Cryptanalysis of Symmetric Key Cipher

Prof. Dipanwita Roy Chowdhury
Department of Computer Science & Engg.
IIT Kharagpur

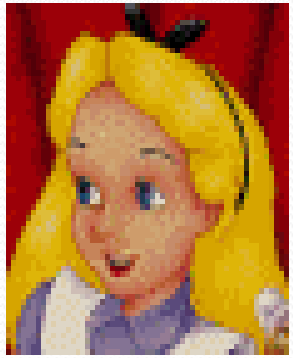
Outline

- Introduction
- Type of Cryptanalysis/ Attacks
- Algebraic Analysis
  - Linear Cryptanalysis
  - Differential Cryptanalysis
- Side Channel Attacks
  - Power attacks
  - Electromagnetic Radiation attacks
  - Timing attacks
  - Fault attacks
  - Scan Attack

What is Cryptography?

Data Integrity

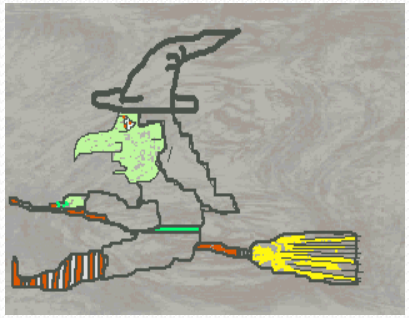
Dear Alice,



Bob

Alice

Authentication

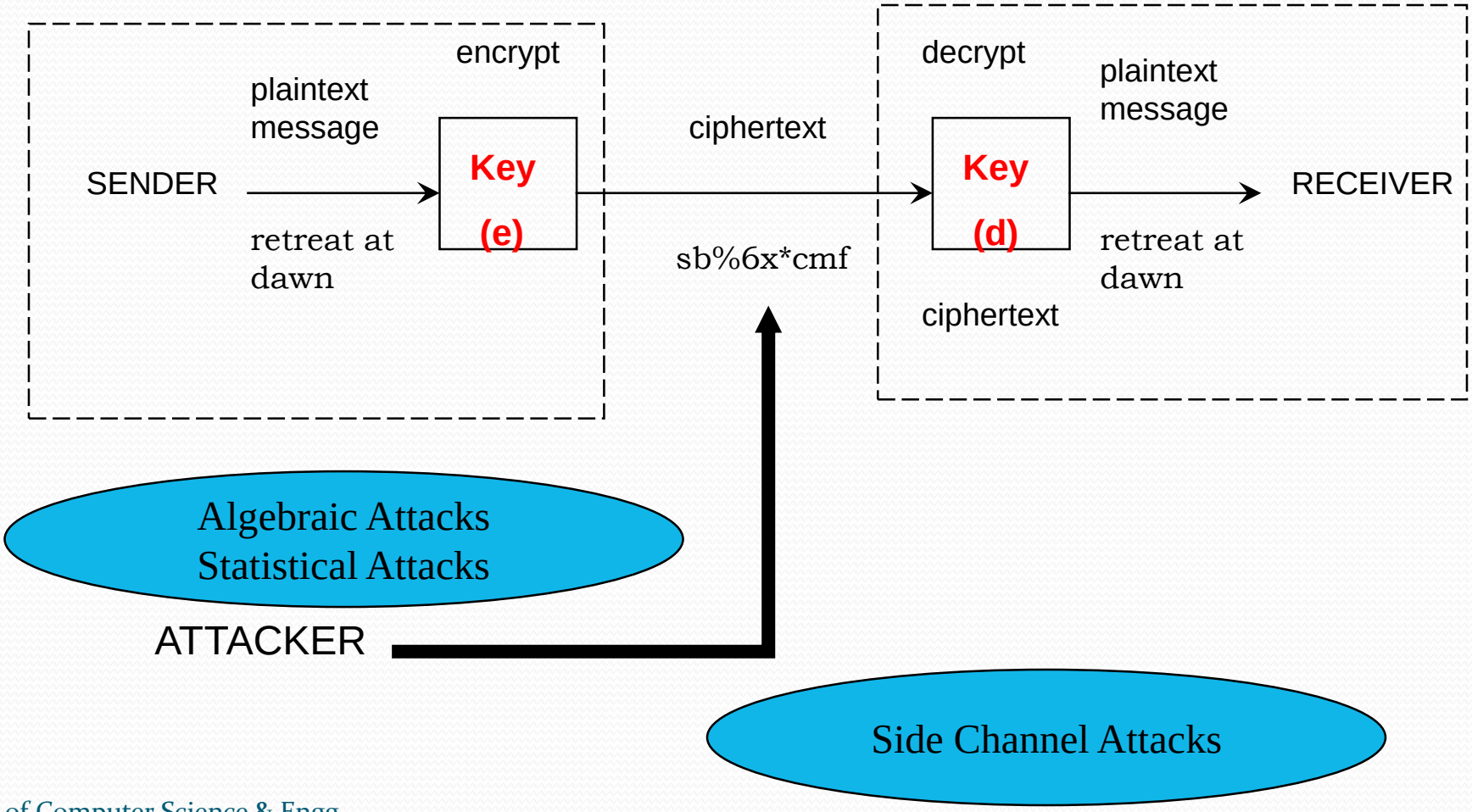


Non Repudiation

Eve

Dept. of Computer Science & Engg.
IIT Kharagpur, India

Cryptographic Algorithms



Types of Algorithms

- Private Key : The encryption key and decryption key are easily derivable from each other
  - Block Cipher : Fixed blocks of data
  - Data Encryption Standard (DES), Triple DES, Advanced Encryption Standard (AES)
  - Stream Cipher : Block Size = 1
  - eStream Winners Trivium, Grain, MICKEY, Rabbit
  - Hash Function : Variable message
  - SHA3 : Keecak, Grostel
- Public Key : Infeasible to determine the decryption key, d from the encryption key, e
  - Diffie- Hellman, RSA, Elliptic Curve Cryptography

Cryptanalysis/Attacks of Ciphers

Parameters to successfully execute the attack.

- Amount of required input data:
  - Number of input/output data required
- Number of necessary operations:
  - Amount of necessary computations required
- Storage complexity:
  - Amount of memory required
- Number of necessary physical actions :
  - Number of necessary measurements in the case of side channel analysis

Types of Cryptanalysis/Attacks

- Algebraic Analysis

  Linear Cryptanalysis, Differential Cryptanalysis

- Algorithmic / Structural Analysis

  Man-in-the-Middle Attack, Related Key Attack

- Side Channel Analysis

  Power Attack, Timing Attack, Fault Analysis etc.

Algebraic Analysis/Attacks

- Introduction
- Substitution-Permutation Networks
- Linear cryptanalysis
- Differential cryptanalysis

Introduction

- **A commonly used design for modern-day 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 key K:** used to construct Nr **round keys** (also called **subkeys**), which are denoted $K^1,\ldots,K^{Nr}$.
    - **key schedule ($K^1,\ldots,K^{Nr}$):** constructed from K using a fixed, public algorithm.
    - **round function g:** takes two inputs: a round key (K^r) and 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 done.

Introduction

- **Encryption operations:**

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

Decryption operations:

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

Note: function g is injective (one-to-one)

Substitution-Permutation Networks (SPN)

- **Cryptosystem : *SPN***
  - l, m and Nr are positive integers
  - $\pi_S : \{0,1\}^l \rightarrow \{0,1\}^l$ is a permutation
  - $\pi_P : \{1,\ldots, lm\} \rightarrow \{1,\ldots, lm\}$ is a permutation.
  - $P = C = \{0,1\}^{lm}$, and $K \subseteq (\{0,1\}^{lm})^{Nr+1}$ consist of all possible key schedules that could be derived from an initial key K using the key scheduling algorithm.
  - For a key schedule $(K^1,\ldots, K^{Nr+1})$, we encrypt the plaintext x using .

Substitution-Permutation Networks

- **Algorithm : SPN** $(x, \pi_S, \pi_P, (K^1, \ldots, K^{Nr+1}))$

$w^0 \leftarrow x$

for $r \leftarrow 1$ to $Nr-1$

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

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

for $i \leftarrow 1$ to m

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

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

output(y)

u^r : input to the S-boxes in round r.
v^r : output of the S-boxes in round r.
w^r : obtained from v^r by applying π_p .
u^{r+1} : 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).

Substitution-Permutation Networks

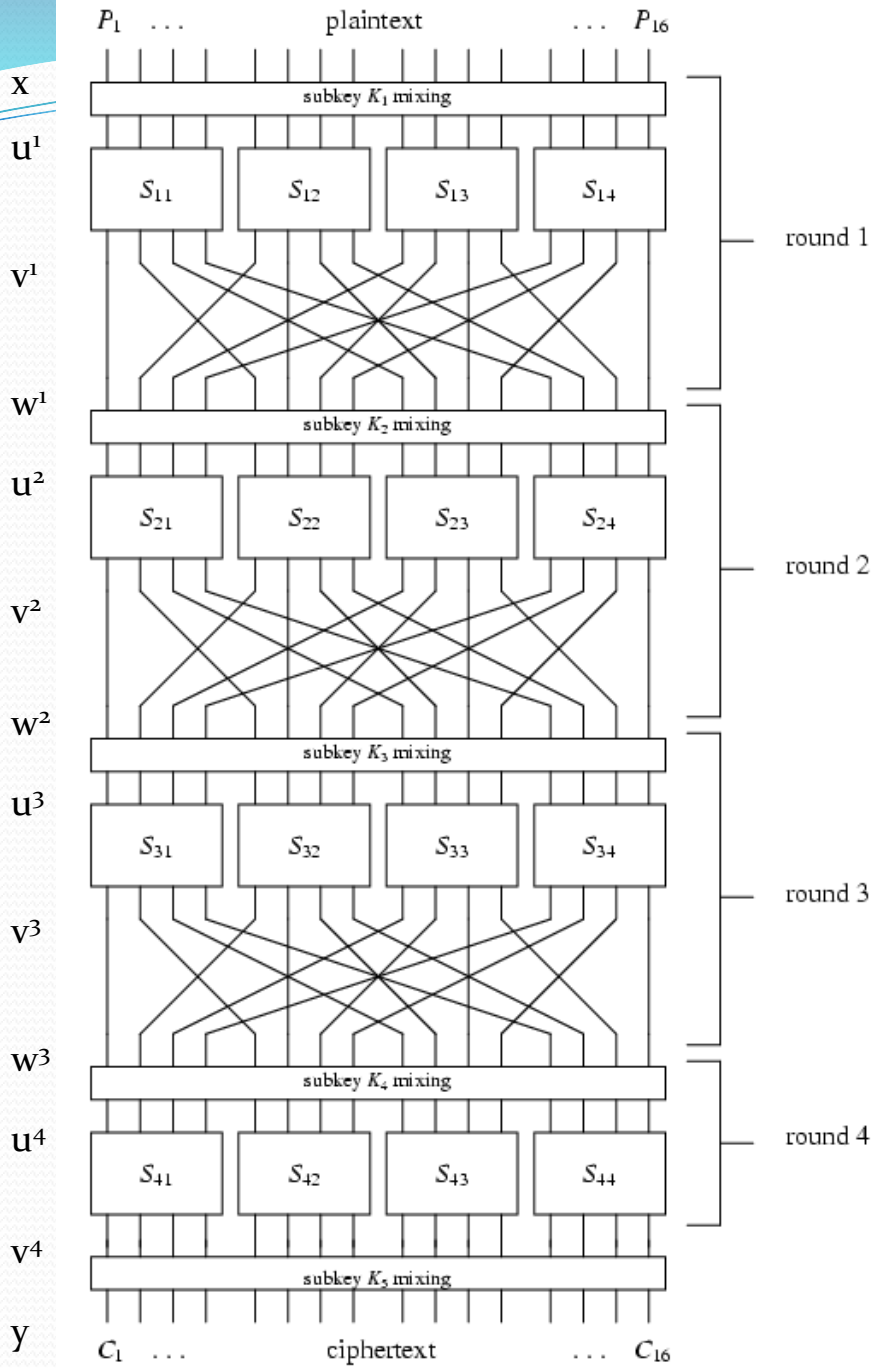
- **Example :**
  - Suppose $l = m = Nr = 4$.
  - Let π_S be defined as follows, where the input and the output are written in hexadecimal:

