Symmetric Cryptography: DES and RC4

Modern Symmetric Ciphers

- We will now look at two examples of modern symmetric key ciphers:
  – DES
  – RC4
- These will serve as the basis for later discussion
- We will also discuss modes of operation
- Other popular ciphers: AES (Rijndael), RC5

Block vs Stream Ciphers

- There are two main classes of symmetric ciphers:
  – Block Ciphers: Break message into blocks and operate on a block-by-block basis
  – Stream Ciphers: Process messages bit-by-bit or byte-by-byte as data arrives.
- Typically, block ciphers may be modified to run in a "stream mode"
- We will examine two of ciphers:
  – DES: The basic block cipher (building block for 3DES)
  – RC4: A popular example of a stream cipher

Block Cipher Principles

- Most symmetric block ciphers are based on a **Feistel Cipher Structure**
  – This structure is desirable as it is easily reversible, allowing for easy encryption and decryption
    - *Just reuse the same code, essentially!*
  – Feistel structures allow us to break the construction of an encryption/decryption algorithm into smaller, manageable building blocks
- Goal of a block cipher:
  – Mix and permute message bits in a way that is parameterized by an encryption key
  – Output should be "statistically" uncorrelated with the key and input message

Substitution & Permutation Ciphers

- In 1949 Claude Shannon introduced idea of substitution-permutation (S-P) networks
  - modern substitution-transposition product cipher
- These form the basis of modern block ciphers
- S-P networks are based on the two primitive cryptographic operations we have seen before:
  - *substitution* (S-box)
  - *permutation* (P-box)
- Provide *confusion* and *diffusion* of message

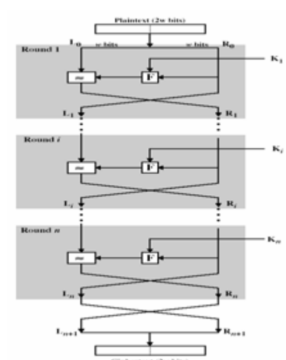
Confusion and Diffusion

- Cipher needs to completely obscure statistical properties of original message
- One approach to accomplish this is to use a one-time pad
  - Drawback: One time pads are impractical!
- More practically Shannon suggested combining elements to obtain:
- **Diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
- **Confusion** – makes relationship between ciphertext and key as complex as possible

Feistel Cipher Building Block

- Horst Feistel devised the **Feistel cipher**
  - It is an example of Shannon's philosophy of substitute and permute
  - Based on concept of invertible product cipher
- Partitions input block into two halves
  - Process through multiple rounds which
  - Perform a substitution on left data half
  - Based on round function of right half & subkey
  - Then have permutation swap halves

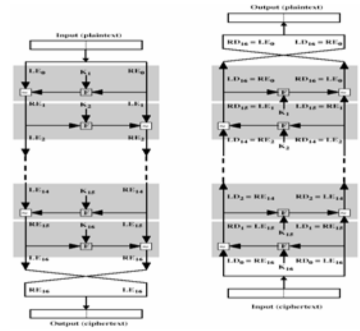
Feistel Cipher Structure



Feistel Cipher Design Parameters

- **Block size:** Amount of message bits input
- **Key size:** Size of the key paramaterizing encryption
  - Increasing size improves security, makes exhaustive key search harder, but may slow cipher
- **Number of rounds**: How many Feistel rounds one does when encrypting
- **Subkey generation:** Keys are generally broken into subkeys
- **Round function:** The operations done during each Feistel Round

Feistel Cipher Decryption (DES)



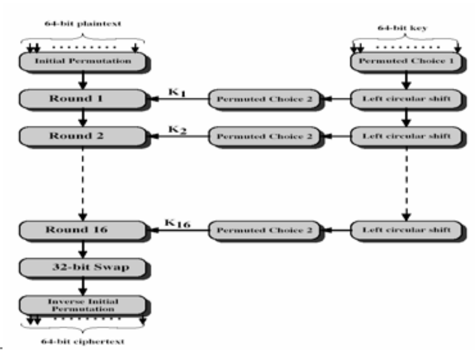
Data Encryption Standard (DES)

- As a building block, it has become the most widely used block cipher in world
- Adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- Encrypts 64-bits of data using a 56-bit key
  - Often the key appears as 64 bits, which is really 56 bits with 8 parity bits
- Recently, DES has experienced significant criticism due to its inability to withstand attacks;
  - Its getting old!

