Introduction to Linear Cryptanalysis and Differential Cryptanalysis

References:

Douglas R. Stinson, *Cryptography - Theory and Practice*

Howard M. Heys, *A Tutorial on Linear and Differential Cryptanalysis*

Content

Introduction

A common design for modern block ciphers is that of an iterated cipher: The cipher requires the specification of a ***round function*** and a ***key schedule***, and the encryption of a plaintext will proceed through Nr similar ***rounds***

- **random binary key** K**:** used to construct Nr ***round keys*** (also called ***subkeys***), which are denoted $K^1, \ldots, K^{\mathrm{Nr}}$
- **key schedule** $(K^1, \ldots, K^{\mathrm{Nr}})$**:** constructed from K using a fixed, public algorithm
- **round function** g**:** takes two inputs
  - a round key (K^r)
  - a current **state** (w^{r-1})
    - $w^r = g(w^{r-1}, K^r)$ is the next state
- **plaintext** x**:** the initial state w^0
- **ciphertext** y**:** the state after all Nr rounds have been performed

Introduction

• Encryption operations:

$$
\begin{aligned}
w^0 &\leftarrow x \\
w^1 &\leftarrow g(w^0, K^1) \\
w^2 &\leftarrow g(w^1, K^2) \\
&\vdots \\
w^{\mathrm{Nr}-1} &\leftarrow g(w^{\mathrm{Nr}-2}, K^{\mathrm{Nr}-1}) \\
w^{\mathrm{Nr}} &\leftarrow g(w^{\mathrm{Nr}-1}, K^{\mathrm{Nr}}) \\
y &\leftarrow w^{\mathrm{Nr}}
\end{aligned}
$$

• Decryption operations :

$$
\begin{aligned}
w^{\mathrm{Nr}} &\leftarrow y \\
w^{\mathrm{Nr}-1} &\leftarrow g^{-1}(w^{\mathrm{Nr}}, K^{\mathrm{Nr}}) \\
&\vdots \\
w^1 &\leftarrow g^{-1}(w^2, K^2) \\
w^0 &\leftarrow g^{-1}(w^1, K^1) \\
x &\leftarrow w^0
\end{aligned}
$$

In order for decryption to be possible, the function g must be injective (i.e., one-to-one).

- Introduction
- **Substitution-Permutation Networks**
- Linear Cryptanalysis
  - The Piling-up Lemma
  - Linear Approximations of S-boxes
  - A Linear Attack on an SPN
- Differential Cryptanalysis

Substitution-Permutation Networks (SPN)

We begin by defining a *substitution-permutation networks*, or *SPN*.

- An SPN is a special type of iterated cipher with a couple of small changes that we will indicate
- Suppose that ℓ and m are positive integers
- A plaintext and ciphertext will both be binary vectors of length ℓm (i.e., ℓm is the *block length* of the cipher)
- A SPN is built from two components, which are denoted π_S and π_P
  - $\pi_S : \{0,1\}^{\ell} \rightarrow \{0,1\}^{\ell}$ is a permutation and called *S-box* (the letter "S" denotes "substitution")
  - $\pi_P : \{1, \ldots, \ell m\} \rightarrow \{1, \ldots, \ell m\}$ is a permutation

Substitution-Permutation Networks (SPN)

Cryptosystem: Substitution-Permutation Networks

- ℓ, m and Nr are positive integers
- $\pi_S : \{0,1\}^{\ell} \rightarrow \{0,1\}^{\ell}$ is a permutation
- $\pi_P : \{1, \ldots, \ell m\} \rightarrow \{1, \ldots, \ell m\}$ is a permutation
- $\mathcal{P} = \mathcal{C} = \{0,1\}^{\ell m}$
- $\mathcal{K}(\{0,1\}^{\ell m})^{\mathrm{Nr}+1}$ consists of all possible key schedules that could be derived from an initial key K using the key scheduling algorithm
- For a key schedule $(K^1, \ldots, K^{\mathrm{Nr}+1})$, the plaintext x is encrypted by **Algorithm 1**

Algorithm 1

$\mathrm{SPN}(x, \pi_S, \pi_P, (K^1, \ldots, K^{\mathrm{Nr}+1}))$

$w^0 \leftarrow x$

for $r \leftarrow 1$ **to** $\mathrm{Nr} - 1$

do
- $u^r \leftarrow w^{r-1} \oplus K^r$
- **for** $i \leftarrow 1$ **to** m
  - **do** $v^r_{<i>} \leftarrow \pi_S(u^r_{<i>})$
- $w^r \leftarrow (v^r_{\pi_P(1)}, \ldots, v^r_{\pi_P(\ell m)})$

$u^{\mathrm{Nr}} \leftarrow w^{\mathrm{Nr}-1} \oplus K^{\mathrm{Nr}}$

for $i \leftarrow 1$ **to** m

do $v^{\mathrm{Nr}}_{<i>} \leftarrow \pi_S(u^{\mathrm{Nr}}_{<i>})$

$y \leftarrow v^{\mathrm{Nr}} \oplus K^{\mathrm{Nr}+1}$

output (y)

u^r is the input to the S-boxes in round r.

v^r is the output of the S-boxes in round r.

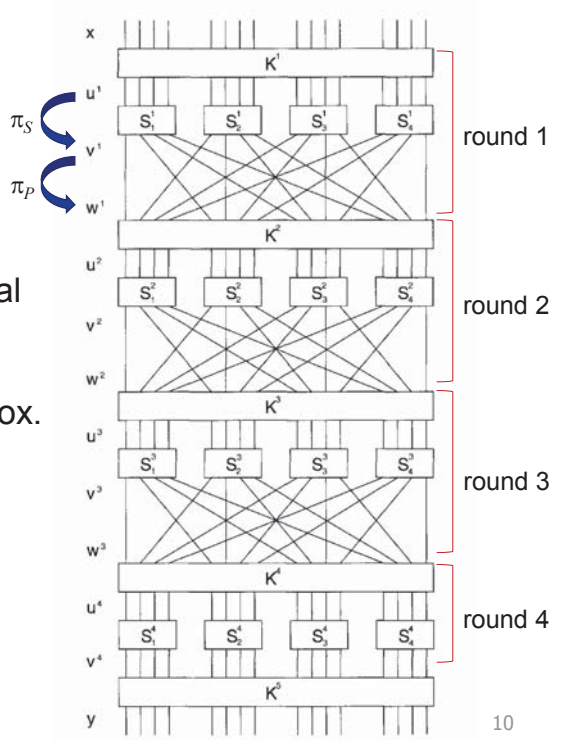
w^r is obtained from v^r by applying π_P.

$\Rightarrow u^{r+1}$ is constructed from w^r by xor-ing with the round key K^{r+1} (called round key mixing).

