B.c) DES

B.61 Remark

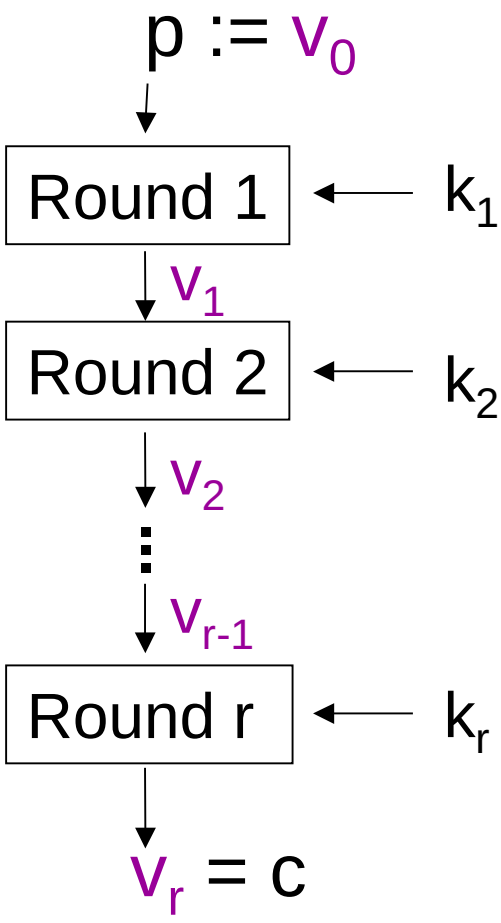
- There exist $(2^n)!$ permutations $\{0,1\}^n \to \{0,1\}^n$.
- Clearly, $|K| \leq (2^n)!$ for any block cipher with block length n.
- In a *true random block cipher* the encryption transformation is selected according to the uniform distribution on the set of all permutations on $\{0,1\}^n$.
- For all widespread block ciphers the number of encryption transformations $|K|$ is much smaller than $(2^n)!$.
- However, roughly speaking, the encryption transformations should have similar statistical properties as randomly chosen permutations.

B.62 Round Based Block Ciphers

- For any reasonable block size n it is infeasible to implement a large set of arbitrary permutations efficiently ($\rightarrow$ memory, code, encryption time).
- Instead, block ciphers usually consist of several rounds. The round functions are easy to implement.

B.62 (continued)

- key scheduling: Round keys $k_1,k_2,\ldots,k_r$ are calculated from the key k



$$v_{j+1} = g_{j+1}(v_j,k_{j+1})$$

B.63 Round Functions: Significant Properties

- Typically, all round functions (maybe apart from the last one) are identical.
- Single round functions are cryptographically weak.
- Roughly speaking, the strength of a block cipher increases but its efficiency decreases with the number of rounds.
- Designers of cryptosystems try to determine a parameter r
  - **w** that is sufficiently large
  - **w** that is not significantly larger than necessary.

B.64 Feistel Cipher

A *Feistel cipher* is specific type of round-based block cipher.

- More precisely, let $v_j := (L_j, R_j)$ where
  - L_j denotes the left half of v_j (consisting of n/2 bits)
  - R_j denotes the right half of v_j (consisting of n/2 bits).

  then $v_{j+1} = (R_j , f_{j+1}(R_j,k_{j+1}) \oplus L_j) =: (L_{j+1}, R_{j+1})$
  for a suitable function f_{j+1} (usually $f_1 = \ldots = f_r$).
- After the final round the halves L_r and R_r are swapped (or, equivalently, there is no swap in the final round; see B.71)

Note: The function f need not be injective.

Details: Blackboard

B.65 Feistel Cipher: Significant Properties

From

$(L_{j+1}, R_{j+1}) = (R_j , f_{j+1}(R_j,k_{j+1}) \oplus L_j)$ [encryption]

we immediately obtain

$(L_{j+1}, f_{j+1}(R_j,k_{j+1}) \oplus R_{j+1}) = (R_j , L_j)$.

The Feistel structure implies $R_j = L_{j+1}$.

This leads to

$(L_{j+1}, f_{j+1}(L_{j+1},k_{j+1}) \oplus R_{j+1}) = (R_j , L_j)$.

B.65 (continued)

Consequence: For Feistel ciphers encryption and decryption are the same apart from the order of the round keys (cf. B.78).

This property is relevant especially for smart cards as it saves code, memory and often also hardware. The benefit was even more important in the early years of smart cards.

B.66 DES (Data Encryption Standard)

DES is a symmetric block cipher with

- plaintext space P = ciphertext space $C = \{0,1\}^{64}$
- key space $K = \{0,1\}^{56}$ (effective key space)

DES is a Feistel cipher with r = 16 rounds.

