DD2448 Foundations of Cryptography
Lecture 2

Douglas Wikström
KTH Royal Institute of Technology
`dd2448@kth.se`

March 20, 2024

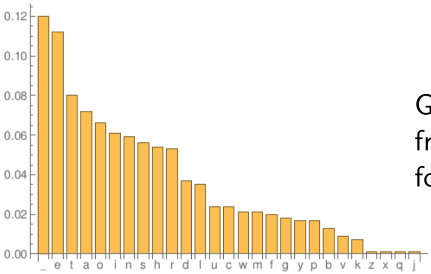
Generic Attack Against Substitution Cipher

1. Compute symbol/digram/trigram frequency tables for the candidate language and the ciphertext.
2. Try to match symbols/digrams/trigrams with similar frequencies.
3. Try to recognize words to confirm your guesses (we would use a dictionary (or Google!) here).
4. Backtrack/repeat until the plaintext can be guessed.

This is hard when several symbols have similar frequencies. A large amount of ciphertext is needed. How can we ensure this?

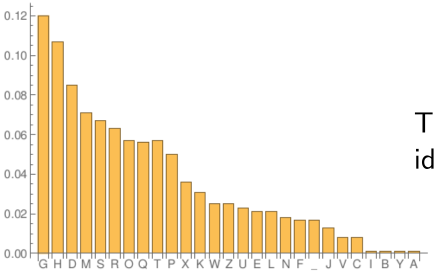
Cryptanalysis of the Substitution Cipher

Sorted English Frequencies



Group by similar frequency and brute force within each group!

Sorted Ciphertext Frequencies



This bar chart is not identical to the above!

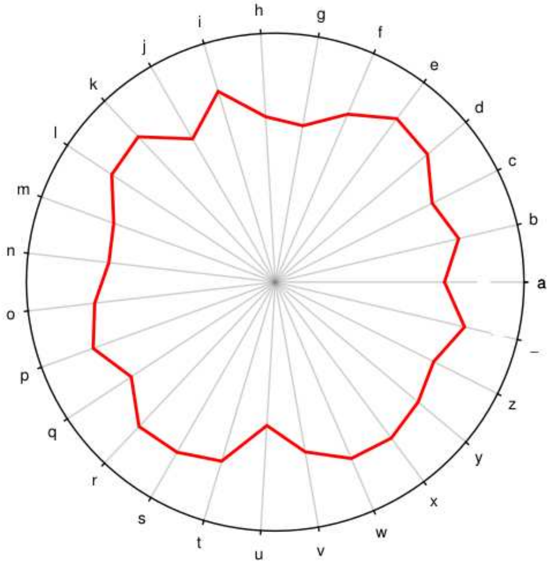
Vigénère

Vigénère Cipher.

- **Key.** $\mathsf{k} = (\mathsf{k}_0, \ldots, \mathsf{k}_{l-1})$, where $\mathsf{k}_i \in \mathbb{Z}_{27}$ is random.
- **Encrypt.** Plaintext $m = (m_1, \ldots, m_n) \in \mathbb{Z}_{27}^n$ gives ciphertext $c = (c_1, \ldots, c_n)$, where $c_i = m_i + \mathsf{k}_{i \bmod l} \bmod 27$.
- **Decrypt.** Ciphertext $c = (c_1, \ldots, c_n) \in \mathbb{Z}_{27}^n$ gives plaintext $m = (m_1, \ldots, m_n)$, where $m_i = c_i - \mathsf{k}_{i \bmod l} \bmod 27$.

More uniform frequency table due to averaging :-)

Vigénère



Attack Vigénère (1/2)

Index of Coincidence.

- Each probability distribution $p_1, \ldots, p_n$ on n symbols may be viewed as a point $p = (p_1, \ldots, p_n)$ on a $n-1$ dimensional hyperplane in $\mathbb{R}^n$ orthogonal to the vector $\overline{1}$
- Such a point $p = (p_1, \ldots, p_n)$ is at distance $\sqrt{F(p)}$ from the origin, where $F(p) = \sum_{i=1}^{n} p_i^2$.
- It is clear that p is closest to the origin, when p is the uniform distribution, i.e., when $F(p)$ is minimized. (Draw picture!)
- $F(p)$ is invariant under permutation of the underlying symbols $\longrightarrow$ tool to check if a set of symbols is the result of **some** substitution cipher (for non-uniform plaintext sources).

