Block Ciphers and the DES

Outline

- Block vs Stream Ciphers
- Substitution-Permutation Ciphers
- Feistel Cipher Structure
- DES Encryption Algorithm
- Strength of DES
- Modes of Operation

Block vs Stream Ciphers

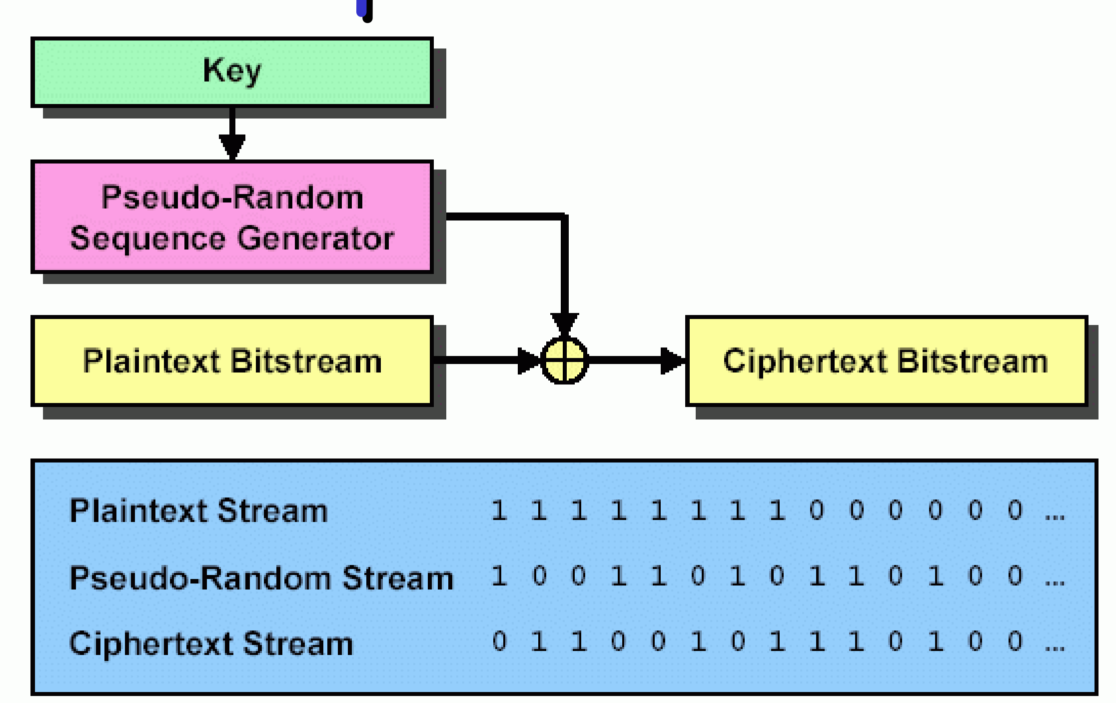
- **Block ciphers**
  - Process messages in relatively large blocks, each of which is then en/decrypted
  - Like a substitution on very big characters, e.g 64-bits or more
  - Same function is used to encrypt successive blocks ⇒memoryless

Block vs Stream Ciphers

- **Stream ciphers**
  - Process messages a bit or byte at a time when en/decrypting
  - encryption function may vary as plaintext is processed $\Rightarrow$ have memory
  - sometimes called state ciphers since encryption depends on not only the key and plaintext, but also on the current state

Stream Cipher

- *Stream* ciphers



- Rather than divide bit stream into discrete blocks, as block ciphers do, XOR each bit of your plaintext continuous stream with a bit from a pseudo-random sequence
- At receiver, use same symmetric key, XOR again to extract plaintext

Block vs Stream Ciphers

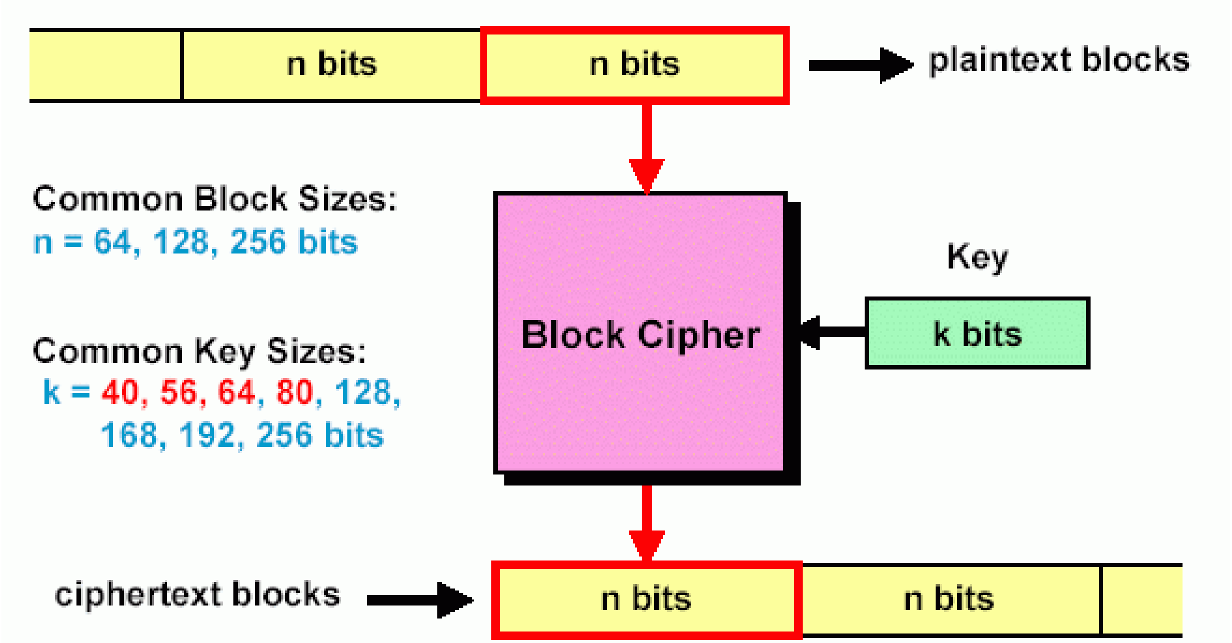
- Distinction between block and stream ciphers is not definitive
  - adding memory to a block cipher (as in CBC) results in a stream cipher
- Many current ciphers are block ciphers
- Hence are focus of this course