DES History

- IBM developed Lucifer cipher
  - by team led by Feistel
  - used 64-bit data blocks with 128-bit key
- Then redeveloped as a commercial cipher with input from NSA and others
- In 1973 NBS issued request for proposals for a national cipher standard
- IBM submitted their revised Lucifer which was eventually accepted as the DES

DES Encryption



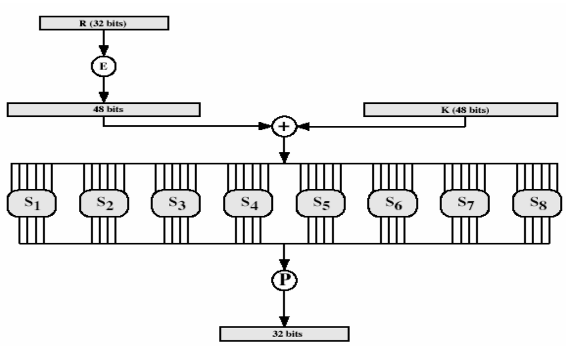
DES Round Structure

- Uses two 32-bit L & R halves
- As for any Feistel cipher can be described as:

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$

- Takes 32-bit R half and 48-bit subkey and:
  1. Expands R to 48-bits using an expansion function
  2. Adds to subkey
  3. Applies 8 S-boxes to get 32-bit result
  4. Permutes this using 32-bit perm P

DES Round Structure



S Boxes and Key Schedule

- There are eight S-boxes in DES, each one maps 6 bits to 4 bits
- How to use the S-boxes:
  - S-box is 4 rows by 16 columns
  - Input is 6 bits. The first and last bit determine the row, the middle 4 bits determine the column
  - Example (011011): 01 = 2nd row, 1101 = 14th column
  - Output value is simply a table look-up for that S-box
- The subkeys used in DES are governed by the Key Schedule:
  - First key bits are shifted depending on which round you are in
  - Next, 48 bits are chosen out of the 56 bits according to a table
- (See Trappe and Washington Book)

DES Decryption

- Decrypting DES is easy since it uses Feistel Cipher Structure
- Observation: How to undo a step of encryption…

$$L_i = R_{i-1} \longleftrightarrow R_{i-1} = L_i$$
$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i) \longleftrightarrow L_{i-1} = R_i \oplus F(R_{i-1}, K_i)$$

- Thus, decryption involves:
  - Using subkeys in reverse order (SK16 … SK1)
  - 1st round with SK16 undoes 16th encrypt round, … , 16th round with SK1 undoes 1st encrypt round
  - Finally undo IP

WINLAB

The Many Modes of Block Cipher Operation

- Block ciphers encrypt fixed size blocks, but typically our data is not a fixed block size or not the same size as the encryption block size
- Four were defined for DES in ANSI standard **ANSI X3.106-1983 Modes of Use,** a 5^{th} mode is now commonly used also
- These modes may operate in either a block-by-block fashion, or in a streaming mode

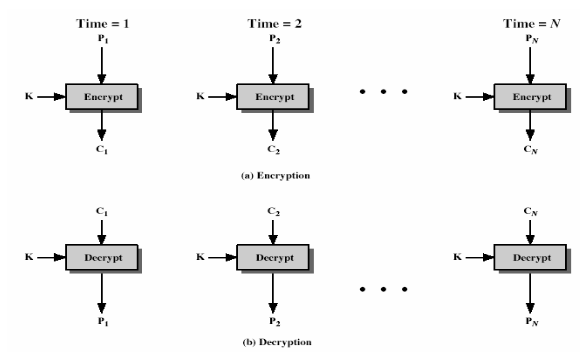
WINLAB

Electronic Codebook Book (ECB)