B.67 DES: Effective Key Length

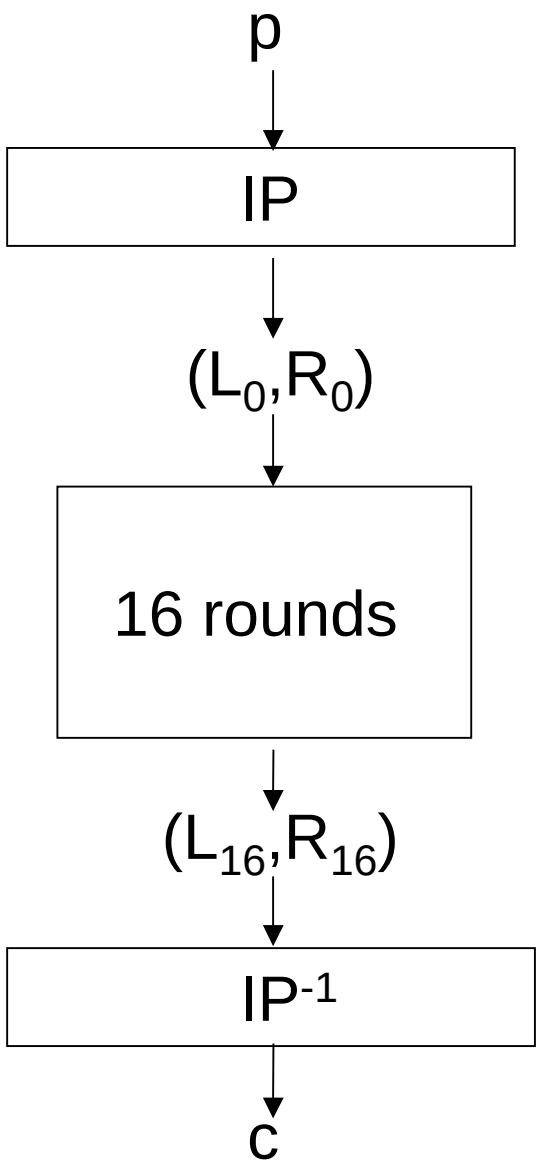
Note: DES keys consist of 64 bits, of which yet 8 bits are control bits (last bit of each byte). More precisely, each key byte has odd parity, and the control bits are not used for encryption. That is, the effective key length is 56 bit.

Example: F1 F4 32 10 75 80 08 01 (hexadecimal) is a valid DES key.

B.68 Remark

- The DES algorithm and the Triple-DES algorithm (see B.88) have worldwide been used for almost 30 years.
- DES was standardized by NIST from 1977 to 2005. In the last years the use of Triple-DES was recommended.
- Although the NIST standard already expired especially financial applications almost exclusively use the DES algorithm or the Triple-DES algorithm.
- The DES algorithm is maybe the mostly studied cryptographic algorithm worldwide.
- Although the DES algorithm has been publicly known since 1977 its design criteria have not been made public.

B.69 DES (coarse structure)



