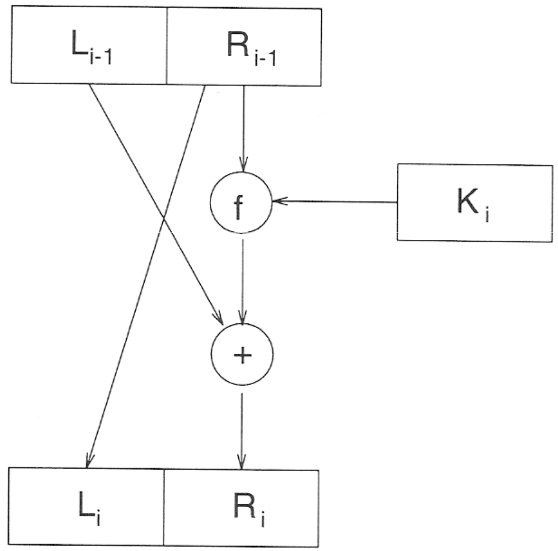
Introduction to Cryptography

Lecture 5

Benny Pinkas

Feistel Networks

- Encryption:
- *Input:* $P = L_{i-1} \mid R_{i-1}$. $|L_{i-1}|=|R_{i-1}|$
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus F(K_i, R_{i-1})$
- Decryption?

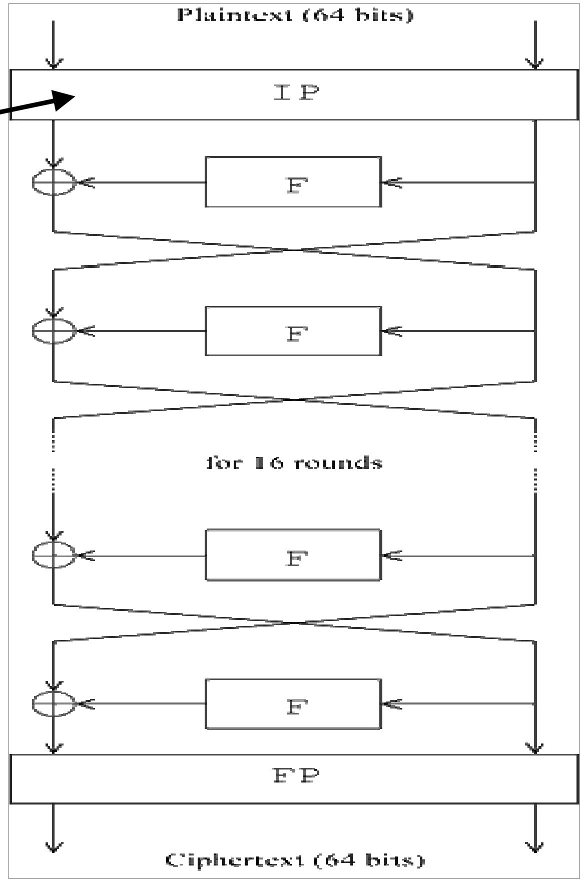


- No matter which function is used as F, we obtain a permutation (i.e., F is reversible even if *f* is not).
- The same code/circuit, with keys in reverse order, can be used for decryption.
- Theoretical result [LubRac]: If *f* is a pseudo-random *function* then a 4 rounds Feistel network gives a pseudo-random *permutation*

DES (Data Encryption Standard)

- A Feistel network encryption algorithm:
  - How many rounds?
  - How are the round keys generated?
  - What is F?
- DES (Data Encryption Standard)
  - Designed by IBM and the NSA, 1977.
  - 64 bit input and output
  - 56 bit key
  - 16 round Feistel network
  - Each round key is a 48 bit subset of the key