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

See **Figure** for a pictorial representation of this particular SPN, where S_{ir} means i-th round, r-th S-box.



A substitution-permutation network

Substitution-Permutation Networks

- **Key schedule:** 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}.
- K= 0011 1010 1001 0100 1101 0110 0011 1111
- Round keys:

  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

- We want to find a probability linear relationship between a subset of plaintext bits and a subset of data bits preceding the last round. This relation behaves in a non-random fashion.
- The attacker has a lot of plaintext-ciphertext pairs (**known plaintext attack**).
- For each candidate subkey, we partially decrypt the cipher and check if the relation holds. If the relation holds then increment its corresponding counter. At the end, the candidate key that counts furthest from ½ is the most likely subkey.

Linear Cryptanalysis

- **3.3.1 The Piling-up Lemma**
  - Suppose $X_1, X_2, \ldots$ are independent random variables from {0,1}. And

$$\Pr[X_i = 0] = p_i, \quad i = 1,2,\ldots \text{ Hence,}$$

$$\Pr[X_i = 1] = 1 - p_i, \quad i = 1,2,\ldots$$

  - The independence of X_i, X_j implies

$$\Pr[X_i = 0, X_j = 0] = p_i p_j$$

$$\Pr[X_i = 0, X_j = 1] = p_i(1 - p_j)$$

$$\Pr[X_i = 1, X_j = 0] = (1 - p_i) p_j$$

$$\Pr[X_i = 1, X_j = 1] = (1 - p_i)(1 - p_j)$$

Linear Cryptanalysis

- Now consider $X_i \oplus X_j$

$$\Pr[X_i \oplus X_j = 0] = p_i p_j + (1-p_i)(1-p_j)$$

$$\Pr[X_i \oplus X_j = 1] = p_i (1-p_j) + (1-p_i) p_j$$

- The **bias** of X_i is defined to be the quantity

$$\varepsilon_i = p_i - \tfrac{1}{2}$$

- And we have

$$-\tfrac{1}{2} \le \varepsilon_i \le \tfrac{1}{2},$$

$$\Pr[X_i = 0] = \tfrac{1}{2} + \varepsilon_i,$$

$$\Pr[X_i = 1] = \tfrac{1}{2} - \varepsilon_i.$$

Linear Cryptanalysis

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

- **Lemma (Piling-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

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

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

Suppose that $\varepsilon_{i_j} = 0$ for some j. Then $\varepsilon_{i_1,i_2,\ldots,i_k} = 0$.

Linear 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 $X=(x_1,\ldots,x_m)$. And the output n-tuple be $Y=(y_1,\ldots,y_n)$.
  - We can see that

$$\Pr[X_1 = x_1,\ldots, X_m = x_m, Y_1 = y_1,\ldots, Y_n = y_n] = 0$$

if $(y_1,\ldots, y_n) \neq \pi_S(x_1,\ldots, x_m)$; and

$$\Pr[X_1 = x_1,\ldots, X_m = x_m, Y_1 = y_1,\ldots, Y_n = y_n] = 2^{-m}$$

if $(y_1,\ldots, y_n) = \pi_S(x_1,\ldots, x_m)$.

  - Now we can compute the bias of the form

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

Linear Cryptanalysis

- **Example** : We use the S-box as **Example**.

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

Linear Cryptanalysis

- Consider $X_1 \oplus X_4 \oplus Y_2$. 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.
- It is seen that

$$\Pr[X_1 \oplus X_4 \oplus Y_2 = 0] = \tfrac{1}{2}$$

  Hence, the bias is 0.
- If we instead analyze $X_3 \oplus X_4 \oplus Y_1 \oplus Y_4$, we find that the bias is –3/8.

□

Linear Cryptanalysis

- We can record the bias of all 2^8=256 possible random variables.
- We represent the relevant random variable in the form

  where $\left(\bigoplus_{i=1}^{4} a_i X_i\right) \oplus \left(\bigoplus_{i=1}^{4} b_i Y_i\right)$.

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

- We treat (a_1,a_2,a_3,a_4) and (b_1,b_2,b_3,b_4) as hexadecimal digit (they are called **input sum** and **output sum**, respectively)

Linear Cryptanalysis

- Let $N_L(a,b)$ denote the number of binary eight-tuples $(x_1,x_2,x_3,x_4,y_1,y_2,y_3,y_4)$ s.t

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

and

$$\left(\bigoplus_{i=1}^{4} a_i X_i\right) \oplus \left(\bigoplus_{i=1}^{4} b_i Y_i\right) = 0$$

The bias is computed as $\varepsilon(a,b) = (N_L(a,b) - 8)/16$

- The table of all N_L is called the **linear approximation table** .

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

Figure: Linear approximation table values of $N_L(a,b)-8$

Linear Cryptanalysis

- **Linear Attack on an SPN**
  - Linear cryptanalysis requires 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).
  - **Figure** illustrates the structure of the approximation we will use.
    - Arrows are the random variables involved in the approximations and the labeled S-boxes (**active S-boxes**) are used in the approximations.

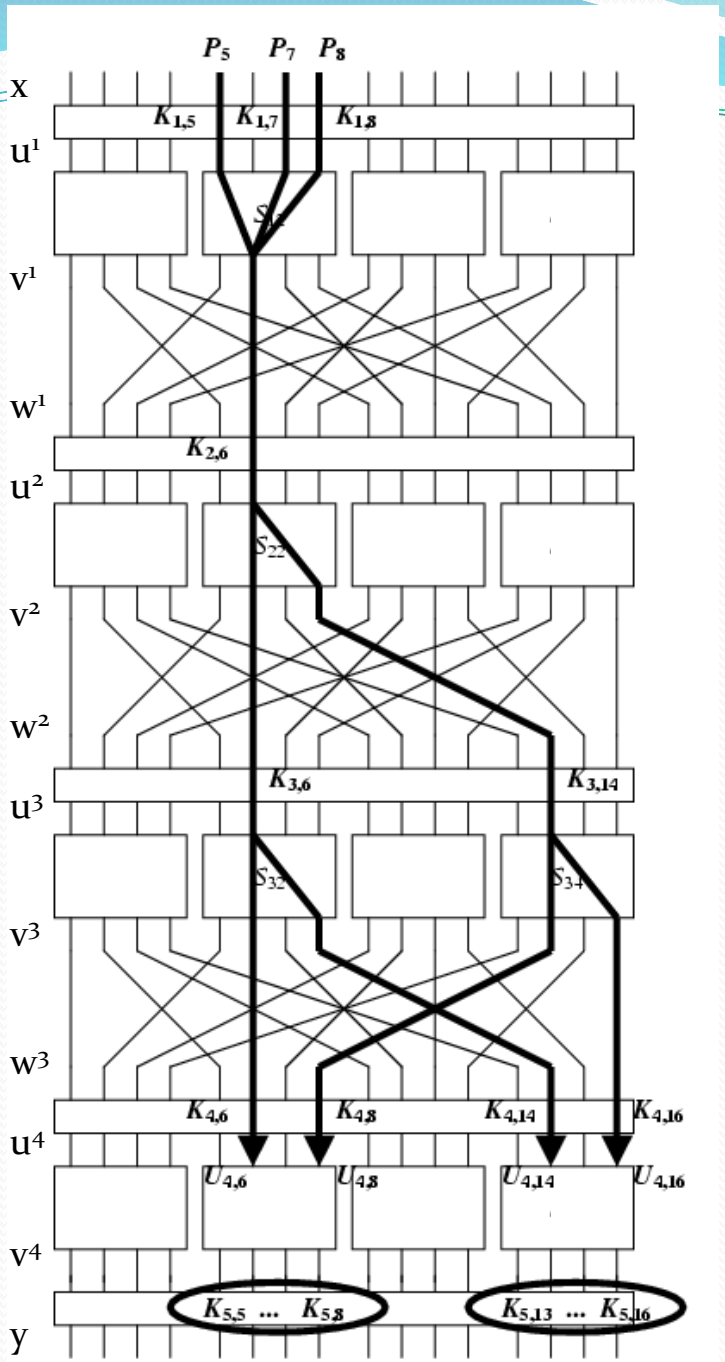


Figure: A linear approximation of an SPN

Linear Cryptanalysis

- The approximation incorporates four active S-boxes:
  - In S_{12}, $T_1 = U_5^1 \oplus U_7^1 \oplus U_8^1 \oplus V_6^1$ has bias ¼
  - In S_{22}, $T_2 = U_6^2 \oplus V_6^2 \oplus V_8^2$ has bias -¼
  - In S_{32}, $T_3 = U_6^3 \oplus V_6^3 \oplus V_8^3$ has bias -¼
  - In S_{34}, $T_4 = U_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3$ has bias -¼
- 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.