Attack Vigénère (2/2)

1. For $l = 1, 2, 3, \ldots$, we form

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{l-1} \end{pmatrix} = \begin{pmatrix} c_0 & c_l & c_{2l} & \cdots \\ c_1 & c_{l+1} & c_{2l+1} & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ c_{l-1} & c_{2l-1} & c_{3l-1} & \cdots \end{pmatrix}$$

   and compute $f_l = \frac{1}{l}\sum_{i=0}^{l-1} F(C_i)$.

2. The local maximum with smallest l is probably the right length.
3. Then attack each C_i separately to recover k_i, using the attack against the Ceasar cipher.

Hill Cipher

Hill Cipher.

- **Key.** $\mathrm{k} = A$, where A is an invertible $l \times l$-matrix over $\mathbb{Z}_{27}$.
- **Encrypt.** Plaintext $m = (m_1, \ldots, m_n) \in \mathbb{Z}_{27}^n$ gives ciphertext $c = (c_1, \ldots, c_n)$, where (computed modulo 27):

$$(c_{i+0}, \ldots, c_{i+l-1}) = (m_{i+0}, \ldots, m_{i+l-1})A \ .$$

- **Decrypt.** Ciphertext $c = (c_1, \ldots, c_n) \in \mathbb{Z}_{27}^n$ gives plaintext $m = (m_1, \ldots, m_n)$, where (computed modulo 27):

$$(m_{i+0}, \ldots, m_{i+l-1}) = (c_{i+0}, \ldots, c_{i+l-1})A^{-1} \ .$$

for $i = 1, l+1, 2l+1, \ldots$

Hill Cipher

Hill Cipher.

- **Key.** $\mathrm{k} = A$, where A is an invertible $l \times l$-matrix over $\mathbb{Z}_{27}$.
- **Encrypt.** Plaintext $m = (m_1, \ldots, m_n) \in \mathbb{Z}_{27}^n$ gives ciphertext $c = (c_1, \ldots, c_n)$, where (computed modulo 27):

$$(c_{i+0}, \ldots, c_{i+l-1}) = (m_{i+0}, \ldots, m_{i+l-1})A \enspace .$$

- **Decrypt.** Ciphertext $c = (c_1, \ldots, c_n) \in \mathbb{Z}_{27}^n$ gives plaintext $m = (m_1, \ldots, m_n)$, where (computed modulo 27):

$$(m_{i+0}, \ldots, m_{i+l-1}) = (c_{i+0}, \ldots, c_{i+l-1})A^{-1} \enspace .$$

  for $i = 1, l+1, 2l+1, \ldots$

The Hill cipher is easy to break using a known plaintext attack.

Permutation Cipher (Transposition Cipher)

The permutation cipher is a special case of the Hill cipher.

Permutation Cipher.

- **Key.** Random permutation $\pi \in S$ for some subset S of the set of permutations of $\{0, 1, 2, \ldots, l-1\}$.
- **Encrypt.** Plaintext $m = (m_1, \ldots, m_n) \in \mathbb{Z}_{27}^n$ gives ciphertext $c = (c_1, \ldots, c_n)$, where $c_i = m_{\lfloor i/l \rfloor + \pi(i \bmod l)}$.
- **Decrypt.** Ciphertext $c = (c_1, \ldots, c_n) \in \mathbb{Z}_{27}^n$ gives plaintext $m = (m_1, \ldots, m_n)$, where $m_i = c_{\lfloor i/l \rfloor + \pi^{-1}(i \bmod l)}$.

Summary of Simple Ciphers

- Caesar cipher and affine cipher: $m_i \mapsto am_i + b$.
- Substitution cipher (generalize Ceasar/affine):
$$m_i \mapsto \sigma(m_i)$$
- Vigénère cipher (more uniform frequency table):
$$m_i \mapsto m_i + k_{i \bmod l}$$
- Hill cipher (invertible linear map):
$$(m_1, \ldots, m_l) \mapsto (m_1, \ldots, m_l)A$$
- Transposition cipher (permutation):
$$\begin{aligned} &(m_1, \ldots, m_l) \mapsto (m_{\pi(1)}, \ldots, m_{\pi(l)}) \\ &(m_1, \ldots, m_l) \mapsto (m_1, \ldots, m_l)M_\pi \qquad \text{(equivalently)} \end{aligned}$$

Simple Ciphers are Bad, What is a Good Block Cipher?

- For every key a block-cipher with plaintext/ciphertext space $\{0,1\}^n$ gives a permutation of $\{0,1\}^n$.

