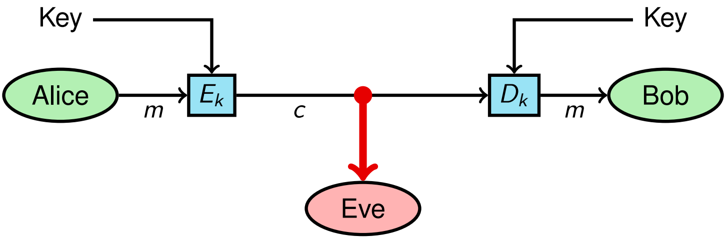
Cryptography Lecture 4

Block ciphers, DES, breaking DES

LINKÖPING UNIVERSITY

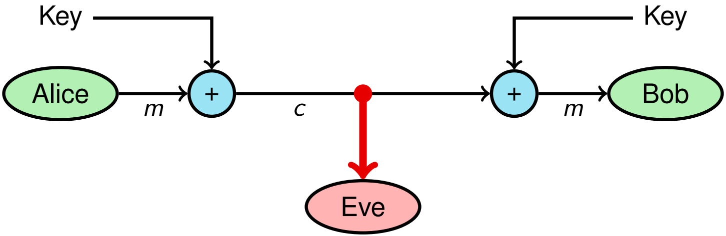
Breaking a cipher

- Eavesdropper recieves n cryptograms created from n plaintexts in sequence, using the same key
- Redundancy exists in the messages
- There is always one n (the unicity distance) where only one value for the key recreates a possible plaintext, unless we use OTP



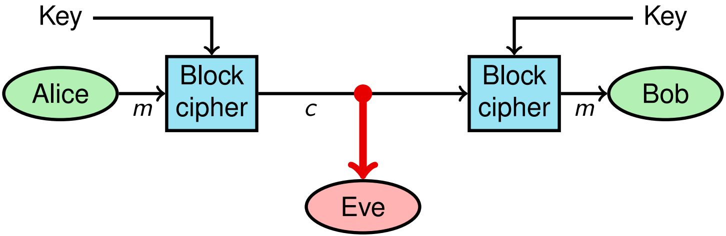
Defence against breaking a cipher through exhaustive search

- Change key often enough, so that unicity distance is not reached
  - OTP
  - Approximation of OTP: Stream ciphers
- Make sure there are too many possible keys for exhaustive search
  - Single-letter substitution is not enough, even though there are $26! \approx 4 * 10^{26} \approx 2^{88}$ combinations
  - Encrypt larger blocks (than one-, two-, or three-letter combinations)

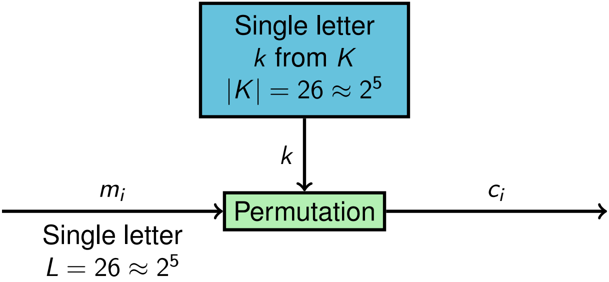


Defence against breaking a cipher through exhaustive search

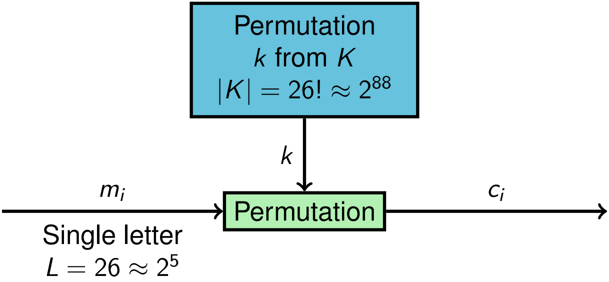
- Change key often enough, so that unicity distance is not reached
  - OTP
  - Approximation of OTP: Stream ciphers
- Make sure there are too many possible keys for exhaustive search
  - Single-letter substitution is not enough, even though there are $26! \approx 4 * 10^{26} \approx 2^{88}$ combinations
  - Encrypt larger blocks (than one-, two-, or three-letter combinations)



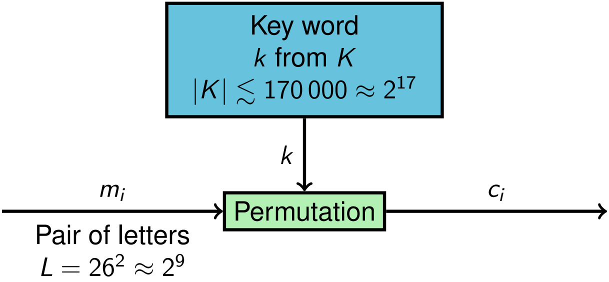
Caesar cipher



Substitution cipher

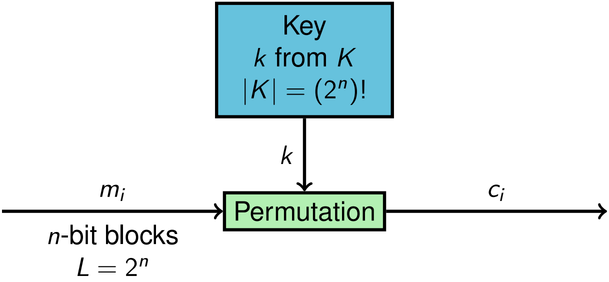


Playfair

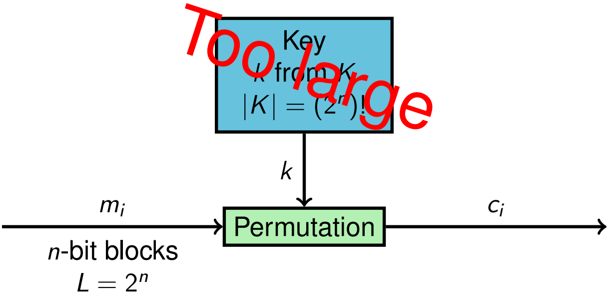


Key word
k from K
$|K| \lesssim 170\,000 \approx 2^{17}$
k
m_i
Permutation
c_i
Pair of letters
$L = 26^2 \approx 2^9$

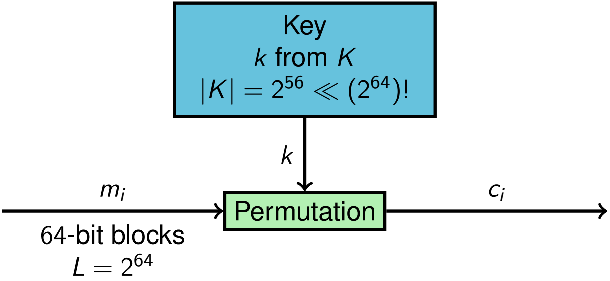
Generic block cipher



Generic block cipher



Data Encryption Standard (1975)



Block ciphers v. codes

- The same block with the same key always produces the same cryptogram, independent of its position in a sequence
- This is simple substitution on the block level
- An attacker could, in principle, create a table of all plaintext values and their corresponding cryptograms, one table for each key, and use this for cryptanalysis
- As defence, blocks and keys must be so large that there are too many values to list in the table