Linear Cryptanalysis

- Using **Piling-up lemma**, $T_1 \oplus T_2 \oplus T_3 \oplus T_4$ has bias equal to $2^3(1/4)(-1/4)^3=-1/32$.
  - Note: we assume the four r.v are independent.
- Then T_1, T_2, T_3, T_4 can be expressed in terms of plaintext bits, bits of u^4 (input to the last round) and key bits as follows:

$$T_1 = U_5^1 \oplus U_7^1 \oplus U_8^1 \oplus V_6^1 = X_5 \oplus K_5^1 \oplus X_7 \oplus K_7^1 \oplus X_8 \oplus K_8^1 \oplus V_6^1$$
$$T_2 = U_6^2 \oplus V_6^2 \oplus V_8^2 \qquad = V_6^1 \oplus K_6^2 \oplus V_6^2 \oplus V_8^2$$
$$T_3 = U_6^3 \oplus V_6^3 \oplus V_8^3 \qquad = V_6^2 \oplus K_6^3 \oplus V_6^3 \oplus V_8^3$$
$$T_4 = U_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3 \qquad = V_8^2 \oplus K_{14}^3 \oplus V_{14}^3 \oplus V_{16}^3$$

Linear Cryptanalysis

- XOR the right side and we get

$$X_5 \oplus X_7 \oplus X_8 \oplus V_6^3 \oplus V_8^3 \oplus V_{14}^3 \oplus V_{16}^3 \\ \oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3 \qquad (3.1)$$

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

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

- Now substitute them into 3.1:

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4 \\ \oplus K_5^1 \oplus K_7^1 \oplus K_8^1 \oplus K_6^2 \oplus K_6^3 \oplus K_{14}^3 \oplus K_6^4 \oplus K_8^4 \oplus K_{14}^4 \oplus K_{16}^4 \quad (3.2)$$

Linear Cryptanalysis

- The expression above only involves plaintext bits, bits of u^4 and key bits.
- Suppose the key bits 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

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4 \qquad (3.3)$$

  has bias -1/32 or 1/32 where the sign depends on the key bits (=0 or =1).

Linear Cryptanalysis

- The fact that (3.3) has bias bounded away from 0 allows us to carry out linear attack.
- Suppose that we have T plaintext-ciphertext pairs (denoted by τ), all use the same unknown key, K. The attack will allow us to obtain the eight key bits,

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

- There are 2^8=256 possibilities for the eight key bits. We refer to a binary 8-tuple as a **candidate subkey**.

Linear Cryptanalysis

- For each $(x, y) \in \tau$ and for each candidate subkey, we compute a partial decryption of y and obtain the resulting value for $u^4_{(2)}, u^4_{(4)}$.
- Then we compute the value

$$x_5 \oplus x_7 \oplus x_8 \oplus u^4_6 \oplus u^4_8 \oplus u^4_{14} \oplus u^4_{16} \qquad (3.4)$$

- We maintain an array of counters indexed by the 256 possible candidate subkeys, and increment the counter corresponding to a particular subkey when (3.4) has the value 0.
- In the end, we expect most counters will have a value close to T/2, but the correct candidate subkey will close to T/2±T/32.

Linear Cryptanalysis

- The attack is presented as **Algorithm**.
  - 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.

Algorithm: LINEARATTACK(τ, T, π_S^{-1})

for $(L_1, L_2) \leftarrow (0,0)$ to (F, F)
  do $Count[L_1, L_2] \leftarrow 0$
for each $(x, y) \in \tau$
  do
    for $(L_1, L_2) \leftarrow (0,0)$ to (F, F)
      do
        $v^4_{(2)} \leftarrow L_1 \oplus y_{(2)}$
        $v^4_{(4)} \leftarrow L_2 \oplus y_{(4)}$
        $u^4_{(2)} \leftarrow \pi_S^{-1}(v^4_{(2)})$
        do $u^4_{(4)} \leftarrow \pi_S^{-1}(v^4_{(4)})$
        $z \leftarrow x_5 \oplus x_7 \oplus x_8 \oplus u^4_6 \oplus u^4_8 \oplus u^4_{14} \oplus u^4_{16}$
        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 $(maxkey)$

Differential Cryptanalysis

- The main difference from linear attack is that differential attack involves comparing the XOR of two inputs to the XOR of the corresponding 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 key and determine if their XOR has a certain value. Whenever it does, increment the corresponding counter. At the end, we expect the largest one is the most likely subkey.

Differential Cryptanalysis

- **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 $\pi_S(x) \oplus \pi_S(x*)$

  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

- It is easy to see that any set $\Delta(x')$ contains 2^m pairs, and that

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

- For each pair in $\Delta(x')$ we can compute the output XOR of the S-box. Then we can tabulate the 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 attack.

Differential Cryptanalysis

- **Example:**
  - We use the same S-box as before. Suppose we consider input XOR x'=1011. Then

$$\Delta(1011) = \{(0000,1011), (0001,1010), \ldots, (1111,0100)\}$$

  - We compute the following table, where

$$x \oplus x^* = 1011,$$

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

$$y' = y \oplus y^*$$

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

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

Distribution table for x'=1011

Differential Cryptanalysis

- In **Example**, only 5 of the 16 possible output XORs occur. It has a very non-uniform distribution.
- We can compute all possible input XORs as **Example**.
- Define

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

  - $N_D(x',y')$ counts the number of pairs with input XOR equal to x' and output XOR equal to y'. (**Figure**)

Input Difference	Output Difference 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

Example

Figure: Difference distribution table: values of $N_D(x',y')$

Differential Cryptanalysis

- An input XOR is computed as

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

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

  - 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.

- Let a' denote the input XOR and let b' denote the output XOR. (a',b') is called a **differential**.

Differential Cryptanalysis

- **propagation ratio $R_p(a', b')$**:

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

  - $R_p(a', b')$ can be interpreted as a conditional probability:

$$\Pr[\text{output XOR} = b' \mid \text{input XOR} = a'] = R_p(a', b')$$

- We combine differentials in consecutive rounds to form a **differential trail**. A particular differential trail is shown in **Figure**.

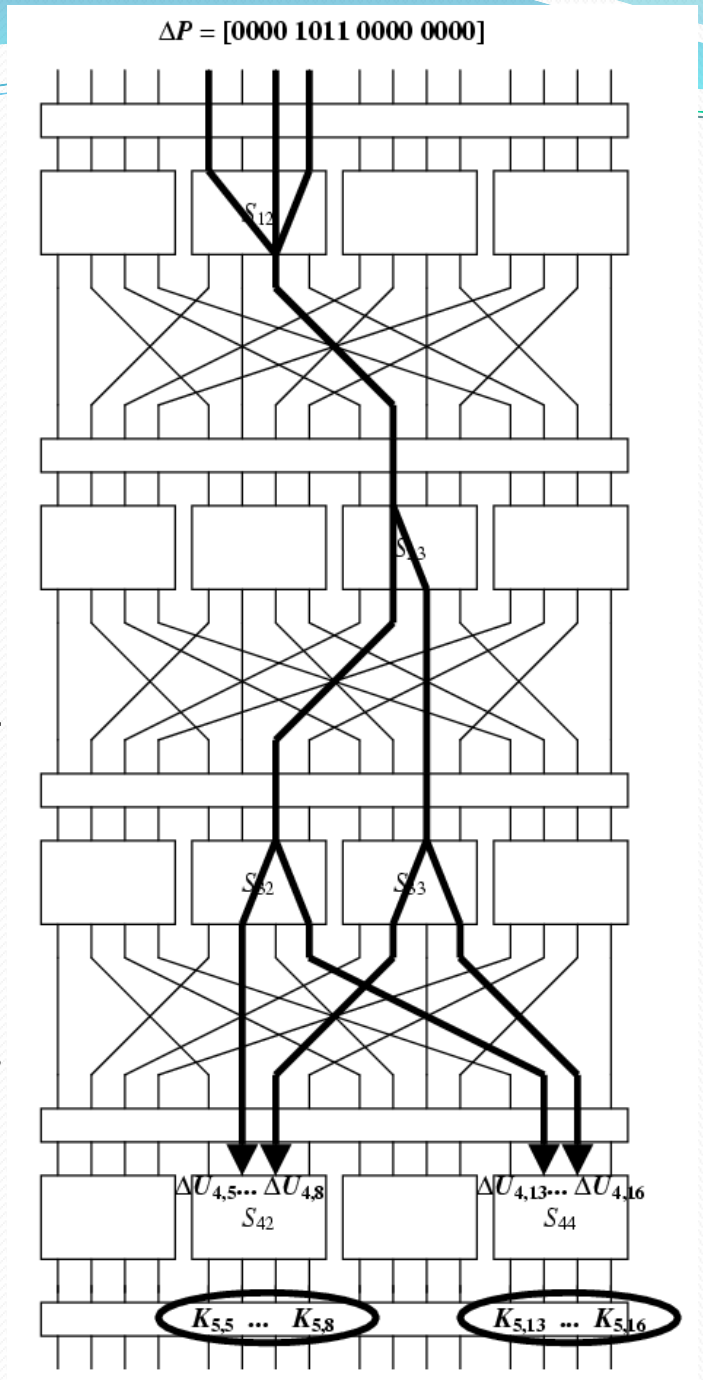


Figure: A differential trail for a SPN

Differential Cryptanalysis

- The differential attack arising from **Figure** uses the following propagation ratios of differentials:
  - In S_{12}, $R_p(1011,0010) = 1/2$
  - In S_{23}, $R_p(0100,0110) = 3/8$
  - In S_{32}, $R_p(0010,0101) = 3/8$
  - In S_{33}, $R_p(0010,0101) = 3/8$
- We therefore obtain a propagation ratio for a differential trail of the first three rounds of the SPN:

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

Differential Cryptanalysis

- 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

$$x' = 0000\ 1011\ 0000\ 0000 \Rightarrow (u^4)' = 0000\ 0110\ 0000\ 0110$$

with probability 27/1024.

Differential Cryptanalysis

- **Algorithm** presents the attack algorithm.
- The input and output are similar to linear attack, except that τ is a set (x,x*,y,y*), where x' is fixed.
- **Algorithm** 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 $y_{(3)} = (y_{(3)})^*$.

Algorithm : DIFFERENTIALATTACK(τ, T, π_S^{-1})

for $(L_1, L_2) \leftarrow (0,0)$ to (F, F)
do $Count[L_1, L_2] \leftarrow 0$
for each $(x, y, x^*, y^*) \in \tau$
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 $(maxkey)$

Differential Cryptanalysis

- A differential attack based on a differential trail having propagation ratio equal to ε will often be successful if the number of tuples (x,x*,y,y*), which we denote by T, is approximately $c\varepsilon^{-1}$, for a "small" constant c.