  What would be an good cipher?

Simple Ciphers are Bad, What is a Good Block Cipher?

- For every key a block-cipher with plaintext/ciphertext space $\{0,1\}^n$ gives a permutation of $\{0,1\}^n$.

  What would be an good cipher?

- A good cipher is one where each key gives a **randomly chosen permutation** of $\{0,1\}^n$.

  Why is this not possible?

Simple Ciphers are Bad, What is a Good Block Cipher?

- For every key a block-cipher with plaintext/ciphertext space $\{0,1\}^n$ gives a permutation of $\{0,1\}^n$.

  What would be an good cipher?

- A good cipher is one where each key gives a **randomly chosen permutation** of $\{0,1\}^n$.

  Why is this not possible?

- The representation of a single typical function $\{0,1\}^n \to \{0,1\}^n$ requires roughly $n2^n$ bits ($147 \times 10^{6 \cdot 3}$ for $n = 64$)

Simple Ciphers are Bad, What is a Good Block Cipher?

- For every key a block-cipher with plaintext/ciphertext space $\{0,1\}^n$ gives a permutation of $\{0,1\}^n$.

  What would be an good cipher?

- A good cipher is one where each key gives a **randomly chosen permutation** of $\{0,1\}^n$.

  Why is this not possible?

- The representation of a single typical function $\{0,1\}^n \rightarrow \{0,1\}^n$ requires roughly $n2^n$ bits ($147 \times 10^{6 \cdot 3}$ for $n = 64$)

- What should we look for instead?

Something Smaller

Idea. Compose smaller weak ciphers into a large one. Mix the components “thoroughly”.

Something Smaller

Idea. Compose smaller weak ciphers into a large one. Mix the components "thoroughly".

Shannon (1948) calls this:

- **Diffusion.** "In the method of diffusion the statistical structure of M which leads to its redundancy is dissipated into long range statistics..."
- **Confusion.** "The method of confusion is to make the relation between the simple statistics of E and the simple description of K a very complex and involved one."

Substitution-Permutation Networks

Substitution-Permutation Networks (1/2)

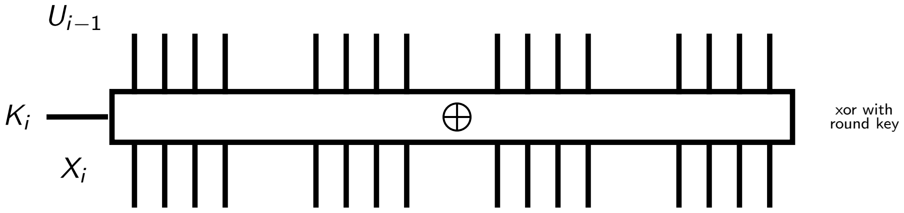
- **Block-size.** We use a block-size of $n = \ell \times m$ bits.
- **Key Schedule.** Round r uses its own round key K_r derived from the key K using a key schedule.
- **Each Round.** In each round we invoke:
  1. **Round Key.** xor with the round key.
  2. **Substitution.** ℓ substitution boxes each acting on one m-bit word (m-bit S-Boxes).
  3. **Permutation.** A permutation π_i acting on $\{1, \ldots, n\}$ to reorder the n bits.

Substitution-Permutation Networks (2/2)

