Confusion and diffusion

Claude Shannon (inventor of information theory) gave design criteria for symmetric ciphers (late 1940s):

- Kerckhoff's principle: only the key should be secret, not the algorithm
- use confusion: hide local patterns in the language (simplest: digrams)
- use diffusion: spread statistical structure of plaintext in long-range statistics of ciphertext

The Vigenère cipher hides small-scale structure: digrams are encoded by different keys, but fails at diffusion: nothing is moved/spread.

Simple transposition ciphers are good at diffusion: they move around, but difficult to achieve confusion.

Modern ciphers get both confusion (small-scale) and diffusion (large-scale), e.g. by combining substitution and transposition in a product cipher.

Feistel

Feistel (1970s) used Shannons ideas in designing ciphers. Standard design since then!

Make product cipher by alternating substitution and permutation:

$$c = E_k(m) = S_n \circ P_{n-1} \circ \cdots \circ S_2 \circ P_1 \circ S_1(m)$$

- Substitution: replace one (symbol/bitstring) by another - but reversibly
- Permutation: change the order of (symbol/bitstring)s (reversible)

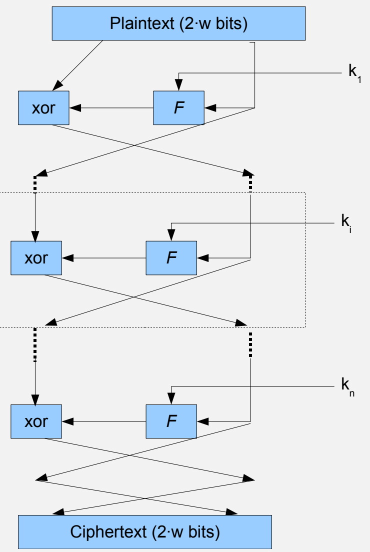
Feistel networks

A Feistel network:

1. split input in two halves L_0, R_0
2. perform n rounds:
   - $L_i \oplus F(R_i, k_i)$
   - swap halves
3. end with a swap

where k_i is a *subkey* for round i, generated from the main key k.

Feistel network



Feistel decryption

Same algorithm to decrypt, but keys in reverse order.

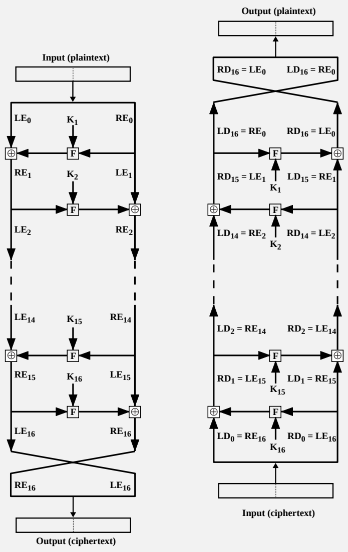


Figure 3.6 Feistel Encryption and Decryption

Feistel decryption

Same algorithm to decrypt, but keys in reverse order. Works regardless of F!

$$
\begin{aligned}
LE_{16} &= RE_{15} = RD_0 = LD_1 \\
RE_{16} &= LE_{15} \oplus F(RE_{15}, k_{16}) \\
RD1 &= LD_0 \oplus F(RD_0, k_{16}) \\
&= RE_{16} \oplus F(RE_{15}, k_{16}) \\
&= LE_{15} \oplus F(RE_{15}, k_{16}) \oplus F(RE_{15}, k_{16}) \\
&= LE_{15} \\
&\vdots \\
RD_{16} &= LE_0 \\
LD_{16} &= RE_0
\end{aligned}
$$

Feistel net parameters

- Block size (64-256 bits)
- Key size (56-256 bits)
  Larger $\Rightarrow$ greater security (diffusion) but slower.
- Number of rounds (10-16)
  One is too little, more increase security, to a limit. (Depends on F.)
- Subkey generation algorithm
- Round function F
  Both should be complex to resist cryptanalysis.