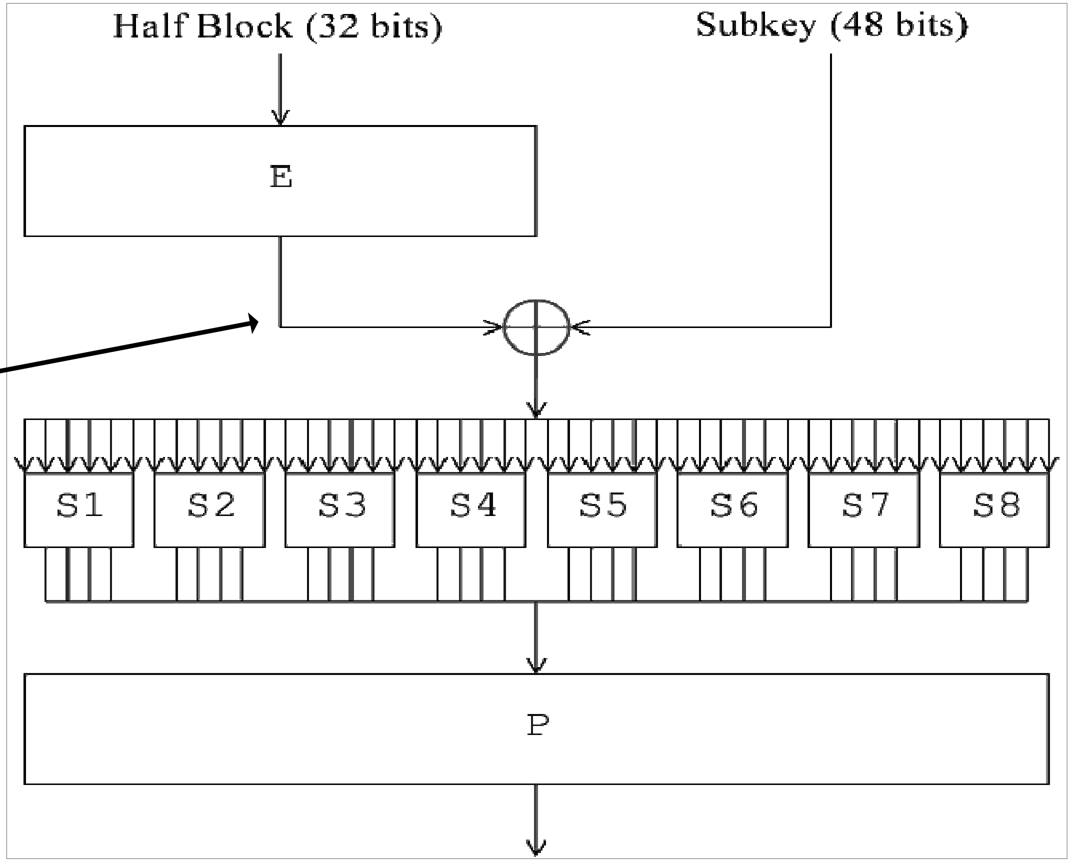
Internals of DES



Initial permutation of bit locations:

- not secret
- makes implementations in software less efficient

DES F functions



Expansion to 48 bits

The S-boxes

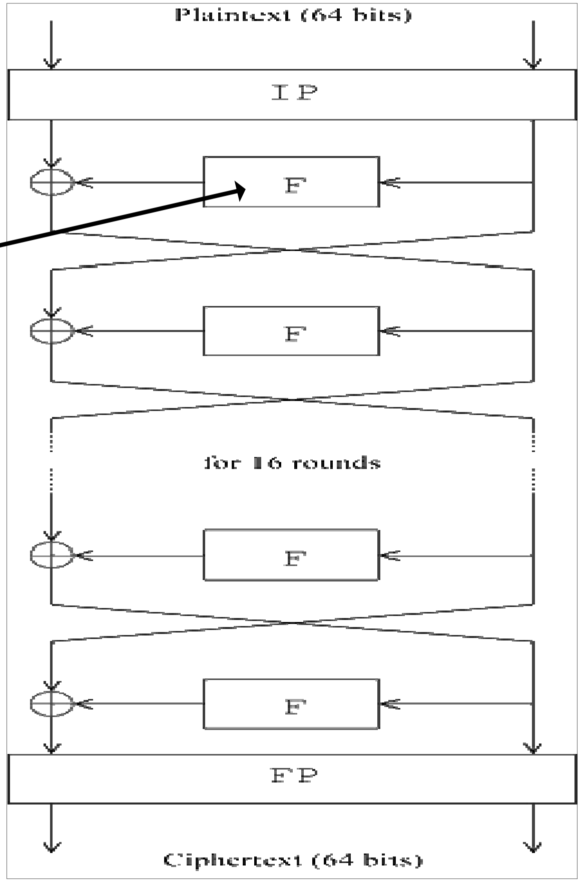
- Very careful design (it is now known that random choices for the S-boxes result in weak encryption).