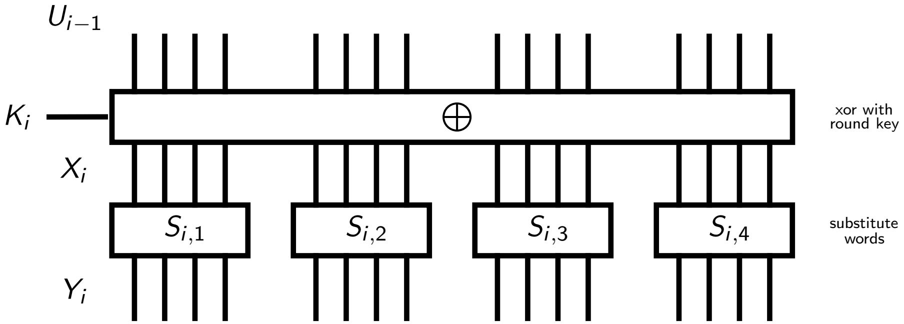
U_{i-1}

K_i

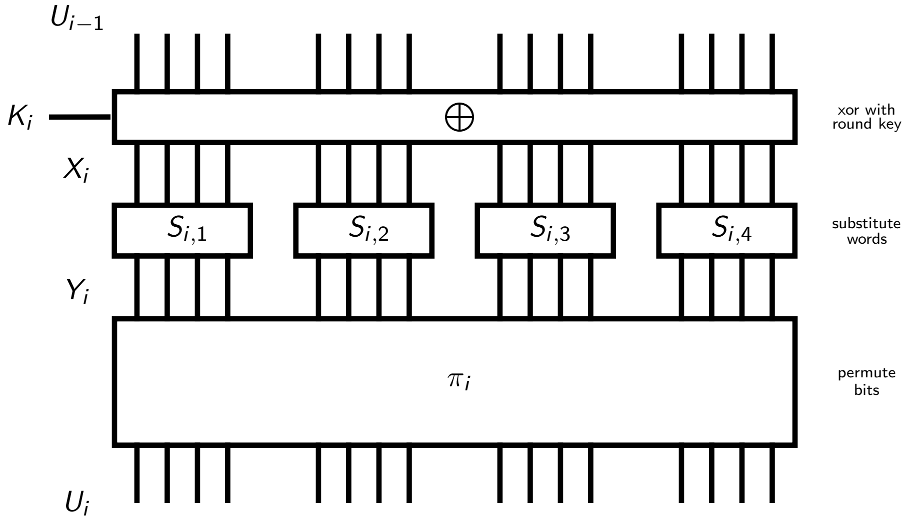
Substitution-Permutation Networks (2/2)



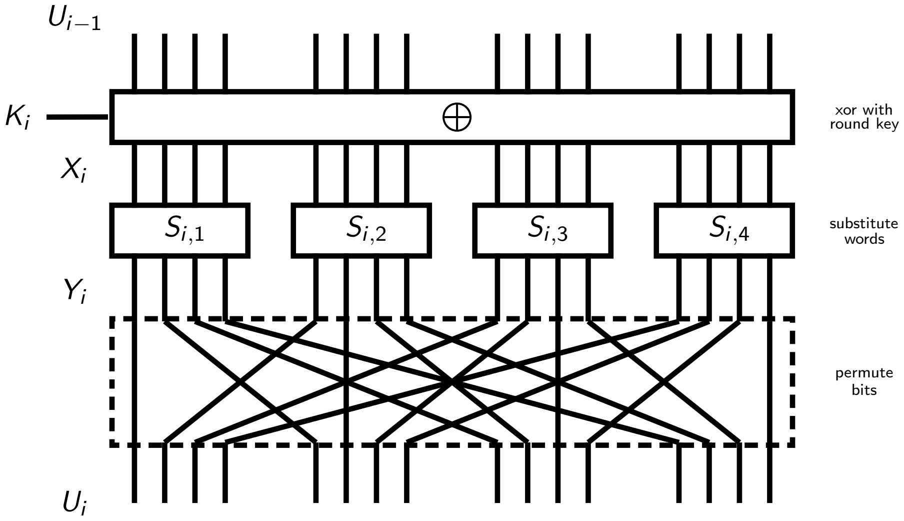
Substitution-Permutation Networks (2/2)



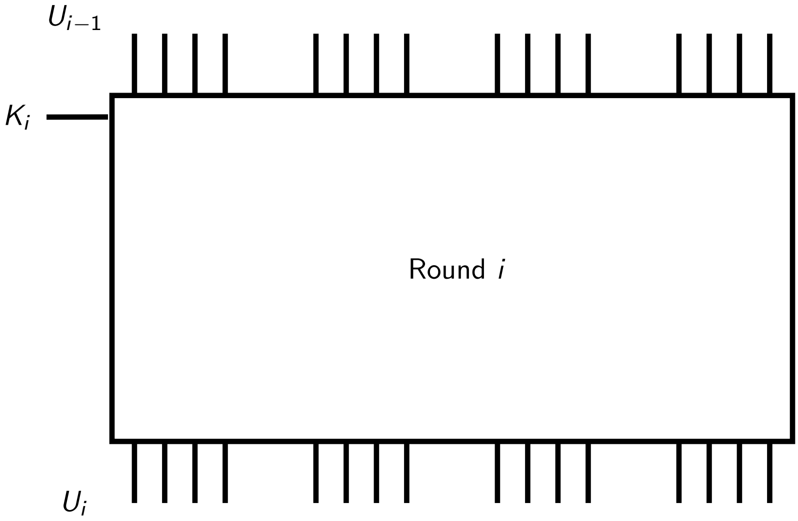
Substitution-Permutation Networks (2/2)