Block Cipher

- Maps n-bit plaintext blocks to n-bit ciphertext blocks (n: block length)
- Use of plaintext and ciphertext of equal size avoids data expansion (bijection: doesn't expand the input. 16 B in, 16 B out)
- To allow unique decryption, encryption function must be 1-1(invertible)
  - For n-bit plaintext and ciphertext blocks and a fixed key, the encryption function is a bijection defining a permutation on n-bit vectors
  - Each key potentially defines a different bijection

Block Cipher

- Divide input bit stream into n-bit sections, encrypt only that section, no dependency/history between sections



Courtesy: Andreas Steffen

- In a good block cipher, each output bit is a function of all n input bits and all k key bits

Block Cipher Principles

- Most symmetric block ciphers are based on a **Feistel Cipher Structure**
- Needed since must be able to **decrypt** ciphertext to recover messages efficiently
- Block ciphers look like an extremely large substitution
- Would need table of 2^{64} entries for a 64-bit block

Block Cipher Principles

- Instead create from smaller building blocks
- Using idea of a product cipher

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

Substitution-Permutation 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

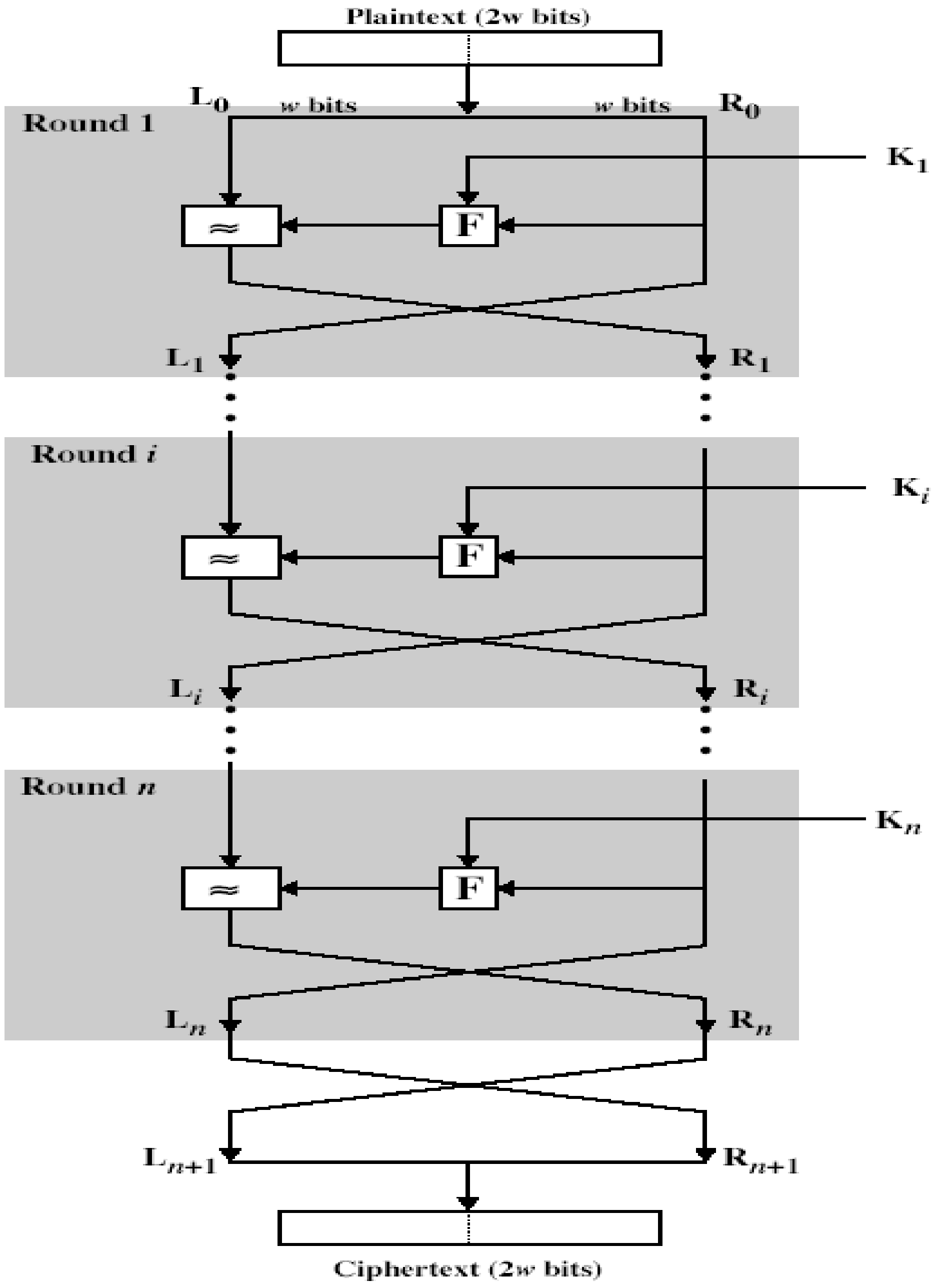
Confusion and Diffusion

- Cipher needs to completely obscure statistical properties of original message
- One-time pad does this
- 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 Structure

- Horst Feistel devised the **feistel cipher**
  - Based on concept of invertible product cipher
- Partitions input block into two halves
- Process through multiple rounds which
  - Perform a substitution on left half of data
  - Substitution based on round function of right half & subkey
  - Then have permutation swapping halves
- Implements Shannon's substitution-permutation network concept