key-independent (fixed) permutation

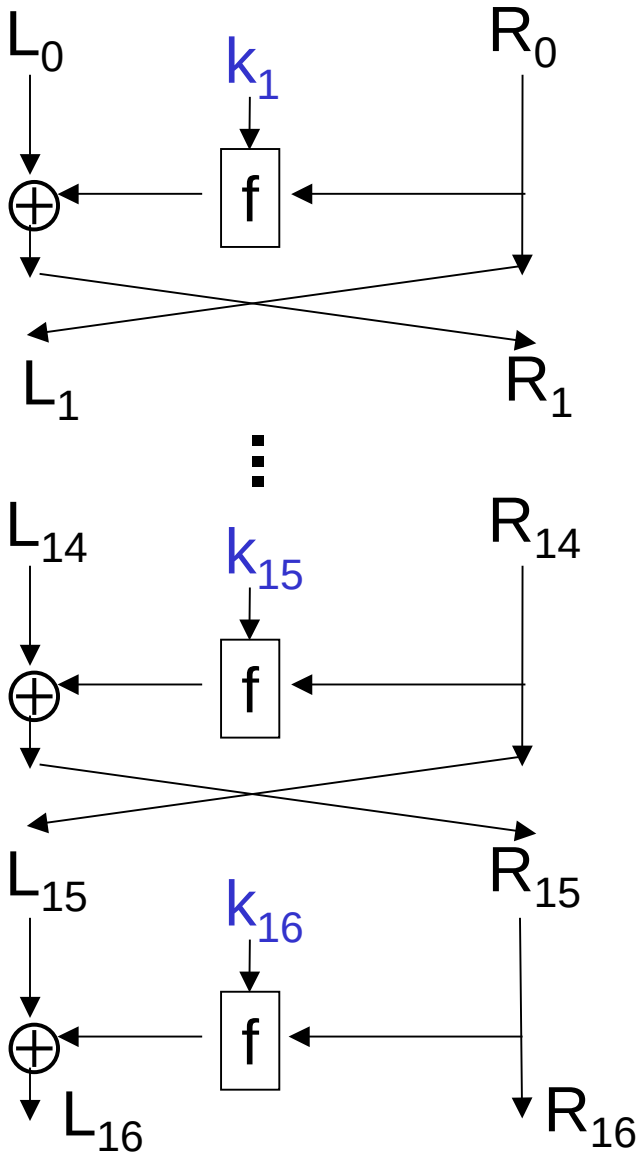
Feistel structure

key-independent (fixed) permutation

B.70 Initial permutation IP

- IP: $\{0,1\}^{64} \rightarrow \{0,1\}^{64}$ defines a key-independent permutation (initial permutation).
- After the final round its inverse IP^{-1} is applied.

B.71 DES: Feistel Structure



1st round

2st – 14th round

15th round

16th round (exceptional;
no switching)

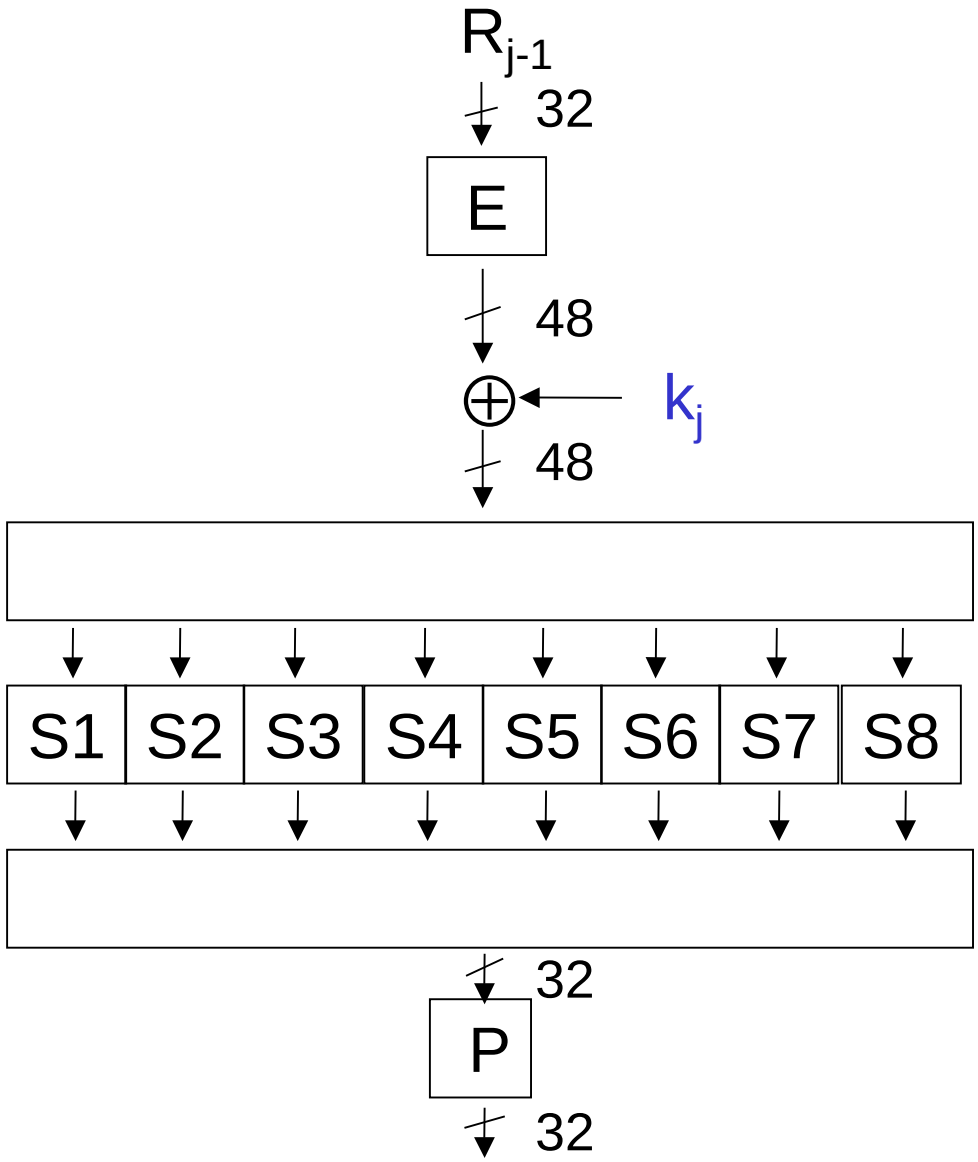
B.72 DES: Key Scheduling

- From the key $k \in \{0,1\}^{56}$ sixteen round keys $k_1, k_2, \ldots, k_{16}$ are deduced. Each of these round keys consists of 48 bits.
- Therefore, the 56 key bits are read in two 28 bit registers. Then

  for j=1 to 16 do {
  - Depending on j both registers are rotated by 1 or 2 positions
  - From each register 24 bits are selected and permuted, forming a 48 bit round key k_j

  }

B.73 DES: Round Function f



$f: \{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$

expansion

8 x 6 = 48 bits

S-boxes

8 x 4 = 32 bits

round permutation

B.73 (continued)

- E: $\{0,1\}^{32} \rightarrow \{0,1\}^{48}$ expands the 32 bit vector R_{j-1} to 48 bits. More precisely, 16 input bits are doubled.