Block cipher criteria

Diffusion If a plaintext character changes, several ciphertext characters should change. This is a basic demand on a block cipher, and ensures that the statistics used need to be block statistics (as opposed to letter statistics)

Confusion Every bit of the ciphertext should depend on several bits in the key. This can be achieved by ensuring that the system is nonlinear

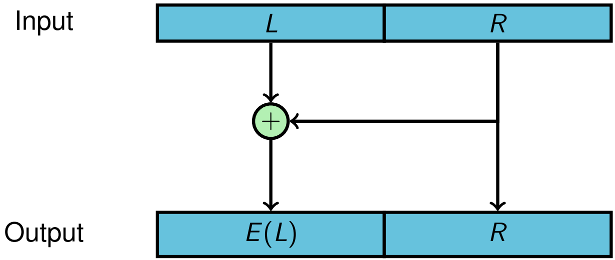
Diffusion: the avalanche effect

- A change in one bit in the input should propagate to many bits in the output

The strict avalanche criterion

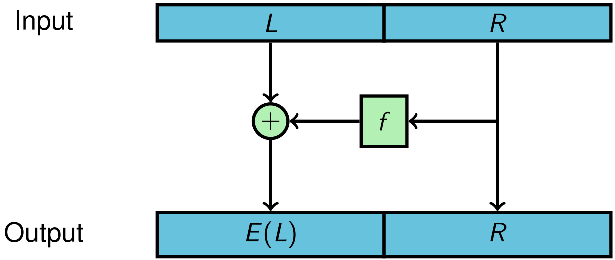
- A change in one bit in the input should change each output bit with probability $\frac{1}{2}$
- If this does not hold, an attacker can make predictions on the input, given only the output

Build the system from components



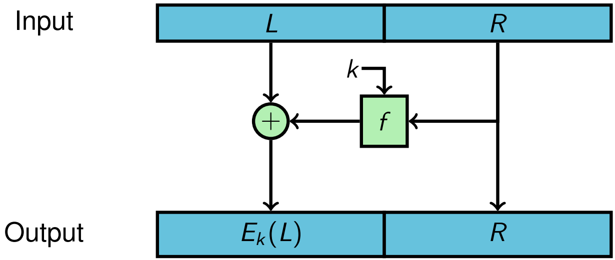
- Diffusion: A change in one bit in the input should change each output bit with probability $\frac{1}{2}$
- This is done by mixing the bits

Build the system from components



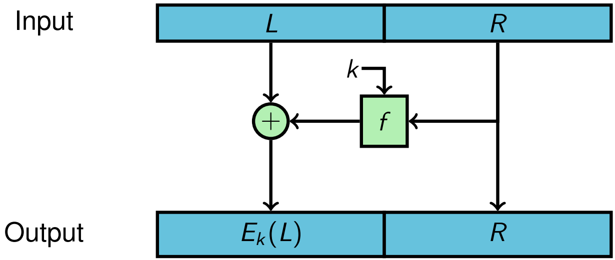
- Diffusion: A change in one bit in the input should change each output bit with probability $\frac{1}{2}$
- This is done by mixing the bits

Build the system from components



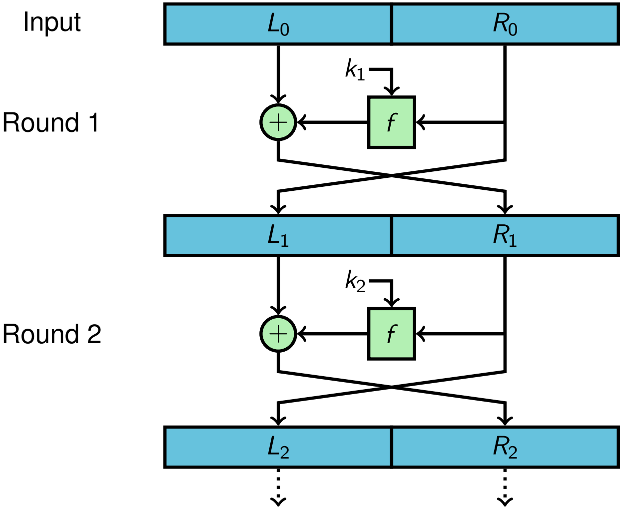
- Diffusion: A change in one bit in the input should change each output bit with probability $\frac{1}{2}$
- This is done by mixing the bits
- Use different functions depending on the key

Build the system from components

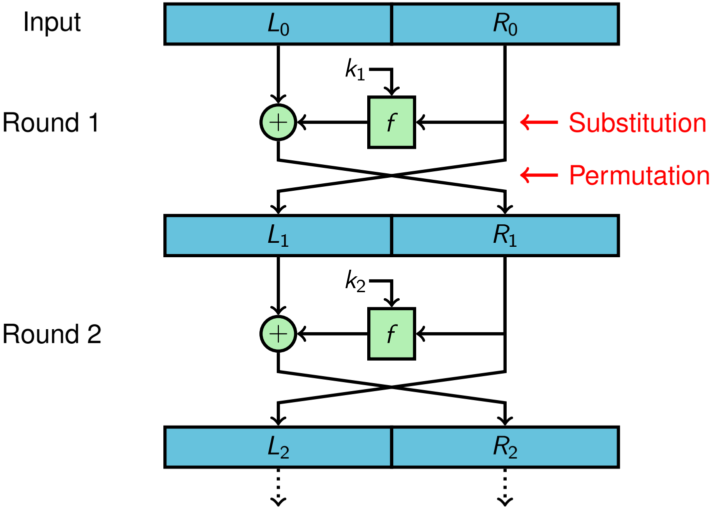


- Diffusion: A change in one bit in the input should change each output bit with probability $\frac{1}{2}$
- This is done by mixing the bits
- Use different functions depending on the key
- Confusion is created by using a nonlinear f