The very first and last operations are xors with subkeys (called whitening).

A Substitution-Permutation Network

See the figure for a pictorial representation of this particular SPN, where S^r_i means r-th round, i-th S-box.



Example

Suppose $\ell = m = \mathrm{Nr} = 4$. Let π_S be defined as follows, where the input and the output are written in hexadecimal:

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

Let π_P be defined as follows:

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

Substitution-Permutation Networks

In order to complete the description of SPN, we need to specify a key scheduling algorithm.

- Suppose we begin with a 32-bit key $K = (k_1, \ldots, k_{32}) \in \{0, 1\}^{32}$
- For $1 \le r \le 5$, define K^r to consist of 16 consecutive bits of K, beginning with k_{4r-3}
- Suppose the key is

  K = 0011 1010 1001 0100 1101 0110 0011 1111

  $\Rightarrow$ The round keys are as follows:

  K^1 = 0011 1010 1001 0100
  K^2 = 1010 1001 0100 1101
  K^3 = 1001 0100 1101 0110
  K^4 = 0100 1101 0110 0011
  K^5 = 1101 0110 0011 1111

Substitution-Permutation Networks

- Suppose the plaintext is x = 0010 0110 1011 0111
- Then the encryption of x proceeds as follows:

w^0 = 0010 0110 1011 0111
K^1 = 0011 1010 1001 0100
u^1 = 0001 1100 0010 0011
v^1 = 0100 0101 1101 0001
w^1 = 0010 1110 0000 0111
K^2 = 1010 1001 0100 1101
u^2 = 1000 0111 0100 1010
v^2 = 0011 1000 0010 0110
w^2 = 0100 0001 1011 1000

Substitution-Permutation Networks

K^3 = 1001 0100 1101 0110
u^3 = 1101 0101 0110 1110
v^3 = 1001 1111 1011 0000
w^3 = 1110 0100 0110 1110
K^4 = 0100 1101 0110 0011
u^4 = 1010 1001 0000 1101
v^4 = 0110 1010 1110 1001
K^5 = 1101 0110 0011 1111, and
y = 1011 1100 1101 0110 is the ciphertext

Linear Cryptanalysis

- Suppose that it is possible to find a probability linear relationship between a subset of plaintext bits and a subset of state bits preceding the substitution performed in the last round
  - There exists a subset of bits whose exclusive-or behaves in a non-random fashion
- Assume that an attacker has a large number of plaintext-ciphertext pairs, all of which are encrypted using the same unknown key K (i.e., we consider a **known-plaintext attack**)

Linear Cryptanalysis

- For each candidate subkey, we compute the values of the relevant state bits involved in the linear relationship, and determine if the above-mentioned linear relationship holds
- If the relation holds then we increment a counter corresponding to the particular candidate key
- At the end of the process, the candidate key that has a frequency count that is furthest from ½ times the number of pairs contains the correct values for these key bits

The Piling-up Lemma

- Suppose that X_1, X_2, … are independent random variables taking on values from the set {0, 1}
- p_1, p_2, … are real numbers such that $0 \le p_i \le 1$, $\forall i$
- Suppose that $\Pr[X_i = 0] = p_i$, $\forall i = 1, 2, \ldots$
- Hence $\Pr[X_i = 1] = 1 - p_i$, $\forall i = 1, 2, \ldots$
- The independence of X_i and X_j, $\forall i \neq j$ implies that
  - $\Pr[X_i = 0, X_j = 0] = p_i p_j$
  - $\Pr[X_i = 0, X_j = 1] = p_i (1 - p_j)$
  - $\Pr[X_i = 1, X_j = 0] = (1 - p_i) p_j$
  - $\Pr[X_i = 1, X_j = 1] = (1 - p_i)(1 - p_j)$

The Piling-up Lemma

- Consider the discrete random variable $X_i \oplus X_j$ (this is the same as $X_i + X_j$ mod 2), which has the following probability distribution
  - $\Pr[X_i \oplus X_j = 0] = p_i p_j + (1 - p_i)(1 - p_j)$
  - $\Pr[X_i \oplus X_j = 1] = p_i (1 - p_j) + (1 - p_i) p_j$
- The **bias** of X_i is defined as the quantity

$$\varepsilon_i = p_i - ½ , \forall i = 1, 2, \ldots$$

$$\Rightarrow -½ \le \varepsilon_i \le ½ , \forall i = 1, 2, \ldots$$

$$\Pr[X_i = 0] = ½ + \varepsilon_i, \forall i = 1, 2, \ldots$$

$$\Pr[X_i = 1] = ½ - \varepsilon_i, \forall i = 1, 2, \ldots$$

Lemma (Pilling-up Lemma)

Let $\varepsilon_{i_1, i_2, \ldots, i_k}$ denote the bias of the random variable $X_{i_1} \oplus \cdots \oplus X_{i_k}$.

Then $\varepsilon_{i_1, i_2, \ldots, i_k} = 2^{k-1} \prod_{j=1}^{k} \varepsilon_{i_j}$.

Proof

By induction on k,

$$k = 1,\ \varepsilon_{i_1} = 2^{1-1} \prod_{j=1}^{1} \varepsilon_{i_j} = \varepsilon_{i_1}$$

$k = 2$, determine the bias of $X_{i_1} \oplus X_{i_2}$

$$\Rightarrow \Pr[X_{i_1} \oplus X_{i_2} = 0] = p_{i_1} p_{i_2} + (1 - p_{i_1})(1 - p_{i_2})$$

$$= (\frac{1}{2} + \varepsilon_{i_1})(\frac{1}{2} + \varepsilon_{i_2}) + (\frac{1}{2} - \varepsilon_{i_1})(\frac{1}{2} - \varepsilon_{i_2}) = \frac{1}{2} + 2\varepsilon_{i_1} \varepsilon_{i_2}$$

Hence, the bias of $X_{i_1} \oplus X_{i_2}$ is $2\varepsilon_{i_1} \varepsilon_{i_2}$ $\Rightarrow \varepsilon_{i_1, i_2} = 2\varepsilon_{i_1} \varepsilon_{i_2} = 2^{2-1} \prod_{j=1}^{2} \varepsilon_{i_j}$

As an induction hypothesis, assume that the result is true for $k = l$, for some positive integer $l \geq 2$.

We will prove that the formula is true for $k = l+1$.

We want to determine the bias of $X_{i_1} \oplus \cdots \oplus X_{i_{l+1}}$.

We spilt this random variable $X_{i_1} \oplus \cdots \oplus X_{i_{l+1}}$ into two parts, as follows:

$$X_{i_1} \oplus \cdots \oplus X_{i_{l+1}} = (X_{i_1} \oplus \cdots \oplus X_{i_l}) \oplus X_{i_{l+1}}$$

The bias of $X_{i_1} \oplus \cdots \oplus X_{i_l}$ is $2^{l-1} \prod_{j=1}^{l} \varepsilon_{i_j}$ and the bias of $X_{i_{l+1}}$ is $\varepsilon_{i_{l+1}}$.