- S1, S2, …, S8: $\{0,1\}^{6} \rightarrow \{0,1\}^{4}$ are (different) non-GF(2)-linear mappings.

- P: $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$ is a fixed permutation.

Note: As IP also E, S1,…,S8 and P are key-independent.

B.74 Remark

- The so-called S-boxes S1, S2, …, S8 are non-linear mappings. Their values are stored in 8 tables. Each table has 64 four-bit-entries.
- The choice of the S-boxes is crucial for the security of DES. Already reordering the S-boxes may increase its vulnerability against particular attacks.
- Precise definitions of IP, E, S1,…,S8, P and the key scheduling are given (e.g.) in “Handbook of Applied Cryptography”.

B.75 Further Properties

- A key k is called a *weak key* if $DES(p,k) = DES^{-1}(p,k)$. DES has four weak keys.

- $DES(\overline{p},\overline{k}) = \overline{DES(p,k)}$ (*inversion property*) where the bar stands for bitwise inversion

B.76 Cryptographic Strength of Single Rounds

- A single DES round and also the composition of a small number of DES rounds are cryptographically weak.

B.77 Example: 1 - Round DES

1st Step: Apply IP and IP^{-1} to the plaintext p and the ciphertext c, resp., to obtain (L_0,R_0) and (L_1,R_1)

2nd Step: We have $(L_1,R_1) = (L_0 \oplus f(R_0,k_1), R_0)$ [Note that the first round is at the same time the last round in 1-round DES!] More precisely, we have
$L_0 \oplus P(S(E(R_0) \oplus k_1)) = L_1$ with $S := S1 \times \ldots \times S8$
and hence

$S(E(R_0) \oplus k_1) = P^{-1}(L_1 \oplus L_0)$.

Note that apart from k_1 all functions and all vectors are known.

B.77 (continued)

This equation falls into eight independent equations, each containing a 6-bit subkey. That is, we have to solve nonlinear equations

$S_j(e_j \oplus k_{1,j}) = v_j$. for $j = 1,\ldots,8$

with known 6-bit vector e_j and a known 4 bit vector v_j. Each equation has 4 solutions, reducing the size of the search space for k_1 from 2^{48} to 2^{16}.

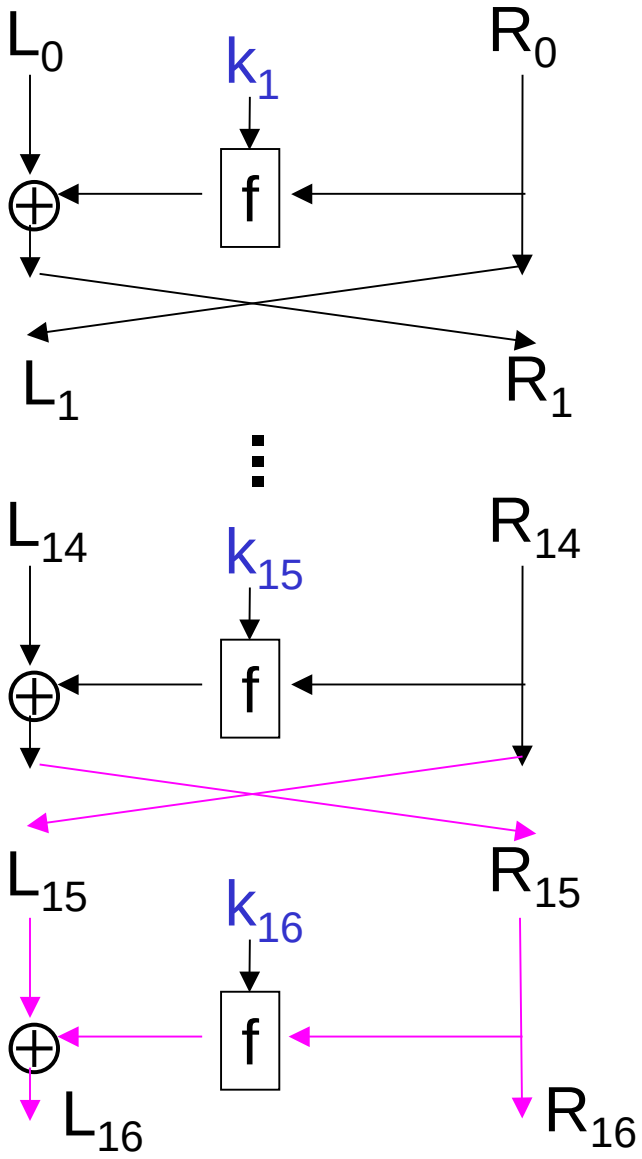
Consequence: Two known-plaintext pairs (p_1,c_1), (p_2,c_2) are sufficient to recover k_1.