- The basic, natural (and insecure) method for encryption is the Electronic Code Book Mode
- Here, the message is broken into independent blocks which are each encrypted
- Weakness:
  - Allows for correlation between subsequent encryptions of the same plaintext
  - Hence, dictionary-type attacks are feasible

WINLAB

Electronic Codebook Book (ECB)



WINLAB

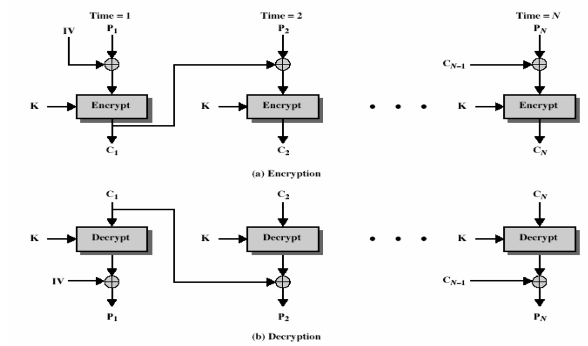
Cipher Block Chaining (CBC)

- A more advanced method is to encrypt the next block based upon the output of the previous encryption
- This idea is called chaining since it links previous ciphertexts with current plaintexts
- The basic scheme starts with setting C0 to be an Initial Vector (IV) to start process, and then:

$$C_i = E_K(P_i \oplus C_{i-1})$$
$$C_0 = IV$$

- Practical applications are in bulk data encryption, and authentication

Cipher Block Chaining (CBC)



Advantages and Limitations of CBC

- Each ciphertext block depends on **all** message blocks
- Thus a change in the message affects all ciphertext blocks after the change as well as the original block
- Need the **Initial Value** (IV) known to sender & receiver
  - However if IV is sent in the clear, an attacker can change bits of the IV
  - 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 (eg nulls)
  - Or pad last block with count of pad size

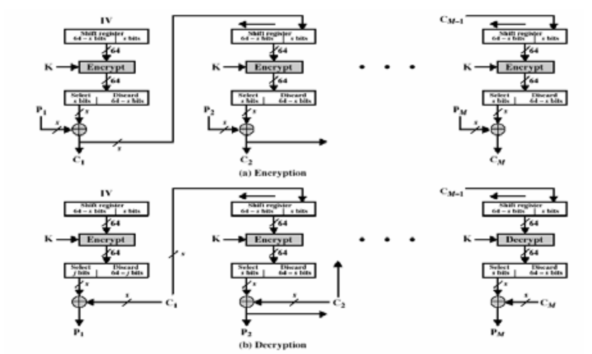
Cipher FeedBack (CFB)

- Message is treated as a stream of bits (or stream of small chunks of bits)
- Plaintext is added to previous output of the cipher:

$$C_i = P_i \oplus E_K(C_{i-1})$$
$$C_0 = IV$$

- Any number of bit (1,8 or 64 or whatever) to be feedback
  - denoted CFB-1, CFB-8, CFB-64 etc
- CFB is a stream mode of operation

Cipher FeedBack (CFB)



Advantages and Limitations of CFB

- Appropriate when data arrives in bits/bytes
- Most common stream mode and is effective in turning a block cipher algorithm, like DES, into a stream cipher algorithm
- Observe that the block cipher is used in **encryption** mode at **both** ends
- Errors propagate for several blocks after the error, but eventually flush themselves

Output FeedBack (OFB)

- The message is treated as a stream of bits, and the output of the cipher is added to the plaintext message
- Output is then feed back to produce next output
- Feedback is independent of the message, and therefore can be computed in advance of data arriving

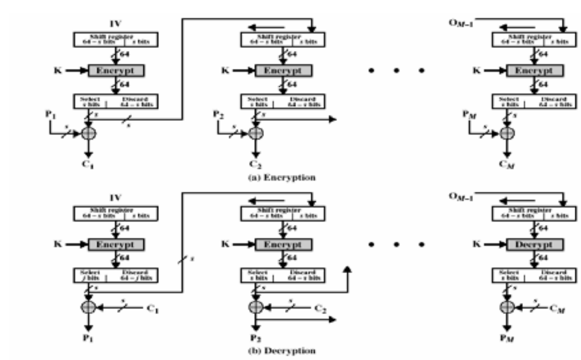
  $C_i = P_i \text{ XOR } O_i$

  $O_i = DES_{K1}(O_{i-1})$

  $O_{-1} = IV$

- Is commonly used for stream encryption over noisy channels

Output FeedBack (OFB)