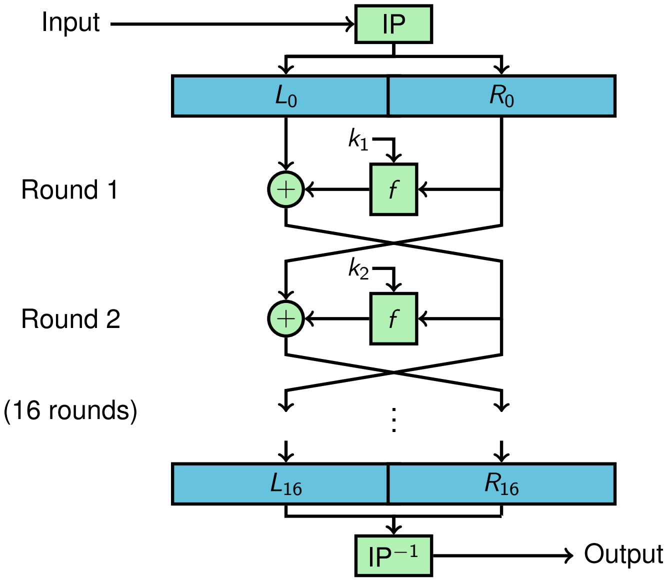
Feistel network



Feistel network



DES



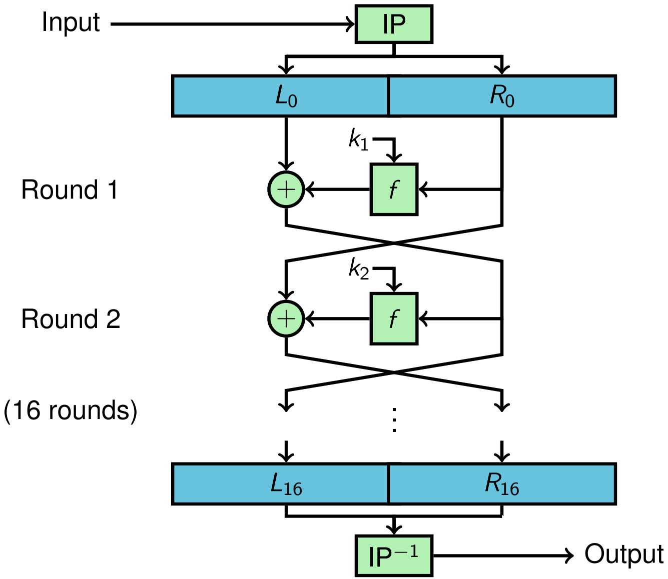
Input
IP
L_0
R_0
k_1
Round 1
f
k_2
Round 2
f
(16 rounds)
L_{16}
R_{16}
IP^{-1}
Output

DES

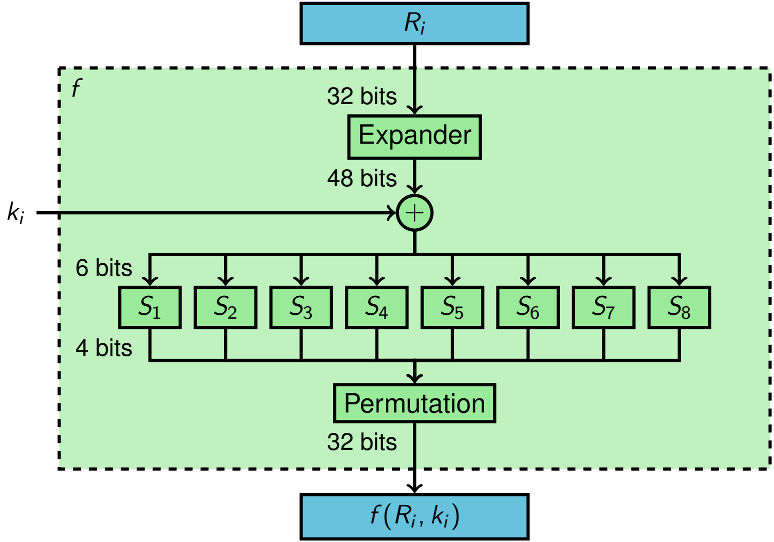
IP

b_{58}	b_{50}	b_{42}	b_{34}	b_{26}	b_{18}	b_{10}	b_2
b_{60}	b_{52}	b_{44}	b_{36}	b_{28}	b_{20}	b_{12}	b_4
b_{62}	b_{54}	b_{46}	b_{38}	b_{30}	b_{22}	b_{14}	b_6
b_{64}	b_{56}	b_{48}	b_{40}	b_{32}	b_{24}	b_{16}	b_8
b_{57}	b_{49}	b_{41}	b_{33}	b_{25}	b_{17}	b_9	b_1
b_{59}	b_{51}	b_{43}	b_{35}	b_{27}	b_{19}	b_{11}	b_3
b_{61}	b_{53}	b_{45}	b_{37}	b_{29}	b_{21}	b_{13}	b_5
b_{63}	b_{55}	b_{47}	b_{39}	b_{31}	b_{23}	b_{15}	b_7

DES


Input
IP
L_0
R_0
k_1
Round 1
f
k_2
Round 2
f
(16 rounds)
L_{16}
R_{16}
IP^{-1}
Output

DES



R_i
f
32 bits
Expander
48 bits
k_i
6 bits
S_1
S_2
S_3
S_4
S_5
S_6
S_7
S_8
4 bits
Permutation
32 bits
$f(R_i, k_i)$

DES

Expander

b_{32}	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9
b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}
b_{16}	b_{17}	b_{18}	b_{19}	b_{20}	b_{21}	b_{22}	b_{23}	b_{24}	b_{25}
b_{24}	b_{25}	b_{26}	b_{27}	b_{28}	b_{29}	b_{30}	b_{31}	b_{32}	b_1

Permutation

b_{16}	b_7	b_{20}	b_{21}	b_{29}	b_{12}	b_{28}	b_{17}
b_1	b_{15}	b_{23}	b_{26}	b_5	b_{18}	b_{31}	b_{10}
b_2	b_8	b_{24}	b_{14}	b_{32}	b_{27}	b_3	b_9
b_{19}	b_{13}	b_{30}	b_6	b_{22}	b_{11}	b_4	b_{25}

DES

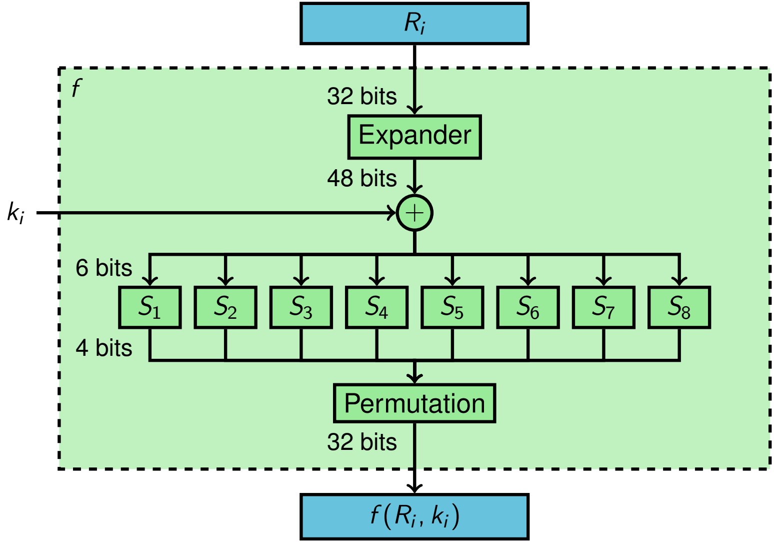
$b_1 b_2 b_3 b_4 b_5 b_6$

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

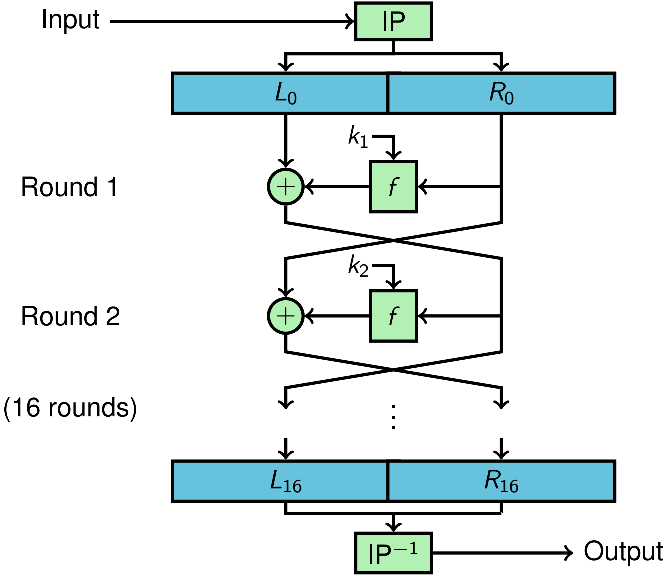
row index: $b_1 b_6$, column index: $b_2 b_3 b_4 b_5$