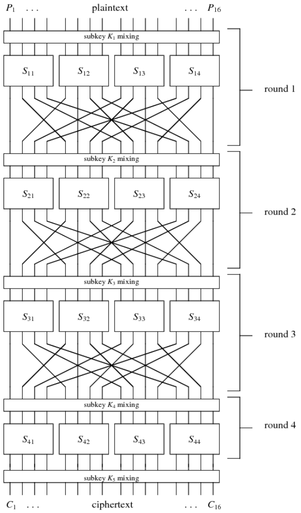
Substitution-Permutation Networks (2/2)



Substitution-Permutation Networks (2/2)



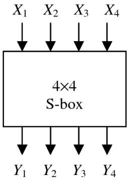
A Simple Block Cipher (1/2)



- $|P| = |C| = 16$
- 4 rounds
- $|K| = 32$
- rth round key K_r consists of the $4r$th to the $(4r + 16)$th bits of key K.
- 4-bit S-Boxes

A Simple Block Cipher (2/2)

S-Boxes the same ($S \neq S^{-1}$)

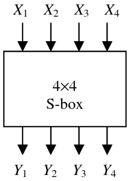


- $Y = S(X)$
- Can be described using 4 boolean functions

Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

A Simple Block Cipher (2/2)

S-Boxes the same ($S \neq S^{-1}$)



- $Y = S(X)$
- Can be described using 4 boolean functions

Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

16-bit permutation ($\pi = \pi^{-1}$)

Input	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Output	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

AES

Advanced Encryption Standard (AES)

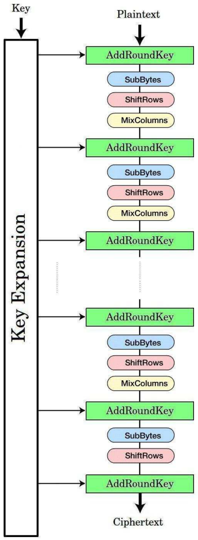
- Chosen in worldwide **public competition** 1997-2000. Probably no backdoors. Increased confidence!
- Winning proposal named "Rijndael", by Rijmen and Daemen
- Family of 128-bit block ciphers:

Key bits	128	192	256
Rounds	10	12	14

- The first key-recovery attacks on full AES due to Bogdanov, Khovratovich, and Rechberger, published **2011**, is faster than brute force by a factor of about **4**.

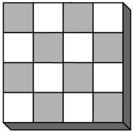
AES

- **AddRoundKey**: xor with round key.
- **SubBytes**: substitution of bytes.
- **ShiftRows**: permutation of bytes.
- **MixColumns**: linear map.



Similar to SPN

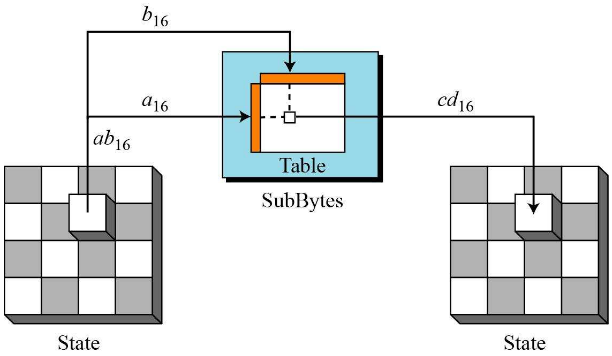
The 128 bit state is interpreted as a 4×4 matrix of bytes.



Something like a mix between substitution, permutation, affine version of Hill cipher. In each round!

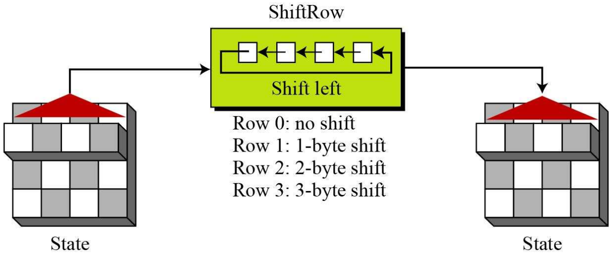
SubBytes

SubBytes is field inversion in $\mathbb{F}_{2^8}$ plus affine map in $\mathbb{F}_2^8$.



