Data Encryption Standard (DES)

One of the early, and still well-regarded block ciphers

Feistel Cipher

- Forms basis of most block ciphers, including DES
- Series of steps based on *substitutions* and *permutations*
  - Substitution: Each plaintext bit or group of bits replaced by ciphertext
  - Permutations: Order of plaintext bits is changed

Feistel Cipher

- Shannon's terminology: *diffusion* and *confusion*
- Diffusion – achieved through permutation
  - Make each block of ciphertext derived from many bits of plaintext
  - Complex statistical relationship between plaintext and ciphertext
- Confusion – achieved through substitution
  - Many rounds of substitution
  - Complex statistical relationship between ciphertext and encryption key

Feistel Encryption and Decryption Rounds

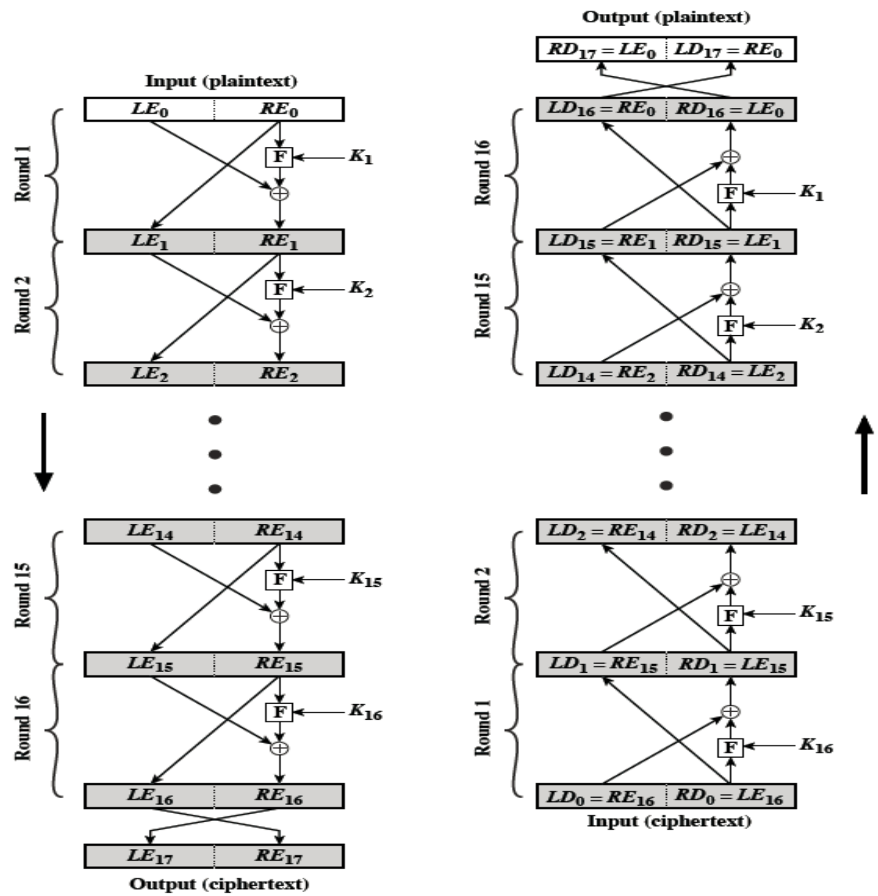
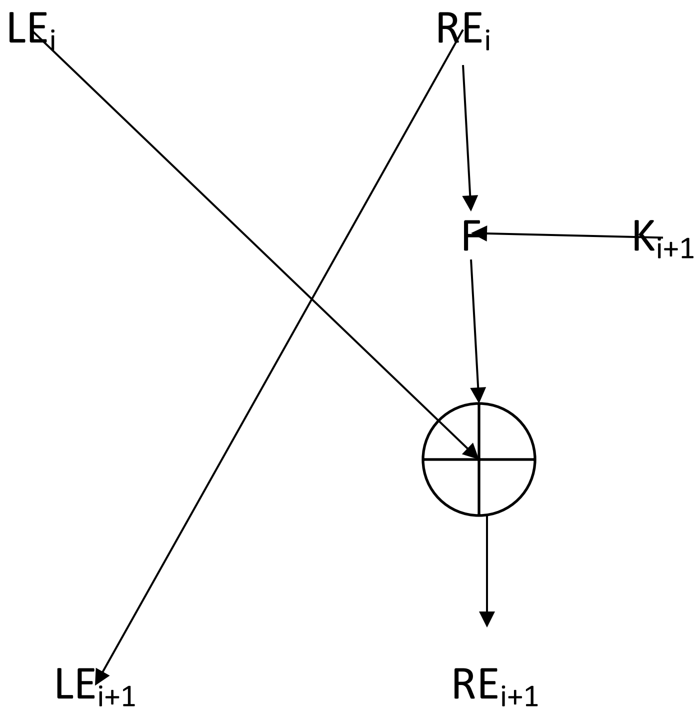