DES



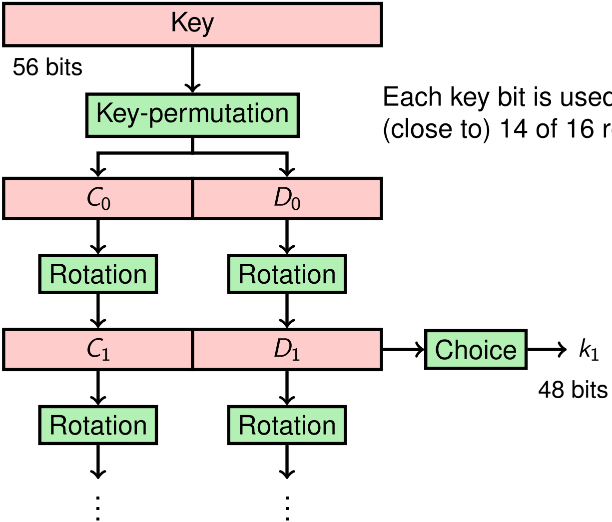
R_i
f
32 bits
Expander
48 bits
k_i
6 bits
S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8
4 bits
Permutation
32 bits
$f(R_i, k_i)$

DES



Input
IP
L_0
R_0
k_1
Round 1
f
k_2
Round 2
f
(16 rounds)
L_{16}
R_{16}
IP^{-1}
Output

DES key schedule



Each key bit is used in (close to) 14 of 16 rounds

DES

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

row index: $b_1 b_6$, column index: $b_2 b_3 b_4 b_5$

- There was a lot of controversy surrounding the S-box construction
- People were worried there were backdoors in the system
- But in the late eighties it was found that even small changes in the S-boxes gave a weaker system

DES

After the (re-)discovery of *differential cryptanalysis*, in 1994 IBM published the construction criteria

- Each S-box has 6 input bits and four output bits (1970's hardware limit)
- The S-boxes should not be linear functions, or even close to linear
- Each row of an S-box contains all numbers from 0 to 15
- Two inputs that differ by 1 bit should give outputs that differ by at least 2 bits
- Two inputs that differ in the first 2 bits but are equal in the last 2 bits should give unequal outputs
- There are 32 pairs of inputs with a given XOR. No more than eight of the corresponding outputs should have equal XORs
- A similar criterion involving three S-boxes

DES

After the (re-)discovery of *differential cryptanalysis*, in 1994 IBM published the construction criteria

- Each S-box has 6 input bits and four output bits (1970's hardware limit)
- The S-boxes should not be linear functions, or even close to linear
- Each row of an S-box contains all numbers from 0 to 15
- Two inputs that differ by 1 bit should give outputs that differ by at least 2 bits
- Two inputs that differ in the first 2 bits but are equal in the last 2 bits should give unequal outputs
- There are 32 pairs of inputs with a given XOR. No more than eight of the corresponding outputs should have equal XORs
- A similar criterion involving three S-boxes

Linearity

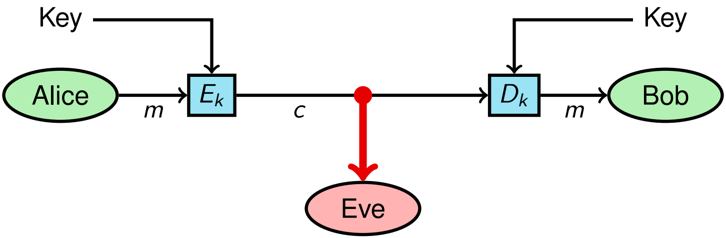
- A function f from is linear if

$$f(ax + by) = a\,f(x) + b\,f(y)$$

- Example: $f(t) = 7t$ is linear
- A (close to) linear system is much easier to analyse
- Therefore, you cannot use only simple mathematical functions

Linear cryptanalysis

- Make a linear approximation of the cipher
- This will have k as parameter
- Use many plaintext-ciphertext pairs to deduce which linear approximation is the best, and this will correspond to the most likely key



Prohibit linear cryptanalysis

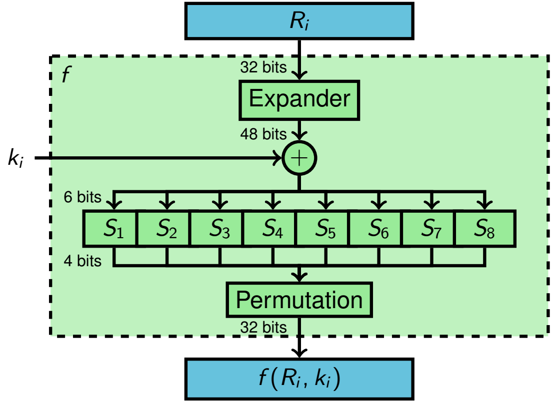
Examples:

- $f(t) = 7t$ is linear
- but $f(t) = (7t \bmod 8)$ in the ring of numbers mod 16
  is nonlinear, because $f(2) \neq 2f(1)$:

$$f(2) = (14 \bmod 8) = 6 \neq 2f(1) = 2(7 \bmod 8) = 14$$

- of course $f(t) = (7t \bmod 8)$ is linear in the ring of numbers mod 8

Prohibit linear analysis



- In DES, smaller blocks are used in each step, and are combined to create non-linearity with respect to the larger blocks
- The S-box itself is also chosen to be non-linear

Linear cryptanalysis of DES

- Make a linear approximation of the S-boxes
- Combine these into a linear approximation of the whole cipher
- This will have k as parameter
- Use many plaintext-ciphertext pairs to deduce which linear approximation is the best, and this will correspond to the most likely key
- Needs 2^{43} plaintext-ciphertext pairs for DES

DES

After the (re-)discovery of *differential cryptanalysis*, in 1994 IBM published the construction criteria

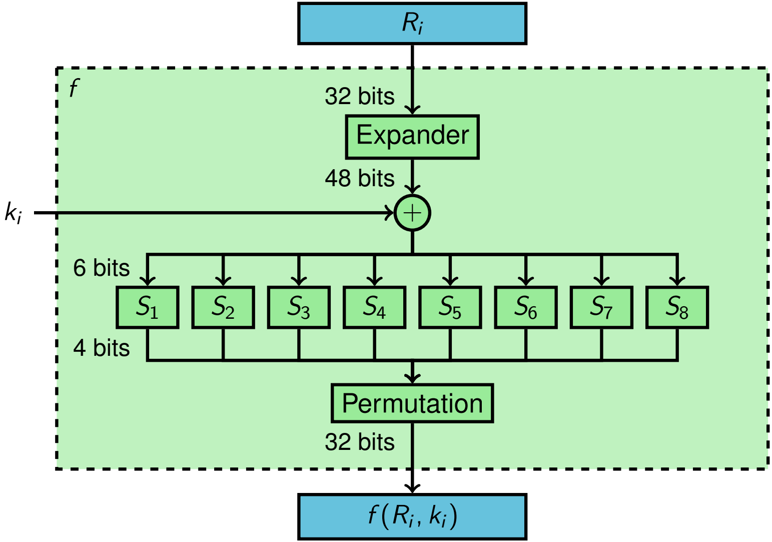
- Each S-box has 6 input bits and four output bits (1970's hardware limit)
- The S-boxes should not be linear functions, or even close to linear
- Each row of an S-box contains all numbers from 0 to 15
- Two inputs that differ by 1 bit should give outputs that differ by at least 2 bits
- Two inputs that differ in the first 2 bits but are equal in the last 2 bits should give unequal outputs
- There are 32 pairs of inputs with a given XOR. No more than eight of the corresponding outputs should have equal XORs
- A similar criterion involving three S-boxes