- Each s-box maps 6 bits to 4 bits:
  - A 4×16 table of 4-bit entries.
  - Bits 1 and 6 choose the row, and bits 2-5 choose column.
  - Each row is a *permutation* of the values 0,1,…,15.
    - Therefore, given an output there are exactly 4 options for the input
  - Curcial property: Changing one input bit changes at least two output bits $\Rightarrow$ avalanche effect.

Differential Cryptanalysis of DES

DES diagram:

S-boxes



Differential Cryptanalysis [Biham-Shamir 1990]

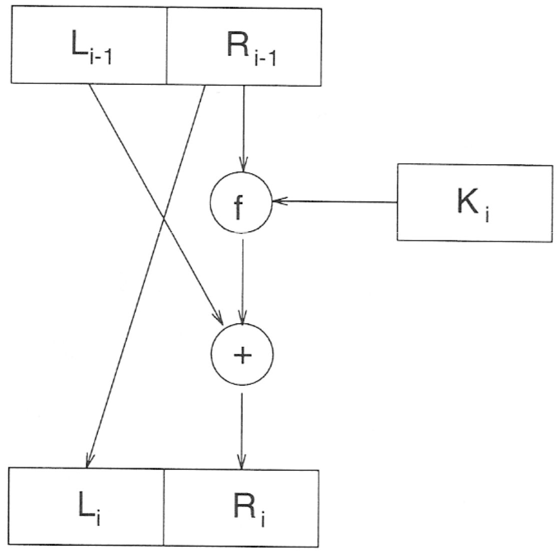
- The first attack to reduce the overhead of breaking DES to below exhaustive search
- Very powerful when applied to other encryption algorithms

- Depends on the structure of the encryption algorithm
- Observation: all operations except for the s-boxes are linear
- Linear operations:
  - $a = b \oplus c$
  - a = the bits of b in (a known) permuted order
- Linear relations can be exposed by solving a system of linear equations

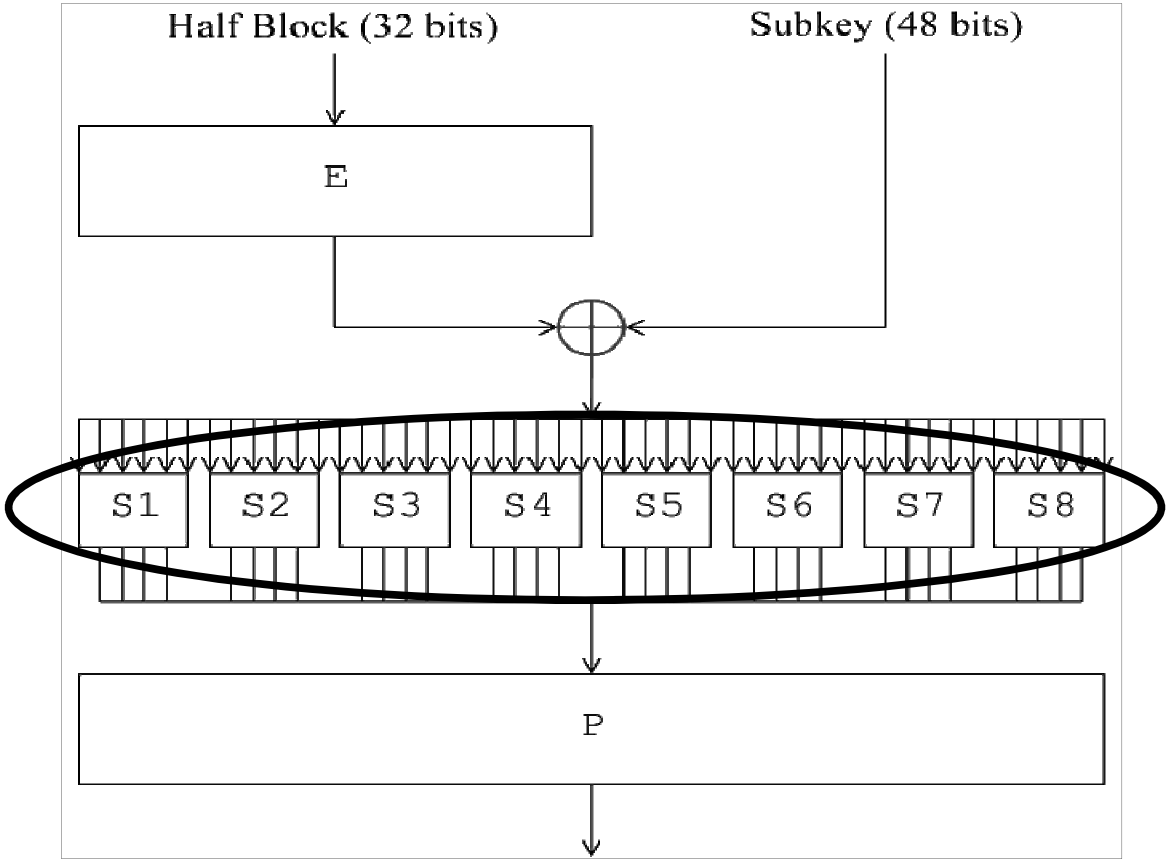
Is a Linear F in a Feistel Network secure?

