Fundamentals of Cryptography

David Jao

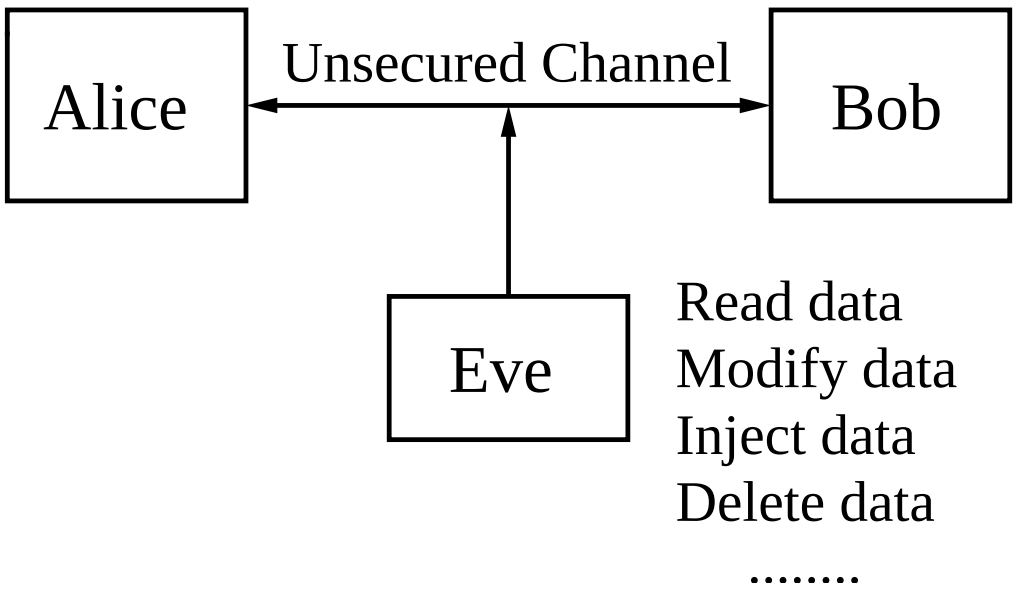
CryptoWorks21

UNIVERSITY OF WATERLOO

August 7–10, 2018

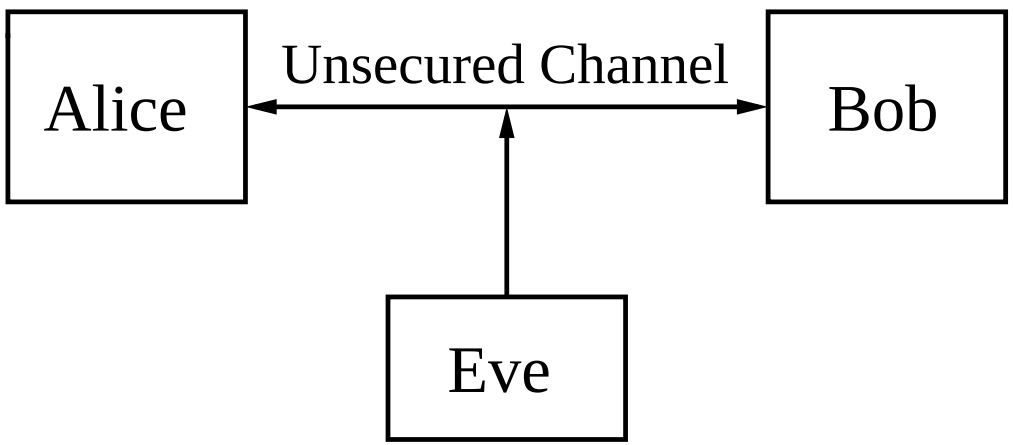
What is cryptography?

Cryptography is about securing communication in the presence of malicious adversaries.



Fundamental goals of cryptography

- Confidentiality: Keep data secret from all but those authorized to see it
- Data integrity: Ensure data has not been altered by unauthorized means
- Data origin authentication: Corroborate the source of data
- Non-repudiation: Prevent an entity from denying previous commitments or actions



Cryptography vs. security

Information security encompasses the concepts, techniques, technical measures, and administrative measures used to protect information assests from deliberate or inadvertent unauthorized acquisition, damage, disclosure, manipulation, modification, loss, or use.

The real challenge is an engineering one: building high confidence systems which

- Behave in a well-understood and predictable fashion
- Withstand malicious attacks as well as naturally occurring hazards
- Must not cause or contribute to accidents or unacceptable losses

Cryptography vs. security

Information security includes the study of subjects like:

- Computer security
- Network security
- Software security

Cryptography $\neq$ Security

- Cryptography provides some mathematical tools that can assist with the provision of information security services. It is a small but essential part of a complete solution.
- Security is a chain
  - Weak links become targets
  - One flaw is all it takes (Door locks $\neq$ Home security)
  - Cryptography is usually not the weakest link (however, when the crypto fails the damage can be catastrophic)