By induction, the bias of $X_{i_1} \oplus \cdots \oplus X_{i_{l+1}}$ is

$$\varepsilon_{i_1,i_2,\cdots,i_l,i_{l+1}} = 2\varepsilon_{i_1,i_2,\cdots,i_l}\varepsilon_{i_{l+1}} = 2 \times (2^{l-1} \prod_{j=1}^{l} \varepsilon_{i_j}) \times \varepsilon_{i_{l+1}} = 2^l \prod_{j=1}^{l+1} \varepsilon_{i_j}$$

The proof is complete.

Corollary

Denote the bias of the random variable $X_{i_1} \oplus \cdots \oplus X_{i_k}$ as $\varepsilon_{i_1,i_2,\ldots,i_k}$. Suppose $\varepsilon_{i_j} = 0$ for some j. Then $\varepsilon_{i_1,i_2,\ldots,i_k} = 0$.

Example

Suppose that $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 1/4$.

Applying Pilling-up Lemma, we see that

$$\varepsilon_{1,2} = \varepsilon_{2,3} = \varepsilon_{1,3} = 2 \times 1/4 \times 1/4 = 1/8.$$

Consider the random variable $X_1 \oplus X_3$.

$\Rightarrow X_1 \oplus X_3 = (X_1 \oplus X_2) \oplus (X_2 \oplus X_3)$

If the two random variables $X_1 \oplus X_2$ and $X_2 \oplus X_3$ were independent, then Pilling-up Lemma would say that

$$\varepsilon_{1,3} = 2\varepsilon_{1,2}\varepsilon_{2,3} = 2 \times (1/8) \times (1/8) = 1/32$$

$\Rightarrow X_1 \oplus X_2$ and $X_2 \oplus X_3$ are not independent.

- Introduction
- Substitution-Permutation Networks
- **Linear Cryptanalysis**
  - The Piling-up Lemma
  - **Linear Approximations of S-boxes**
  - A Linear Attack on an SPN
- Differential Cryptanalysis

Linear Approximations of S-boxes

- Consider an S-box $\pi_S : \{0, 1\}^m \rightarrow \{0, 1\}^n$
- Let the input m-tuple be $X = (x_1, \ldots, x_m)$
  - This m-tuple is chosen uniformly at random from $\{0, 1\}^m$, which means that each coordinate x_i defines a random variable X_i taking on values 0 and 1, having bias $\varepsilon_i = 0$
  - These m random variables are independent
- Let the output n-tuple be $Y = (y_1, \ldots, y_n)$
  - Each coordinate y_j defines a random variable Y_j taking on values 0 and 1
  - These n random variables are not independent from each other or from the X_i's

Linear Approximations of S-boxes

- We can see that the following formula holds:
  - $\Pr[X_1 = x_1, \ldots, X_m = x_m, Y_1 = y_1, \ldots, Y_n = y_n] = 0$ if $(y_1, \ldots, y_n) \neq \pi_S(x_1, \ldots, x_m)$
  - $\Pr[X_1 = x_1, \ldots, X_m = x_m, Y_1 = y_1, \ldots, Y_n = y_n] = 2^{-m}$ if $(y_1, \ldots, y_n) = \pi_S(x_1, \ldots, x_m)$
    - $\because \Pr[X_1 = x_1, \ldots, X_m = x_m] = 2^{-m}$ and $\Pr[Y_1 = y_1, \ldots, Y_n = y_n] = 1$ if $(y_1, \ldots, y_n) = \pi_S(x_1, \ldots, x_m)$
- Now we can compute the bias of the form

$$X_{i_1} \oplus \cdots \oplus X_{i_k} \oplus Y_{j_1} \oplus \cdots \oplus Y_{j_l}$$

using the formulas stated above

Example

Consider an S-box $\pi_S : \{0, 1\}^4 \rightarrow \{0, 1\}^4$.

We record the possible values taken on by the eight random variables $X_1, \ldots, X_4, Y_1, \ldots, Y_4$ in the rows of the following table:

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4
0	0	0	0	1	1	1	0
0	0	0	1	0	1	0	0
0	0	1	0	1	1	0	1
0	0	1	1	0	0	0	1
0	1	0	0	0	0	1	0
0	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1
0	1	1	1	1	0	0	0
1	0	0	0	0	0	1	1
1	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0
1	0	1	1	1	1	0	0
1	1	0	0	0	1	0	1
1	1	0	1	1	0	0	1
1	1	1	0	0	0	0	0
1	1	1	1	0	1	1	1

Consider the random variable $X_1 \oplus X_4 \oplus Y_2$.

The probability that $X_1 \oplus X_4 \oplus Y_2 = 0$ can be determined by counting the number of rows in which $X_1 \oplus X_4 \oplus Y_2 = 0$, and then dividing by 16.

$\Rightarrow \Pr[X_1 \oplus X_4 \oplus Y_2 = 0] = \frac{8}{16} = \frac{1}{2}$

$\Rightarrow$ The bias of $X_1 \oplus X_4 \oplus Y_2$ is 0.

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_1 \oplus X_4 \oplus Y_2$
0	0	0	0	1	1	1	0	0 + 0 + 1 = 1
0	0	0	1	0	1	0	0	0 + 1 + 1 = 0
0	0	1	0	1	1	0	1	0 + 0 + 1 = 1
0	0	1	1	0	0	0	1	0 + 1 + 0 = 1
0	1	0	0	0	0	1	0	0 + 0 + 0 = 0
0	1	0	1	1	1	1	1	0 + 1 + 1 = 0
0	1	1	0	1	0	1	1	0 + 0 + 0 = 0
0	1	1	1	1	0	0	0	0 + 1 + 0 = 1
1	0	0	0	0	0	1	1	1 + 0 + 0 = 1
1	0	0	1	1	0	1	0	1 + 1 + 0 = 0
1	0	1	0	0	1	1	0	1 + 0 + 1 = 0
1	0	1	1	1	1	0	0	1 + 1 + 1 = 1
1	1	0	0	0	1	0	1	1 + 0 + 1 = 0
1	1	0	1	1	0	0	1	1 + 1 + 0 = 0
1	1	1	0	0	0	0	0	1 + 0 + 0 = 1
1	1	1	1	0	1	1	1	1 + 1 + 1 = 1

Analyze the random variable $X_3 \oplus X_4 \oplus Y_1 \oplus Y_4$ instead.

The probability that $X_3 \oplus X_4 \oplus Y_1 \oplus Y_4 = 0$ can be determined by counting the number of rows in which $X_3 \oplus X_4 \oplus Y_1 \oplus Y_4 = 0$, and then dividing by 16.