Figure 3.3 Feistel Encryption and Decryption (16 rounds)

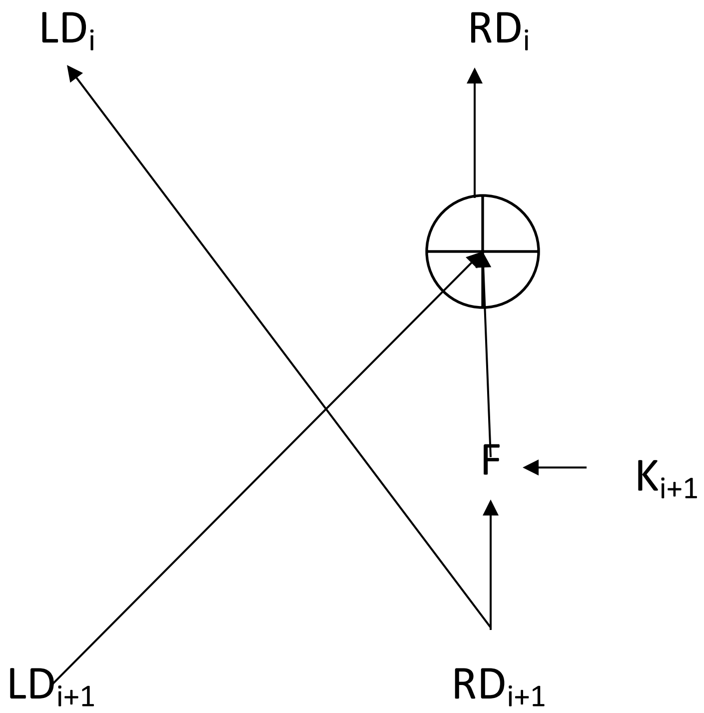
Figure from W. Stallings, Cryptography and Network Security,

Generalizing Feistel Cipher Round

Round of Encryption



Round of Decryption



Metrics for Feistel Ciphers (and more generally for Block Ciphers)

- Block size
  - Larger block size = more permutations = more security. But less speed
  - Block size inversely proportional to speed
  - Standard block size 128 bits

- Key size
  - Larger key size = more keyspace = more resistant to brute force attacks. But less speed
  - Key size inversely proportional to speed
  - Standard key size 128 bits

Metrics for Feistel Ciphers (and more generally for Block Ciphers)

- Number of rounds
  - More rounds = more security
  - Typical for Feistel ciphers – 16 rounds

- Ease of analysis: heads-I-win-tails-you-lose
  - We want easy-to-analyze algorithms, but will be easy-to-cryptanalyze for adversary too
  - Hard-to-understand and hard-to-examine algorithms for attacker are ideal, but will be so for us too

Metrics for Feistel Ciphers (and more generally for Block Ciphers)

- Other metrics
  - Key (and subkey) generation algorithm -- must be complex
  - Function F – must be complex

- High speed
  - Speed important metric
  - Decryption slower than encryption (especially in PKC)
  - Algorithms implemented in hardware faster than software-only implementations

History of DES

- Started with NIST putting out a request, circa 1973
- 1974 – IBM’s Lucifer chosen
- 1976 - Lucifer renamed as Data Encryption Standard (DES), FIPS 46
- 1992 – DES broken in principle (in theory) by Shamir-Biham
  - Differential cryptanalysis
- 1998 – EFF break DES in around 2 days
  - Linear cryptanalysis
- 1999 – EFF with collaborators break DES in under a day
- 2000 – NIST puts out request for a new standard
- 2004-2005 NIST retires DES after AES becomes new standard

Key Length and Block Length

- Recall this

$$E : \{0,1\}^{k} \times \{0,1\}^{l} \rightarrow \{0,1\}^{l}$$

- and this

$$\underbrace{E_K}_{\text{k bits}} \; \underbrace{(M)}_{\text{l bits}} \; = \underbrace{(K, M)}_{\text{l bits}}$$

- In DES k = 56 bits, l = 64 bits
- k = key length, l = block length

Cryptographic Strength of DES

- 56-bit keys, so keyspace is $2^{56} \approx 7.2 \times 10^{16}$ keys

- Current speeds of multicore processors = 10^{12} encryptions/second
  - Breaking time = 2-3 hours

- On supercomputers, 10^{17} encryptions/second
  - Breaking time = 0.25-0.5 hour

Cryptographic Strength of DES

- Actually DES was broken way back in 90’s
  - Michael Weiner, 1993 – 3.5 hours, cost $1 million (impractical)
  - EFF, 1999 – 56 hours, $250,000

- Other avenues of attack?
  - S-boxes – design criteria made public in ‘94; possible weaknesses?... maybe, maybe not[1]

- Brute forcing key only way to break DES till date. No algorithmic weaknesses ever found

At least nobody has publically acknowledged any

2DES

- Double DES key length

- k = 56*2 = 112 bits, l = 64 bits
- So, 2DES : $\{0,1\}^{112} \times \{0,1\}^{64} \rightarrow \{0,1\}^{64}$
- In other words

$$\underbrace{E_K}_{112\text{ bits}} \; \underbrace{(M)}_{64\text{ bits}} = \underbrace{(K, M)}_{64\text{ bits}}$$

- How does it work?