Part I

Symmetric key cryptosystems

Basic concepts

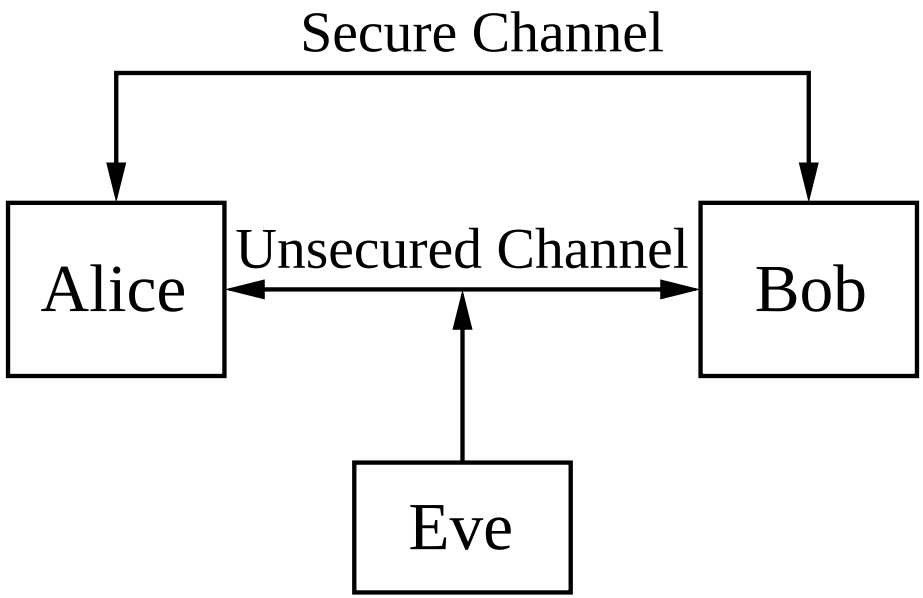
Definition: A symmetric-key encryption scheme (SKES) consists of:

- M – the plaintext space,
- C – the ciphertext space,
- K – the key space,
- a family of encryption functions, $E_k \colon M \to C$, $\forall k \in K$,
- a family of decryption functions, $D_k \colon C \to M$, $\forall k \in K$,

such that $D_k(E_k(m)) = m$ for all $m \in M$, $k \in K$.

Equivalently: $E \colon K \times M \to C$ and $D \colon K \times C \to M$.

Using a SKES to achieve confidentiality



1. Alice and Bob agree on a secret key $k \in K$ by communicating over the secure channel.
2. Alice computes $c = E_k(m)$ and sends the ciphertext c to Bob over the unsecured channel.
3. Bob retrieves the plaintext by computing $m = D_k(c)$.

What Does it Mean for a SKES to be Secure?

Three key questions:

1. How does the adversary interact with the communicating parties?
2. What are the computational powers of the adversary?
3. What is the adversary's goal?

- Basic assumption: The adversary knows everything about the SKES, except the particular key k chosen by Alice and Bob. (Avoid security by obscurity!!)

- Security model: Defines the computational abilities of the adversary, and how she interacts with the communicating parties.

1. Adversary's Interaction

- Passive attacks:
  - Ciphertext-only attack.
  - Known-plaintext attack: The adversary also knows some plaintext and the corresponding ciphertext.
- Active attacks:
  - Chosen-plaintext attack: The adversary can also choose some plaintext(s) and obtain the corresponding ciphertext(s).
  - Chosen-ciphertext attack: The adversary can also choose some ciphertext(s) and obtain the corresponding plaintext(s).
- Other attacks:
  - Side-channel attacks: monitor the encryption and decryption equipment (timing attacks, power analysis attacks, electromagnetic-radiation analysis, etc.)
  - Physical attacks: bribery, blackmail, rubber hose, etc.

2. Computational Power of the Adversary

- Information-theoretic security: Eve has infinite computational resources.
- Complexity-theoretic security: Eve is a "polynomial-time Turing machine".
- Computational security: Eve has X number of real computers/workstations/supercomputers. (Eve is "computationally bounded")

3. Adversary's Goal

1. Recover the secret key.
2. Systematically recover plaintext from ciphertext (without necessarily learning the secret key).
3. Learn some partial information about the plaintext from the ciphertext (other than its length).

- If the adversary can achieve 1 or 2, the SKES is said to be totally insecure (or totally broken).
- If the adversary cannot learn any partial information about the plaintext from the ciphertext (except possibly its length), the SKES is said to be semantically secure.

Hiding length information is very hard. This topic falls under the heading of *traffic analysis*.

Definition of a Secure SKES

Definition

A symmetric-key encryption scheme is said to be secure if it is semantically secure against a chosen-plaintext attack by a computationally bounded adversary.

To break a symmetric-key encryption scheme, the adversary has to accomplish the following:

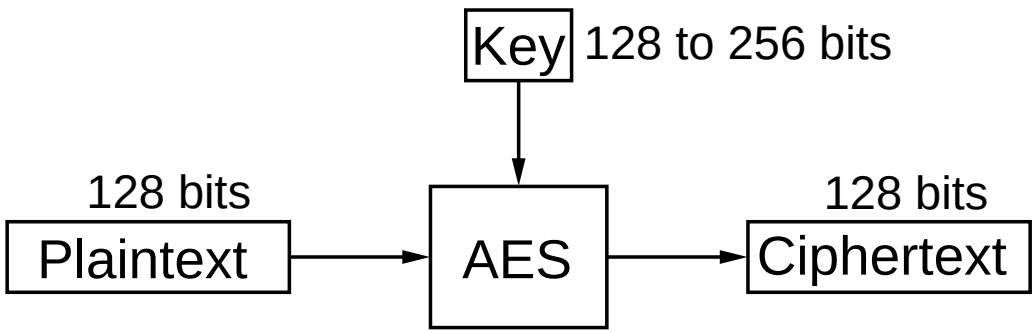
1. The adversary is given a challenge ciphertext c (generated by Alice or Bob using their secret key k).
2. During its computation, the adversary can select plaintext and obtains (from Alice or Bob) the corresponding ciphertext.
3. After a feasible amount of computation, the adversary obtains some information about the plaintext corresponding to c (other than the length of m).