$\Rightarrow \Pr[X_3 \oplus X_4 \oplus Y_1 \oplus Y_4 = 0] = \frac{2}{16} = \frac{1}{8}$

$\Rightarrow$ The bias of $X_3 \oplus X_4 \oplus Y_1 \oplus Y_4$ is $\frac{1}{8} - \frac{1}{2} = \frac{-3}{8}$.

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_3 \oplus X_4 \oplus Y_1 \oplus Y_4$
0	0	0	0	1	1	1	0	0 + 0 + 1 + 0 = 1
0	0	0	1	0	1	0	0	0 + 1 + 0 + 0 = 1
0	0	1	0	1	1	0	1	1 + 0 + 1 + 1 = 1
0	0	1	1	0	0	0	1	1 + 1 + 0 + 1 = 1
0	1	0	0	0	0	1	0	0 + 0 + 0 + 0 = 0
0	1	0	1	1	1	1	1	0 + 1 + 1 + 1 = 1
0	1	1	0	1	0	1	1	1 + 0 + 1 + 1 = 1
0	1	1	1	1	0	0	0	1 + 1 + 1 + 0 = 1
1	0	0	0	0	0	1	1	0 + 0 + 0 + 1 = 1
1	0	0	1	1	0	1	0	0 + 1 + 1 + 0 = 0
1	0	1	0	0	1	1	0	1 + 0 + 0 + 0 = 1
1	0	1	1	1	1	0	0	1 + 1 + 1 + 0 = 1
1	1	0	0	0	1	0	1	0 + 0 + 0 + 1 = 1
1	1	0	1	1	0	0	1	0 + 1 + 1 + 1 = 1
1	1	1	0	0	0	0	0	1 + 0 + 0 + 0 = 1
1	1	1	1	0	1	1	1	1 + 1 + 0 + 1 = 1

Linear Cryptanalysis

- Represent the relevant random variable in the form

$$\left(\bigoplus_{i=1}^{4} a_i \mathrm{X}_i\right) \oplus \left(\bigoplus_{i=1}^{4} b_i \mathrm{Y}_i\right)$$

where $a_i \in \{0, 1\}$, $b_i \in \{0, 1\}$, $i = 1, 2, 3, 4$

- Treat each of the binary vectors (a_1, a_2, a_3, a_4) and (b_1, b_2, b_3, b_4) as a hexadecimal digit (they are called the **input sum** and **output sum**, respectively)
  - Example: Consider the random variable $\mathrm{X}_1 \oplus \mathrm{X}_4 \oplus \mathrm{Y}_2$

    The input sum is (1, 0, 0, 1), which is 9 in hexadecimal.

$$1\times 2^3 + 0\times 2^2 + 0\times 2^1 + 1\times 2^0 = 9$$

    The output sum is (0, 1, 0, 0), which is 4 in hexadecimal.

$$0\times 2^3 + 1\times 2^2 + 0\times 2^1 + 0\times 2^0 = 4$$

Linear Cryptanalysis

- For a random variable having (hexadecimal) input sum a and output sum b (where $a = (a_1, a_2, a_3, a_4)$ and $b = (b_1, b_2, b_3, b_4)$, in binary), let $N_L(a, b)$ denote **the number of binary eight-tuples** $(x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4)$ such that

$$(y_1, y_2, y_3, y_4) = \pi_S(x_1, x_2, x_3, x_4)$$

and $$\left(\bigoplus_{i=1}^{4} a_i \mathrm{X}_i\right) \oplus \left(\bigoplus_{i=1}^{4} b_i \mathrm{Y}_i\right) = 0$$

- The bias of the random variable having input sum a and output sum b is computed as

$$\varepsilon(a,b) = \frac{N_L(a,b) - 8}{16}$$

Example

Consider the random variable $\mathrm{X}_1 \oplus \mathrm{X}_4 \oplus \mathrm{Y}_2$.

The input sum is a = (1, 0, 0, 1), which is 9 in hexadecimal.

The output sum is b = (0, 1, 0, 0), which is 4 in hexadecimal.

Since the number of rows in which $\mathrm{X}_1 \oplus \mathrm{X}_4 \oplus \mathrm{Y}_2 = 0$ is 8.

$\Rightarrow N_L(a, b) = N_L((1, 0, 0, 1), (0, 1, 0, 0)) = N_L(9, 4) = 8$

$\Rightarrow$ The bias of the random variable having input sum a and output sum b is

$$\varepsilon(9,4) = \frac{N_L(9,4) - 8}{16} = \frac{8-8}{16} = 0$$

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$\mathrm{X}_1 \oplus \mathrm{X}_4 \oplus \mathrm{Y}_2$
0	0	0	0	1	1	1	0	0 + 0 + 1 = 1
0	0	0	1	0	1	0	0	0 + 1 + 1 = 0
0	0	1	0	1	1	0	1	0 + 0 + 1 = 1
0	0	1	1	0	0	0	1	0 + 1 + 0 = 1
0	1	0	0	0	0	1	0	0 + 0 + 0 = 0
0	1	0	1	1	1	1	1	0 + 1 + 1 = 0
0	1	1	0	1	0	1	1	0 + 0 + 0 = 0
0	1	1	1	1	0	0	0	0 + 1 + 0 = 1
1	0	0	0	0	0	1	1	1 + 0 + 0 = 1
1	0	0	1	1	0	1	0	1 + 1 + 0 = 0
1	0	1	0	0	1	1	0	1 + 0 + 1 = 0
1	0	1	1	1	1	0	0	1 + 1 + 1 = 1
1	1	0	0	0	1	0	1	1 + 0 + 1 = 0
1	1	0	1	1	0	0	1	1 + 1 + 0 = 0
1	1	1	0	0	0	0	0	1 + 0 + 0 = 1
1	1	1	1	0	1	1	1	1 + 1 + 1 = 1

a \ b	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
1	8	8	6	6	8	8	6	14	10	10	8	8	10	10	8	8
2	8	8	6	6	8	8	6	6	8	8	10	10	8	8	2	10
3	8	8	8	8	8	8	8	8	10	2	6	6	10	10	6	6
4	8	10	8	6	6	4	6	8	8	6	8	10	10	4	10	8
5	8	6	6	8	6	8	12	10	6	8	4	10	8	6	6	8
6	8	10	6	12	10	8	8	10	8	6	10	12	6	8	8	6
7	8	6	8	10	10	4	10	8	6	8	10	8	12	10	8	10
8	8	8	8	8	8	8	8	8	6	10	10	6	10	6	6	2
9	8	8	6	6	8	8	6	6	4	8	6	10	8	12	10	6
A	8	12	6	10	4	8	10	6	10	10	8	8	10	10	8	8
B	8	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8
C	8	6	12	6	6	8	10	8	10	8	10	12	8	10	8	6
D	8	10	10	8	6	12	8	10	4	6	10	8	10	8	8	10
E	8	10	10	8	6	4	8	10	6	8	8	6	4	10	6	8
F	8	6	4	6	6	8	10	8	8	6	12	6	6	8	10	8