Feistel Cipher Structure



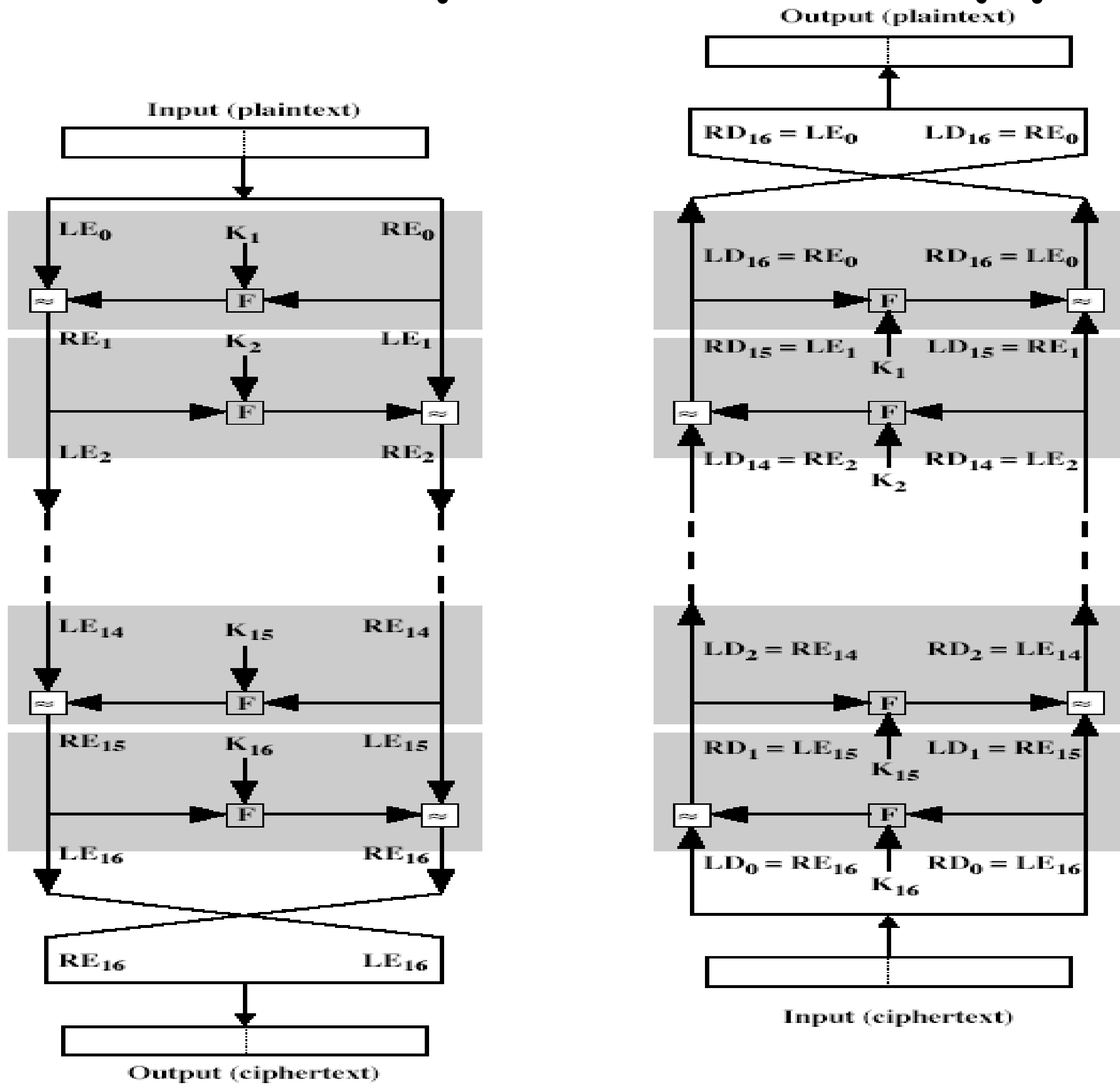
Feistel Cipher Structure

- **Block size** -larger block sizes mean greater security
- **Key Size** -larger key size means greater security
- **Number of rounds** - multiple rounds offer increasing security
- **Subkey generation algorithm** - greater complexity will lead to greater difficulty of cryptanalysis
- **Fast software encryption/decryption** -the speed of execution of the algorithm becomes a concern

Feistel Cipher Decryption

- Same as encryption
- Use the ciphertext as input to the algorithm
- But, use the subkeys K_i in reverse order
  - i.e., use K_n in the first round
- Need not implement two different algorithms for encryption and decryption

Feistel Cipher Decryption



Conventional Encryption Algorithms

Data Encryption Standard (DES)

