Introduction to Cryptology

4.2 - DES

Federico Pintore

Mathematical Institute, University of Oxford (UK)

The Data Encryption Standard (DES)

DES is a 16-round Feistel Network, where:

- the block length ℓ is 64;
- the key length n is 56;
- the key schedule derives 16 sub-keys of 48-bit size, $k_1, \cdots, k_{16}$, from the key k.

The Data Encryption Standard (DES)

DES is a 16-round Feistel Network, where:

- the block length ℓ is 64;
- the key length n is 56;
- the key schedule derives 16 sub-keys of 48-bit size, $k_1, \cdots, k_{16}$, from the key k.

A simple animation to illustrate DES:
https://kathrynneugent.com/des-animation/

The Data Encryption Standard (DES)

- A mixing permutation IP precedes the first round, while its inverse follows the last one.
- The key is specified as a 64-bit string, but 8 bits are discarded or used as parity check bits;
- the 56 bits of the key are selected with the Permuted Choice 1 (PC-1) and split into two 28-bit strings: C and D;
- in each round, C and D are rotated to the left by one or two steps (specified for each round);
- each 48-bit sub-key is constructed taking 24 bits from C and 24 from D, by means of the Permuted Choice 2 (PC-2).

The Data Encryption Standard (DES)

- In each of the 16 rounds, a round function

$$f_i : \{0,1\}^{32} \times \{0,1\}^{48} \to \{0,1\}^{32}$$

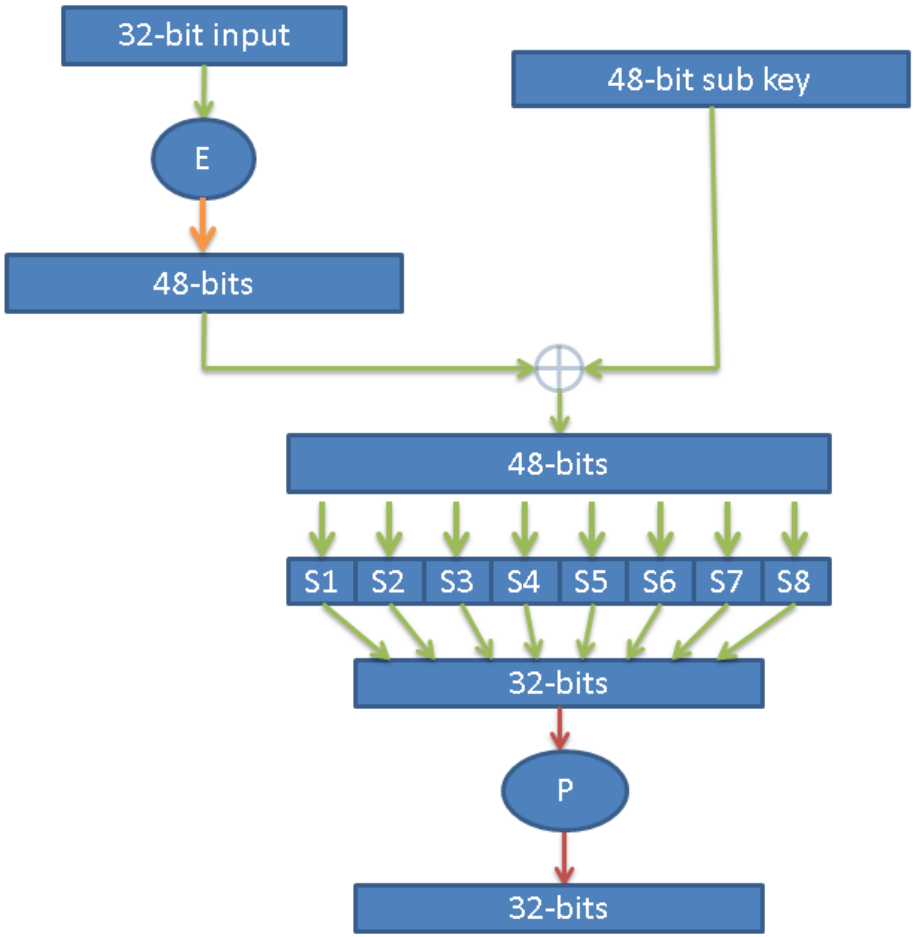
  is used;

