DES, Modes of Operation, DES-Variants, and Luby-Rackoff(2/2)

Douglas Wikström
KTH Stockholm
dog@csc.kth.se

February 2

- DES
- Modes of Operation
- Variants of DES
- Luby-Rackoff (2/2)

Quote of the Day (XKCD)

Data Encryption Standard (DES)

- Developed at IBM in 1975, or perhaps...

Data Encryption Standard (DES)

- Developed at IBM in 1975, or perhaps...
- at National Security Agency (NSA). Nobody knows for certain.

Data Encryption Standard (DES)

- Developed at IBM in 1975, or perhaps...
- at National Security Agency (NSA). Nobody knows for certain.
- 16-round Feistel network.

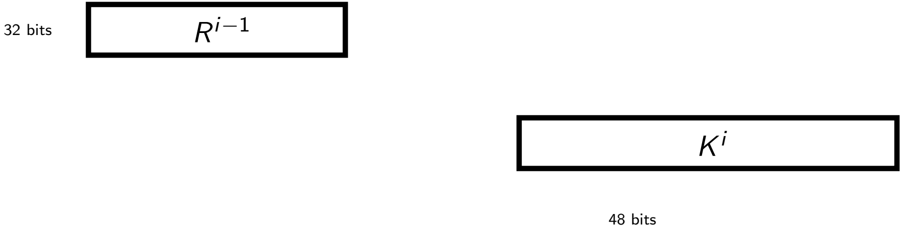
Data Encryption Standard (DES)

- Developed at IBM in 1975, or perhaps...
- at National Security Agency (NSA). Nobody knows for certain.
- 16-round Feistel network.
- Key schedule derives permuted bits for each round key from a 56-bit key. Supposedly not 64-bit due to parity bits.

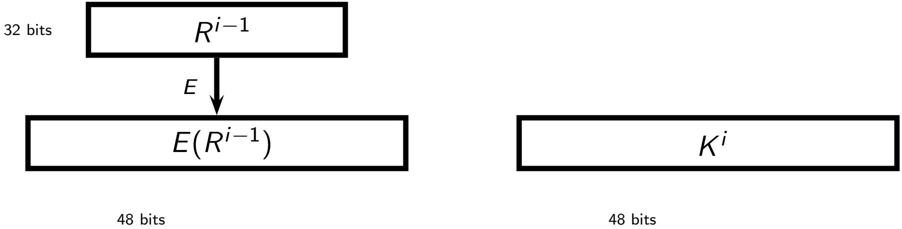
Data Encryption Standard (DES)

- Developed at IBM in 1975, or perhaps...
- at National Security Agency (NSA). Nobody knows for certain.
- 16-round Feistel network.
- Key schedule derives permuted bits for each round key from a 56-bit key. Supposedly not 64-bit due to parity bits.
- Let us look a little at the Feistel-function f.