Linear Approximation Table:
Values of $N_L(a, b)$

$N_L(9, 4) = 8$

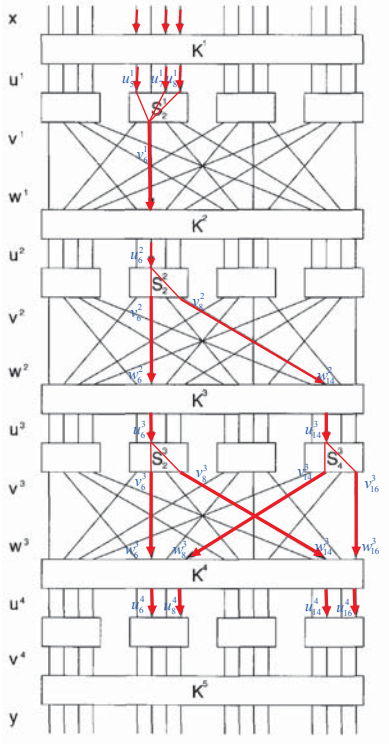
- Introduction
- Substitution-Permutation Networks
- **Linear Cryptanalysis**
  - The Piling-up Lemma
  - Linear Approximations of S-boxes
  - **A Linear Attack on an SPN**
- Differential Cryptanalysis

A Linear Attack on an SPN

- Linear cryptanalysis requires finding a set of linear approximations of S-boxes that can be used to derive a linear approximation of the entire SPN (excluding the last round)

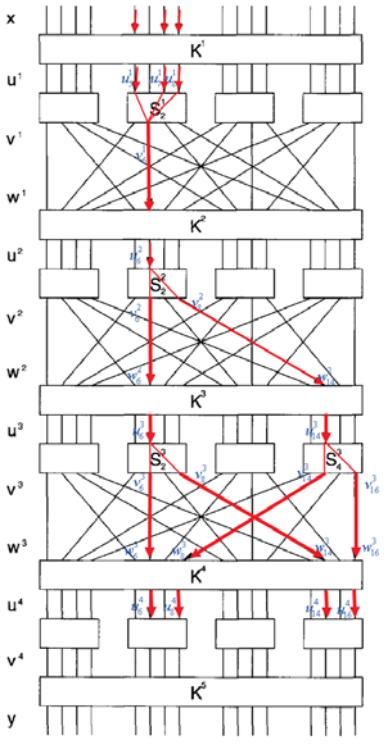
A Linear Approximation of a Substitution-Permutation Network

- The Figure illustrates the structure of the approximation we will use
- Lines with arrows correspond to the random variables which will be involved in linear approximations and the labeled S-boxes (**active S-boxes**) are used in the approximations



A Linear Attack on an SPN

- The approximation incorporates four active S-boxes:
  - In $S_2^1, T_1 = U_5^1 \oplus U_7^1 \oplus U_8^1 \oplus V_6^1$ has bias ¼
  - In $S_2^2, T_2 = U_6^2 \oplus V_6^2 \oplus V_8^2$ has bias –¼
  - In $S_2^3, T_3 = U_6^3 \oplus V_6^3 \oplus V_8^3$ has bias –¼
  - In $S_4^3, T_4 = U_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3$ has bias –¼



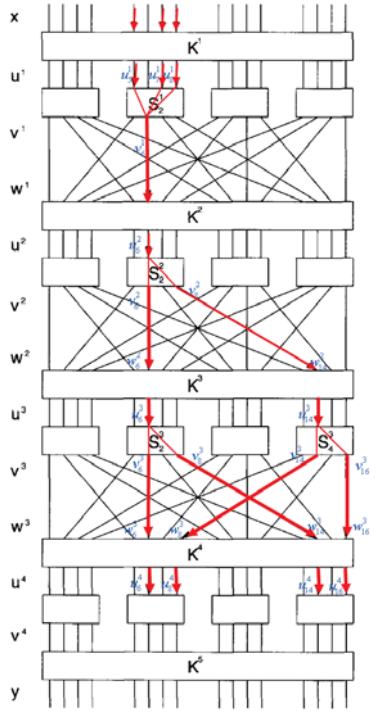
A Linear Attack on an SPN

- The four random variables T_1, T_2, T_3, T_4 have biases that are high in absolute value
- Further, we will see their XOR will lead to **cancellations of "intermediate"** random variables
- Assume that these four random variables T_1, T_2, T_3, T_4 are independent, by pilling-up lemma, the bias of $T_1 \oplus T_2 \oplus T_3 \oplus T_4$ is

$$2^3 \times \frac{1}{4} \times (-\frac{1}{4})^3 = -\frac{1}{32}$$

A Linear Attack on an SPN

- The random variables T_1, T_2, T_3 and T_4 have the property that their XOR can be expressed in terms of plaintext bits, bits of u^4 and key bits as follows:
  - $T_1 = U_5^1 \oplus U_7^1 \oplus U_8^1 \oplus V_6^1$
    $= X_5 \oplus K_5^1 \oplus X_7 \oplus K_7^1 \oplus X_8 \oplus K_8^1 \oplus V_6^1$
  - $T_2 = U_6^2 \oplus V_6^2 \oplus V_8^2$
    $= V_6^1 \oplus K_6^2 \oplus V_6^2 \oplus V_8^2$
  - $T_3 = U_6^3 \oplus V_6^3 \oplus V_8^3$
    $= V_6^2 \oplus K_6^3 \oplus V_6^3 \oplus V_8^3$
  - $T_4 = U_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3$
    $= V_8^2 \oplus K_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3$