B.77 (continued)

Details: Blackboard

Exercise: Work out an attack on 2-Round-DES.

B.78 Encryption and Decryption



Encryption

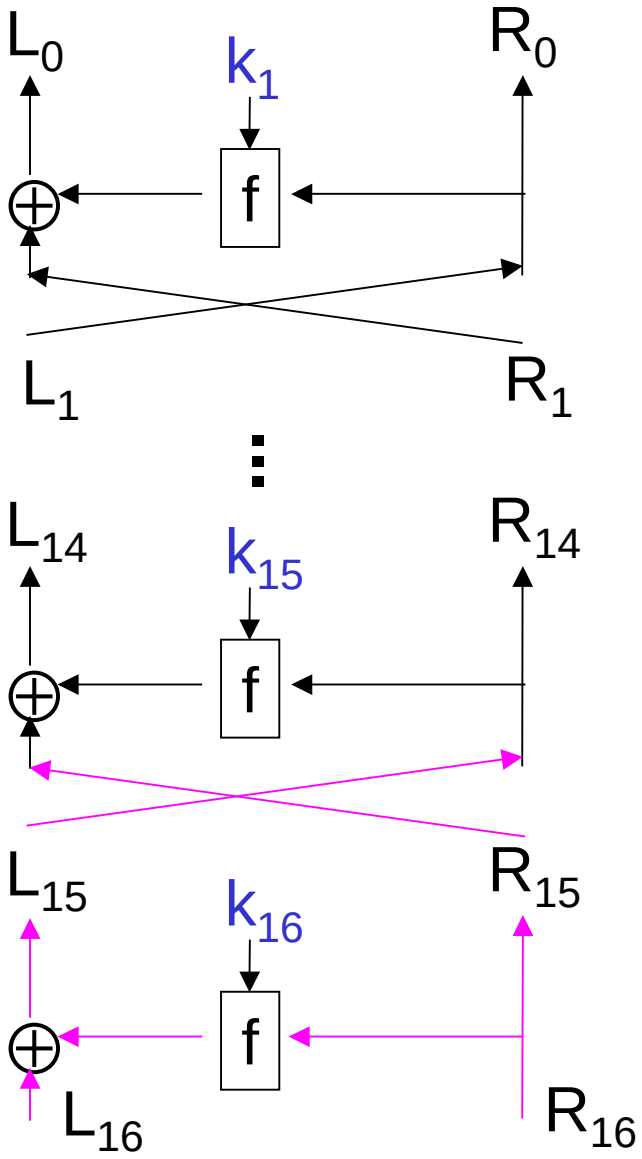
1st round

2st – 14th round

15th round

16th round (exceptional; no switching)

B.78 (continued)



Decryption

16th round (exceptional)

3st – 15th round

2nd round

1st round

B.79 Remark

Encryption and Decryption may be carried out using a common software- or hardware implementation. Only the order of the round keys has to be reversed.

B.80 Remark

- In many scenarios the initial and the final permutation have no cryptographic meaning (e.g., when the DES is used in EBC or CBC mode) since the adversary can simply “remove” IP and IP^{-1} (cf. Example B.77).
- It is easy to implement fixed permutations in hardware. Unlike in software implementations these permutations do not reduce the throughput.
- It has been conjectured that one reason to apply the initial and the final permutation was to prevent efficient software implementations ($\rightarrow$ late seventies). The DES algorithm has always been royalty-free.

B.81 Security: Exhaustive Key Search

- The DES key space K only contains 2^{56} keys. An exhaustive key search requires one known (plaintext, ciphertext) pair (in rare cases two pairs) and 2^{55} DES encryptions in average.

- When the DES was adopted standard in 1977 an exhaustive key search (if feasible at all) had demanded giantic efforts. Technical progress changed the case. Hence the DES algorithm has not been viewed secure against powerful adversaries for many years.

B.81 (continued): Milestones

- Wiener (1993): describes an ASIC design at gate level but does not provide “real” hardware
  - est. average search time per DES key: 3.5 hours
  - estimated costs: 1 million $
- EFF (Electronic Frontier Foundation, 1998): real hardware
  - average search time per DES key: 5 days
  - costs: 250 000 $
- University of Bochum (chair of Prof. Paar, 2006): real hardware (FPGAs)
  - average search time per DES key: 9 days
  - costs: < 9000 €

B.82 Consequences

• In sensitive applications the DES algorithm has been substituted by the Triple-DES algorithm (see B.88). The key space of Triple-DES equals $\{0,1\}^{112}$ or $\{0,1\}^{168}$.