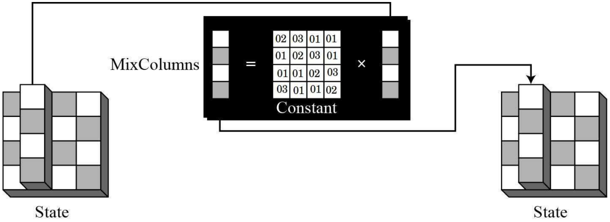
ShiftRows

`ShiftRows` is a cyclic shift of bytes with offsets: 0, 1, 2, and 3.



MixColumns

`MixColumns` is an invertible linear map over $\mathbb{F}_{2^8}$ (with irreducibile polynomial $x^8 + x^4 + x^3 + x + 1$) with good diffusion.



Decryption

Uses the following transforms:

- **AddRoundKey**
- **InvSubBytes**
- **InvShiftRows**
- **InvMixColumns**

Feistel Networks

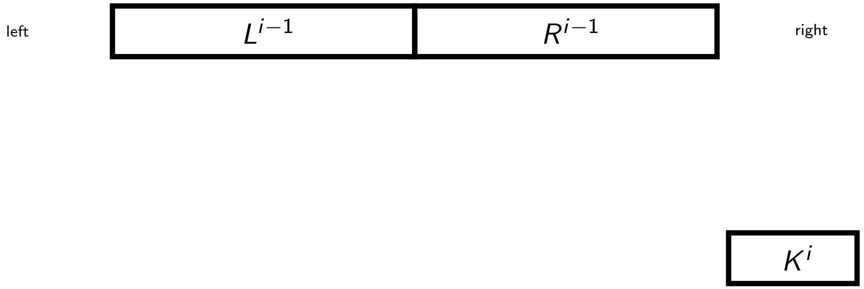
Feistel Networks

- Identical rounds are iterated, but with different round keys.
- The input to the *i*th round is divided in a left and right part, denoted L^{i-1} and R^{i-1}.
- f is a function for which it is somewhat hard to find pre-images, but f is **not invertible**!
- One round is defined by:

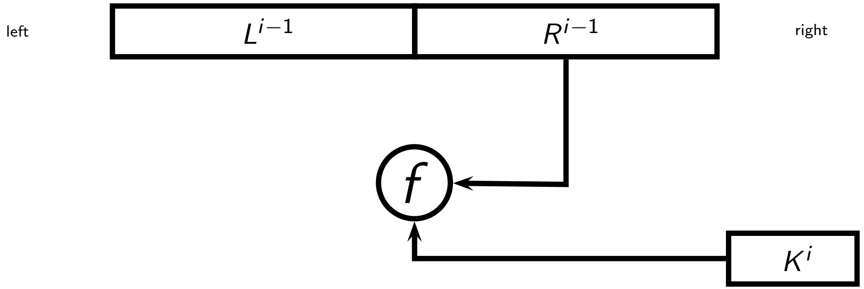
$$\begin{aligned} L^i &= R^{i-1} \\ R^i &= L^{i-1} \oplus f(R^{i-1}, K^i) \end{aligned}$$

where K^i is the *i*th round key.