References

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

  November 1, 2005 by Chapman and Hall/CRC, Textbook - 616

  Pages – 27 B/W Illustrations, ISBN 9781584885085 - CAT# C5084
  Series: Discrete Mathematics and Its Applications

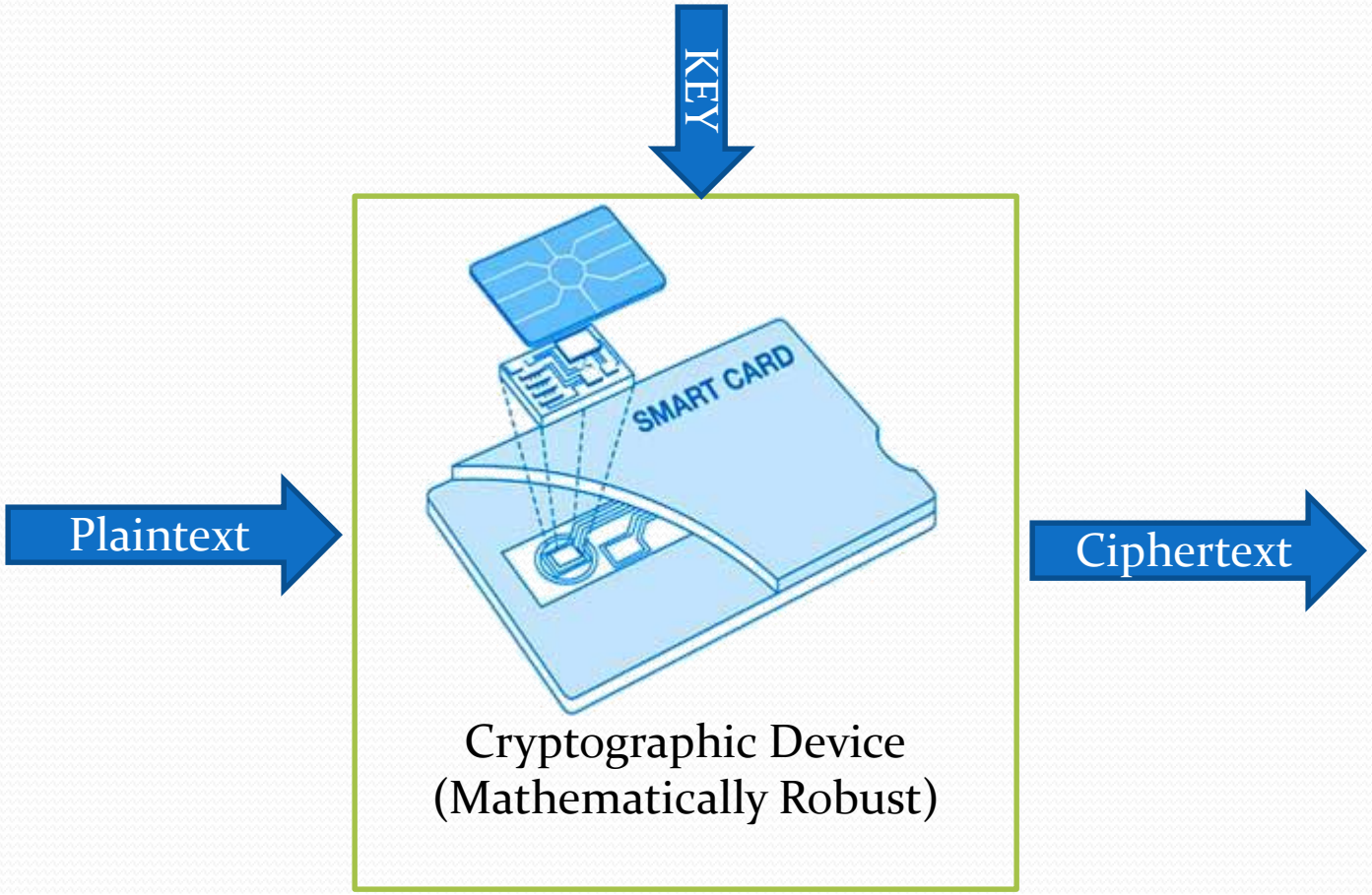
- **A Tutorial on Linear and Differential Cryptanalysis**

  https://www.engr.mun.ca/~howard/PAPERS/ldc_**tutorial**.pdf

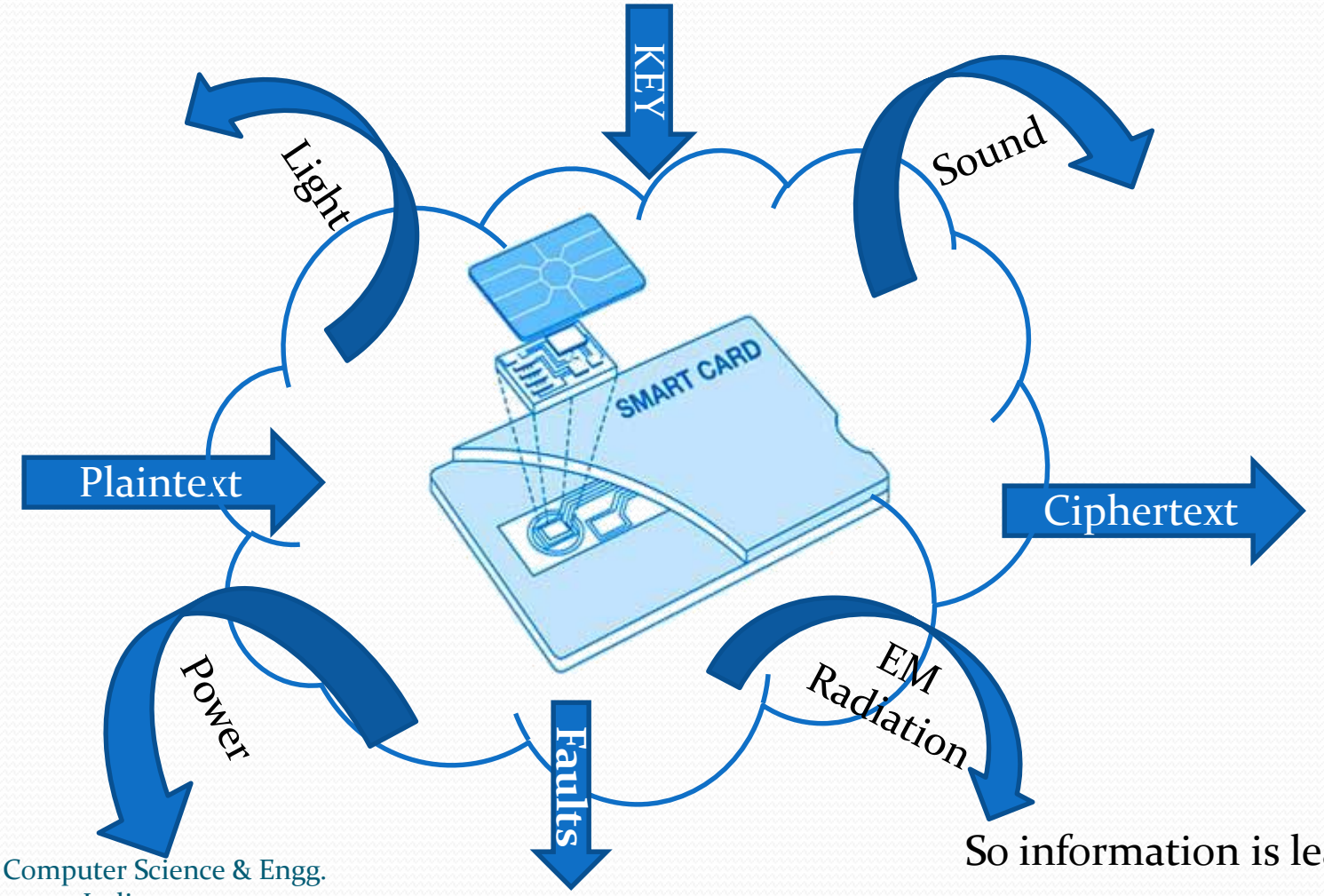
What is a Side Channel Attack(SCA)?

- Most cryptographic algorithms are theoretically or computationally secure but their implementations may be vulnerable
- A 'side channel' is a source of information that is inherent to the physical implementation of a primitive viz, light, sound, power, etc.
- SCA exploits such vulnerabilities
- Implementation based attacks

Conventional Viewpoint



Side Channel Viewpoint



So information is leaking.

Types of SCA

- Power Attacks
- Radiation Monitoring Attacks
- Acoustic Attacks
- Scan-Chain Based Attacks
- Fault Attacks

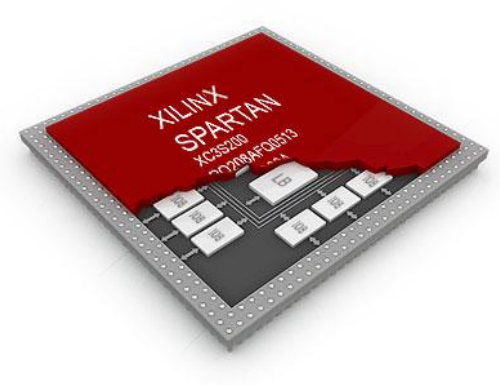
Our Focus

Underlying Idea:
Information leaked by these side channels can give useful information about the secret key

Hardware targets

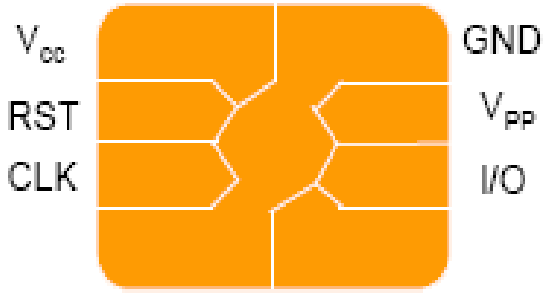
- The most common hardware that are targetted are
  - the smart cards (SC)

  - The FPGAs



Smart Cards

- It has a small processor (8bit or 32bit) long with ROM, EEPROM and a small RAM



- There are eight wires connecting the process
- Power supply: SCs have no internal batteries, the current provided by the reader
- Clock: SCs do not have an internal clock

Type of Attack classification

- Many possible attacks, the attacks are often not mutually exclusive
- Invasive vs. noninvasive attacks
- Active vs. passive
  - Active attacks tamper with device's proper functionality, either temporary or permanently
- Passive, Non-invasive attacks and relatively inexpensive