A Linear Attack on an SPN

- Then

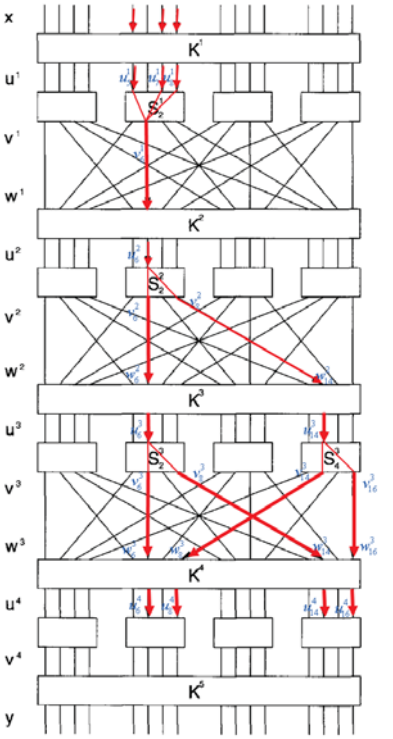
$$T_1 \oplus T_2 \oplus T_3 \oplus T_4$$
$$= (X_5 \oplus K_5^1 \oplus X_7 \oplus K_7^1 \oplus X_8 \oplus K_8^1 \oplus V_6^1) \oplus (V_6^1 \oplus K_6^2 \oplus V_6^2 \oplus V_8^2)$$
$$\oplus (V_6^2 \oplus K_6^3 \oplus V_6^3 \oplus V_8^3) \oplus (V_8^2 \oplus K_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3)$$
$$= X_5 \oplus X_7 \oplus X_8 \oplus V_6^3 \oplus V_8^3 \oplus V_{14}^3 \oplus V_{16}^3$$
$$\oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3$$

has bias equal to –1/32

A Linear Attack on an SPN

- Then replace the terms V_i^3 by U_i^4 and key bits:

$$V_6^3 = U_6^4 \oplus K_6^4$$
$$V_8^3 = U_{14}^4 \oplus K_{14}^4$$
$$V_{14}^3 = U_8^4 \oplus K_8^4$$
$$V_{16}^3 = U_{16}^4 \oplus K_{16}^4$$



A Linear Attack on an SPN

- Then

$$T_1 \oplus T_2 \oplus T_3 \oplus T_4$$
$$= X_5 \oplus X_7 \oplus X_8 \oplus V_6^3 \oplus V_8^3 \oplus V_{14}^3 \oplus V_{16}^3$$
$$\oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3$$
$$= X_5 \oplus X_7 \oplus X_8 \oplus (U_6^4 \oplus K_6^4) \oplus (U_{14}^4 \oplus K_{14}^4)$$
$$\oplus (U_8^4 \oplus K_8^4) \oplus (U_{16}^4 \oplus K_{16}^4)$$
$$\oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3$$
$$= X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_{14}^4 \oplus U_8^4 \oplus U_{16}^4$$
$$\oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3 \oplus K_6^4 \oplus K_8^4 \oplus K_{14}^4 \oplus K_{16}^4$$

bits of u^4

plaintext bits

key bits

A Linear Attack on an SPN

- Suppose that the key bits

$$K_5^1, K_7^1, K_8^1, K_6^2, K_6^3, K_{14}^3, K_6^4, K_8^4, K_{14}^4 \text{ and } K_{16}^4$$

are fixed, then

$$K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3 \oplus K_6^4 \oplus K_8^4 \oplus K_{14}^4 \oplus K_{16}^4$$

has the (fixed) value 0 or 1

- It follows that the random variable

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4$$

has bias equal to $\pm 1/32$, where the sign of this bias depends on the values of unknown key bits

- Note that the random variable

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4$$

involves only plaintext bits and bits of u^4

A Linear Attack on an SPN

- The random variable

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4$$

has bias bounded away from 0 allows us to carry out linear attack

- Suppose that we have T plaintext-ciphertext pairs (denoted by $\mathcal{T}$), all use the same unknown key, K
- The attack will allow us to obtain the eight key bits in $K_{<2>}^5$ and $K_{<4>}^5$, namely,

$$K_5^5, K_6^5, K_7^5, K_8^5, K_{13}^5, K_{14}^5, K_{15}^5, \text{ and } K_{16}^5$$

  - These are the eight key bits that are XOR with the output of the S-boxes S_2^4 and S_4^4

A Linear Attack on an SPN

- Notice that there are $2^8 = 256$ possibilities for the eight key bits
- We refer to a binary 8-tuple as a **candidate subkey**
- For each $(x, y) \in \mathcal{T}$ and for each candidate subkey, we compute a partial decryption of y and obtain the resulting value for $u_{<2>}^4$ and $u_{<4>}^4$
- Then we compute the value

$$x_5 \oplus x_7 \oplus x_8 \oplus u_6^4 \oplus u_8^4 \oplus u_{14}^4 \oplus u_{16}^4$$

A Linear Attack on an SPN

- We maintain an array of counters indexed by the 256 possible candidate subkeys, and increment the counter corresponding to a particular subkey whenever

$$x_5 \oplus x_7 \oplus x_8 \oplus u_6^4 \oplus u_{14}^4 \oplus u_8^4 \oplus u_{16}^4$$

  has the value 0
- At the end of this counting process, we expect most counters will have a value close to $T/2$, but the counter for the correct candidate subkey will have the value that is close to $T/2 \pm T/32$

Algorithm 2

- The algorithm for this particular attack is present as follows:
  - L_1 and L_2 are hexadecimal value
  - π_S^{-1} is the inverse of the S-box
  - The output, maxkey, contains the most likely subkey
- In general, it is suggested that a linear attack based on a linear approximation having bias ε will be successful if the number of plaintext-ciphertext pairs is approximately $c\varepsilon^{-2}$ for some "small" constant c

for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do $Count[L_1, L_2] \leftarrow 0$
for each $(x, y) \in \mathcal{T}$
  do { **for** $(L_1, L_2) \leftarrow (0,0)$ to (F, F)
    do { $v_{(2)}^4 \leftarrow L_1 \oplus y_{<2>}$
      $v_{(4)}^4 \leftarrow L_2 \oplus y_{<4>}$
      $u_{(2)}^4 \leftarrow \pi_S^{-1}(v_{<2>}^4)$
      $u_{(4)}^4 \leftarrow \pi_S^{-1}(v_{<4>}^4)$
      $z \leftarrow x_5 \oplus x_7 \oplus x_8 \oplus u_6^4 \oplus u_8^4 \oplus u_{14}^4 \oplus u_{16}^4$
      if $z = 0$
        then $Count[L_1, L_2] \leftarrow Count[L_1, L_2] + 1$
$max \leftarrow -1$
for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do { $Count[L_1, L_2] \leftarrow |Count[L_1, L_2] - T/2|$
    if $Count[L_1, L_2] > max$
      then { $max \leftarrow Count[L_1, L_2]$
        $maxkey \leftarrow (L_1, L_2)$
output $(maxkey)$

A Linear Attack on an SPN

- As can be seen from the partial results in the table, the largest bias occurs for partial subkey value $[K_{5,5} \ldots K_{5,8}, K_{5,13} \ldots K_{5,16}] = [2, 4]$
- The experimentally determined bias value of 0.0336 is very close to the expected value of 1/32 = 0.03125

partial subkey $[K_{5,5}\ldots K_{5,8}, K_{5,13}\ldots K_{5,16}]$	| bias |	*partial subkey* $[K_{5,5}\ldots K_{5,8}, K_{5,13}\ldots K_{5,16}]$	| bias |
1 C	0.0031	2 A	0.0044
1 D	0.0078	2 B	0.0186
1 E	0.0071	2 C	0.0094
1 F	0.0170	2 D	0.0053
2 0	0.0025	2 E	0.0062
2 1	0.0220	2 F	0.0133
2 2	0.0211	3 0	0.0027
2 3	0.0064	3 1	0.0050
2 4	**0.0336**	3 2	0.0075
2 5	0.0106	3 3	0.0162
2 6	0.0096	3 4	0.0218
2 7	0.0074	3 5	0.0052
2 8	0.0224	3 6	0.0056
2 9	0.0054	3 7	0.0048