- 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

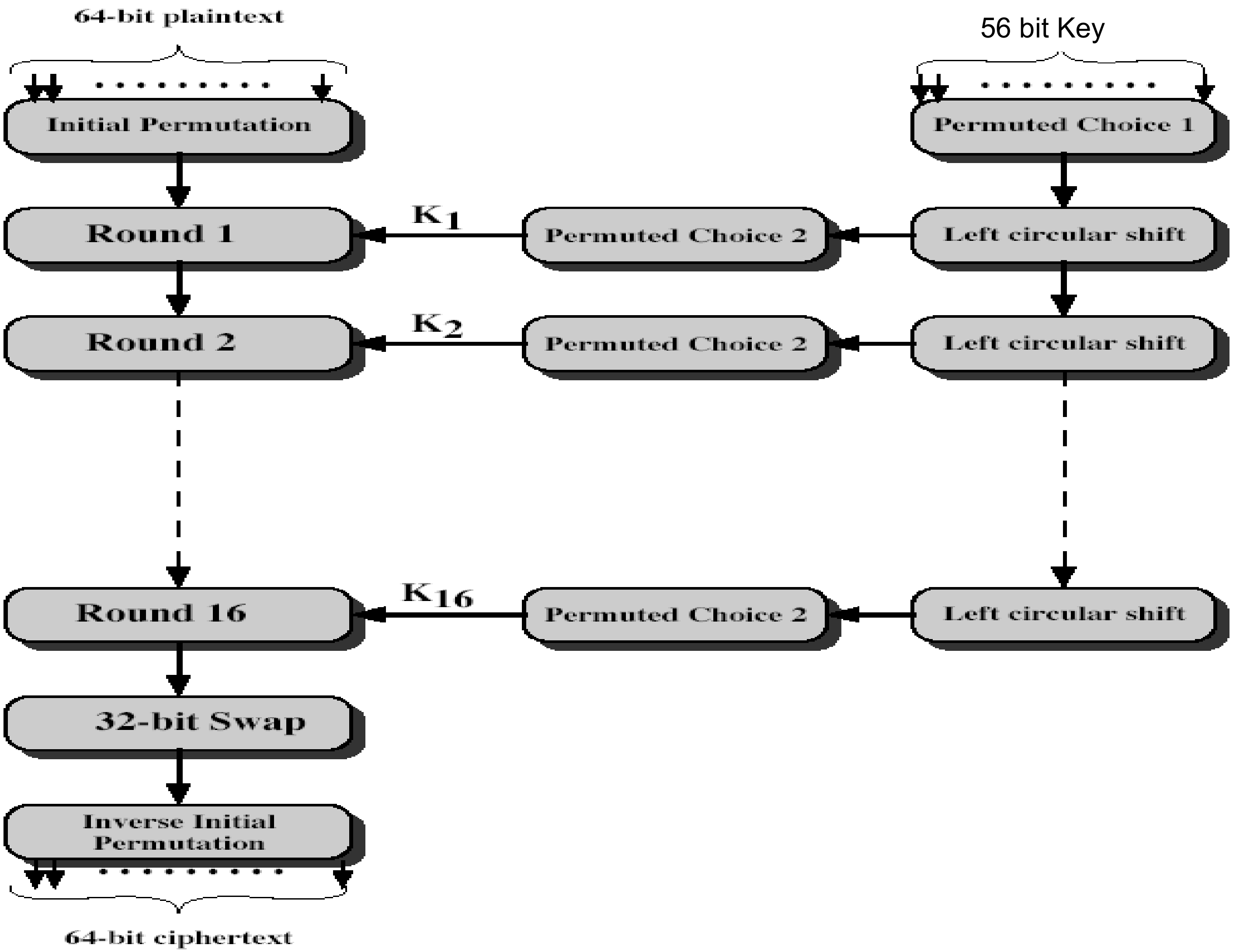
Data Encryption Standard (DES)

- Most widely used encryption scheme
- The algorithm is reffered to as Data Encryption Algorithm (DEA)
- DES is a block cipher
  – The plaintext is processed in 64-bit blocks
- The key is 56-bits in length

DES Design Controversy

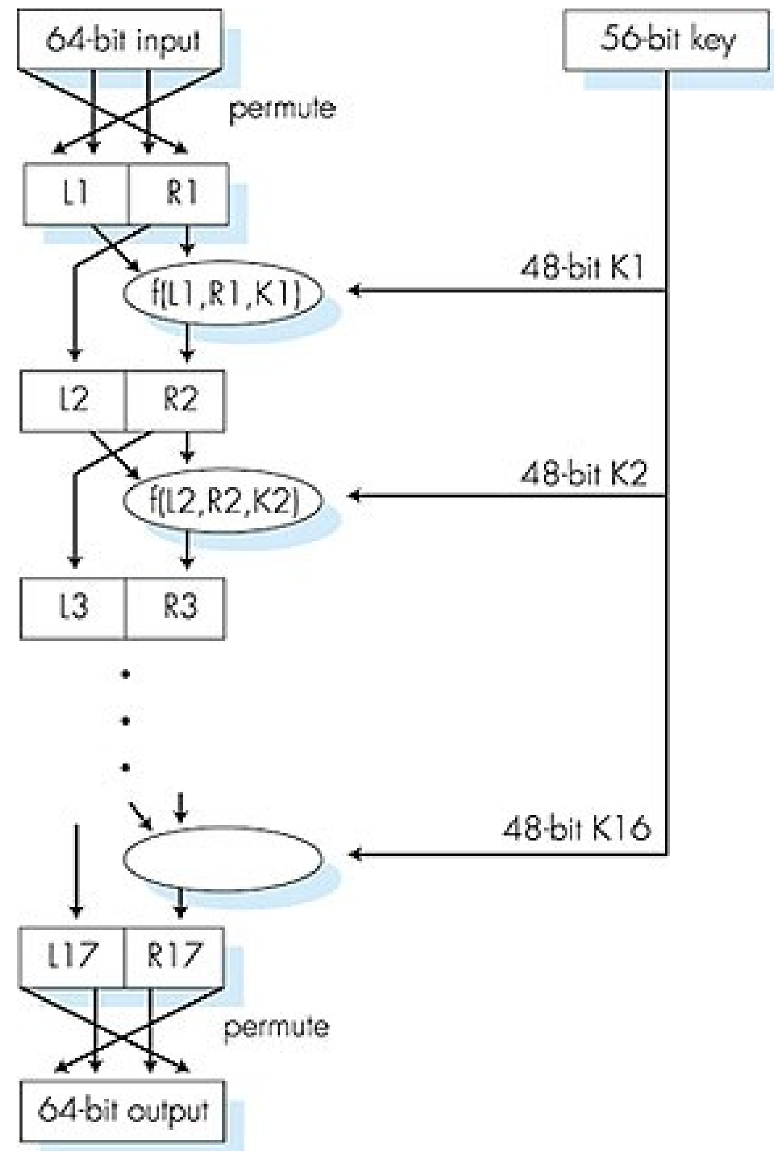
- Although DES standard is public
- Was considerable controversy over design
  - In choice of 56-bit key (vs Lucifer 128-bit)
  - And because design criteria were classified
- Subsequent events and public analysis show in fact design was appropriate
- DES has become widely used, especially in financial applications