- Suppose $F(R_{i-1},K_i) = R_{i-1} \oplus K_i$
  - Namely, F is linear
- Then $R_i = L_{i-1} \oplus R_{i-1} \oplus K_i$

  $L_i = R_{i-1}$
- Write L_{16}, R_{16} as linear functions of L_0, R_0 and K.
  - Given L_0R_0 and $L_{16}R_{16}$ Solve and find K.
- F must therefore be non-linear.
- F is the only source of non-linearity in DES.



DES F functions



Source of non-linearity

Differential Cryptanalysis

- The S-boxes are non-linear
- We study the differences between two encryptions of two different plaintexts

- Notation:
  - Denote two different plaintexts as P and P*
  - Their difference is $dP = P \oplus P^*$
  - Let X and X* be two intermediate values, for P and P*, respectively, in the encryption process.
  - Their difference is $dX = X \oplus X^*$
    - Namely, dX is always the result of two inputs

Differences and S-boxes

- S-box: a function (table) from 6 bit inputs to 4 bit output
- X and X* are inputs to the same S-box. We can compute their difference $dX = X \oplus X^*$.
- $Y = S(X)$
- When $dX=0$, $X=X^*$, and therefore $Y=S(X)=S(X^*)=Y^*$, and $dY=0$.
- When $dX\neq0$, $X\neq X^*$ and we don't know dY for sure, but we can investigate its distribution.
- For example,

Distribution of Y' for S1

- dX=110100
- There are 2^6=64 input pairs with this difference, { (000000,110100), (000001,110101),…}
- For each pair we can compute the xor of outputs of S1
- E.g., S1(000000)=1110, S1(110100)=1001. dY=0111.
- Table of frequencies of each dY:

0000	0001	0010	0011	0100	0101	0110	0111
0	8	16	6	2	0	0	12
1000	1001	1010	1011	1100	1101	1110	1111
6	0	0	0	0	8	0	6