- an expansion function $E : \{0,1\}^{32} \to \{0,1\}^{48}$ expands the 32-bit input of f_i. The output is xor'ed with the sub-key k_i;

- f_i uses 8 different and non invertible S-boxes, $S_1, \cdots, S_8$, where S_i takes a 6-bit input and produces a 4-bit output.

- the execution of f_i ends with a 32-bit mixing permutation P.

The Data Encryption Standard (DES)



The DES function

The Data Encryption Standard (DES)

An S-box takes a 6-bit input:

- the first bit and the last one identify the row;
- the middle bits identify the column;
- the output is the entry of the identified cell.

The Data Encryption Standard (DES)

An S-box takes a 6-bit input:

- the first bit and the last one identify the row;
- the middle bits identify the column;
- the output is the entry of the identified cell.

Here is an example for the input 011011:

S_5		Middle 4 bits of input															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Outer bits	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Security of DES

- 1970: Horst Feistel designs Lucifer (precursor of DES) at IBM, with $n = \ell = 128$.
- 1976: NIST (at that time NBS) adopts DES as a federal standard, with $n = 56, \ell = 64$.
- 1997: first successful brute-force attack on DES (DESCHALL project, approximately 96 days of computation).
- State-of-the-art: brute-force attack takes less than a day.

Security of DES

- 1970: Horst Feistel designs Lucifer (precursor of DES) at IBM, with $n = \ell = 128$.
- 1976: NIST (at that time NBS) adopts DES as a federal standard, with $n = 56, \ell = 64$.
- 1997: first successful brute-force attack on DES (DESCHALL project, approximately 96 days of computation).
- State-of-the-art: brute-force attack takes less than a day.

The key length used by DES is too short!

Security of DES

Anything better than brute-force?

Differential cryptanalysis (Biham-Shamir, late 1980s):

- time 2^{37} (DES computations),
- it requires 2^{47} chosen plaintexts.

Security of DES

Anything better than brute-force?

Differential cryptanalysis (Biham-Shamir, late 1980s):

- time 2^{37} (DES computations),
- it requires 2^{47} chosen plaintexts.

Linear cryptanalysis (Matsui, mid 1990s):

- time 2^{43},
- it requires 2^{42} known plaintexts.

2DES

The main problem of DES is its short key length.

2DES

The main problem of DES is its short key length.

- Changing the internal structure of DES is not recommended.
- What if we double the key length defining

$$F'_{k_1,k_2} \leftarrow F_{k_2} \circ F_{k_1}\,?$$

- Not a great idea! A meet-in-the-middle attack takes time $\mathcal{O}(n \cdot 2^n)$ and requires space $\mathcal{O}((n+\ell) \cdot 2^n)$.

2DES: meet-in-the-middle attack

Suppose a pair $(x, y = F_{k_2^*}(F_{k_1^*}(x)))$ is known.

2DES: meet-in-the-middle attack

Suppose a pair $(x, y = F_{k_2^*}(F_{k_1^*}(x)))$ is known.

Maintain two lists, L_1 and M, as follows:

- $\forall k_1 \in \{0,1\}^n$, compute $z \leftarrow F_{k_1}(x)$, and store (z, k_1) in L_1;
- $\forall k_2 \in \{0,1\}^n$, compute $z \leftarrow F_{k_2}^{-1}(y)$. If there exists (z, k_1) in L_1, store (k_1, k_2) in M.

2DES: meet-in-the-middle attack

Suppose a pair $(x, y = F_{k_2^*}(F_{k_1^*}(x)))$ is known.