B.83 Merkle's Time-Memory Trade-off

Assume that an adversary aims to find several keys of a block cipher Enc (and not just one). If he has sufficient storage he can accelerate the search for individual keys.

Setup-Step (to be performed only once): The adversary initializes a table T that contains about $|K|^{2/3}$ keys.

Search Step (to be performed in each key search): The adversary uses the table T to find a particular key.

B.83 (continued)

Efficiency:

- Setup costs
  - memory: $O(|K|^{2/3})$ keys
  - time: $O(|K|)$ operations
- Search Step
  - time: $O(|K|^{2/3})$ operations

DES: $|K| = 2^{56}$

B.84 Remark

• Apart from exhaustive key search also other types of cryptanalytic attacks on DES have been investigated, e.g. the *linear attack* (see B.85) and the *differential attack* (see B.86).

B.85 Linear Attack

The linear attack was introduced by Matsui (1993).

Basic idea: Let X denote random plaintext block. The adversary searches a GF(2)-linear functional

L: $P \times C \times K \rightarrow \{0,1\}$ (= XOR sum of plaintext bits, ciphertext bits and key bits) such that

$\text{Prob}(L(X,DES_0(X,k),k) = 0) = 0.5 + \varepsilon$ with $\varepsilon \neq 0$ (*)

for (at least a large subset) of the key space. Here $DES_0(.,.)$ denotes the DES cipher without IP and IP^{-1}.

B.85 (continued)

Note: (i) An adversary can easily “remove” the effect of the initial and final permutation: From the (plaintext, ciphertext) pair (p, DES(p,k)) he simply computes (IP(p),IP(DES(p,k))).
(ii) $L(p,c,k) = L_1(p) \oplus L_2(c) \oplus L_3(k)$ for suitable linear functionals on *P*, *C* and *K*.

The adversary substitutes known (plaintext, ciphertext) pairs $(p_1,c_{1:})$, …, (p_N,c_N) (for DES_0) into $L(\cdot,\cdot,\cdot)$.

B.85 (continued)

- Decision rule (for $\varepsilon > 0$):

Set $L_3(k) := 0$ if

$$(L_1(p_1) \oplus L_2(c_1)) + \ldots + (L_1(p_1) \oplus L_2(c_1)) < N / 2$$

and $L_3(k) := 1$ else.

Note: If this decision is correct it gives one bit of information on the key, halving the key space. Applying this procedure to m linear independent linear functionals reduces the key space by the factor 2^m .

Details: Blackboard

B.85 (continued)

- Goal: Find linear functionals L with large $|\varepsilon|$
- This is difficult.
- The known functionals are compositions of several functionals over a small number of rounds. Their overall probability decreases exponentially with the number of rounds.
- Property (*) can usually only be shown for random subkeys ($\rightarrow$ average of individual probabilities over all keys). However, this seems to imply (*).

B.85 (continued)

- Matsui combined a linear functional L with nonlinear terms (expressing the 1^{st} and the 16^{th} round, restricted to one particular S-box).
- At cost of evaluating the decision rule 2^{12} times (substitution of two 6-bit subkey candidates into the non-linear terms) this advanced attack provides 13 bits of information on the key space.
- Matsui used two linear functionals (in combination with nonlinear terms), reducing the key space from 2^{56} to 2^{30}.

B.85 (continued)

- Efficiency: known plaintext attack, requires about 2^{43} (plaintext, ciphertext) pairs to obtain a success probability $\approx$ 85 %)
- This limits the practical applicability of the linear attack on the DES cipher.

B.86 Differential Attack

The differential attack was introduced by Biham and Shamir (1991)

Basic idea: Let X denote random plaintext after the initial permutation and $DES_{(15)}(.,.)$ the intermediate result after 15 rounds. Find "differences" $\Delta, \Delta' \in \{0,1\}^{64}$ for which

$$Prob(DES_{(15)}(X+\Delta,k) \oplus DES_{(15)}(X,k) = \Delta') = 2^{-64} + \varepsilon$$

with $\varepsilon > 0$ for (at least a large subset) of the key space.

The adversary uses this relation to estimate 6-bit subkeys. Details: Blackboard

B.86 (continued)

• Efficiency: requires about 2^{47} chosen (plaintext, ciphertext) pairs

• This limits the practical applicability of the differential attack on the DES cipher.

B.87 Remark

- The differential attack is a universal tool which was very efficient against other block ciphers. FEAL-8, for instance, could be broken with only 128 chosen (plaintext,ciphertext) pairs.
- In 1994 D. Coppersmith, one of the designers of DES, published a paper that states that the resistance against differential attacks was one of the (unpublished) design criteria of DES.