Feistel features

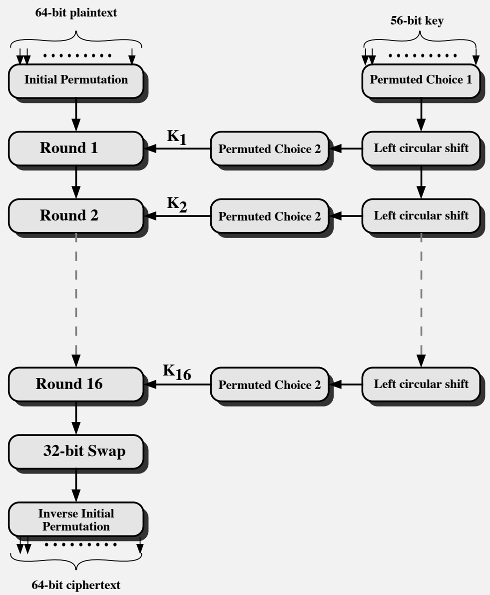
- Fast implementation
  Feistel nets can typically be implemented efficiently (fast en/decryption) in both hardware *and software*. This makes it easier to use.
- Ease of analysis
  Concise and clear description $\Rightarrow$ easier to analyse $\Rightarrow$ safer to trust.

Data Encryption Standard (1977)

No longer standard, but instructive example of a Feistel net.

- encrypts 64-bit blocks with 56-bit key
- hardware and software implementations
- one of the most analysed algorithms
  - unknown criteria for design (but more is known now)
  - US export control on implementations for long time (lifted now)
- based on Lucifer algorithm (Feistel/IBM, 1973) with 128-bit keys and blocks, but scaled down by NSA