DES

After the (re-)discovery of *differential cryptanalysis*, in 1994 IBM published the construction criteria

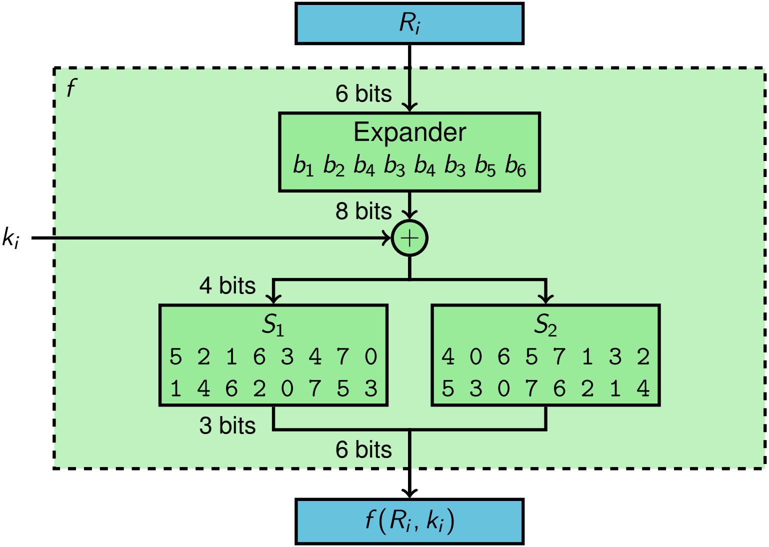
- Each S-box has 6 input bits and four output bits (1970's hardware limit)
- The S-boxes should not be linear functions, or even close to linear
- Each row of an S-box contains all numbers from 0 to 15
- Two inputs that differ by 1 bit should give outputs that differ by at least 2 bits
- Two inputs that differ in the first 2 bits but are equal in the last 2 bits should give unequal outputs
- There are 32 pairs of inputs with a given XOR. No more than eight of the corresponding outputs should have equal XORs
- A similar criterion involving three S-boxes

DES

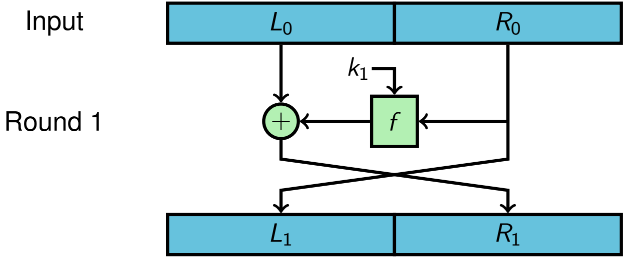


R_i
f
32 bits
Expander
48 bits
k_i
6 bits
S_1
S_2
S_3
S_4
S_5
S_6
S_7
S_8
4 bits
Permutation
32 bits
$f(R_i, k_i)$

Simple(r) Encryption System

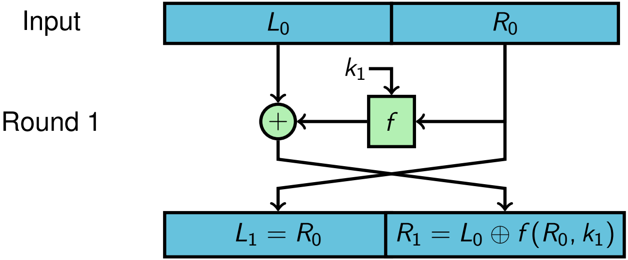


A one-round Feistel network is trivial to break



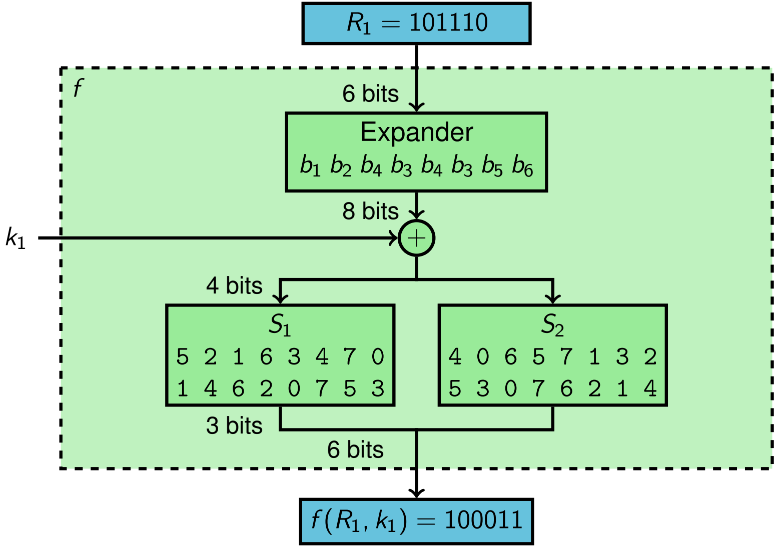
A known-plaintext attack breaks the system, because then you know R_0 and $f(R_0, k_1) = R_1 \oplus L_0$, so you can find k_1

A one-round Feistel network is trivial to break

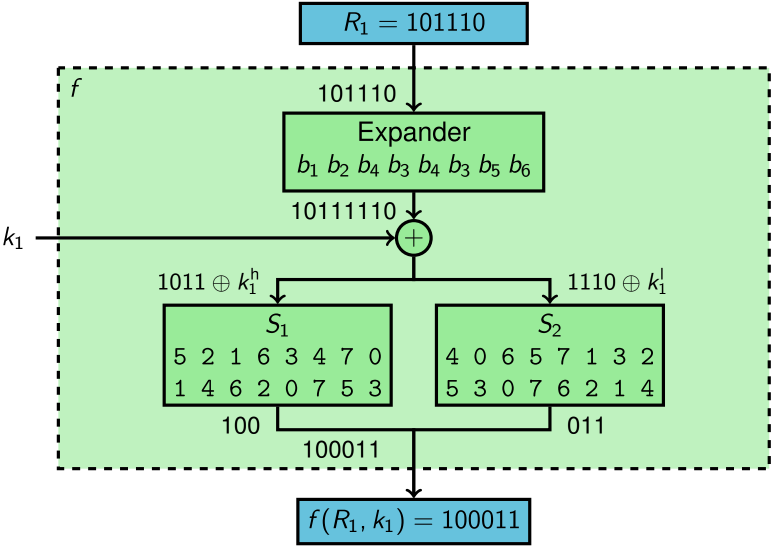


A known-plaintext attack breaks the system, because then you know R_0 and $f(R_0, k_1) = R_1 \oplus L_0$, so you can find k_1