Part II

Block ciphers

Block ciphers and stream ciphers

- A block cipher is a SKES which breaks up the plaintext into blocks of a fixed length (e.g. 128 bits), and encrypts the blocks one at a time.
- In contrast, a stream cipher encrypts the plaintext one character (usually a bit) at a time.
- Example of a block cipher:
  The Advanced Encryption Standard (AES)



Key size: 128 to 256 bits; Size of key space: 2^{128} to 2^{256}; Block size: 128 bits.

20th-century history of symmetric-key encryption schemes

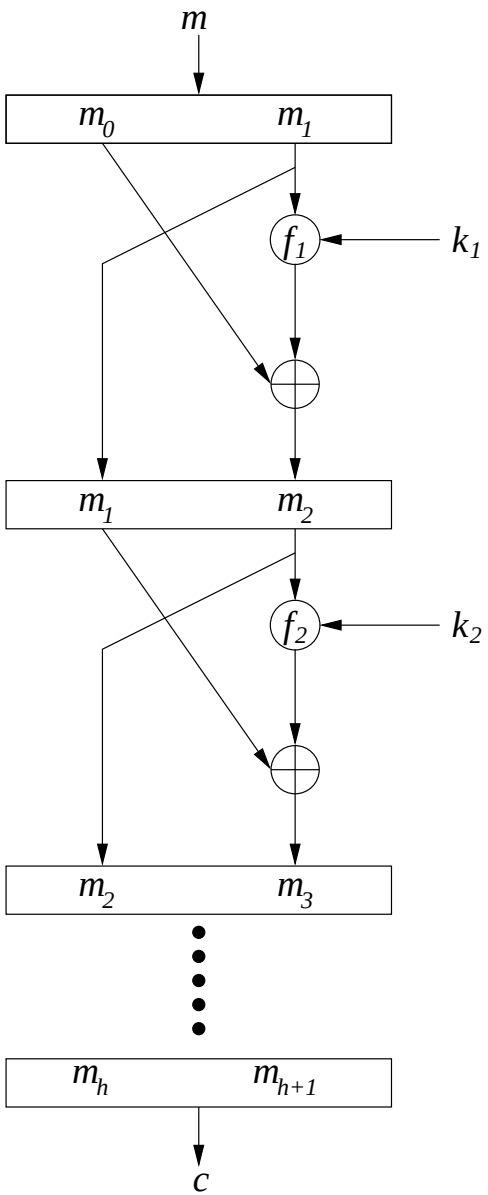
- One-time pad (Vernam, 1917)
- Electro-mechanical (stream) ciphers of World War II:
  - Enigma (Germany)
  - Lorenz (Germany)
  - PURPLE (Japan)
  - Navajo code (USA)
- Late 1960's: Feistel network, substitution-permutation network, and LUCIFER designed at IBM.

20th-century history of symmetric-key encryption schemes

- 1972: NBS (now NIST: National Institute of Standards and Technology) solicits proposals for encryption algorithms for the protection of computer data.
- 1974: IBM submits a variant of Lucifer (based on a Feistel network) as a DES candidate.
- 1975: NSA (National Security Agency) (allegedly) "fixes" DES
  - Reduces the key size from 64 bits to 56 bits.

    "We sent the S-boxes off to Washington. They came back and were all different."
- 1977: DES adopted as US Federal Information Processing Standard (FIPS 46).
- 1981: DES adopted as a US banking standard (ANSI X3.92).

The Feistel network design



- DES uses a Feistel network design.
- Plaintext is divided into two halves.
- Key is used to generate subkeys $k_1, k_2, \ldots, k_h$
- f_i is a *component function* whose output value depends on k_i and m_i

Components of a Feistel Cipher

Encryption takes h rounds:

- Plaintext is $m = (m_0, m_1)$, where $m_i \in \{0,1\}^n$.
- Round 1: $(m_0, m_1) \mapsto (m_1, m_2)$, where $m_2 = m_0 \oplus f_1(m_1)$.
- Round 2: $(m_1, m_2) \mapsto (m_2, m_3)$, where $m_3 = m_1 \oplus f_2(m_2)$.