Advantages and Limitations of OFB

- Used when error feedback a problem or where we need to encrypt before message is available
- Since feedback is independent of message, pre-computation is facilitated
- A variation of a Vernam cipher
  - Hence must **never** reuse the same sequence (key+IV)
- Synchronization is critical
- If DES is used, all 64 bits should be fed back!

WINLAB

Counter (CTR)

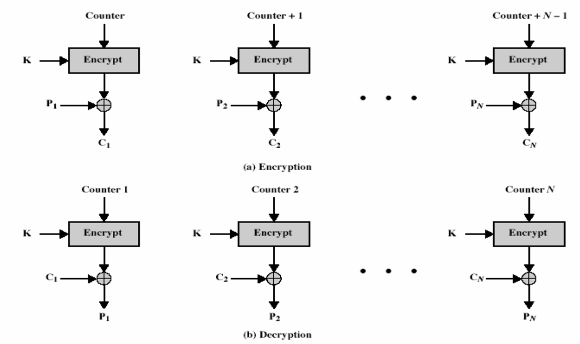
- CTR, or counter mode, is a “new” encryption mode that finds application in networking, and is particularly suited for wireless applications
  - Has been employed in sensor network applications
- It is basically OFB but we encrypt a counter value rather than any feedback value
- Like OFB, we must have a different key & counter value for every plaintext block (never reused)

$C_i = P_i \text{ XOR } O_i$

$O_i = DES_{K1}(i)$

WINLAB

Counter (CTR)



WINLAB

Advantages and Limitations of CTR

- Efficiency
  - Can do parallel encryptions in advance of need
  - Good for bursty high speed links
- Its security can be shown to be as good (or bad) as the other modes
- Again, must never reuse key/counter values, otherwise CTR is easily breakable

WINLAB

Another way to get a Stream Cipher

- Operating a Block Cipher in a feedback mode allows for streaming
- Another approach is to directly build a streaming cipher
- Applications for stream ciphers arise due to the need to operate on a bit-by-bit manner
- Design issues:
  - Long period with no repetitions and statistically random
  - Correlation immunity
  - How to we make them efficient, yet complicated?

WINLAB

RC4

- The "pure" stream cipher we will talk about is RC4, which is a proprietary cipher owned by RSA Inc.
- The "R" is for Ron Rivest (MIT), its inventor
- Facts about RC4:
  - Variable key size, byte-oriented stream cipher
  - Widely used (web SSL/TLS, wireless WEP)
  - Design was secret until 1994 when reverse engineered and source code posted to Cypherpunks mailing list
  - Two components: Key-Scheduling Algorithm and a pseudo-random generator

WINLAB

RC4 Key Schedule

- Starts with an array S of numbers: 0..255
- Use key to truly shuffle S
- S forms **internal state** of the cipher
- Given a key k of length l bytes

```
for i = 0 to 255 do
    S[i] = i
j = 0
for i = 0 to 255 do
    j = (j + S[i] + k[i mod l]) (mod 256)
    swap (S[i], S[j])
```

WINLAB

RC4 Encryption

- Encryption involves XORing data bytes with output of the PRGA
- The PRGA initializes i and j to 0 and then loops over 4 basic operations: increase j, increase j using s[i], swap and output s[i]+s[j]
- Pseudocode is:

```
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 XOR S[t]
```

WINLAB

RC4 Issues

- RC4 has been claimed to be secure, though there is no strong evidence to back this up
- Since RC4 is a stream cipher, must **never reuse a key**
- Recently, a variation of RC4 has been used in 802.11 encryption (WEP)
  - There has been significant news about WEP being weak
  - This does not mean RC4 is weak
  - WEP was poorly designed, the weakness is due to a key handling issue rather than RC4 itself