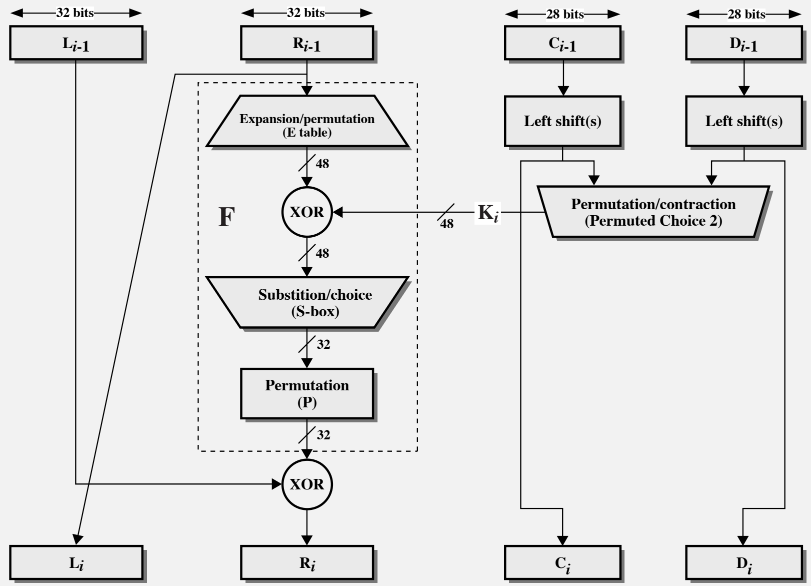
DES: Overview



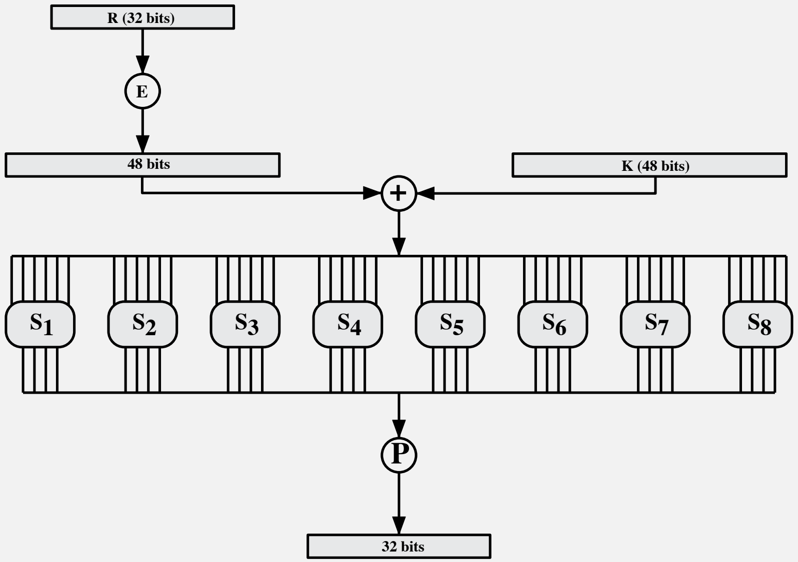
DES: subkey generation

- Each round uses subkey k_i based on k, which is 64 bits
- Permuted Choice 1 (PC1) discards parity bits (so 56 remain) and permutes
- Split in two 28-bit halves C_0, D_0
- Each round:
  - $C_i = LS_i(C_{i-1})$, $D_i = LS_i(D_{i-1})$ where LS_i is a left circular shift of $\langle 1, 1, 2, 2, \ldots, 2, 1, 2, \ldots, 2, 2, 1 \rangle$ bits
  - $k_i = PC2(C_i \cdot D_i)$ where $PC2$ is Permuted Choice 2 (permutes and discards) $56 \Rightarrow 48$ bits

DES: one round



DES: the F-box



Breaking DES by brute force

- 1977: estimated breakable in 1 day by \$20M machine
- 1981: estimated breakable in 2 days by \$50M machine
- 1997: broken in 96 days by 70 000 machines, testing 7 billion keys/s (DESCHALL project)
- 1998: less than 56 hours by special hardware, \$250K incl design and development ("Deep Crack")
- 1999: 22 h 15 min, "Deep Crack" + 100 000 machines, testing 245 billion keys/s
- 2007: 6.4 days, \$10K hardware, 120 FPGAs (COPACOBANA project)

Properties of DES

- Decryption: like Feistel (keys in reverse order)
- Symmetry:
  - $c = DES(m, k)$ iff $\overline{c} = DES(\overline{m}, \overline{k})$ where $\overline{x}$ is the bitwise negation of x
  - cuts search space in half
- Weak keys:
  - key cause involution: $E_k(E_k(m)) = m$
  - four for DES: $\langle 0, 0\rangle$, $\langle -1, 0\rangle$, $\langle 0, -1\rangle$, $\langle -1, -1\rangle$
- Semi-weak keys:
  - such that $E_{k_1}(E_{k_2}(m)) = m$
  - 6 such pairs for DES (few enough to check for)

Avalanche effect

Small changes in m or k give big changes in c, and changes increase for each round

Example: one bit change in plaintext or key

Change in plaintext		Change in key	
Round	Bits that differ	Round	Bits that differ
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
⋮	⋮	⋮	⋮
14	46	14	26
15	29	15	34
16	34	16	35

Feistel net design criteria

- S-box design
  - very careful for DES (see below)
  - in general: randomly, randomly with testing, careful hand-crafting, mathematically
- Number of rounds
  - DES: brute force requires 2^{55} tests
  - for DES with 16 rounds, *differential cryptanalysis* requires $2^{55.1}$ operations due to S-box design
  - with 15 rounds, differential cr.an. would beat brute force
  - Differential cryptanalysis invented ca 1990 – but turns out IBM and NSA had it secretly since 1974
- Round function F:
  - strict avalance criterion: any output bit changes with $p = 1/2$ if a single input bit changes
  - bit independence criterion: any two output bits should change independently when a single input bit changes

Other modern ciphers

E.g. IDEA, Blowfish, CAST, ...

- variable key length
- mixed operations (not only xor, not distributive/associative)
- data dependent rotations instead of S-boxes
- key dependent rotations and S-boxes
- variable F, block length, number of rounds
- operations on both halves

but (many) are just improvements of Feistel nets!