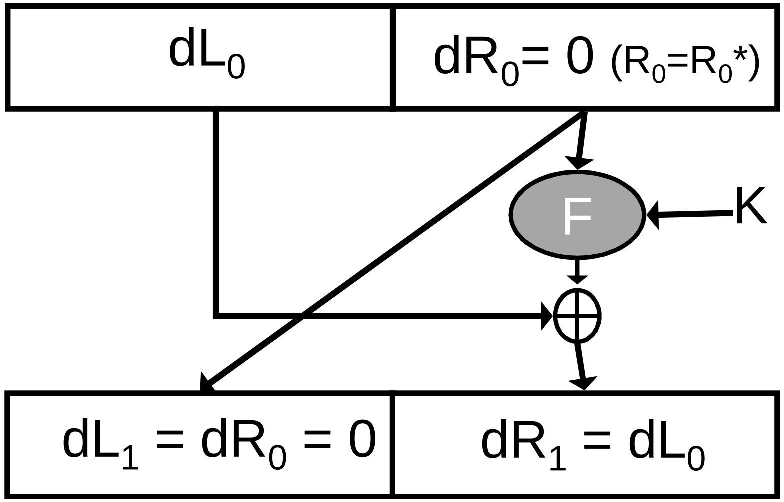
Differential Probabilities

- The probability of $dX \Rightarrow dY$ is the probability that a pair of inputs whose xor is dX, results in a pair of outputs whose xor is dY (for a given S-box).
- Namely, for dX=110100 these are the entries in the table divided by 64.

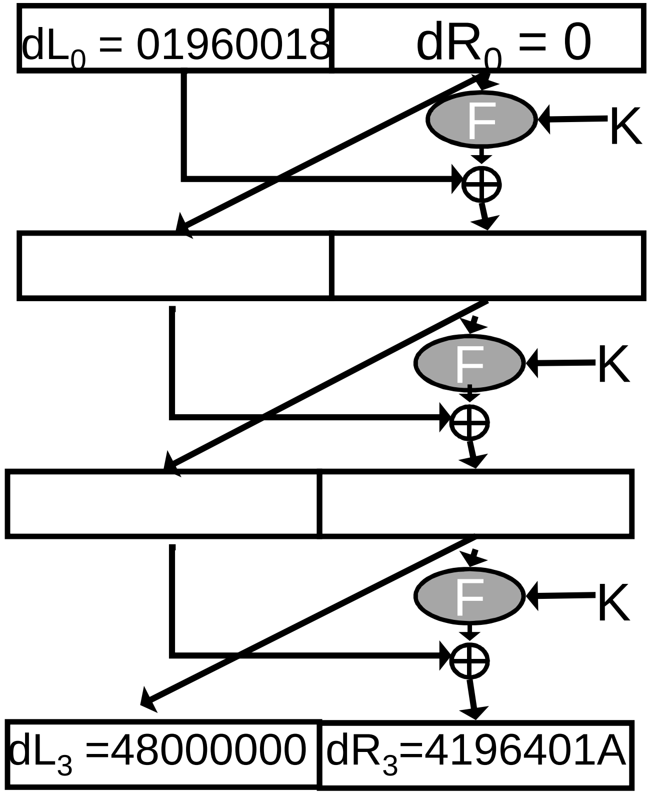
- Differential cryptanalysis uses entries with large values
  - $dX=0 \Rightarrow dY=0$
  - Entries with value 16/64
  - (Recall that the outputs of the S-box are uniformly distributed, so the attacker gains a lot by looking at differentials rather than the original values.)

Warmup

Inputs: L_0R_0, $L_0^*R_0^*$, s.t. $R_0=R_0^*$.
Namely, inputs whose xor is dL_0 0