Major Attack Groups

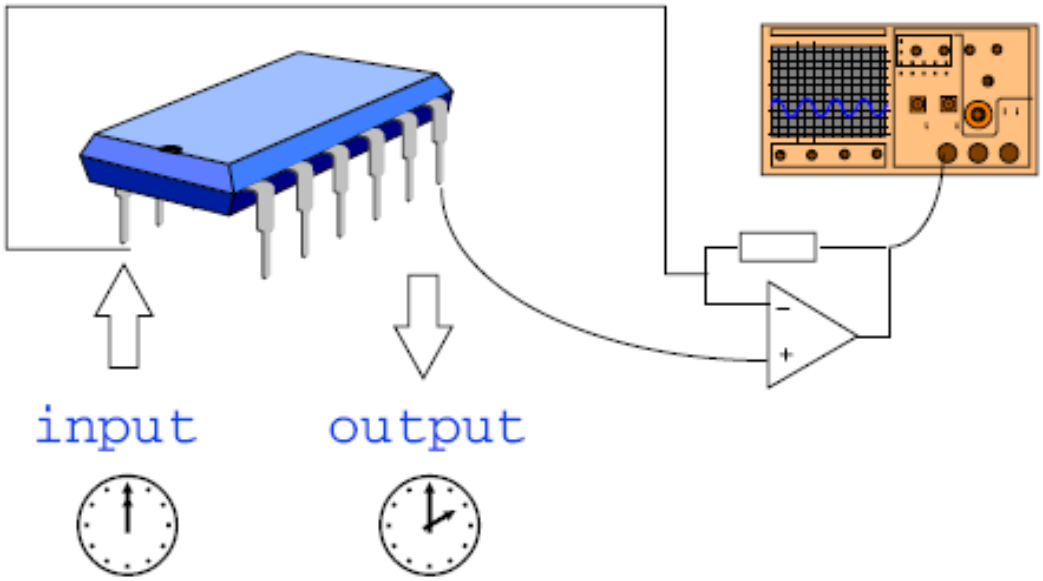
- Probing attack (invasive)
- Fault injection attacks – active attacks , maybe invasive or noninvasive
- Timing attacks exploit device's running time
- Power analysis attack
- Electromagnetic radiation attacks

Outline

- Introduction
- Type of Side Channel Attacks
- **Power attacks**
- Electromagnetic Radiation attacks
- Timing attacks
- Fault attacks
- Scan Attack

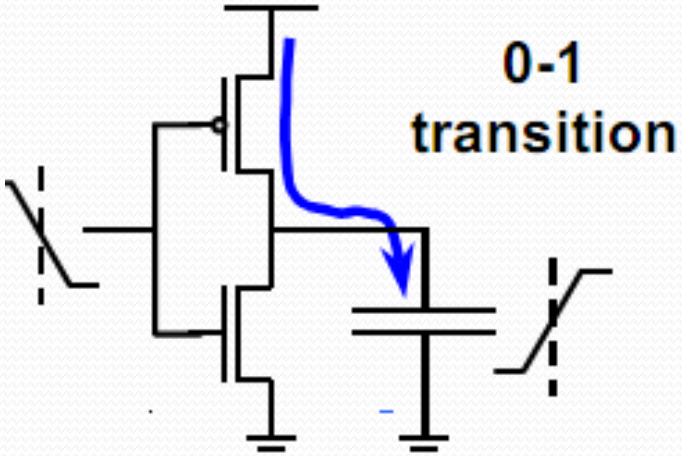
Power attacks

- Measure the circuit's processing time and current consumption to infer what is going on inside it.



Basic Principle of Power Analysis

Input	Output	Dynamic Power Consumption
$0 \rightarrow 0$	1	No
$0 \rightarrow 1$	$1 \rightarrow 0$	Discharge
$1 \rightarrow 0$	$0 \rightarrow 1$	Charge
$1 \rightarrow 1$	0	No



The CMOS Inverter

- CMOS – Most popular logic style
- CMOS inverter is the basic building block
- Consumes power from supply when there is a $0 \rightarrow 1$ transition in the output
- Switching Activity – Number of times the output of a logic gate switches
- Basic assumption - Power dissipation is a function of the switching activity

Power Attacks

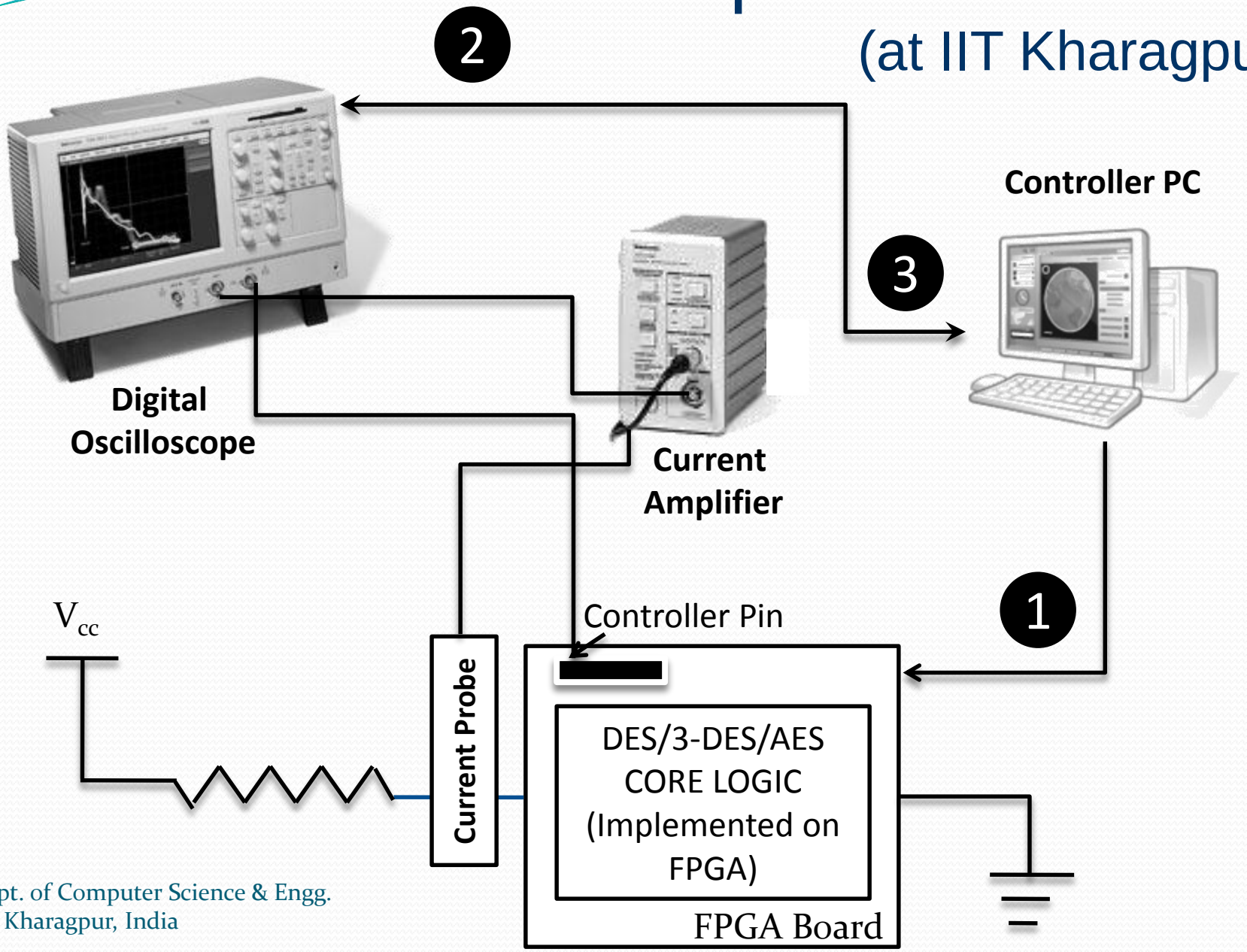
- SPA – Simple Power Analysis Attacks
  - Fact exploited - Power consumption at an instant of time is a function of the operation being carried out by the device

- DPA – Differential Power Analysis Attacks
  - Fact exploited - Power consumption of the same operation at different instants of time depends on the data being processed.

Power Attacks (PA)

- During the last few years (eight ?) lot of research has been conducted on Differential Power Attacks (DPA)
- Exploit the fact that (dynamic) power consumption of chip is correlated to intermediate results of the algorithm
- To measure a ckt's power, a small resistor (50 ohm) is inserted in series with the power or ground input

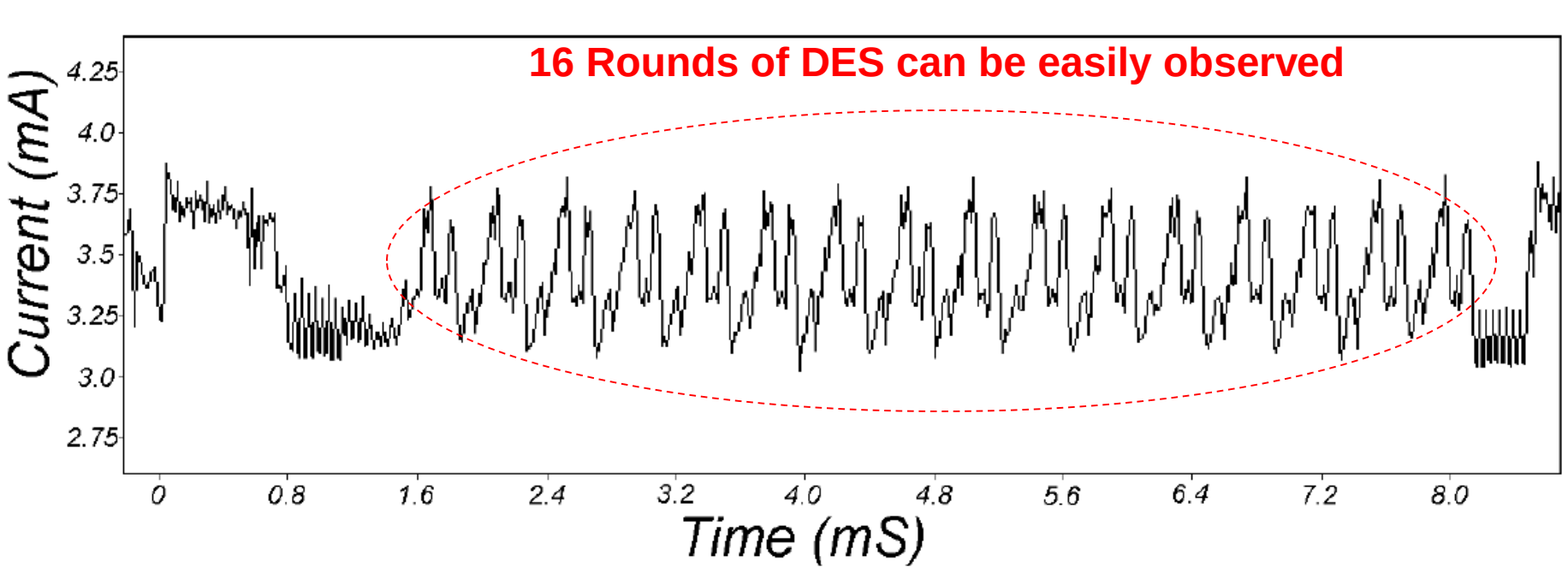
Experiment Set-up
(at IIT Kharagpur)



