stream key" value from permutation
- XOR  $S[t]$  with next byte of message to en/decrypt

```

i = j = 0
for each message byte  $M_i$ 
    i = (i + 1) (mod 256)
    j = (j + S[i]) (mod 256)
    swap( $S[i]$ ,  $S[j]$ )
    t = ( $S[i]$  +  $S[j]$ ) (mod 256)
     $C_i = M_i \text{ XOR } S[t]$ 
    
```

25



26

## 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**
- have a concern with WEP, but due to key handling rather than RC4 itself

27

## Summary

- Triple-DES
- Modes of Operation
  - ECB, CBC, CFB, OFB, CTR
- stream ciphers
- RC4

28