B.88 Triple-DES

• Let **k** = (k_1,k_2,k_3). The *Triple-DES* (*TDES*, *3DES*) algorithm is defined as follows:

3DES(p,**k**):= DES(DES^{-1}(DES(p,k_1),k_2),k_3).

We distinguish two cases:

- two-key Triple DES: $k_1 = k_3$, $K = \{0,1\}^{112}$
- three-key Triple DES: three independent DES keys, $K = \{0,1\}^{168}$

B.89 Remark

- The Triple-DES algorithm counteracts the small key space of the DES algorithm. Both the three-key Triple-DES and the two-key Triple-DES are viewed as secure against strong adversaries.
- The migration from DES to Triple-DES did not require new hardware.

B.89 (continued)

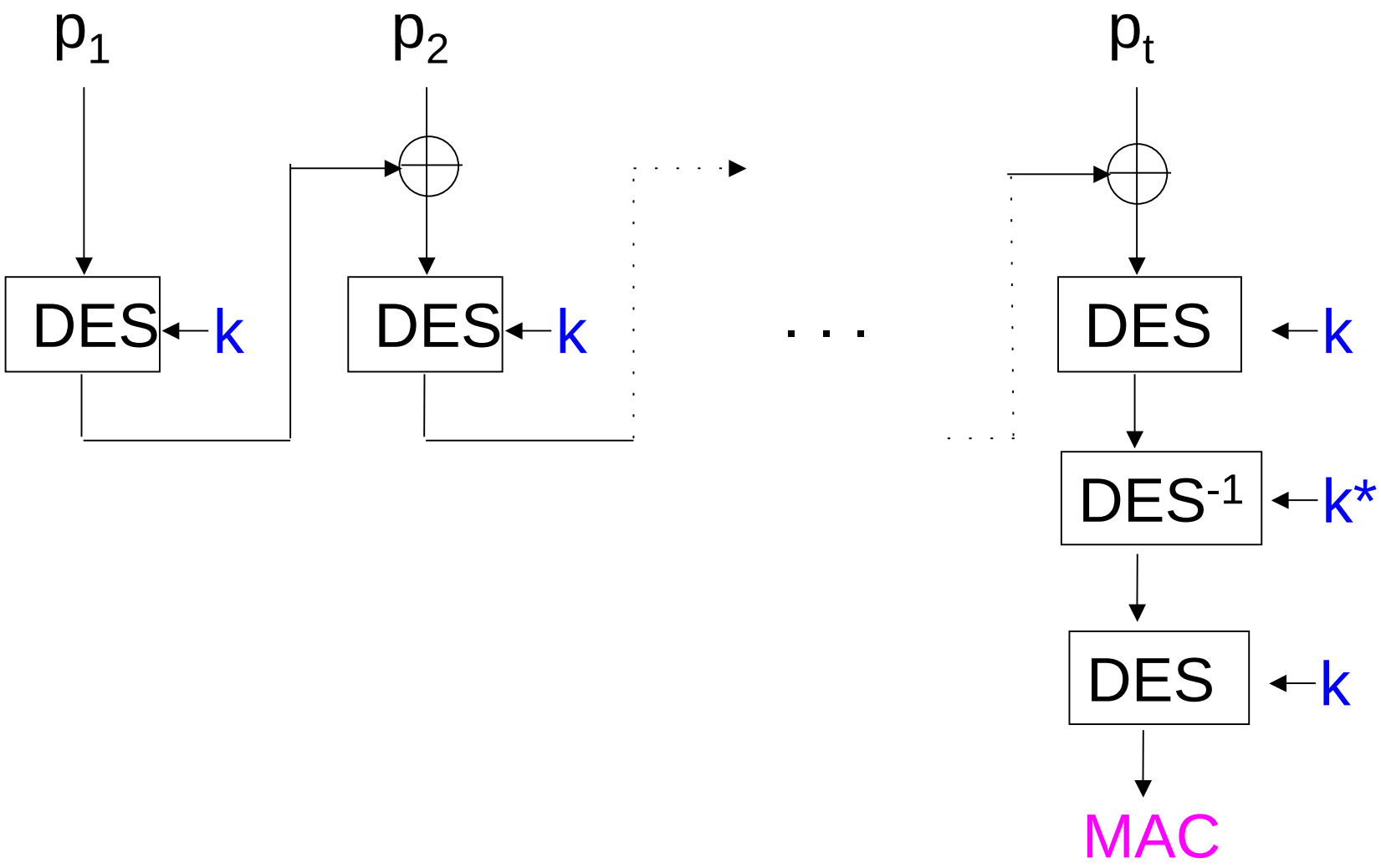
- The definition of the Triple-DES algorithm is surprising at first sight as one would expect DES(DES(DES(p,k_1),k_2),k_3) which seemed more "natural".

The Triple-DES definition from B.88, however, is compatible with the single DES if $k_1 = k_2 = k_3$. This was an important aspect for the migration of systems that consisted of many different components.