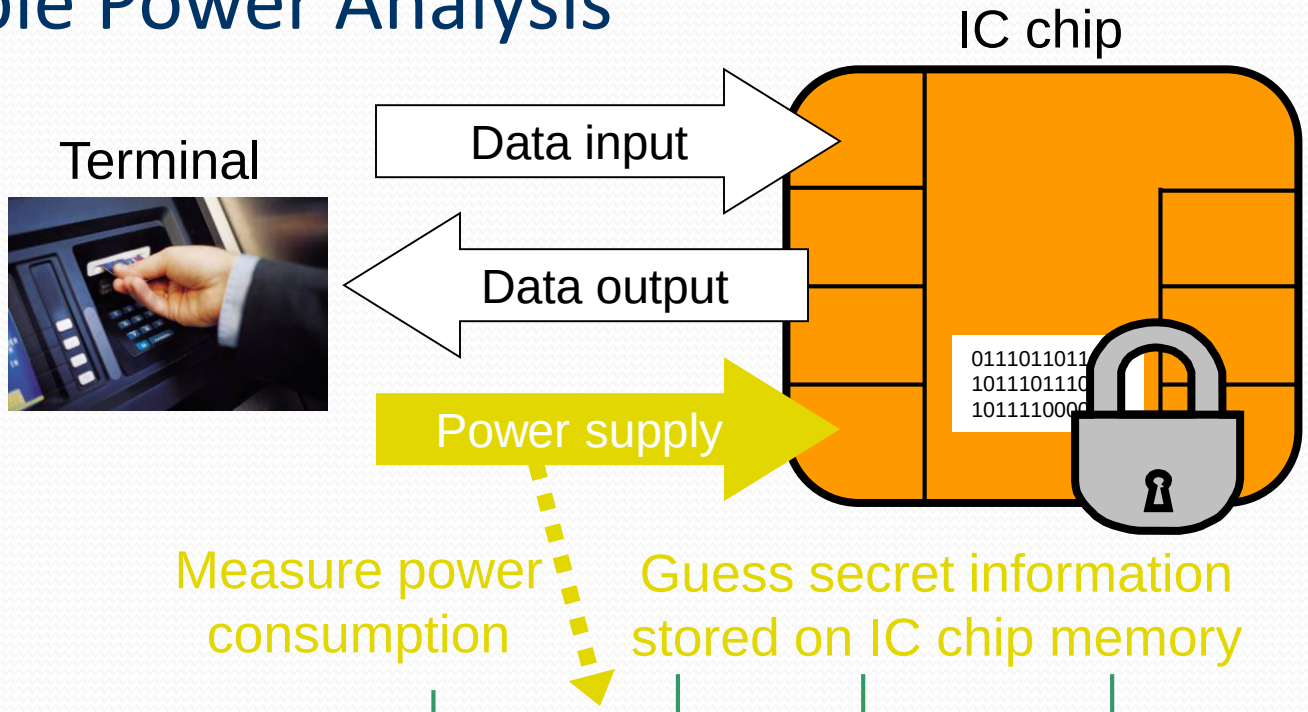
Simple Power Analysis (SPA)

- Directly interprets the power consumption of the device
- Looks for the operations taking place and also the key!
- Trace: A set of power consumptions across a cryptographic process
- As an example: 1 millisecond operation sampled at 5MHz yield a trace with 5000 points

Power Traces of DES



Simple Power Analysis



Differential Power Analysis (DPA)

DPA Overview

Introduced by P. Kocher and colleagues

More powerful and more difficult to prevent than SPA

Different power consumption for different state(0 or 1)

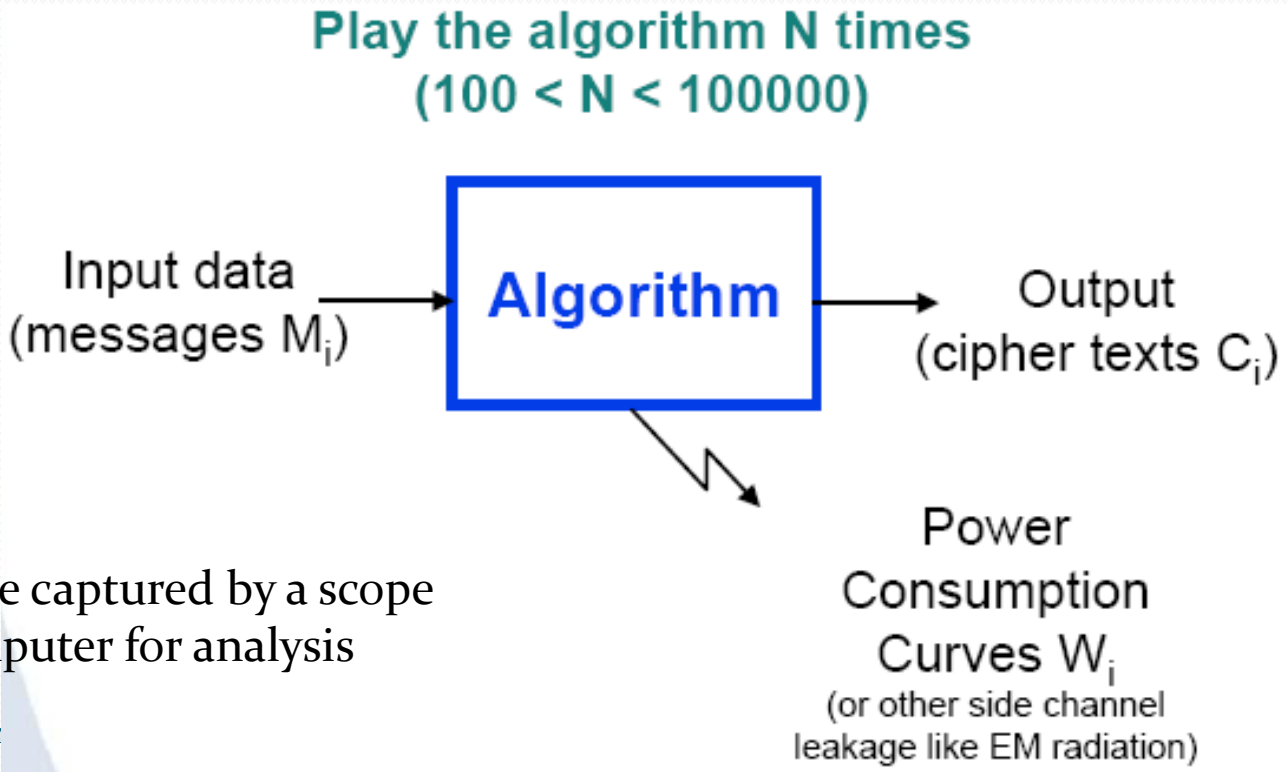
Data collection phase and data analysis phase

Procedure

- Gather many power consumption curves
- Assume a key value
- Divide data into two groups(0 and 1)
- Calculate mean value curve of each group
- Correct key assumption → not negligible difference

DPA (cont'd)

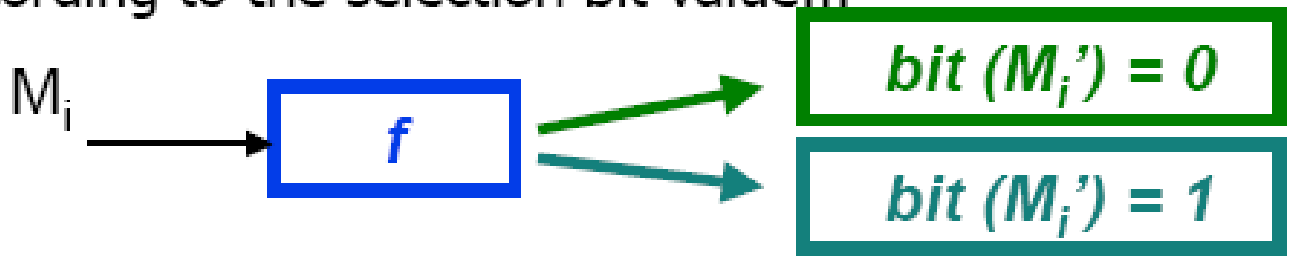
- DPA can be performed in any algorithm that has the operation $C=S(P \oplus K)$,
  - P is known and K is the segment key



The waveforms are captured by a scope and Sent to a computer for analysis

DPA (cont'd)

- Partition the data and related curves into two packs, according to the selection bit value…