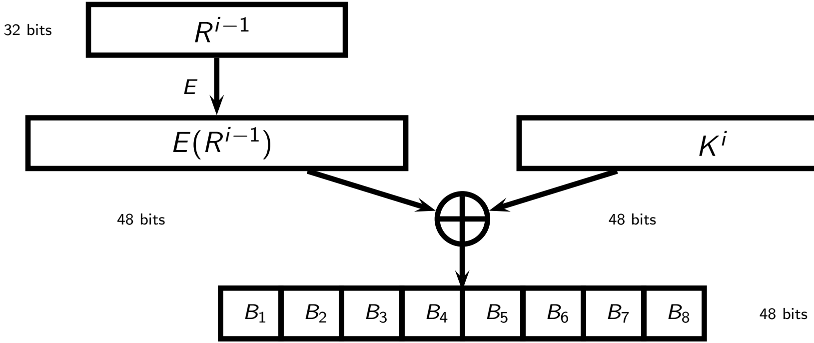
DES' f-Function



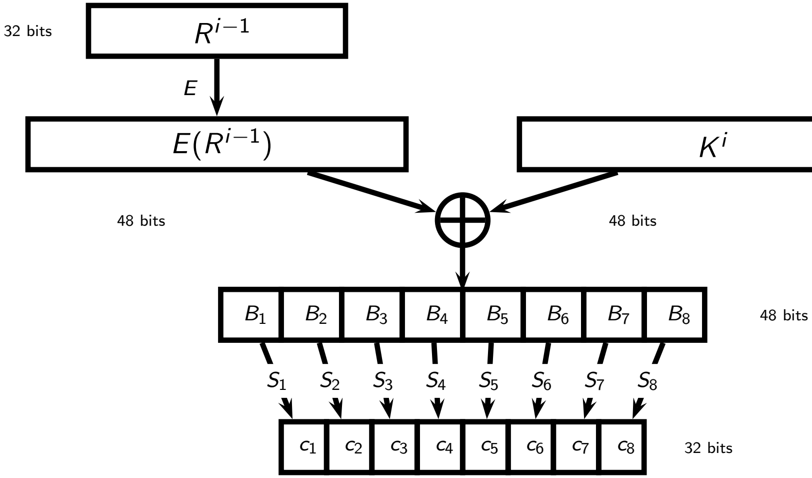
DES' f-Function



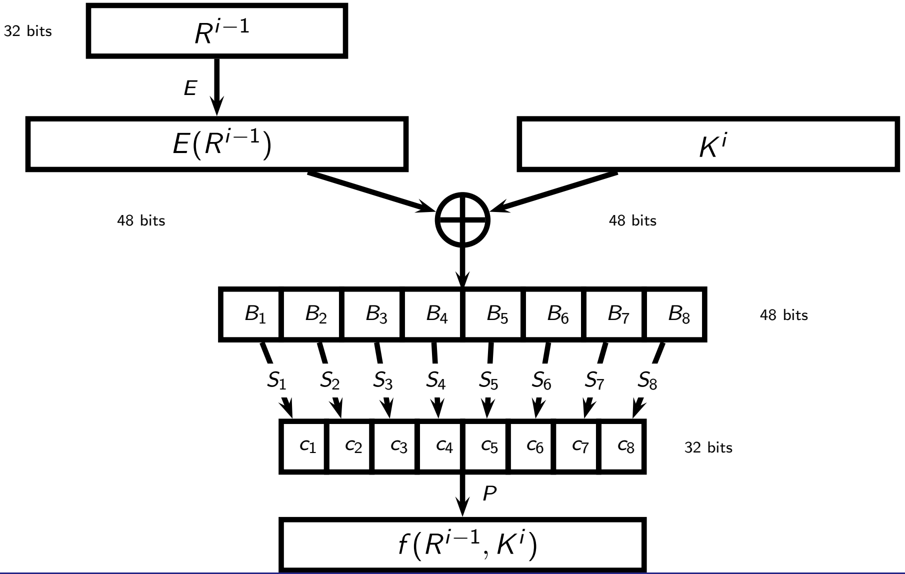
DES' f-Function



DES' f-Function



DES' f-Function



Security of DES

- **Brute Force.** Try all 2^{56} keys. Done in practice with special chip by Electronic Frontier Foundation, 1998. Likely much earlier by NSA and others.
- **Differential Cryptanalysis.** 2^{47} chosen plaintexts, Biham and Shamir, 1991. (approach: late 80'ies). Known earlier by IBM and NSA. DES is surprisingly resistant!
- **Linear Cryptanalysis.** 2^{43} known plaintexts, Matsui, 1993. Probably **not** known by IBM and NSA!

Modes of Operation

- Electronic codebook mode (ECB mode).
- Cipher feedback mode (CFB mode).
- Cipher block chaining mode (CBC mode).
- Output feedback mode (OFB mode).

ECB Mode

Encrypt each block independently:

$$c_i = \mathrm{E}_k(m_i)$$

ECB Mode

Encrypt each block independently:

$$c_i = \mathrm{E}_k(m_i)$$

- Identical plaintext blocks give identical ciphertext blocks.

ECB Mode

Encrypt each block independently:

$$c_i = \mathrm{E}_k(m_i)$$

- Identical plaintext blocks give identical ciphertext blocks.
- How can we avoid this?

CFB Mode

xor plaintext block with previous ciphertext block **after** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = m_i \oplus \mathsf{E}_k(c_{i-1})$$

CFB Mode

xor plaintext block with previous ciphertext block **after** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = m_i \oplus \mathsf{E}_k(c_{i-1})$$

- Sequential.

CFB Mode

xor plaintext block with previous ciphertext block **after** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = m_i \oplus \mathsf{E}_k(c_{i-1})$$

- Sequential.
- Self-synchronizing.

CBC Mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k\left(c_{i-1} \oplus m_i\right)$$

CBC Mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k\left(c_{i-1} \oplus m_i\right)$$

- Sequential.

CBC Mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k(c_{i-1} \oplus m_i)$$

- Sequential.
- Self-synchronizing.

OFB Mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$s_0 = \text{initialization vector}$$
$$s_i = \mathsf{E}_k(s_{i-1})$$
$$c_i = s_i \oplus m_i$$

OFB Mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathrm{E}_k(s_{i-1}) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Sequential.

OFB Mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$s_0 = \text{initialization vector}$$
$$s_i = \mathrm{E}_k(s_{i-1})$$
$$c_i = s_i \oplus m_i$$

- Sequential.
- Synchronous.

OFB Mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$s_0 = \text{initialization vector}$$
$$s_i = \mathsf{E}_k(s_{i-1})$$
$$c_i = s_i \oplus m_i$$

- Sequential.
- Synchronous.
- Allows batch processing.