Experimental Results for Linear Attack with 10000 known plaintext-ciphertext values

- Introduction
- Substitution-Permutation Networks
- Linear Cryptanalysis
  - The Piling-up Lemma
  - Linear Approximations of S-boxes
  - A Linear Attack on an SPN
- **Differential Cryptanalysis**

Differential Cryptanalysis

- The main difference from linear cryptanalysis is that differential cryptanalysis involves comparing the XOR of two inputs to the XOR of the corresponding two outputs
- Differential attack is a **chosen-plaintext attack**
- We consider inputs x and x* having a specified XOR value denoted by $x' = x \oplus x^*$
  - We decrypt y and y* using all possible keys and determine if their XOR has a certain value
  - Whenever it does, we increment a counter corresponding to the particular candidate key
- At the end of this process, we hope that the candidate key that has the ***highest*** frequency count contains the correct values for these key bits

Definition

Let $\pi_S : \{0, 1\}^m \rightarrow \{0, 1\}^n$ be an S-box.

Consider an (ordered) pair of bitstrings of length m, say (x, x*). We say that the ***input XOR*** of the S-box is $x \oplus x^*$ and the ***output XOR*** is $\pi_S(x) \oplus \pi_S(x^*)$.

Note that the output XOR is a bitstring of length n.

For any $x' \in \{0, 1\}^m$, define the set $\Delta(x')$ to consist of all the ordered pairs (x, x*) having input XOR equal to x'.

Differential Cryptanalysis

- Apparently any set $\Delta(x')$ contains 2^m pairs, and

$$\Delta(x') = \{(x,\ x \oplus x') : x \in \{0, 1\}^m\}$$

- For each pair in $\Delta(x')$, we can compute the output XOR of the S-box
- Then we can tabulate the resulting distribution of output XORs
- There are 2^m output XORs, which are distributed among 2^n possible values
  - A non-uniform output distribution will be the basis for a successful differential attack

Example

We use the same S-box as before.

Suppose we consider input XOR x' = 1011.

Then

$\Delta(x')$ = Δ(1011)
= {(0000, 1011),
(0001, 1010),
…,
(1111, 0100)}

x	x*
0000	1011
0001	1010
0010	1001
0011	1000
0100	1111
0101	1110
0110	1101
0111	1100
1000	0011
1001	0010
1010	0001
1011	0000
1100	0111
1101	0110
1110	0101
1111	0100

X_1	X_2	X_3	X_4
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

For each ordered pair in the set $\Delta(1011)$, we compute output XORs of π_S. In each row of the following table, we have $x \oplus x^* = 1011$

$y = \pi_S(x)$

$y^* = \pi_S(x^*)$

$y' = y \oplus y^*$

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

Number of output

x	x^*	y	y^*	y'
0000	1011	1110	1100	0010
0001	1010	0100	0110	0010
0010	1001	1101	1010	0111
0011	1000	0001	0011	0010
0100	1111	0010	0111	0101
0101	1110	1111	0000	1111
0110	1101	1011	1001	0010
0111	1100	1000	0101	1101
1000	0011	0011	0001	0010
1001	0010	1010	1101	0111
1010	0001	0110	0100	0010
1011	0000	1100	1110	0010
1100	0111	0101	1000	1101
1101	0110	1001	1011	0010
1110	0101	0000	1111	1111
1111	0100	0111	0010	0101

Distribution table for $x' = 1011$

Here, only 5 of 16 possible output XORs actually occur.

This particular example has a very non-uniform distribution.

Such computation can be carried out for any possible input XORs.

For a bitstring x' of length m and a bitstring y' of length n, define

$N_D(x', y') = |\{(x, x^*) \in \Delta(x') : \pi_S(x) \oplus \pi_S(x^*) = y'\}|$

i.e., $N_D(x', y')$ counts the number of pairs with input XOR equal to x' which also have output XOR equal to y'.

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

$N_D((1011), (0010)) = 8$

b' = **0010**, which is 2 in hexadecimal

a' \ b'	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

this example

Difference Distribution Table: $N_D(a', b')$

a' = **1011**, which is 11 = B in hexadecimal

$N_D((1011), (0010)) = 8$

Differential Cryptanalysis

- Recall that the input to the ith S-box in round r of the SPN is denoted by $u^r_{<i>}$, and

$$u^r_{<i>} = w^{r-1}_{<i>} \oplus K^r_{<i>}$$

- An input XOR is computed as

$$u^r_{<i>} \oplus (u^r_{<i>})^* = (w^{r-1}_{<i>} \oplus K^r_{<i>}) \oplus ((w^{r-1}_{<i>})^* \oplus K^r_{<i>})$$

$$= w^{r-1}_{<i>} \oplus (w^{r-1}_{<i>})^* = (w^{r-1}_{<i>})'$$

- Therefore, the input XOR does not depend on the subkey bits used in round r; it is equal to the (permuted) output XOR of round $r-1$

Differential Cryptanalysis

- Let a' denote an input XOR and let b' denote an output XOR
- The pair (a', b') is called a **differential**
- Each entry in the difference distribution table gives rise to an **XOR propagation ratio** (or simply **propagation ratio**) for the corresponding differential
- The propagation ratio $R_P(a', b')$ for the differential (a', b') is denoted as follows:

$$R_P(a', b') = \frac{N_D(a', b')}{2^m}$$

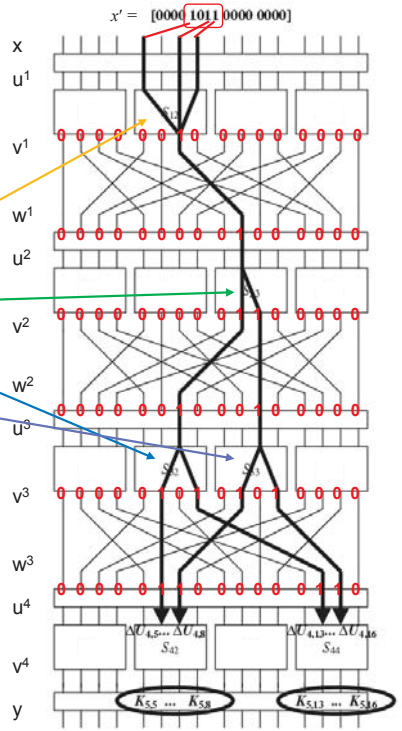
  - $R_P(a', b')$ can be interpreted as a conditional probability:
    Pr[output XOR = b' | input XOR = a'] = $R_P(a', b')$