- … and assign -1 to pack 0 and +1 to pack 1

0	B688EE57BB63E03E	1	+1
1	185D04D77509F36F	0	-1
2	C031A0392DC881E6	1	+1

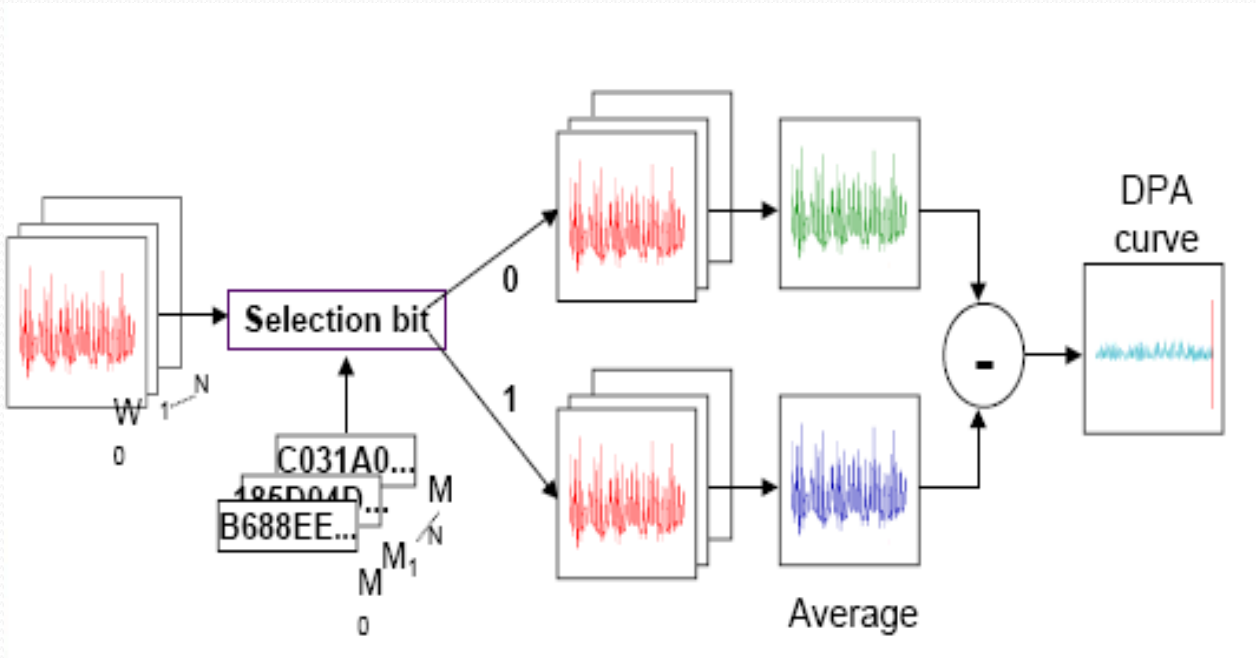
…

- Sum the signed consumption curves and normalise
- <=> Difference of averages

$(N_0 + N_1 = N)$

$$DPA = \frac{\sum W_1}{N_1} - \frac{\sum W_0}{N_0}$$

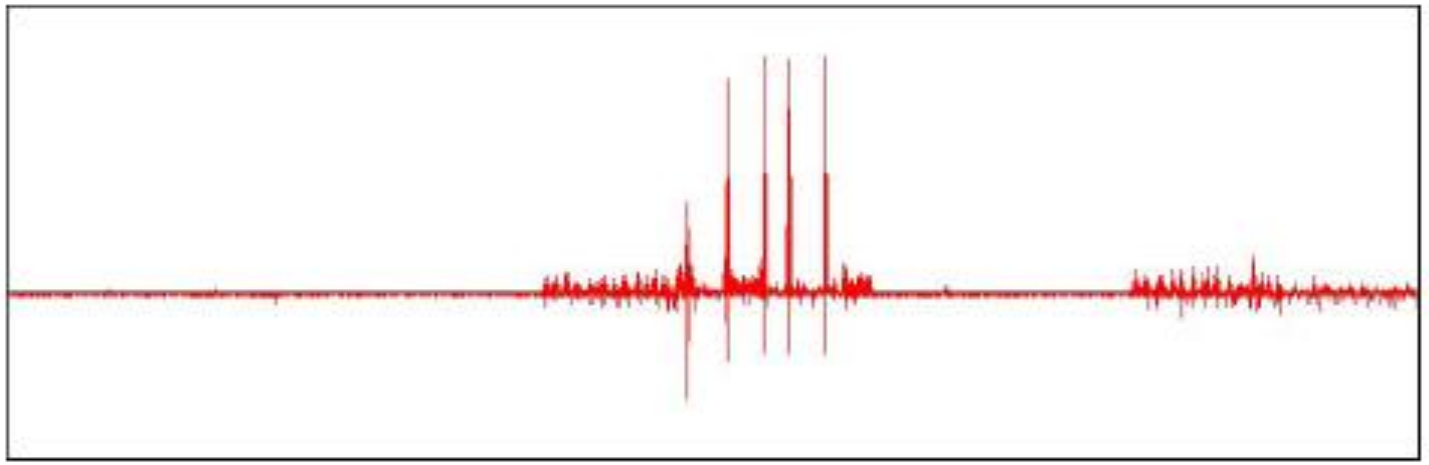
DPA (cont'd)



$$\Delta_n = \frac{\sum_{w_i \in S_0} w_i}{|S_0|} - \frac{\sum_{w_i \in S_1} w_i}{|S_1|}$$

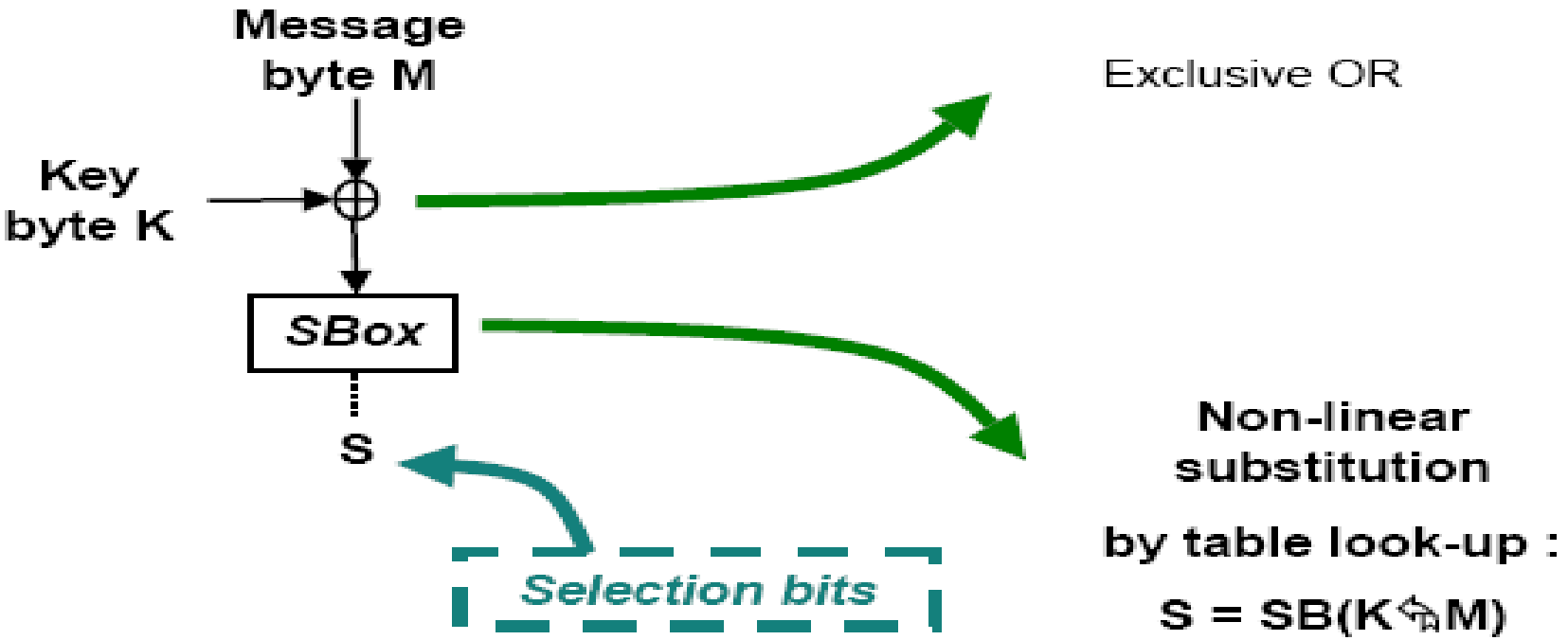
DPA (cont'd)

- The DPA waveform with the highest peak will validate the hypothesis



Typical DPA Target

- Basic mechanism in Secret Key algorithms (AES, DES…)



DPA Procedure for DES

1. Make power consumption measurement of about 1000 DES operations, 100000 data points / curve, ($Ciphertext_i$, $Curve_i$)
2. Assume a key for a S-box of last round
3. Calculate first S-box first bit input for each ciphertext using the assumed key
4. Divide the measurement into 2 groups (output 0 and 1)
5. Calculate the average curve of each group
6. Calculate the difference of two curves
7. Assumed correct key → spikes in the differential curve
8. Repeat 2-7 for other S-boxes
9. Exhaustive search for 8 bits of key