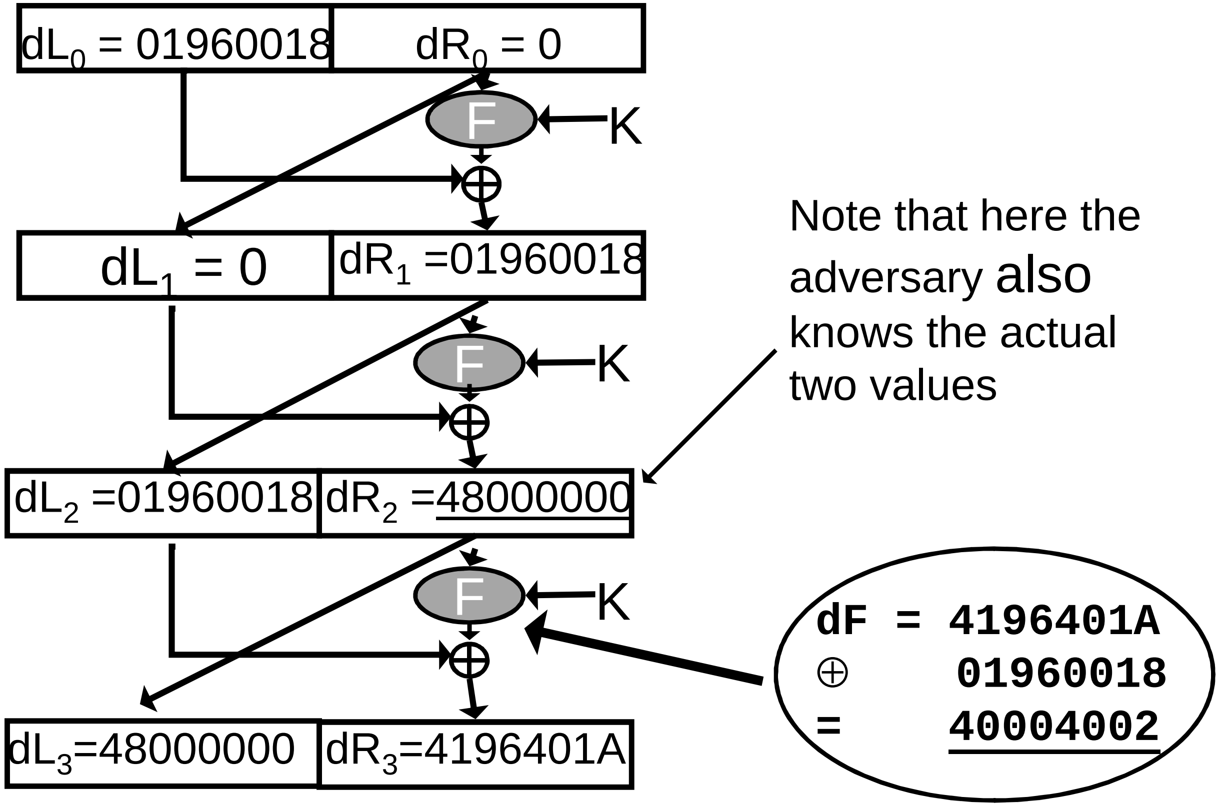


3 Round DES

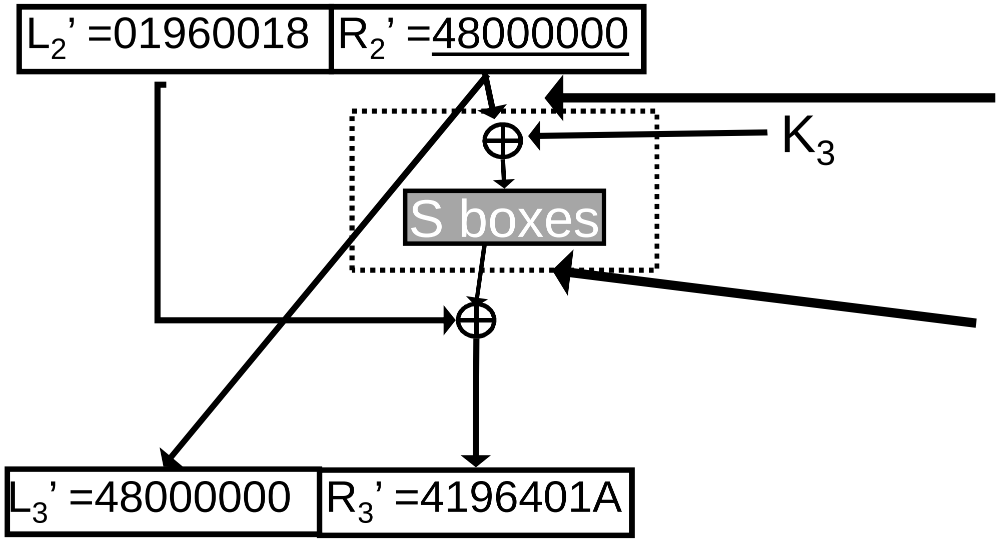


The attacker knows the two plaintext/ciphertext pairs, and therefore also their differences

Intermediate differences equal to plaintext/ciphertext differences



Finding K



The actual two inputs to F are known

Output xor of F (i.e., S boxes) is 40004002

⇒Table enumerates options for the pairs of inputs to S box

Find which K_3 maps the inputs to an s-box input pair that results in the output pair!