  $\vdots$

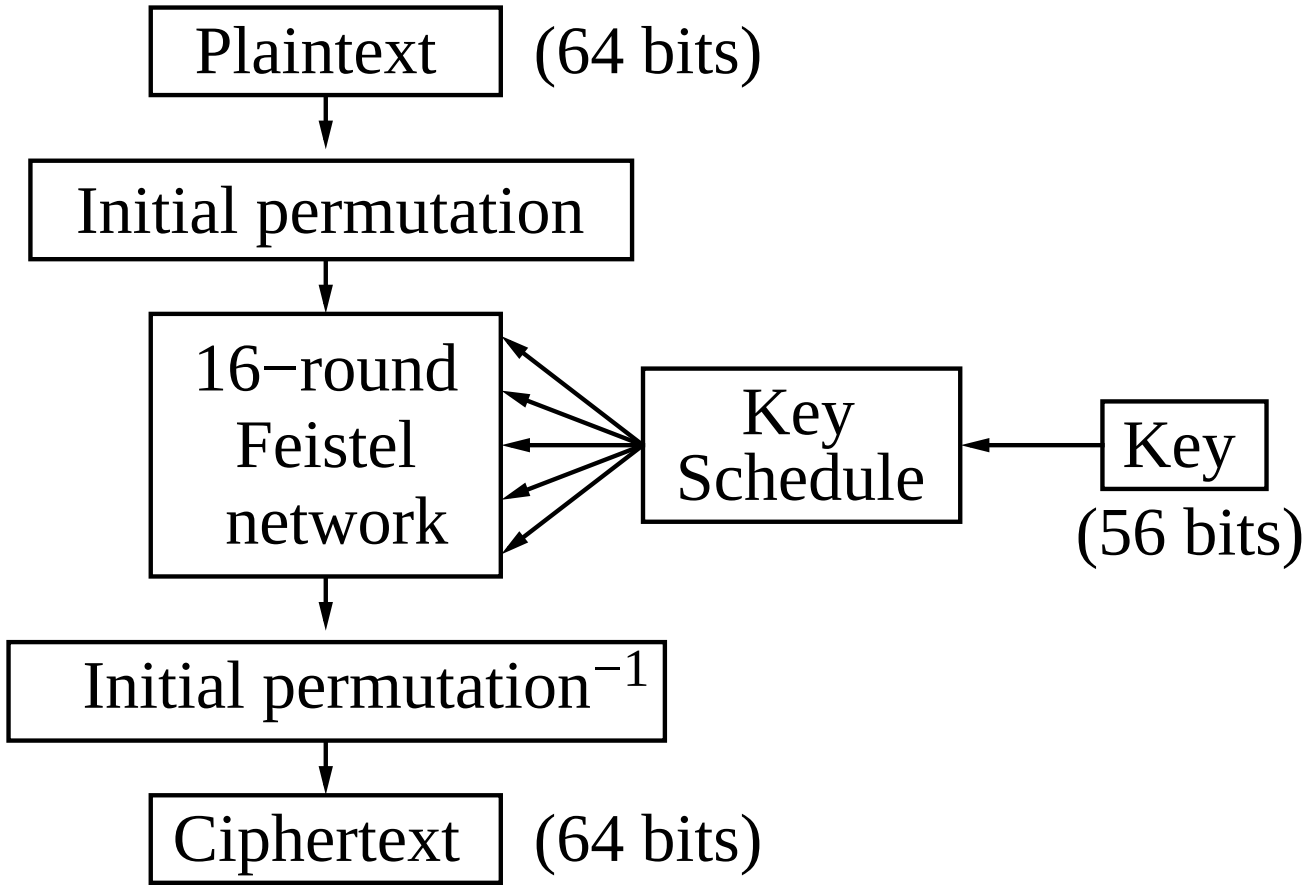
- Round h: $(m_{h-1}, m_h) \mapsto (m_h, m_{h+1})$, where $m_{h+1} = m_{h-1} \oplus f_h(m_h)$.
- Ciphertext is $c = (m_h, m_{h+1})$.

Decryption: Given $c = (m_h, m_{h+1})$ and k, to find $m = (m_0, m_1)$:

- Compute $m_{h-1} = m_{h+1} \oplus f_h(m_h)$.
- Similarly, compute $m_{h-2}, \ldots, m_1, m_0$.

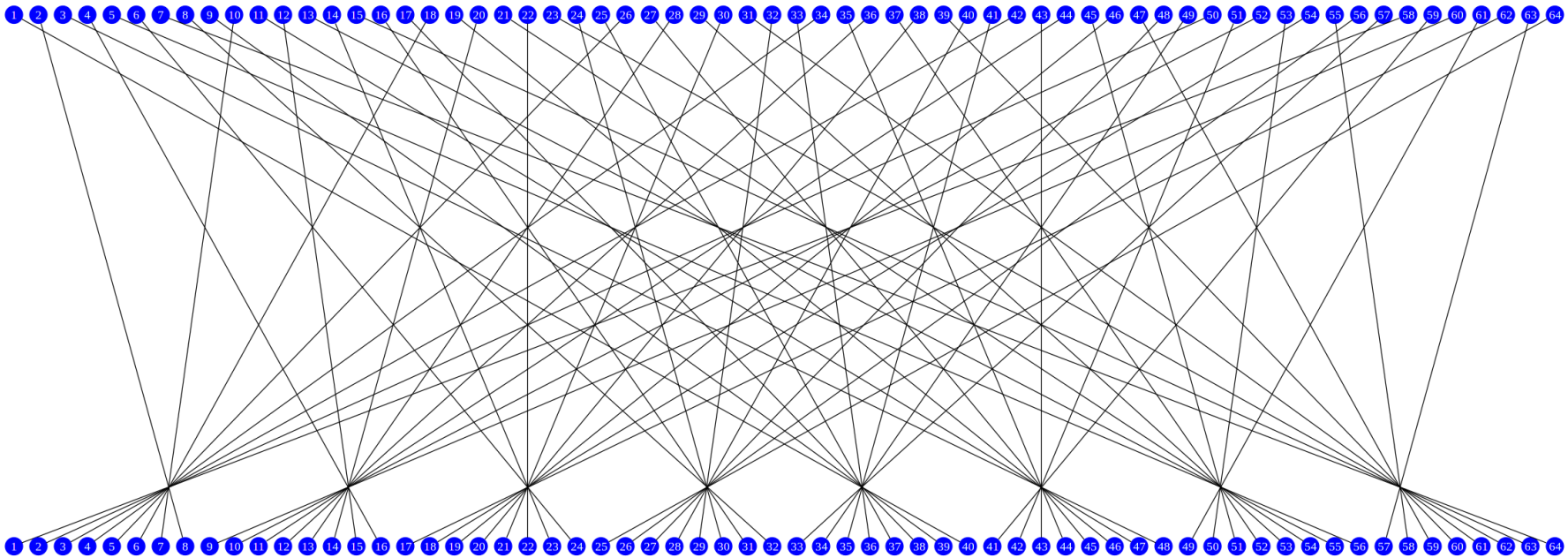
Overview of DES

Feistel cipher with $n = 32$, $h = 16$, $\ell = 56$.