B.89 (continued)

- The Triple-DES algorithm is widely used in many banking applications, e.g. for the PIN validation of German banking cards or to secure payments with electronic purses. Also the SSL cipher suite applies the Triple-DES algorithm.

B.90 Retail CBC – MAC with Enc = DES



B.91 Remark

- The Retail CBC-MAC with Enc = DES was the answer on the fact that exhaustive key search against DES had become feasible.
- Compared to a MAC construction (e.g., the CMAC) with Enc = Triple-DES it saves computation time.
- However, if the attacker knows about 2^{32} (message, MAC) pairs he can mount an instructive attack (cf. B.93).

B.92 The Birthday Paradox

- Suppose that an urn contains m balls that are labelled with numbers 1,…,m.
- Assume that a player draws one ball, reads its label and puts the ball back into the urn. The player repeats this process r times.
- Determine the probability p(r) that the player has drawn r different balls:

$$p(r) = (m/m)*((m-1)/m)*\ldots*((m-r+1)/m)$$
$$= 1*(1-1/m)*\ldots*(1-(r-1)/m)$$

B.92 (continued)

Note: Given a group of at least 23 randomly chosen people the probability that at least two of them have the same birthday is more than 0.5.

For $r<<m$ the Taylor expansion of the natural logarithm log around 1, i.e. $\log(1-x) = -x + O(x^2)$ gives

$$\log(p(r)) \approx 0 - 1/m - \ldots - (r-1)/m$$
$$\approx -r(r-1)/(2m),$$

i.e. $p(r) \approx \exp(-r(r-1) / (2m))$ if $r<<m$.

B.92 (continued)

Note: For large m this formula implies that it is very likely that the player draws at least one ball twice if $r \approx m^{1/2}$.

This fact is important for several areas of cryptography.

B.93 Attacking the Retail-CBC-MAC with Enc=DES

Assumption: The adversary knows two different messages $m_1 = (p_1,\ldots,p_t)$ and $m_2 = (p'_1,\ldots,p'_s)$ with identical Retail-CBC-MACs (for identical but unknown keys k,k*).

Note: Due to B.92 this assumption is reasonable when the adversary observes about 2^{32} known (message,MAC) pairs to the same keys k,k*.

Note: Since the final decryption and encryption are bijectiions the assumption implies CBC-MAC(m_1,k) = CBC-MAC(m_2,k).

B.93 (continued)

Attack:

Step 1: The adversary computes CBC-MAC(m_1,k') and CBC-MAC(m_2,k') for different keys $k' \in \{0,1\}^{56}$ until he finds a key k'' that gives two equal MAC values. The adversary assumes that k'' = k.

Note: For the correct key k both CBC-MACs are indeed equal. The probability that a further key has this property is about $2^{-(56-64)} = 2^{-8}$.

B.93 (continued)

Note: If k'' = k then

$DES(DES^{-1}(m_1,k''),k^*) = CBC\text{-}MAC(m_1,k'')$

Step 2: The adversary uses this equation to find k* by exhaustive key search.

Step 3: The adversary verifies the obtained key pair (k'',k*') at another known (message, Retail-CBC-MAC). If this candidate pair turns out to be wrong he goes back to Step 2 or possibly to Step 1.

B.93 (continued)

Efficiency (average case):

Step 1: $(2^{56} (t+s) / 2)$ DES encryptions $(= 2^{57}$ for t=s=2)
Step 2: $(2^{56} / 2)$ DES encryptions

Note: Provided that the adversary has access to about 2^{32} (message, Retail-CBC-MAC) a key recovery attack is not significantly more difficult than a key recovery attack on DES.
For t = s = 2 this attack requires about 5 times the number of encryptions of an exhaustive key search on DES.

B.94 Remark

Countermeasures:

- The designer takes care that any key pair (k,k*) is used for $r << 2^{32}$ Retail-CBC-MACs. E.g., he may use only
  - session keys
  - a counter
- The DES algorithm may be substituted by a block cipher that does not allow a key recovery attack.

B.95 Why not double DES?

The key space of the two-key Triple-DES is $\{0,1\}^{112}$. Hence it seems to be reasonable to apply Double-DES instead:

$2DES(p,k_1,k_2) := DES(DES(p,k_1),k_2)$.

Double-DES only has the same key space $\{0,1\}^{112}$ but saves one DES encryption.

Is the Double-DES algorithm as secure as the two-key Triple-DES?

B.95 (continued)

Answer: no

Fact: If the adversary has enough storage it requires essentially only 2^{56} DES encryptions and 2^{56} DES decryptions to recover a Double-DES key pair (k_1,k_2).

Attack: meet-in-the-middle attack
Details: Exercises

Hint: $DES(DES(p,k_1),k_2) = c$ is equivalent to $DES(p,k_1) = DES^{-1}(c,k_2)$