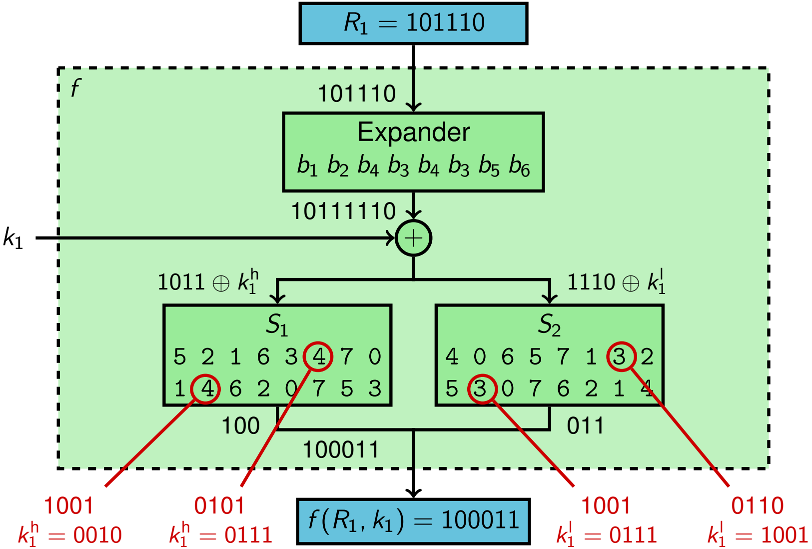
Simple(r) Encryption System, example



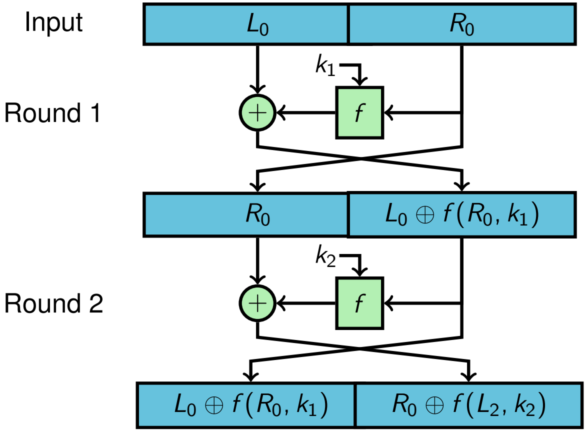
Simple(r) Encryption System, example



Simple(r) Encryption System, example



A two-round Feistel network is trivial to break



Use the same method twice: $(R_0, f(R_0, k_1) = L_2 \oplus L_0)$;
$(L_2, f(L_2, k_2) = R_2 \oplus R_0)$, this gives two alternatives each for k_1 and k_2.
Now, the key schedule may rule out some combinations.

A three-round Feistel network is simple to break

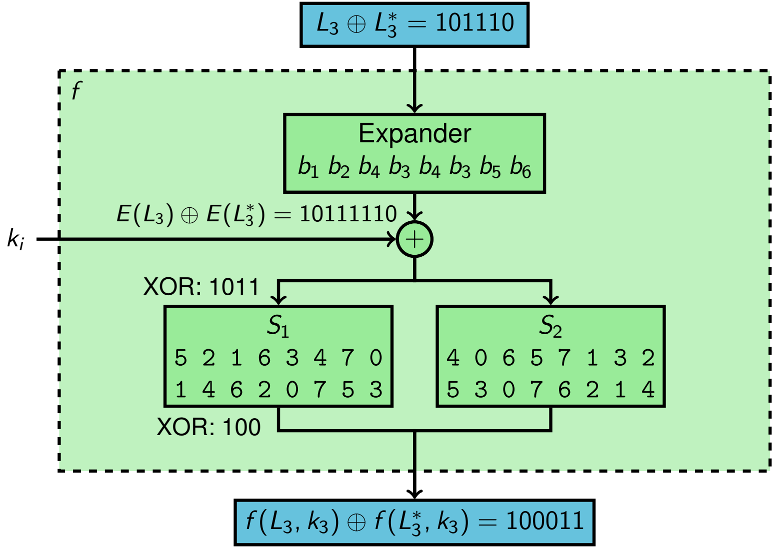
L_0	R_0
R_0	$L_0 \oplus f(R_0, k_1)$
$L_0 \oplus f(R_0, k_1)$	L_3
L_3	$R_3 = L_0 \oplus f(R_0, k_1) \oplus f(L_3, k_3)$

Perform two known-plaintext attacks for $L_0 R_0$ and $L_0^* R_0^*$ with $R_0 = R_0^*$. Then, the outputs have the relation

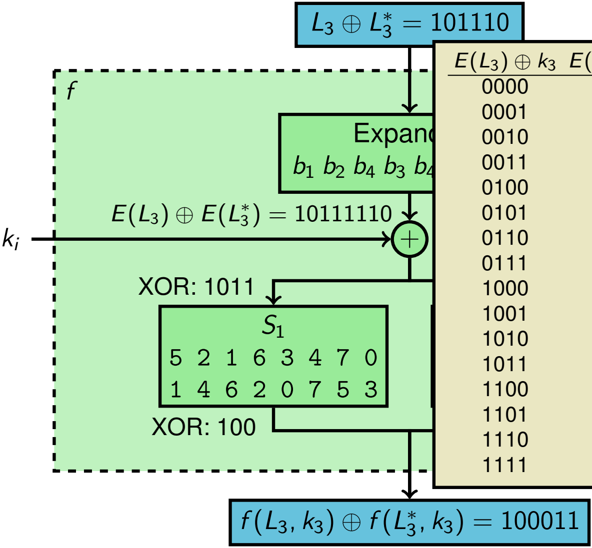
$$R_3 \oplus R_3^* = L_0 \oplus L_0^* \oplus f(L_3, k_3) \oplus f(L_3^*, k_3)$$