Initial permutation

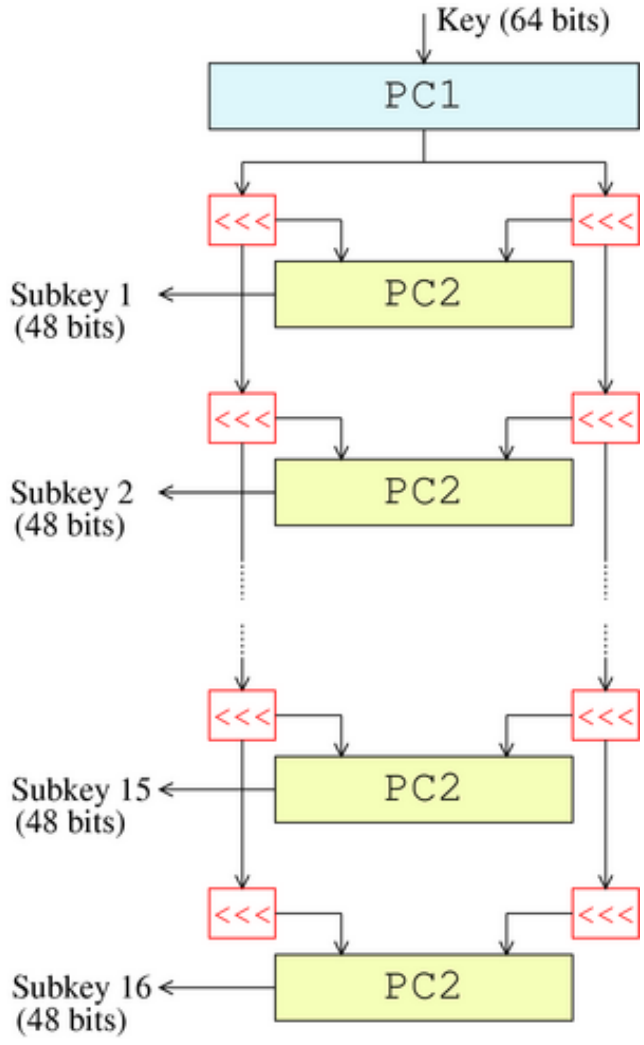
In	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Out	40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
In	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Out	38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
In	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Out	36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
In	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
Out	34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25



Credit: Wikipedia

Key scheduling algorithm

Round number	Left shift each half by this many bits
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1



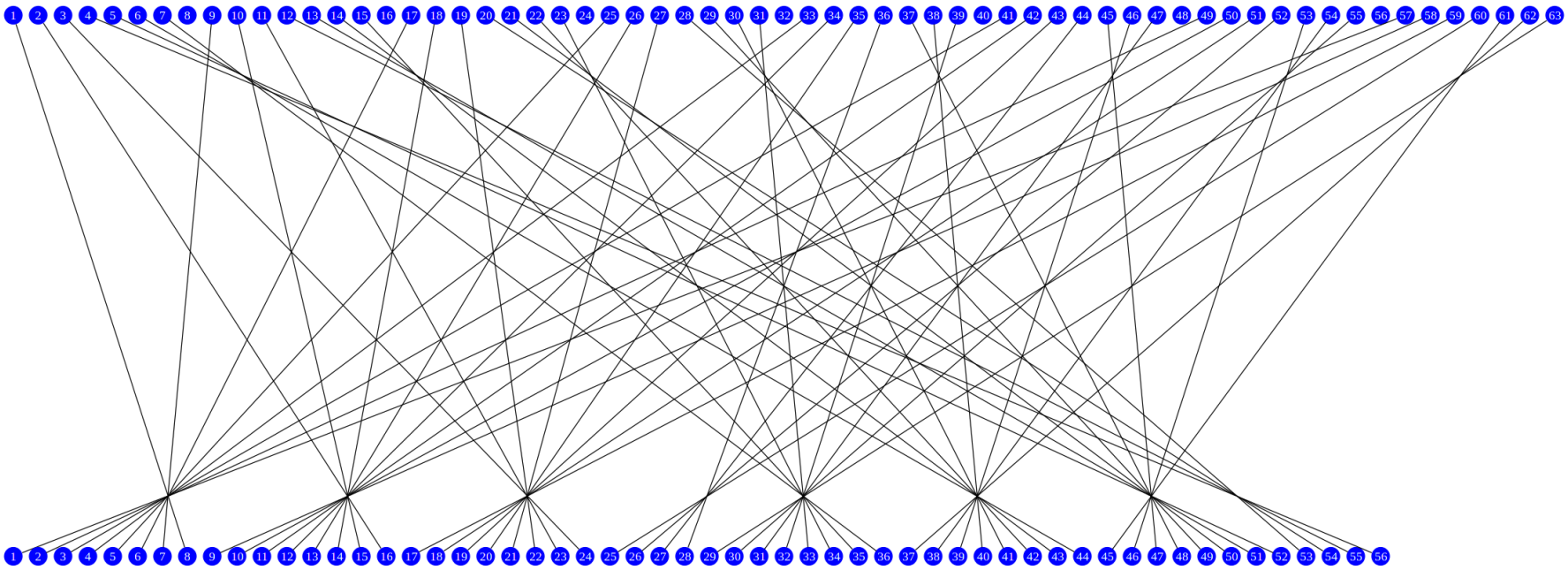
Permuted Choice #1

In	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Out	8	16	24	56	52	44	36		7	15	23	55	51	43	35	

In	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Out	6	14	22	54	50	42	34		5	13	21	53	49	41	33	