DES with more than 3 rounds

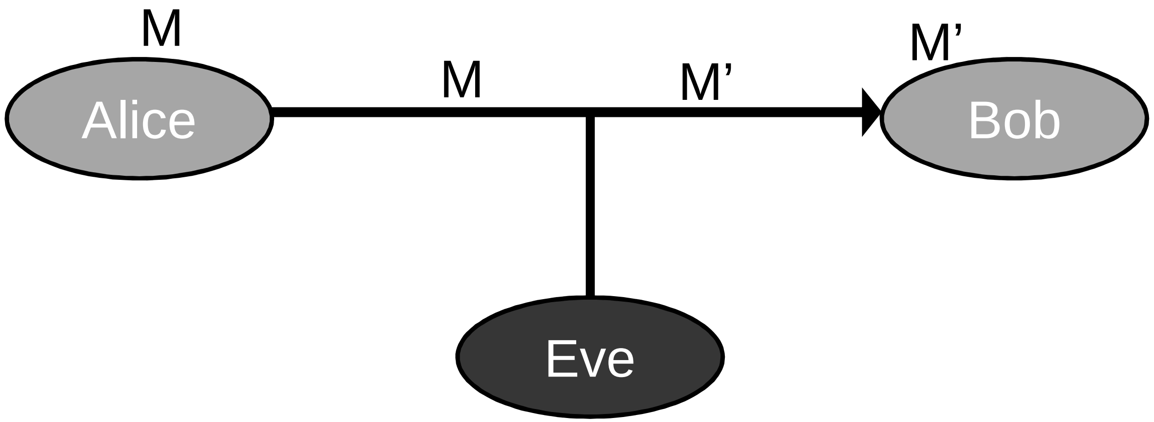
- Carefully choose pairs of plaintexts with specific xor, and determine xor of pairs of intermediate values at various rounds.
- E.g., if $dL_0=40080000_x$, $dR_0=04000000_x$
  Then, with probability ¼, $dL_3=04000000_x$, $dR_3=4008000_x$

- 8 round DES is broken given 2^{14} chosen plaintexts.
- 16 round DES is broken given 2^{47} chosen plaintexts...

Message Authentication

Data Integrity, Message Authentication

- Risk: an *active* adversary might change messages exchanged between Alice and Bob



• Authentication is orthogonal to secrecy. It is a relevant challenge regardless of whether encryption is applied.

One Time Pad

- OTP is a perfect cipher, yet provides no authentication
  – Plaintext $x_1x_2\ldots x_n$
  – Key $k_1k_2\ldots k_n$
  – Ciphertext $c_1=x_1\oplus k_1,\ c_2=x_2\oplus k_2,\ldots,c_n=x_n\oplus k_n$