Differential Cryptanalysis

- Suppose we find propagation ratios for differentials in consecutive rounds of the SPN, such that the input XOR of a differential in any round is the same as the (permuted) output XORs of the differentials in the previous round
- Then these differentials can be combined to form a **differential trail**
- Assume that the various propagation ratios in a differential trail are independent
- This assumption allows us to multiply the propagation ratios of the differentials in order to obtain the propagation ratio of the differential trail

Example

- Consider a differential characteristic involving S_{12}, S_{23}, S_{32}, and S_{33}
- S_{12} : a' = 1011 $\rightarrow$ b' = 0010
- S_{23} : a' = 0100 $\rightarrow$ b' = 0110
- S_{32} : a' = 0010 $\rightarrow$ b' = 0101
- S_{33} : a' = 0010 $\rightarrow$ b' = 0101



- S_{12} : a' = $\underbrace{1011}_{B}$ $\rightarrow$ b' = $\underbrace{0010}_{2}$

  $\Rightarrow R_P(B, 2) = 8/2^4 = 8/16 = 1/2$

- S_{23} : a' = $\underbrace{0100}_{4}$ $\rightarrow$ b' = $\underbrace{0110}_{6}$

  $\Rightarrow R_P(4, 6) = 6/2^4 = 6/16 = 3/8$

- S_{32} and S_{33} : a' = $\underbrace{0010}_{2}$ $\rightarrow$ b' = $\underbrace{0101}_{5}$

  $\Rightarrow R_P(2, 5) = 6/2^4 = 6/16 = 3/8$

- These differentials can be combined to form a differential trail

a' \ b'	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	0	2	2	2	0	2	0	0	2	2	0	0	0	4
8	0	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Difference Distribution Table: $N_D(a', b')$

- Therefore we obtain a propagation ratio for a differential trail of the first three rounds of the SPN:

$$R_P(0000\ 1011\ 0000\ 0000, 0000\ 0101\ 0101\ 0000)$$
$$= \frac{1}{2} \times (\frac{3}{8})^3 = \frac{27}{1024}$$

- In other words,
  x' = 0000 1011 0000 0000 $\Rightarrow$ $(v^3)'$ = 0000 0101 0101 0000 with probability 27/1024
- However,
  $(v^3)'$ = 0000 0101 0101 0000 $\Leftrightarrow$ $(u^4)'$ = 0000 0110 0000 0110
- Hence, it follows that
  x' = 0000 1011 0000 0000 $\Leftrightarrow$ $(u^4)'$ = 0000 0110 0000 0110 with probability 27/1024
- Note that $(u^4)'$ is the XOR of two inputs to the last round of S-boxes

Algorithm 3

for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do $Count[L_1, L_2] \leftarrow 0$
for each $(x, y, x^*, y^*) \in \mathcal{T}$
  do **if** $(y_{<1>} = (y_{<1>})^*)$ **and** $(y_{<3>} = (y_{<3>})^*)$
    then **for** $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
      do $v^4_{<2>} \leftarrow L_1 \oplus y_{<2>}$
      $v^4_{<4>} \leftarrow L_2 \oplus y_{<4>}$
      $u^4_{<2>} \leftarrow \pi_S^{-1}(v^4_{<2>})$
      $u^4_{<4>} \leftarrow \pi_S^{-1}(v^4_{<4>})$
      $(v^4_{<2>})^* \leftarrow L_1 \oplus (y_{<2>})^*$
      $(v^4_{<4>})^* \leftarrow L_2 \oplus (y_{<4>})^*$
      $(u^4_{<2>})^* \leftarrow \pi_S^{-1}((v^4_{<2>})^*)$
      $(u^4_{<4>})^* \leftarrow \pi_S^{-1}((v^4_{<4>})^*)$
      $(u^4_{<2>})' \leftarrow u^4_{<2>} \oplus (u^4_{<2>})^*$
      $(u^4_{<4>})' \leftarrow u^4_{<4>} \oplus (u^4_{<4>})^*$
      if $((u^4_{<2>})' = 0110)$ **and** $((u^4_{<4>})' = 0110)$
        then $Count[L_1, L_2] \leftarrow Count[L_1, L_2] + 1$

$max \leftarrow -1$
for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do **if** $Count[L_1, L_2] > max$
    then $max \leftarrow Count[L_1, L_2]$
    $maxkey \leftarrow (L_1, L_2)$
output $(maxkey)$

Filtering Operation

- **Algorithm 3** presents the attack algorithm
- The input and output are similar to linear attack, except that $\mathcal{T}$ is a set (x, x^*, y, y^*), where x' is fixed
- **Algorithm 3** makes use of a certain **filtering operation**
- Tuples (x, x^*, y, y^*) for which the differential holds are often called **right pairs**, and allow us to determine the key bits
  - A right pair has the form
    $$(u^4_{<1>})' = (u^4_{<3>})' = 0000$$
  - Hence we consider those $y_{<1>} = (y_{<1>})^*$ and $y_{<3>} = (y_{<3>})^*$

Extracting Key Bits

- We have simulated attacking our basic cipher keyed using randomly generated subkeys by generating 5000 chosen plaintext-ciphertext pairs
  - i.e., 10000 encryptions with plaintext pairs satisfying x' = (0000 1011 0000 0000)
- The values in the following table indicate the estimated probability of the occurrence of the right pairs for the candidate partial subkey derived from

  prob = count/5000

  where the count is the count corresponding to the particular partial subkey value

Extracting Key Bits

- In our example, we would expect the probability of the occurrence of the right pair to be

  P_D = 27/1024 = 0.0264

  and we found experimentally the probability for the correct subkey [2, 4] gives P_D = 0.0244
- This indicates that the examination of incorrect target partial subkeys is not precisely equivalent to comparing random differences to the expected differential value

partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob
1 C	0.0000	2 A	0.0032
1 D	0.0000	2 B	0.0022
1 E	0.0000	2 C	0.0000
1 F	0.0000	2 D	0.0000
2 0	0.0000	2 E	0.0000
2 1	0.0136	2 F	0.0000
2 2	0.0068	3 0	0.0004
2 3	0.0068	3 1	0.0000
2 4	**0.0244**	3 2	0.0004
2 5	0.0000	3 3	0.0004
2 6	0.0068	3 4	0.0000
2 7	0.0068	3 5	0.0004
2 8	0.0030	3 6	0.0000
2 9	0.0024	3 7	0.0008

Experimental Results for Differential Attack