2DES: $E_{K2}\,(E_{K1}(M)) = C$; where E = DES

How Secure is 2DES?

- 2DES: $E_{K2}(E_{K1}(M)) = C$
- $E_{K1}(M) = D_{K2}(C)$; where E = D = DES
- Pick a (M,C) pair, construct a table:

M	**C**
EK_1^1 (M) =0x2cdf	DK_2^1 (C) = 0x23de
EK_1^2 (M)=0x32df	DK_2^2 (C) = 0x42ef
EK_1^3 (M)=0x41a2	DK_2^3 (C) = 0x21ad
…	…
EK_1^x (M)=0x3f; $x=2^{55}$	DK_2^x (C) = 0x3f; $x=2^{55}$
EK_1^y (M)=0x2d; $y=2^{56}$	DK_2^y (C) = 0xac21; $y=2^{56}$

Match!

Man-in-the-middle attack idea:

- Deduce that $EK_1^x\ (M) = DK_2^x\ (C)$
- Test if $EK_1^x\ (M') = DK_2^x\ (C')$. If yes, K_1^x, K_2^x is what we are looking for.
- We've broken 2DES, Yay!
- On an average takes 2^{55} trials.

3DES

- Triple DES key length

- k = 56*3 = 168 bits, l = 64 bits
- So, 3DES : $\{0,1\}^{168} \times \{0,1\}^{64} \rightarrow \{0,1\}^{64}$
- In other words

$$\underbrace{E_K}_{\text{168 bits}} \underbrace{(M)}_{\text{64 bits}} = \underbrace{(K, M)}_{\text{64 bits}}$$

- How does it work?

3DES: $E_{K3}(D_{K2}(E_{K1}(M))) = C$, where E = DES

Triple Encryption with Two Keys

- Discovered by Tuchman

$$C = E_{K1}(D_{K2}(E_{K1}(M)))$$
$$M = D_{K1}(E_{K2}(D_{K1}(C)))$$

- No known practical attacks on 3DES (with 2 or 3 keys)
- Brute force attack needs to try 2^{111} keys, while keyspace = 2^{112}
- Merkle and Hellman’s attack needs 2^{56} key searches, but requires 2^{56} (M,C) pairs to be provided to attacker – unlikely to happen

DESX

$$\text{DESX}: K_2 \oplus E_K (K_1 \oplus M) = C$$

- $k = 56 + 64 + 64 = 184$ bits, $l = 64$ bits
- DESX : $\{0,1\}^{184} \times \{0,1\}^{64} \rightarrow \{0,1\}^{64}$
- Usable key length = 120 bits
- Less computationally expensive than 2DES, 3DES

DES in Hindsight

- DES is an extremely well-designed algorithm
- Very well-scrutinized too
- From 1976 on, best known (practical) attack still only brute-forcing key
- No algorithmic/structural weaknesses
- Serves as template/inspiration for algorithms after it

Key Recovery Security

- So far, "breaking" an encryption algorithm:
  - Cryptanalysis goal = Guessing key
  - Brute Force Attack goal = Guessing key

- Is this enough? Consider this:

$$E: \{0,1\}^{512} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$$

- k = 512 bits, l = 256 bits
- Brute force attack – try out 2^{511} keys – key guaranteed not recoverable

Key Recovery Security

- What if we define the encryption algorithm as:

$$E_K(P) = P$$

$$E_K(P) = P \cdot 2$$

$$E_K(P) = P^2$$

$$E_K(P) = P \text{ XOR } 1$$

$$E_K(P) = \sqrt{P}$$

…

Secure, ain't it ?

So what do we Learn

- Security against key recovery alone not enough
- Necessary, but not sufficient
- Strength of encryption algorithm matters too
- Structure and design of encryption algorithm must resist reverse-engineering, or any other attack

Symmetric Crypto Security

- Not based on an underlying hard math problem (unlike public-key crypto)

- Symmetric crypto relies on less well-known assumptions
  - Existence of PRF, pseudo-random permutations

- Security guarantee: Output of DES/AES, etc. indistinguishable from output of a good PRF