DES Encryption



DES

- DES
  - 64-bit input is permuted
  - 16 stages of identical operation
    - differ in the 48-bit key extracted from 56-bit key - complex
    - R2= R1 is encrypted with K1 and XOR'd with L1
    - L2=R1, …
  - Final inverse permutation stage



Initial Permutation IP

- First step of the data computation
- IP reorders the input data bits
- Even bits to LH half, odd bits to RH half
- Quite regular in structure (easy in h/w)
- See table 3.2 in the textbook
- Example:

```
IP(675a6967 5e5a6b5a) = (ffb2194d 004df6fb)
```

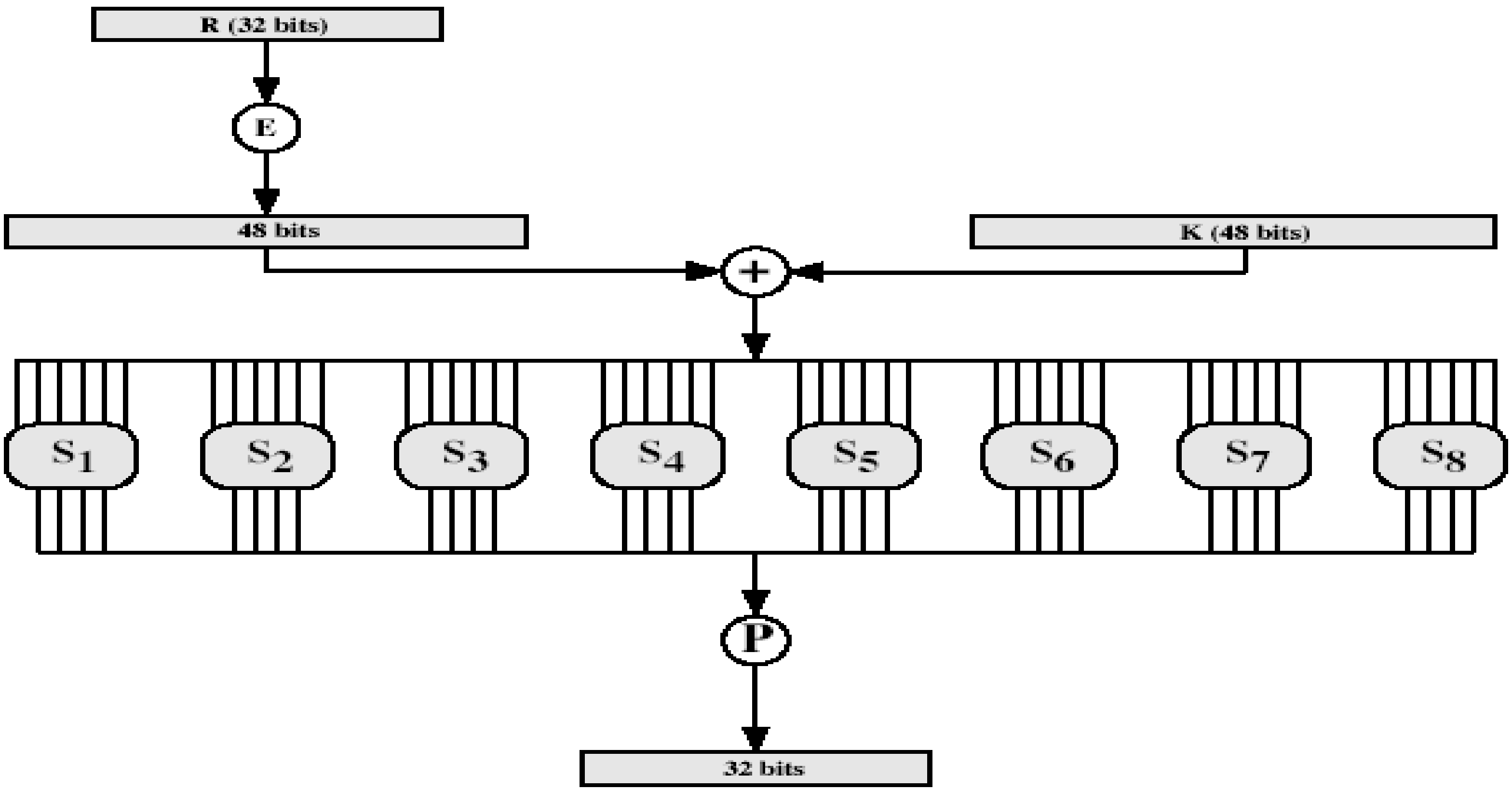
DES Round Structure

- Uses two 32-bit l & r halves
- As for any feistel cipher can describe as:

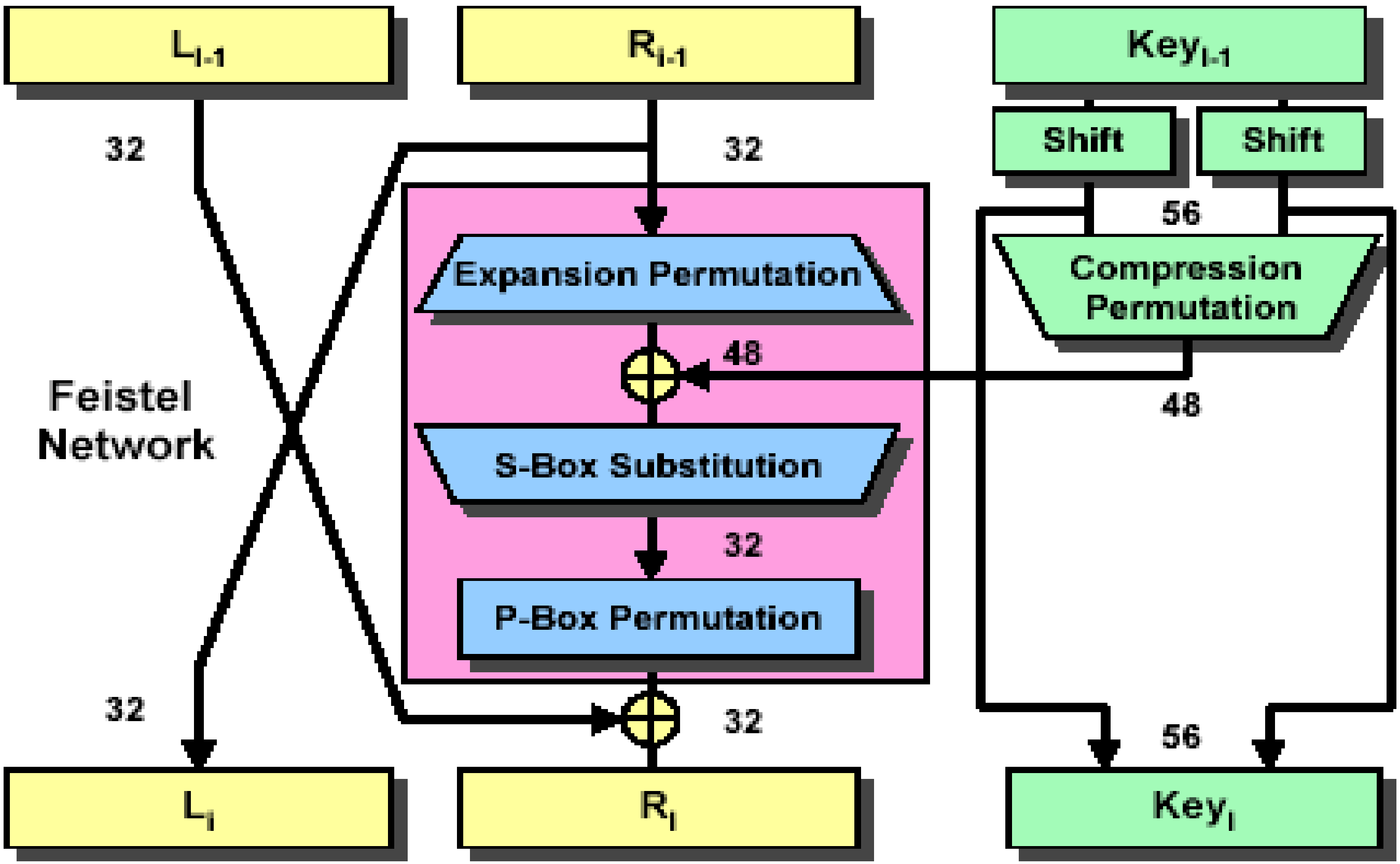
  $L_i = R_{i-1}$

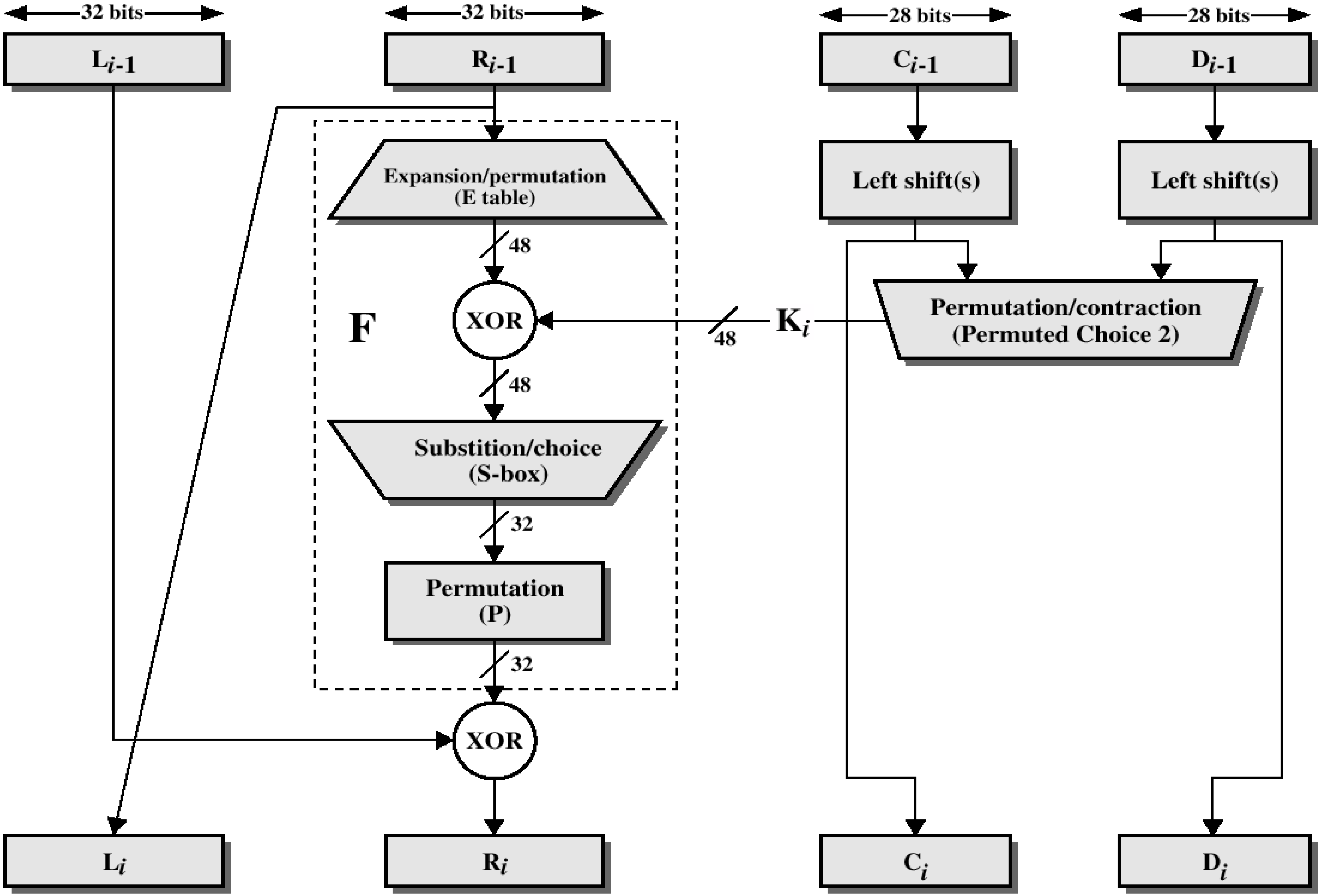
  $R_i = L_{i-1}$ XOR $F(R_{i-1}, K_i)$
- Takes 32-bit R half and 48-bit subkey and:
  - Expands R to 48-bits using perm E
  - Adds to subkey
  - Passes through 8 s-boxes to get 32-bit result
  - Finally permutes this using 32-bit perm P