We have $L_3 \oplus L_3^*$ and $f(L_3, k_3) \oplus f(L_3^*, k_3)$

Simple(r) Encryption System, given XOR

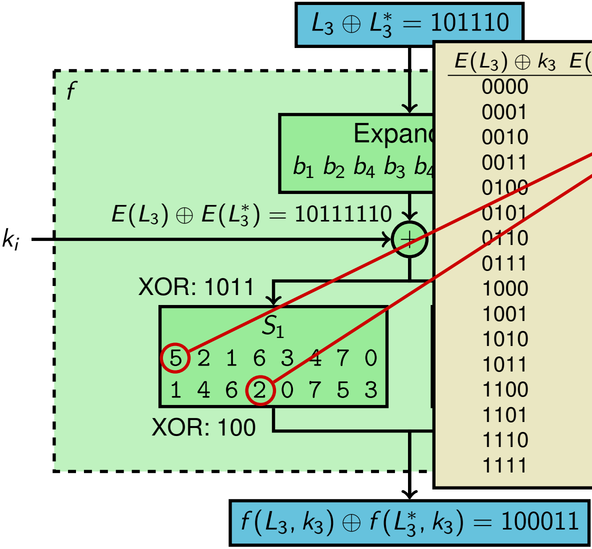


Simple(r) Encryption System, given XOR



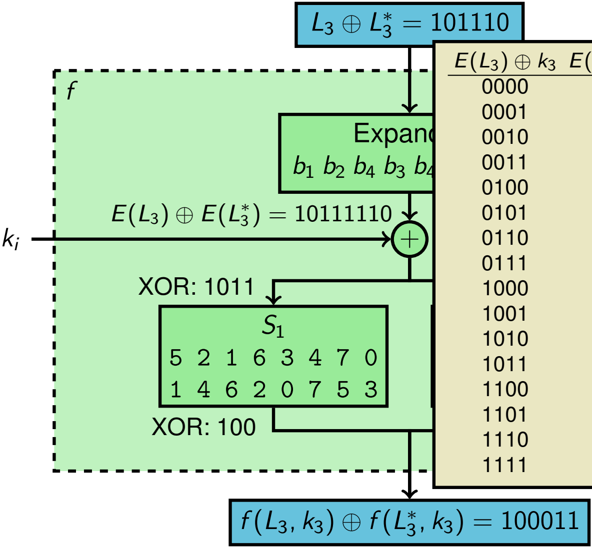
$E(L_3) \oplus k_3$	$E(L_3^*) \oplus k_3$	Out XOR
0000	1011	111
0001	1010	100
0010	1001	101
0011	1000	111
0100	1111	000
0101	1110	001
0110	1101	000
0111	1100	000
1000	0011	111
1001	0010	101
1010	0001	100
1011	0000	111
1100	0111	000
1101	0110	000
1110	0101	001
1111	0100	000

Simple(r) Encryption System, given XOR



$E(L_3) \oplus k_3$	$E(L_3^*) \oplus k_3$	Out XOR
0000	1011	111
0001	1010	100
0010	1001	101
0011	1000	111
0100	1111	000
0101	1110	001
0110	1101	000
0111	1100	000
1000	0011	111
1001	0010	101
1010	0001	100
1011	0000	111
1100	0111	000
1101	0110	000
1110	0101	001
1111	0100	000

Simple(r) Encryption System, given XOR



$E(L_3) \oplus k_3$	$E(L_3^*) \oplus k_3$	Out XOR
0000	1011	111
0001	1010	100
0010	1001	101
0011	1000	111
0100	1111	000
0101	1110	001
0110	1101	000
0111	1100	000
1000	0011	111
1001	0010	101
1010	0001	100
1011	0000	111
1100	0111	000
1101	0110	000
1110	0101	001
1111	0100	000

A three-round Feistel network is simple to break

L_0	R_0
R_0	$L_0 \oplus f(R_0, k_1)$
$L_0 \oplus f(R_0, k_1)$	L_3
L_3	$R_3 = L_0 \oplus f(R_0, k_1) \oplus f(L_3, k_3)$

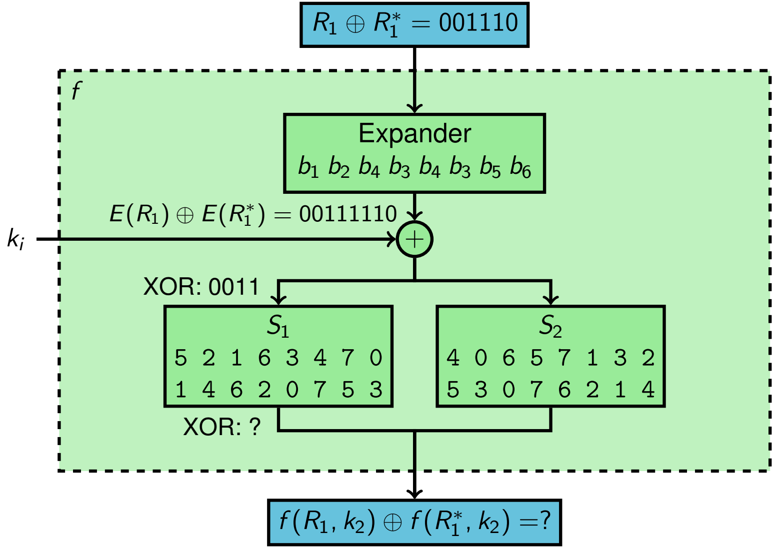
Choose $R_0 = R_0^*$ so that $f(R_0, k_1) \oplus f(R_0^*, k_1) = 0$. Then, we can calculate $f(L_3, k_3) \oplus f(L_3^*, k_3)$

A four-round Feistel network is more complicated to break

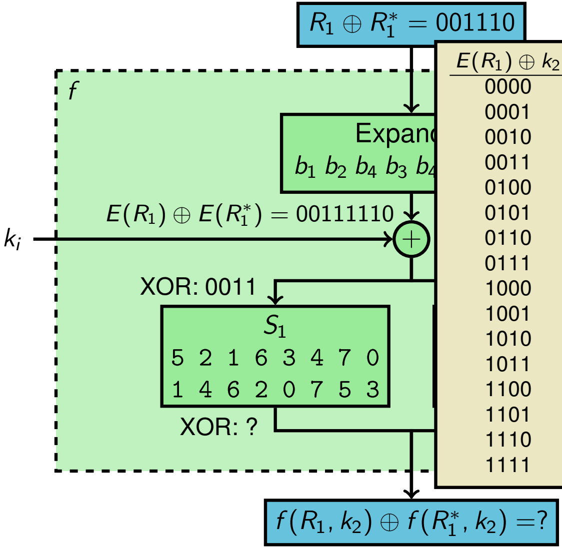
L_0	R_0
L_1	R_1
R_1	$L_1 \oplus f(R_1, k_2)$
$L_1 \oplus f(R_1, k_2)$	$L_4 = L_0 \oplus f(R_0, k_1) \oplus f(L_3, k_3)$
L_4	$R_4 = L_1 \oplus f(R_1, k_2) \oplus f(L_4, k_4)$

Here, if we can guess $f(R_1, k_2) \oplus f(R_1^*, k_2)$ (even if it is $\neq 0$), we can calculate $f(L_4, k_4) \oplus f(L_4^*, k_4)$

Simple(r) Encryption System, given XOR

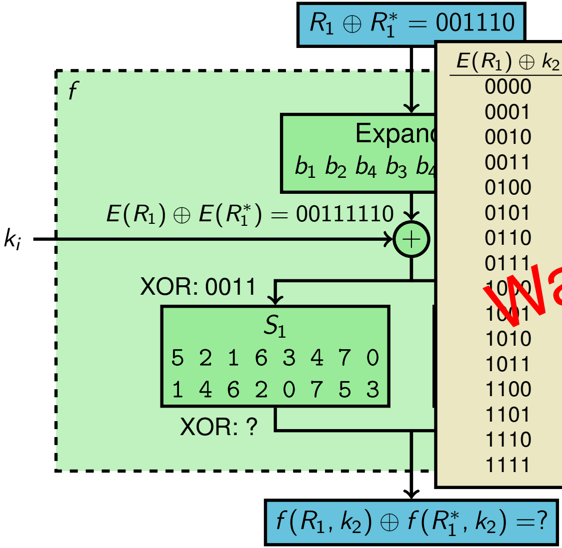


Simple(r) Encryption System, given XOR



$E(R_1) \oplus k_2$	$E(R_1^*) \oplus k_2$	Out XOR
0000	0011	011
0001	0010	011
0010	0001	011
0011	0000	011
0100	0111	011
0101	0110	011
0110	0101	011
0111	0100	011
1000	1011	011
1001	1010	010
1010	1001	010
1011	1000	011
1100	1111	011
1101	1110	010
1110	1101	010
1111	1100	011

Simple(r) Encryption System, given XOR



$E(R_1) \oplus k_2$	$E(R_1^*) \oplus k_2$	Out XOR
0000	0011	011
0001	0010	011
0010	0001	011
0011	0000	011
0100	0111	011
0101	0110	011
0110	0101	011
0111	0100	011
1000	1011	011
1001	1010	010
1010	1001	010
1011	1000	011
1100	1111	011
1101	1110	010
1110	1101	010
1111	1100	011

Warning!