Maintain two lists, L_1 and M, as follows:

- $\forall k_1 \in \{0,1\}^n$, compute $z \leftarrow F_{k_1}(x)$, and store (z, k_1) in L_1;
- $\forall k_2 \in \{0,1\}^n$, compute $z \leftarrow F_{k_2}^{-1}(y)$. If there exists (z, k_1) in L_1, store (k_1, k_2) in M.

2DES: meet-in-the-middle attack

Suppose a pair $(x, y = F_{k_2^*}(F_{k_1^*}(x)))$ is known.

Maintain two lists, L_1 and M, as follows:

- $\forall k_1 \in \{0,1\}^n$, compute $z \leftarrow F_{k_1}(x)$, and store (z, k_1) in L_1;
- $\forall k_2 \in \{0,1\}^n$, compute $z \leftarrow F_{k_2}^{-1}(y)$. If there exists (z, k_1) in L_1, store (k_1, k_2) in M.

$(k_1^*, k_2^*) \in M$ and it can be identified with very high probability.

3DES

Two possible versions:

1. Choose independent keys $k_1, k_2, k_3 \in \{0,1\}^n$ and define

$$F''_{k_1,k_2,k_3} \leftarrow F_{k_3} \circ F_{k_2}^{-1} \circ F_{k_1}$$

   Meet-in-the-middle attack takes time 2^{2n}.

3DES

Two possible versions:

1. Choose independent keys $k_1, k_2, k_3 \in \{0,1\}^n$ and define

$$F''_{k_1,k_2,k_3} \leftarrow F_{k_3} \circ F_{k_2}^{-1} \circ F_{k_1}$$

   Meet-in-the-middle attack takes time 2^{2n}.

2. Use two keys $k_1, k_2 \in \{0,1\}^n$ and define

$$F''_{k_1,k_2} \leftarrow F_{k_1} \circ F_{k_2}^{-1} \circ F_{k_1}$$

   Best attack takes time 2^{2n}.

Security of 3DES

3DES was standardised in 1999.

- Drawbacks: it has a small block length and it runs slowly (it requires three block cipher executions!).
- The best security level that it can offer is 2^{112}, whereas the usual recommendation is 2^{128}.

Can DES be used to achieve higher security levels? Check this:

http://www.iacr.org/conferences/eurocrypt2012/Rump/shamir.pdf

Further Reading I

Nadhem J AlFardan, Daniel J Bernstein, Kenneth G Paterson, Bertram Poettering, and Jacob CN Schuldt.
On the security of RC4 in TLS.
In 22nd USENIX Security Symposium (USENIX Security 13), pages 305–320, 2013.

Boaz Barak and Shai Halevi.
A model and architecture for pseudo-random generation with applications to/dev/random.
In Proceedings of the 12th ACM conference on Computer and communications security, pages 203–212. ACM, 2005.

Daniel J Bernstein.
The Salsa20 Family of Stream Ciphers.
In New stream cipher designs, pages 84–97. Springer, 2008.

Further Reading II


Lenore Blum, Manuel Blum, and Mike Shub.
A simple unpredictable pseudo-random number generator.
SIAM Journal on computing, 15(2):364–383, 1986.


Christian Cachin.
Entropy measures and unconditional security in cryptography.
PhD thesis, ETH Zurich, 1997.


Scott Fluhrer, Itsik Mantin, and Adi Shamir.
Weaknesses in the key scheduling algorithm of RC4.
In Selected areas in cryptography, pages 1–24. Springer, 2001.

Further Reading III

Christina Garman, Kenneth G Paterson, and Thyla Van der Merwe.
Attacks only get better: Password recovery attacks against RC4 in TLS.
In 24th USENIX Security Symposium (USENIX Security 15), pages 113–128, 2015.

Itsik Mantin and Adi Shamir.
A practical attack on broadcast RC4.
In Fast Software Encryption, pages 152–164. Springer, 2002.