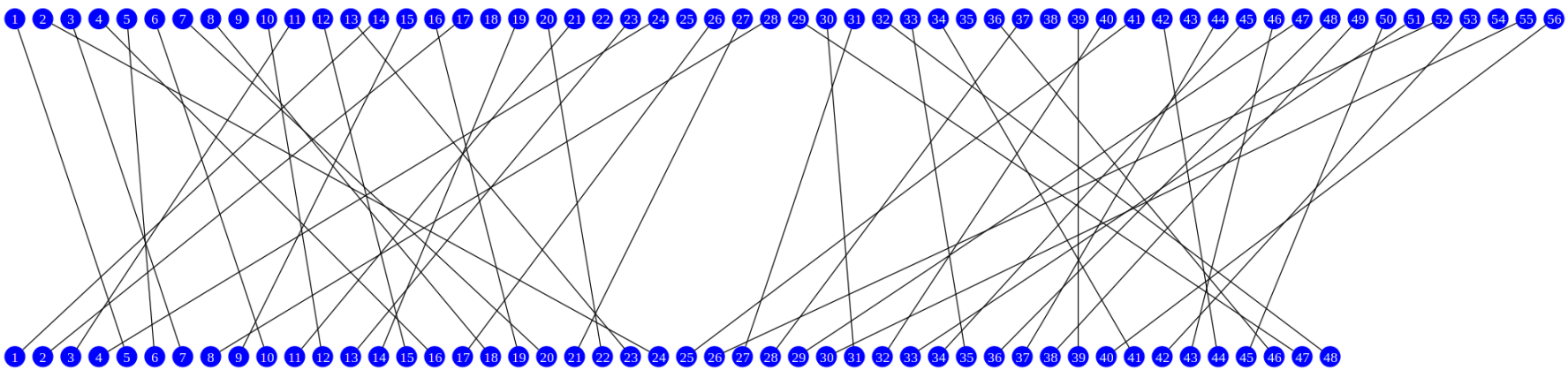
In	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Out	4	12	20	28	48	40	32		3	11	19	27	47	39	31	

In	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
Out	2	10	18	26	46	38	30		1	9	17	25	45	37	29	