DES Round Structure



DES: Single Round





Single Round of DES Algorithm

Substitution Boxes S

- Have eight s-boxes which map 6 to 4 bits
- Each s-box is actually 4 little 4 bit boxes
  - Outer bits 1 & 6 (**row** bits) select one rows
  - Inner bits 2-5 (**col** bits) are substituted
  - Result is 8 lots of 4 bits, or 32 bits
- Row selection depends on both data & key
  - Feature known as autoclaving (autokeying)
- Example:

```
S(18 09 12 3d 11 17 38 39) = 5fd25e03
```

DES Key Schedule

- Forms subkeys used in each round
- Consists of:
  - Initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - Selecting 24-bits from each half
    - Permuting them by PC2 for use in function f,
    - Rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K

DES Decryption

- Decrypt must unwind steps of data computation
- With feistel design, do encryption steps again
- Using subkeys in reverse order (SK16 … SK1)
- Note that IP undoes final FP step of encryption

DES Decryption

- 1st round with SK16 undoes 16th encrypt round
- ….
- 16th round with sk1 undoes 1st encrypt round
- Then final fp undoes initial encryption ip
- Thus recovering original data value

Avalanche Effect

- Key desirable property of encryption algorithm
- Where a change of **one** input or key bit results in changing approx **half** output bits
- Making attempts to “home-in” by guessing keys impossible
- DES exhibits strong avalanche

Strength of DES - Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- Brute force search looks hard
- Recent advances have shown is possible
  - In 1997 on Internet in a few months
  - In 1998 on dedicated h/w (EFF) in a few days
  - In 1999 above combined in 22hrs!
- Still must be able to recognize plaintext
- Now considering alternatives to DES

Strength of DES - Timing Attacks

- Attacks actual implementation of cipher
- Use knowledge of consequences of implementation to derive knowledge of some/all subkey bits
- Specifically use fact that calculations can take varying times depending on the value of the inputs to it
- Particularly problematic on smartcards

Strength of DES – Analytic Attacks

- Now have several analytic attacks on DES
- These utilise some deep structure of the cipher
  - By gathering information about encryptions
  - Can eventually recover some/all of the sub-key bits
  - If necessary then exhaustively search for the rest

Strength of DES – Analytic Attacks

- Generally these are statistical attacks
- Include
  - Differential cryptanalysis
  - Linear cryptanalysis
  - Related key attacks

Differential Cryptanalysis

- One of the most significant recent (public) advances in cryptanalysis
- Known by NSA in 70's cf DES design
- Murphy, Biham & Shamir published 1990
- Powerful method to analyse block ciphers
- Used to analyse most current block ciphers with varying degrees of success
- DES reasonably resistant to it, cf Lucifer

Differential Cryptanalysis

- A statistical attack against Feistel ciphers
- Uses cipher structure not previously used
- Design of S-P networks has output of function *F* influenced by both input & key
- Hence cannot trace values back through cipher without knowing values of the key
- Differential Cryptanalysis compares two related pairs of encryptions

Differential Cryptanalysis Compares Pairs of Encryptions

- With a known difference in the input
- Searching for a known difference in output
- When same subkeys are used

$$\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1}$$

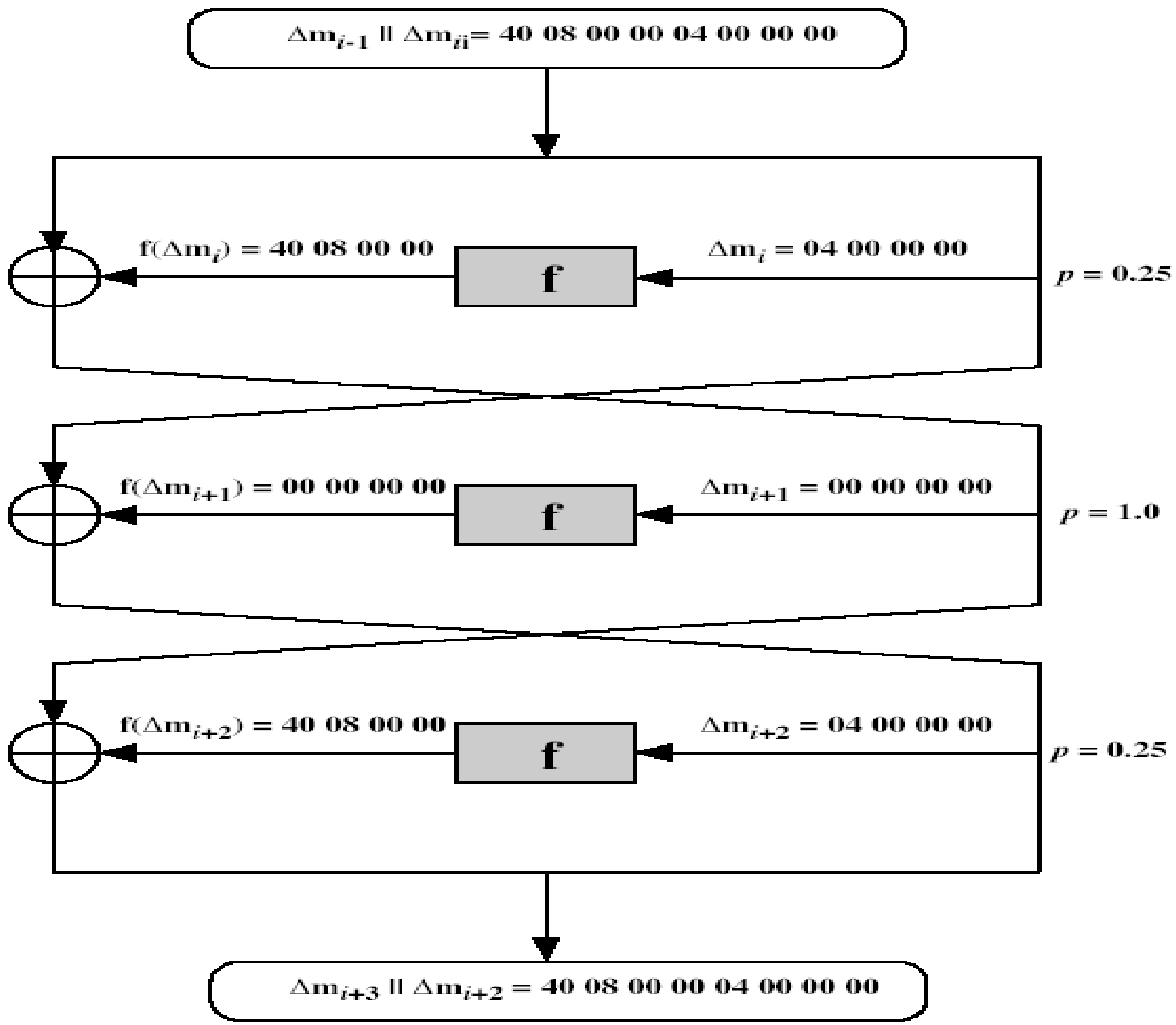
$$= \left[m_{i-1} \oplus \mathrm{f}\left(m_i, K_i\right)\right] \oplus \left[m'_{i-1} \oplus \mathrm{f}\left(m'_i, K_i\right)\right]$$