OFB Mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$\begin{aligned} s_0 &= \text{initialization vector} \\ s_i &= \mathsf{E}_k(s_{i-1}) \\ c_i &= s_i \oplus m_i \end{aligned}$$

- Sequential.
- Synchronous.
- Allows batch processing.
- Malleable!

Double DES

We have seen that the key space of DES is too small. One way to increase it is to use DES twice, so called "double DES".

$$\mathrm{2DES}_{k_1,k_2}(x) = \mathrm{DES}_{k_2}(\mathrm{DES}_{k_1}(x))$$

Is this more secure than DES?

Meet-In-the-Middle Attack

- Get hold of a plaintext-ciphertext pair (m, c)
- Compute $X = \{c \mid k_1 \in \mathcal{K}_{\mathrm{DES}} \wedge c = \mathsf{E}_{k_1}(m)\}$.
- For $k_2 \in \mathcal{K}_{\mathrm{DES}}$ check if $\mathsf{E}_{k_2}^{-1}(c) = \mathsf{E}_{k_1}(m) \in \mathcal{K}$, then (k_1, k_2) is a good candidate.
- Repeat with (m', c'), starting from the set of candidate keys to identify correct key.

Triple DES

What about triple DES?

$$3\mathrm{DES}_{k_1,k_2,k_3}(x) = \mathrm{DES}_{k_3}(\mathrm{DES}_{k_2}(\mathrm{DES}_{k_1}(x)))$$

- Seemingly 112 bit "effective" key size.
- 3 times as slow as DES. DES is slow in software, and this is even worse. One of the motivations of AES.

DESX

DESX

$$\mathrm{DESX}_{k_1,k_2,k_3}(x) = k_1 \oplus \mathrm{DES}_{k_3}(x \oplus k_2)$$

- Seemingly stronger against brute-force attack.
- Not stronger against differential/linear cryptanalysis, since xor is linear.
- The use of the additional keys are called “whitening”.

Negligible Functions

Definition. A function $\epsilon(n)$ is negligible if for every constant $c > 0$, there exists a constant n_0, such that

$$\epsilon(n) < \frac{1}{n^c}$$

for all $n \geq n_0$.

Motivation. Events happening with negligible probability can not be exploited by polynomial time algorithms! (they "never" happen)

Pseudo-Random Function

"Definition". A function is pseudo-random if no efficient adversary can distinguish between the function and a random function.

Pseudo-Random Function

"Definition". A function is pseudo-random if no efficient adversary can distinguish between the function and a random function.

Definition. A family of functions $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is pseudo-random if for all polynomial time oracle adversaries A

$$\left| \Pr_K \left[A^{F_K(\cdot)} = 1 \right] - \Pr_{R:\{0,1\}^n \to \{0,1\}^n} \left[A^{R(\cdot)} = 1 \right] \right|$$

is negligible.

Pseudo-Random Permutation

"Definition". A permutation and its inverse is pseudo-random if no efficient adversary can distinguish between the permutation and its inverse, and a random permutation and its inverse.

Pseudo-Random Permutation

"Definition". A permutation and its inverse is pseudo-random if no efficient adversary can distinguish between the permutation and its inverse, and a random permutation and its inverse.

Definition. A family of permutations $P : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ are pseudo-random if for all polynomial time oracle adversaries A

$$\left| \Pr_K \left[A^{P_K(\cdot), P_K^{-1}(\cdot)} = 1 \right] - \Pr_{\Pi \in \mathcal{S}_{2^n}} \left[A^{\Pi(\cdot), \Pi^{-1}(\cdot)} = 1 \right] \right|$$

is negligible, where $\mathcal{S}_{2^n}$ is the set of permutations of $\{0,1\}^n$.

Idealized Four-Round Feistel Network

Definition. Feistel round (H for "Horst Feistel").

$$H_f(L, R) = (R, L \oplus f(R, K))$$

Idealized Four-Round Feistel Network

Definition. Feistel round (H for "Horst Feistel").

$$H_f(L, R) = (R, L \oplus f(R, K))$$

Theorem. (Luby and Rackoff) If F is a pseudo-random family of functions, then

$$H_{F_{k_1},F_{k_2},F_{k_3},F_{k_4}}(x) = H_{F_{k_4}}(H_{F_{k_3}}(H_{F_{k_2}}(H_{F_{k_1}}(x))))$$

(and its inverse) is a pseudo-random family of permutations.

Idealized Four-Round Feistel Network

Definition. Feistel round (H for "Horst Feistel").

$$H_f(L, R) = (R, L \oplus f(R, K))$$

Theorem. (Luby and Rackoff) If F is a pseudo-random family of functions, then

$$H_{F_{k_1},F_{k_2},F_{k_3},F_{k_4}}(x) = H_{F_{k_4}}(H_{F_{k_3}}(H_{F_{k_2}}(H_{F_{k_1}}(x))))$$

(and its inverse) is a pseudo-random family of permutations.

Why do we need four rounds?