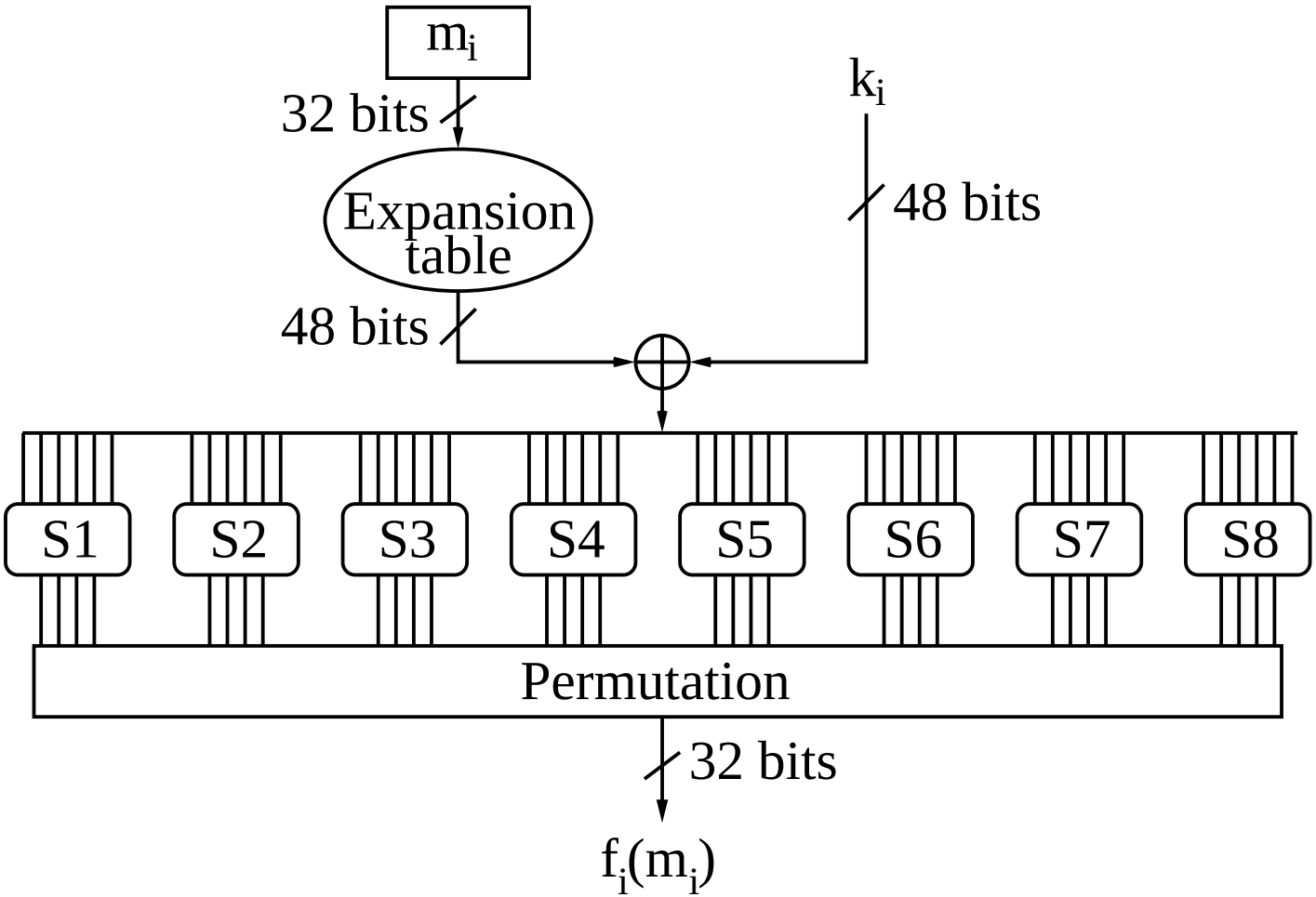
Permuted Choice #2

In	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Out	5	24	7	16	6	10	20	18		12	3	15	23	1
In	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Out	9	19	2		14	22	11		13	4		17	21	8
In	29	30	31	32	33	34	35	36	37	38	39	40	41	42
Out	47	31	27	48	35	41		46	28		39	32	25	44
In	43	44	45	46	47	48	49	50	51	52	53	54	55	56
Out		37	34	43	29	36	38	45	33	26	42		30	40



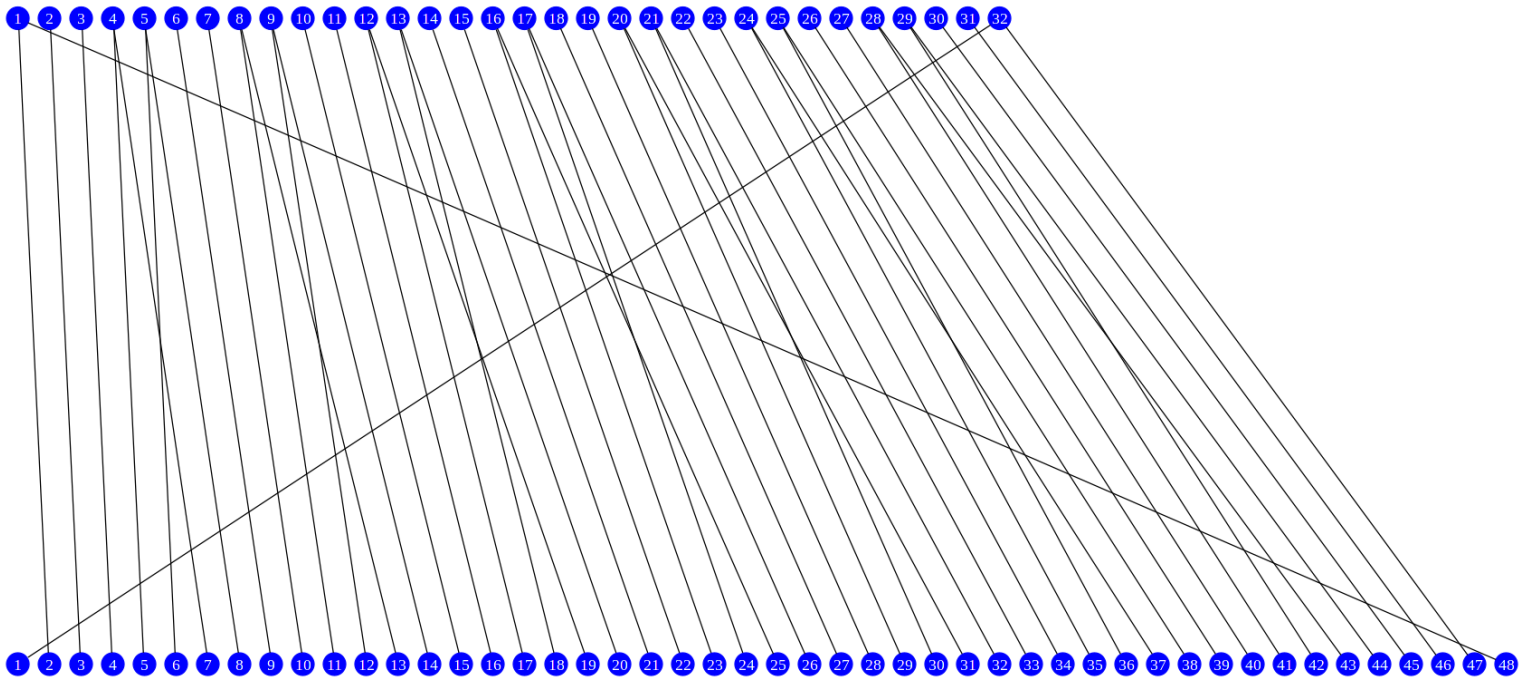
Structure of the component functions

Recall $f_i : \{0,1\}^{32} \to \{0,1\}^{32}$.