$$= \Delta m_{i-1} \oplus \left[\mathrm{f}\left(m_i, K_i\right) \oplus \mathrm{f}\left(m'_i, K_i\right)\right]$$

Differential Cryptanalysis

- Have some input difference giving some output difference with probability *p*
- Find instances of some higher probability input/output difference pairs occurring
- Can infer subkey that was used in round
- Then must iterate process over many rounds (with decreasing probabilities)

Differential Cryptanalysis



Differential Cryptanalysis

- Perform attack by repeatedly encrypting plaintext pairs with known input XOR until obtain desired output XOR
- When found
  - If intermediate rounds match required XOR have a **right pair**
  - If not then have a **wrong pair**, relative ratio is S/N for attack
- Can then deduce keys values for the rounds
  - Right pairs suggest same key bits
  - Wrong pairs give random values

Differential Cryptanalysis

- For large numbers of rounds, probability is so low that more pairs are required than exist with 64-bit inputs
- Biham and Shamir have shown how a 13-round iterated characteristic can break the full 16-round DES

Linear Cryptanalysis

- Another recent development
- Also a statistical method
- Must be iterated over rounds, with decreasing probabilities
- Developed by Matsui et al in early 90's
- Based on finding linear approximations
- Can attack DES with 2^{47} known plaintexts, still in practice infeasible

Linear Cryptanalysis

- Find linear approximations with prob p != ½

```
P[i1,i2,...,ia](+)C[j1,j2,...,jb] =
  K[k1,k2,...,kc]
```

```
where ia,jb,kc are bit locations in P,C,K
```

- Gives linear equation for key bits
- Get one key bit using max likelihood algorithm
- Using a large number of trial encryptions
- Effectiveness given by: `|p–½|`

Block Cipher Design Principles

- Basic principles still like Feistel in 1970's
- Number of rounds
  - More is better, exhaustive search best attack
- Function *F*:
  - Provides "confusion", is nonlinear, avalanche
- Key schedule
  - Complex subkey creation, key avalanche

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

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

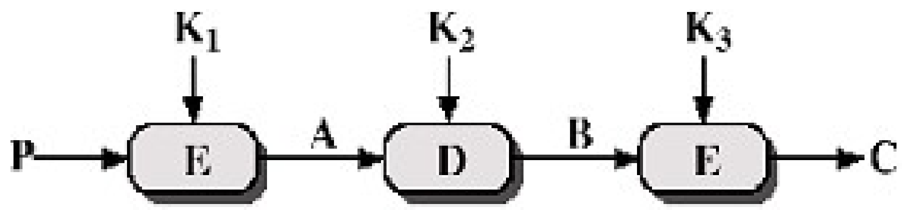
Triple DEA

- Use three keys and three executions of the DES algorithm (encrypt-decrypt-encrypt)

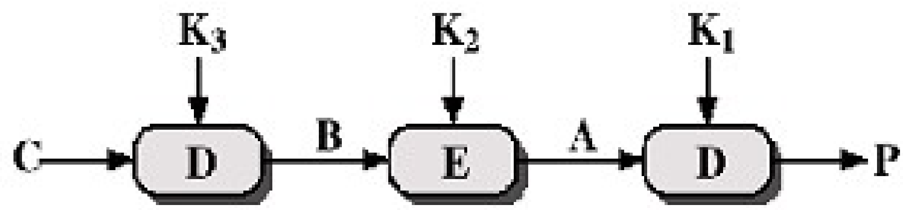
$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

  - C = ciphertext
  - P = Plaintext
  - EK[X] = encryption of X using key K
  - DK[Y] = decryption of Y using key K
- Effective key length of 168 bits

Triple DEA



Encryption



Decryption

Modes of Operation

- Block ciphers encrypt fixed size blocks
- e.g. DES encrypts 64-bit blocks, with 56-bit key
- Need a way to use in practice, usually have arbitrary amount of information to encrypt
- Four modes were defined for DES in ANSI standard **ANSI X3.106-1983 Modes of Use**
- Subsequently now have 5 for DES and AES
- Have **block** and **stream** modes