- Adversary changes, e.g., c_2 to $1\oplus c_2$
- User decrypts $1\oplus x_2$

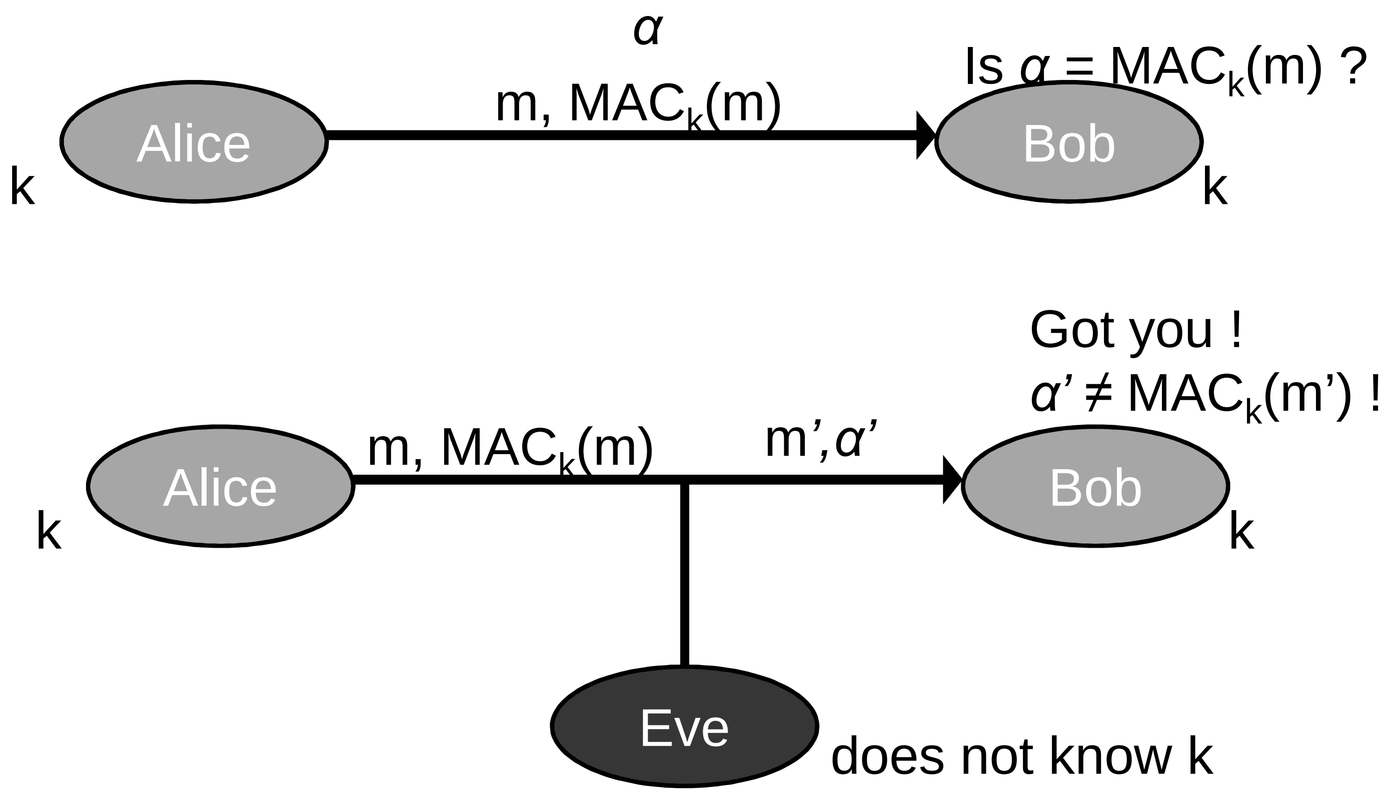
- Error-detection codes are insufficient. (For example, linear codes can be changed by the adversary, even if encrypted.)
  – They were not designed to withstand adversarial behavior.

Definitions

- Scenario: Alice and Bob share a secret key K.
- Authentication algorithm:
  - Compute a Message Authentication Code: $\alpha = MAC_K(m)$.
  - Send m and α
- Verification algorithm: $V_K(m, \alpha)$.
  - $V_K(m, MAC_K(m)) = accept.$
  - For $\alpha \neq MAC_K(m)$, $V_K(m, \alpha) = reject.$

- How does $V_k(m)$ work?
  - Receiver knows k. Receives m and α.
  - Receiver uses k to compute $MAC_K(m)$.
  - $V_K(m, \alpha) = 1$ iff $MAC_K(m) = \alpha$.

Common Usage of MACs for message authentication



Requirements

- Security: The adversary,
  - Knows the MAC algorithm (but not *K*).
  - Is given many pairs $(m_i, MAC_K(m_i))$, where the m_i values might also be chosen by the adversary (chosen plaintext).
  - Cannot compute $(m, MAC_K(m))$ for any new m ($\forall i\ m \neq m_i$).
  - The adversary must not be able to compute $MAC_K(m)$ *even* for a message *m* which is “meaningless” (since we don’t know the context of the attack).

- Efficiency: MAC output must be of fixed length, and as short as possible.
  - $\Rightarrow$ The MAC function is not 1-to-1.
  - $\Rightarrow$ An n bit MAC can be broken with prob. of at least 2^{-n}.

Constructing MACs

- Length of MAC output must be at least n bits, if we do not want the cheating probability to be greater than 2^{-n}

- Constructions of MACs
  - Based on block ciphers (CBC-MAC)

  or,
  - Based on hash functions
    - More efficient
    - At the time, encryption technology was controlled (export restricted) and it was preferable to use other means when possible.