Expansion table

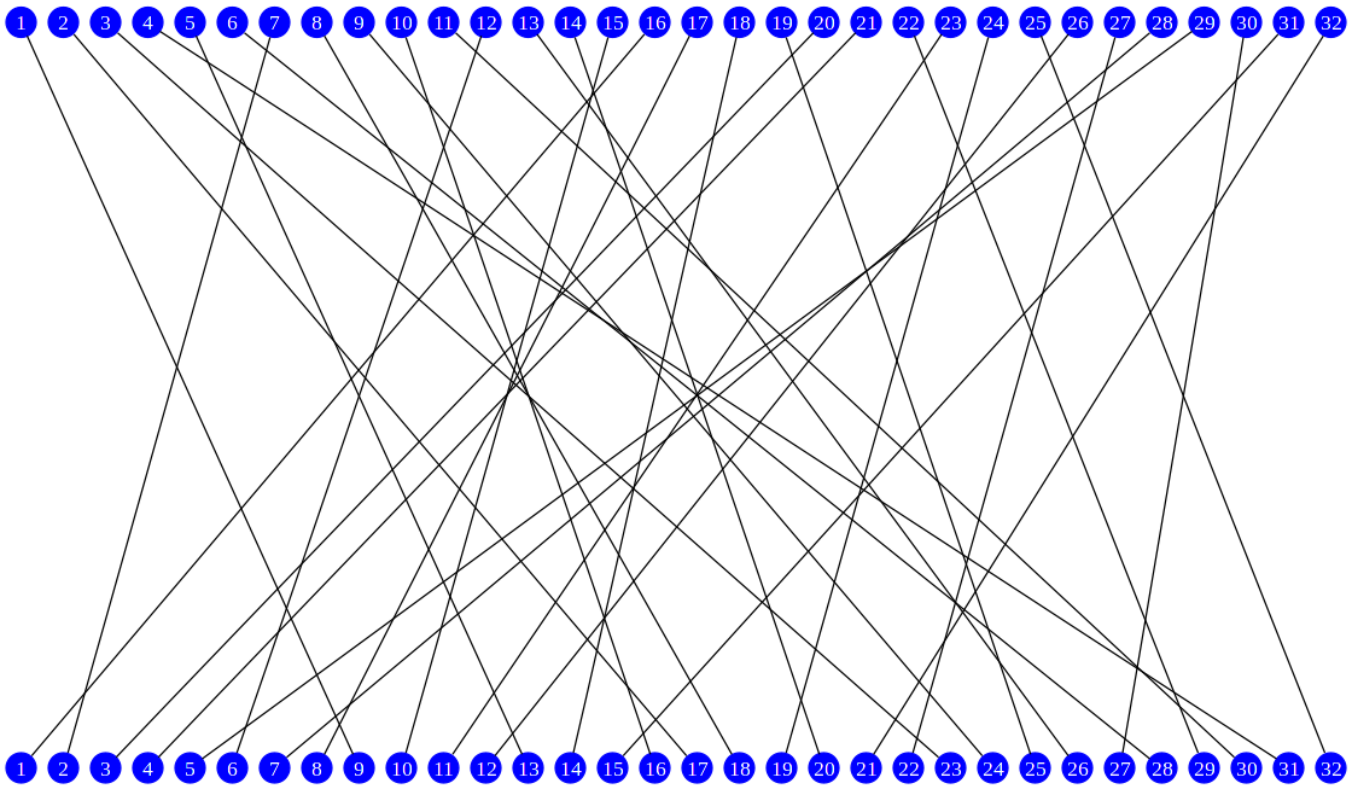
In	1	2	3	4	5	6	7	8
Out	2, 48	3	4	5, 7	6, 8	9	10	11, 13
In	9	10	11	12	13	14	15	16
Out	12, 14	15	16	17, 19	18, 20	21	22	23, 25
In	17	18	19	20	21	22	23	24
Out	24, 26	27	28	29, 31	30, 32	33	34	35, 37
In	25	26	27	28	29	30	31	32
Out	36, 38	39	40	41, 43	42, 44	45	46	1, 47



Permutation

In	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Out	9	17	23	31	13	28	2	18	24	16	30	6	26	20	10	1

In	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Out	8	14	25	3	4	29	11	19	32	12	22	7	5	27	15	21



DES S-boxes

Substitution-boxes or S-boxes (S1, S2, S3, S4, S5, S6, S7, S8):

- Each S-box is a function taking six input bits and producing four output bits.
- S-boxes are the only components of DES that are non-linear. (Without the S-boxes, changing one plaintext bit would change very few ciphertext bits.)
- Security of DES crucially depends on their choice.
- DES with randomly selected S-boxes is easy to break.

DES S-boxes. Columns denote middle four bits of input. Rows denote outer two bits of input.

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Cryptanalysis of DES

"DES did more to galvanize the field of cryptanalysis than anything else. Now there was an algorithm to study."
—Bruce Schneier

- Brute force attacks (try every key):
  - (1977 estimate) $20 million machine to find keys in one day
  - (1993 estimate) $1 million machine to find keys in 7 hours
  - (1999) EFF DES Cracker: $250,000 machine, 4.5 days per key
  - (2006) COPACOBANA: $10,000 machine, 4.5 days per key
  - (2012) Cloudcracker.com: $200 and 11.5 hours per key
- Non-brute-force attacks:
  - Differential cryptanalysis (Eli Biham & Adi Shamir, 1991): 2^{49} chosen plaintexts
  - Linear cryptanalysis (Mitsuru Matsui, 1993): 2^{43} known plaintexts

Attacks on DES

Differential cryptanalysis [Biham & Shamir 1989–1991]:

- Recovers key given 2^{47} chosen plaintext/ciphertext pairs.
- DES was designed to resist this attack.
- Differential cryptanalysis is more effective against other block ciphers.

Linear cryptanalysis [Matsui 1993]:

- Recovers key given 2^{43} known plaintext/ciphertext pairs.
- Storing these pairs takes 131,000 Gbytes.
- Implemented in 1993: 10 days on 12 machines.

Brute force

- June 1997: Broken by Internet search (3 months).
- July 1998: Broken in 3 days by EFF machine ($250,000).
- Jan 1999: Broken in 22 hrs, 15 min (EFF + `distributed.net`).

Implementation attacks on DES

Power analysis attacks:

- Kocher, Jaffe & Jun 1999.
- Processor power consumption depends on instruction.
- Measure power consumption of instructions executed in 16th round of DES.
- $\approx$1000 encryptions suffice to expose the secret key.

Differential fault analysis (DFA) attacks:

- Biham & Shamir 1997.
- Attack: induce random errors in 16th round of DES.
- $\approx$200 erroneous decryptions expose the secret key.