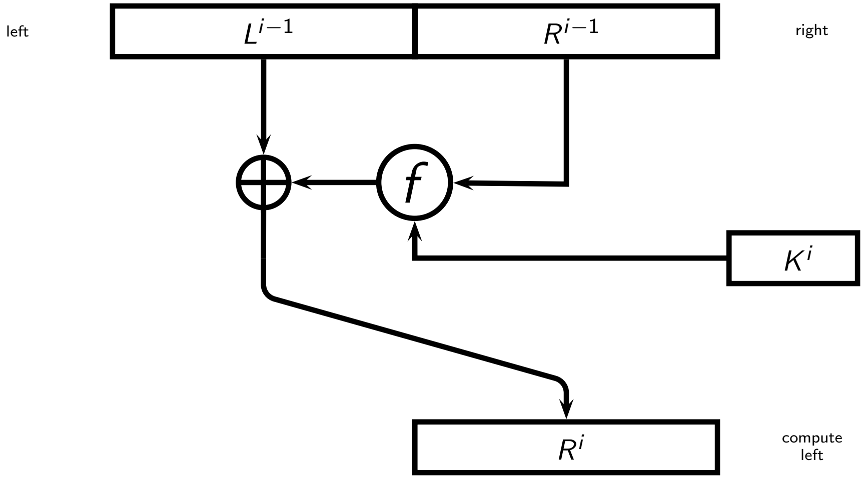
Feistel Round



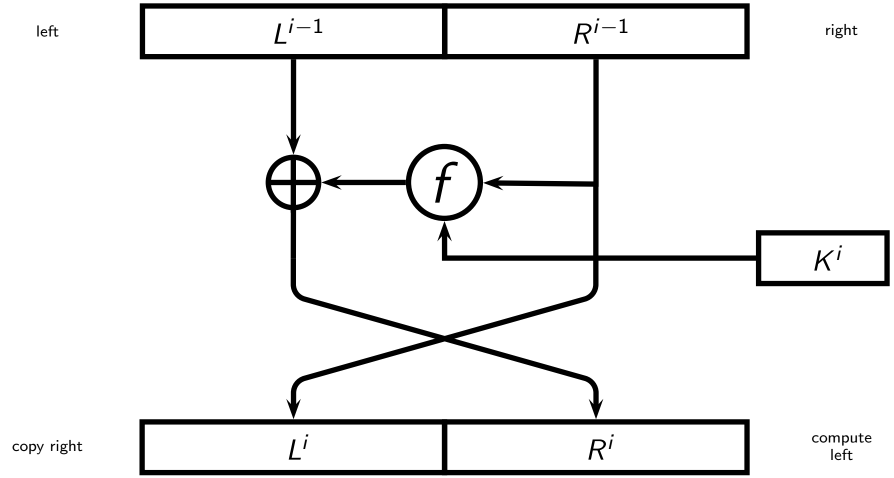
Feistel Round



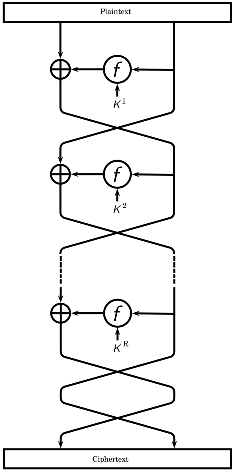
Feistel Round



Feistel Round



Feistel Cipher



Inverse Feistel Round

Feistel Round.

$$
\begin{aligned}
L^i &= R^{i-1} \\
R^i &= L^{i-1} \oplus f(R^{i-1}, K^i)
\end{aligned}
$$

Inverse Feistel Round

Feistel Round.

$$\begin{aligned} L^i &= R^{i-1} \\ R^i &= L^{i-1} \oplus f(R^{i-1}, K^i) \end{aligned}$$

Inverse Feistel Round.

$$\begin{aligned} L^{i-1} &= R^i \oplus f(L^i, K^i) \\ R^{i-1} &= L^i \end{aligned}$$

Reverse direction and swap left and right!

DES

Quote

The news here is not that DES is insecure, that hardware algorithm-crackers can be built, or that a 56-bit key length is too short. ... The news is how long the government has been denying that these machines were possible. As recently as 8 June 98, Robert Litt, principal associate deputy attorney general at the Department of Justice, denied that it was possible for the FBI to crack DES. ... My comment was that the FBI is either incompetent or lying, or both.

– Bruce Schneier, 1998

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.

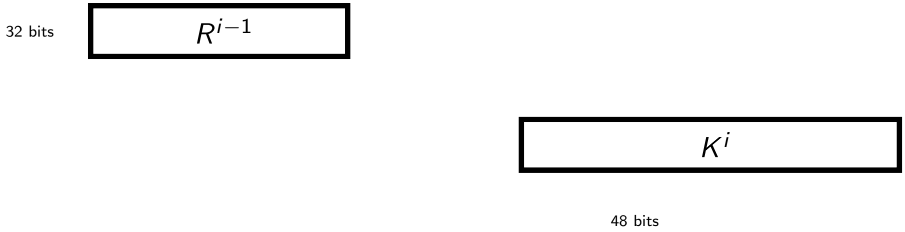
Data Encryption Standard (DES)

- 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.

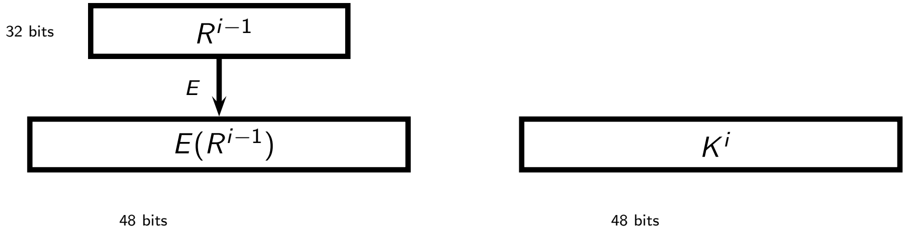
Data Encryption Standard (DES)

- 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.