A four-round Feistel network is more complicated to break

L_0	R_0
L_1	R_1
R_1	$L_1 \oplus f(R_1, k_2)$
$L_1 \oplus f(R_1, k_2)$	$L_4 = L_0 \oplus f(R_0, k_1) \oplus f(L_3, k_3)$
L_4	$R_4 = L_1 \oplus f(R_1, k_2) \oplus f(L_4, k_4)$

Here, if we can guess $f(R_1, k_2) \oplus f(R_1^*, k_2)$ (even if it is $\neq 0$), we can calculate $f(L_4, k_4) \oplus f(L_4^*, k_4)$

Take random input pairs, and use the most likely output XOR to deduce the most likely k_4

DES

The seemingly strange criterion is to prohibit differential cryptanalysis

- There are 32 pairs of inputs with a given XOR. No more than eight of the corresponding outputs should have equal XORs

The designers knew about differential cryptanalysis

Still, it works on DES, and breaks 15-round DES faster than exhaustive search (16-round DES requires 2^{47} chosen plaintexts pairs)

Computational cost of breaking DES

- DES was standardized 1975, and already 1977 there was an estimate that a machine to break it would cost \$20M (1977 dollars)
- DES was recertified in 1992 despite growing concerns
- One can use distributed computing, specialized hardware, or nowadays, cheap FPGAs
- In “the DES challenge” in 1997 the key was found in five months (distributed computation) having searched 25% of the key space (1998: 39 days, 85%)
- 1998: EFF DES cracker, parallelized, \$200k, 4.5 days (on average)

Key length

Table 7.1: Minimum symmetric key-size in bits for various attackers.

Attacker	Budget	Hardware	Min security
“Hacker”	0	PC	58
	$< \$400$	PC(s)/FPGA	63
	0	”Malware”	77
Small organization	\$10k	PC(s)/FPGA	69
Medium organization	\$300k	FPGA/ASIC	69
Large organization	\$10M	FPGA/ASIC	78
Intelligence agency	\$300M	ASIC	84

From “ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012)”

Key length

Table 7.1: Minimum symmetric key-size in bits for various attackers

Attacker	Budget	Hardware	Min security	(1996)
“Hacker”	0	PC	58	45
	$< \$400$	PC(s)/FPGA	63	50
	0	”Malware”	77	
Small organization	$10k	PC(s)/FPGA	69	55
Medium organization	$300k	FPGA/ASIC	69	60
Large organization	$10M	FPGA/ASIC	78	70
Intelligence agency	$300M	ASIC	84	75

From “ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012)”

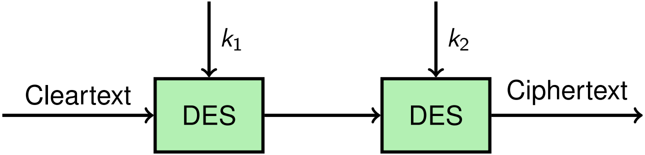
Key length

Table 7.4: Security levels (symmetric equivalent)

Security (bits)	Protection	Comment
32	Real-time, individuals	Only auth. tag size
64	Very short-term, small org	Not for confidentiality in new systems
72	Short-term, medium org Medium-term, small org	
80	Very short-term, agencies Long-term, small org	Smallest general-purpose $<$ 4 years protection (E.g., use of 2-key 3DES, $< 2^{40}$ plaintext/ciphertexts)
96	Legacy standard level	2-key 3DES restricted to 10^6 plaintext/ciphertexts, $\sim$ 10 years protection
112	Medium-term protection	$\sim$ 20 years protection (E.g., 3-key 3DES)
128	Long-term protection	Good, generic application-indep. Recommendation, $\sim$ 30 years
256	”Foreseeable future”	Good protection against quantum computers unless Shor’s algorithm applies.

From “ECRYPT II Yearly Report on Algorithms and Keysizes (2009-2010)”

Double DES



$$E_{k_2}\left(E_{k_1}(m)\right) \neq E_{k_3}(m)$$

Encrypt repeatedly with the keys consisting of all 0s and all 1s. The smallest n such that $(E_0 \circ E_1)^n(m) = m$ is called the cycle length. If DES is a group, then $n < 2^{56}$

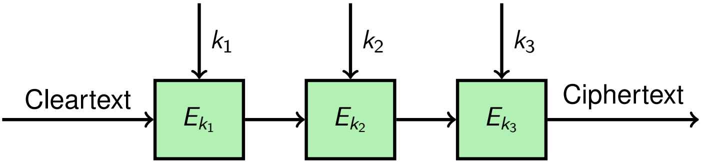
Lemma: the smallest integer N such that $(E_0 \circ E_1)^N(m) = m$ for all m contains all individual cycles as factors

An example has been found where the cycle lengths of 33 messages has the least common multiple of $10^{277} \gg 2^{56}$

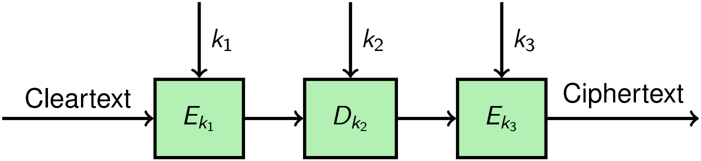
Meet-in-the-middle attacks

- A meet-in-the-middle attack is a known plaintext attack
- Make a list of all 2^{56} possible (single-DES) encryptions of the plaintext, and of all 2^{56} (single-DES) decryptions of the ciphertext
- Match the two lists. The key(s) that give the same middle value is (are) the key (candidates)
- Attack is of complexity 2^{57}

Triple DES



More common:



Breaking three-key triple DES has a complexity of 2^{112}

Key length

Table 7.4: Security levels (symmetric equivalent)

Security (bits)	Protection	Comment
32	Real-time, individuals	Only auth. tag size
64	Very short-term, small org	Not for confidentiality in new systems
72	Short-term, medium org Medium-term, small org	
80	Very short-term, agencies Long-term, small org	Smallest general-purpose $<$ 4 years protection (E.g., use of 2-key 3DES, $< 2^{40}$ plaintext/ciphertexts)
96	Legacy standard level	2-key 3DES restricted to 10^6 plaintext/ciphertexts, $\sim$ 10 years protection
112	Medium-term protection	$\sim$ 20 years protection (E.g., 3-key 3DES)
128	Long-term protection	Good, generic application-indep. Recommendation, $\sim$ 30 years
256	”Foreseeable future”	Good protection against quantum computers unless Shor’s algorithm applies.

From “ECRYPT II Yearly Report on Algorithms and Keysizes (2009-2010)”

Next lecture

- AES
- Mathematics: intro to finite fields
- Modes of operation
- Message Authentication Codes, MACs