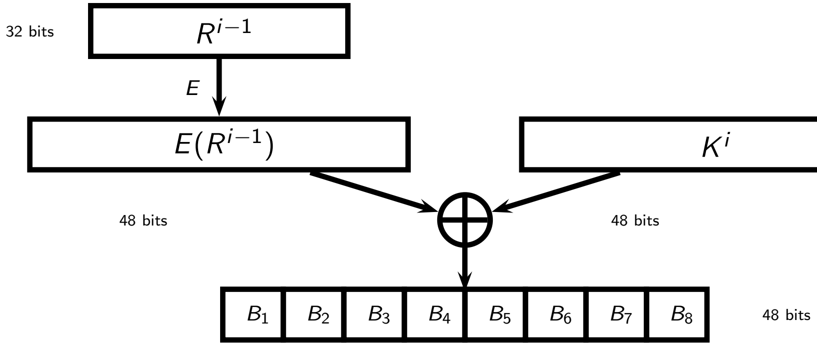
DES's f-Function



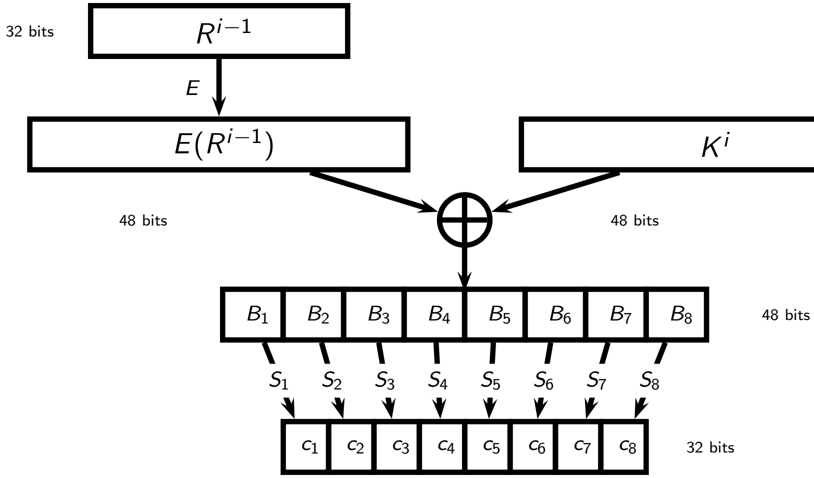
DES's f-Function



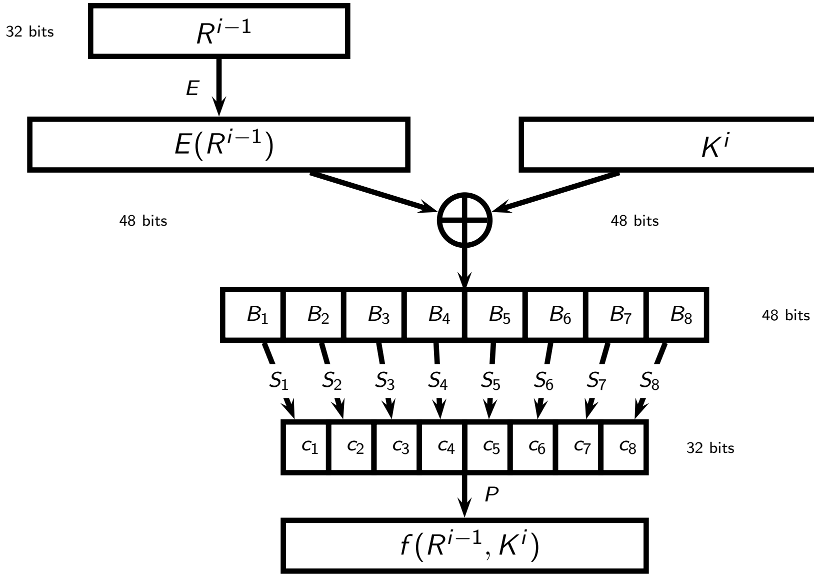
DES's f-Function



DES's f-Function


32 bits
R^{i-1}
E
$E(R^{i-1})$
K^i
48 bits
48 bits
B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8
48 bits
S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8
32 bits

DES's f-Function


32 bits
R^{i-1}
E
$E(R^{i-1})$
K^i
48 bits
48 bits
B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8
48 bits
S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8
32 bits
P
$f(R^{i-1}, K^i)$

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!

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?

This question is valid for any cipher.