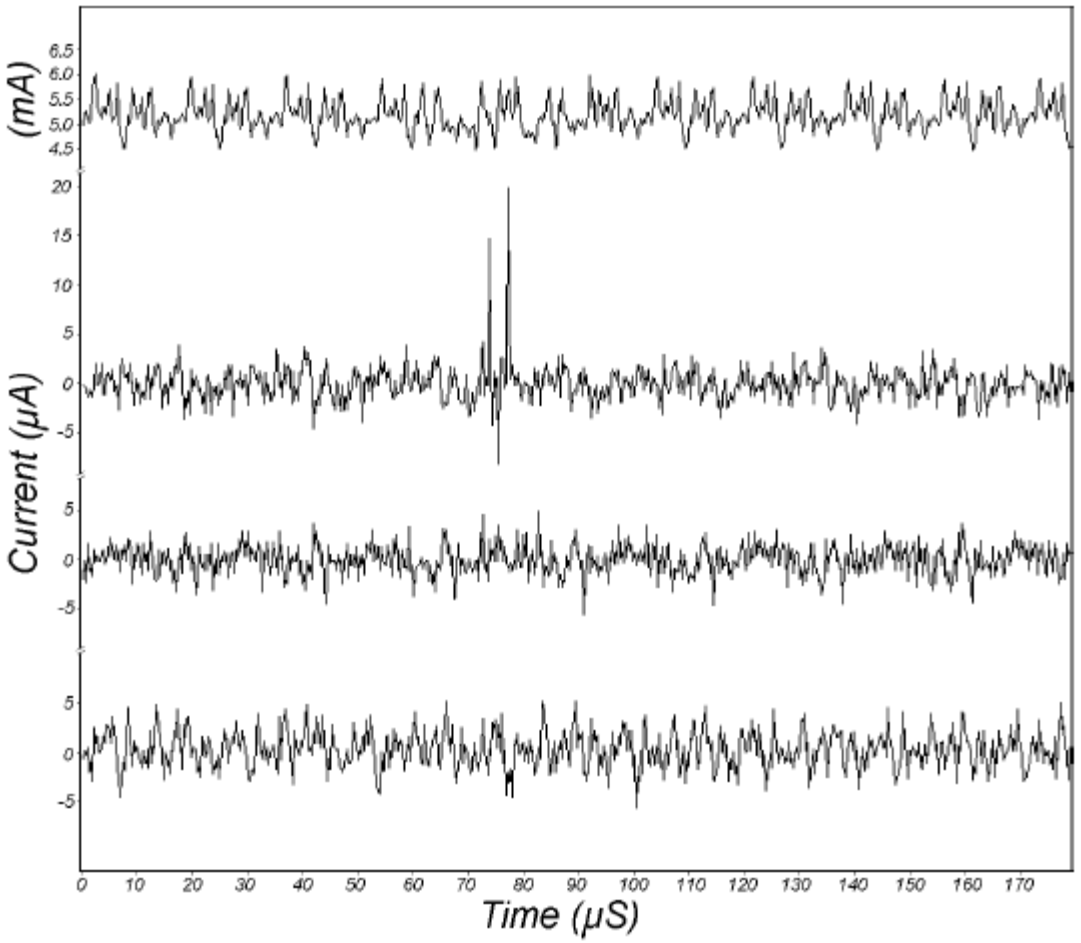
DPA Result Example

Average Power Consumption

Power Consumption Differential Curve With Correct Key Guess

Power Consumption Differential Curve With Incorrect Key Guess

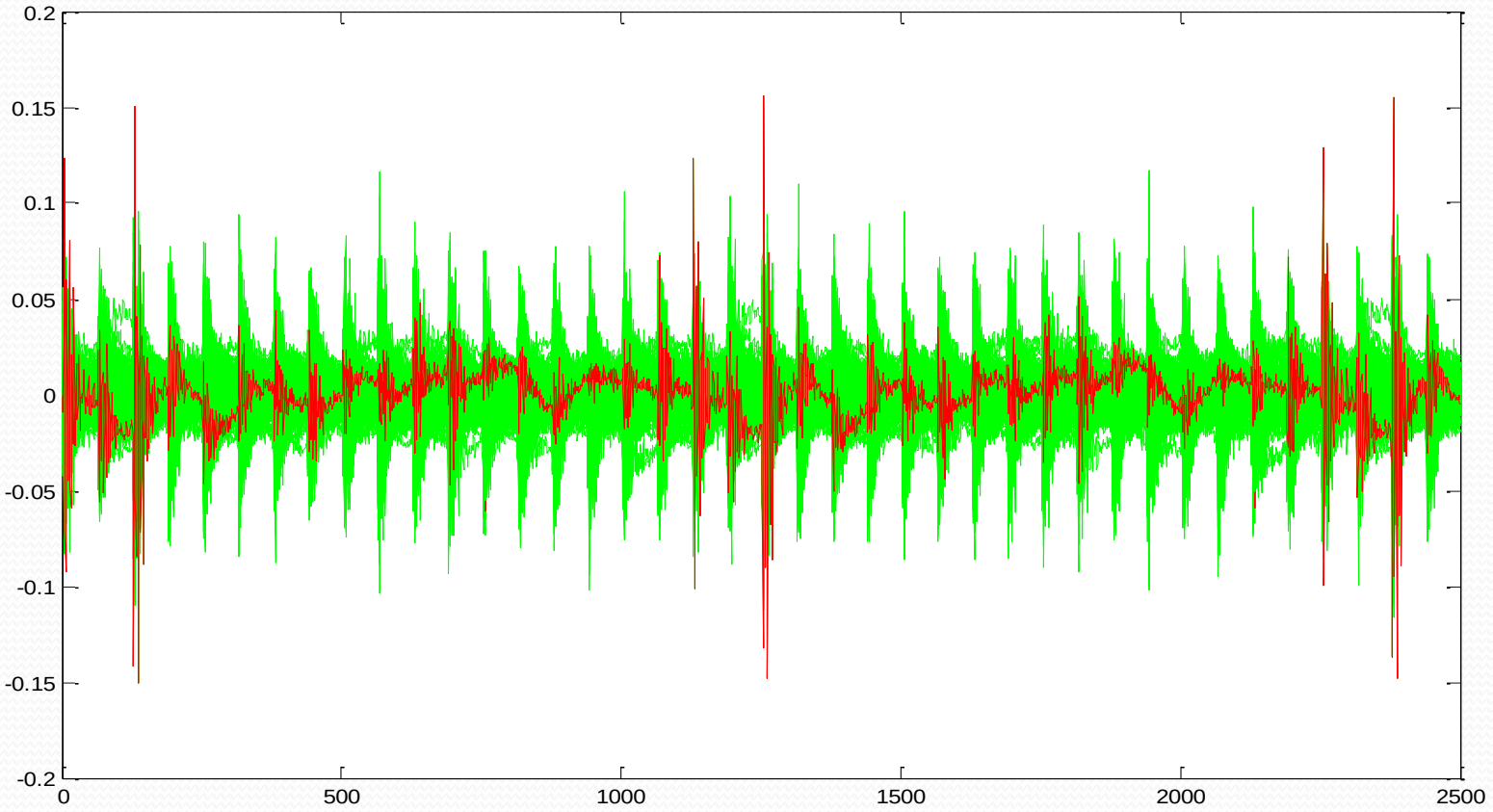
Power Consumption Differential Curve With Incorrect Key Guess



DPA in details

- Attacker obtains m encryption operations and capture power traces, with k sample points each.
- An attacker records the m ciphertexts
- No knowledge of the plaintext is required

DPA Results - DES

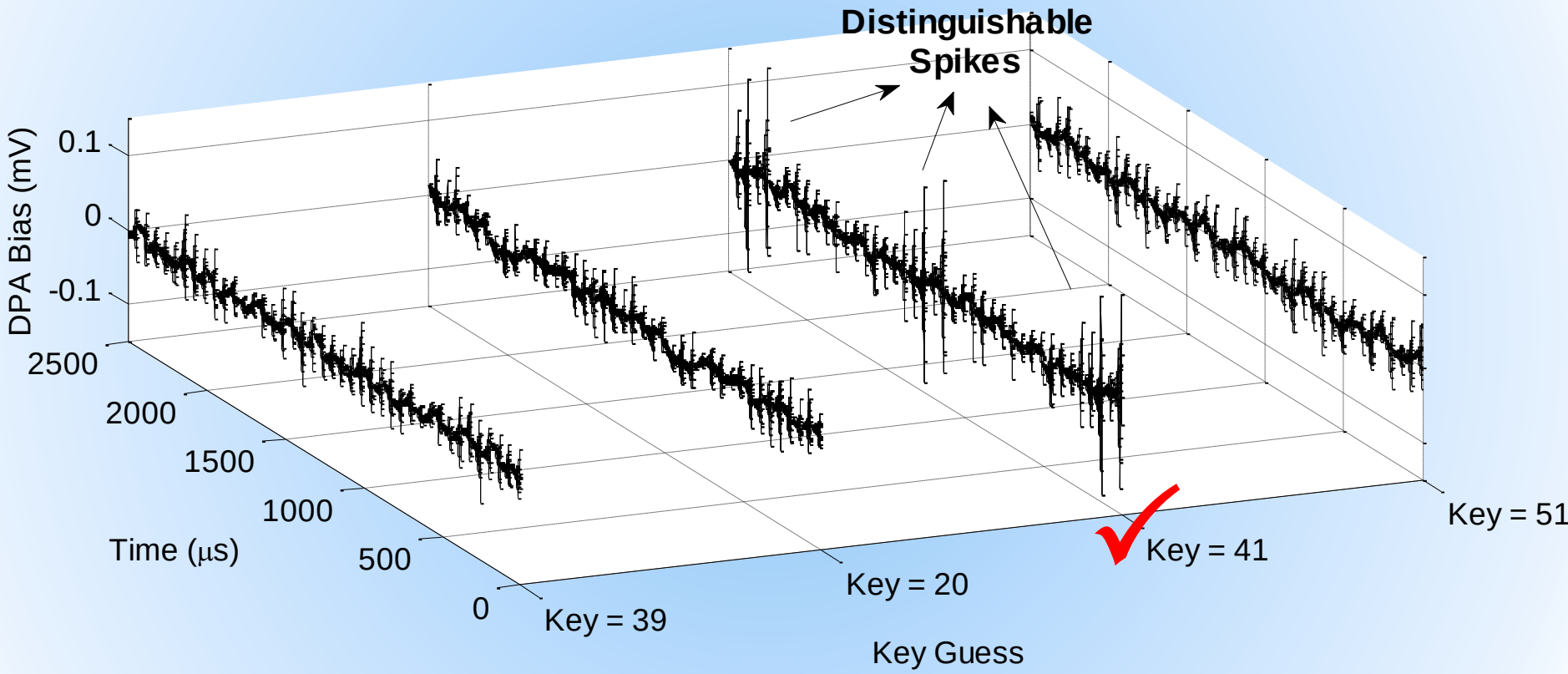


2D Differential Plot

SBOX – 3 **BIT – 3** **TRACE COUNT = 4,000**

Dept. of Computer Science & Engg.
IIT Kharagpur, India

DPA Results - DES

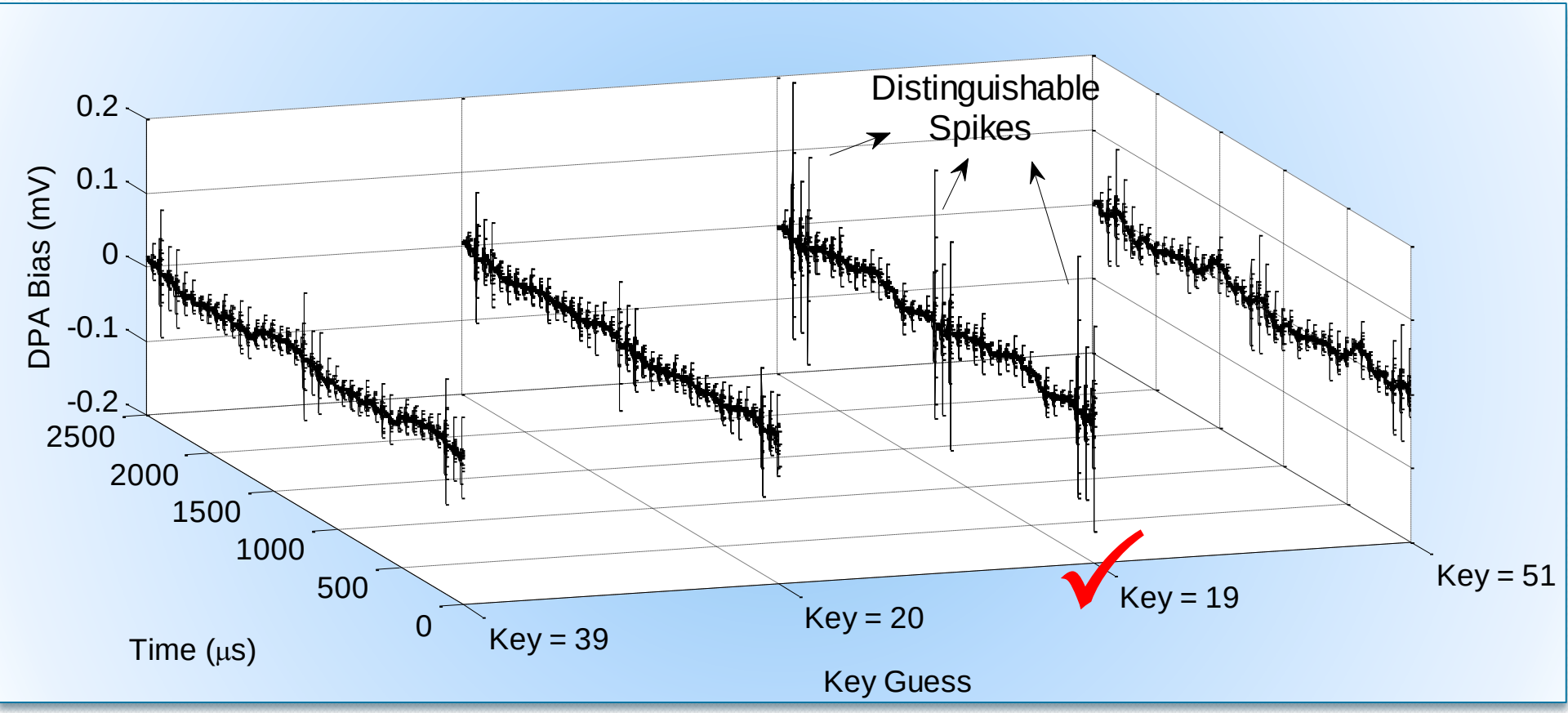


3D Differential Plot

Dept. of Computer Science & Engg.
IIT Kharagpur, India

SBOX – 3 **BIT – 3** **TRACE COUNT = 4,000**

DPA Results - Triple-DES

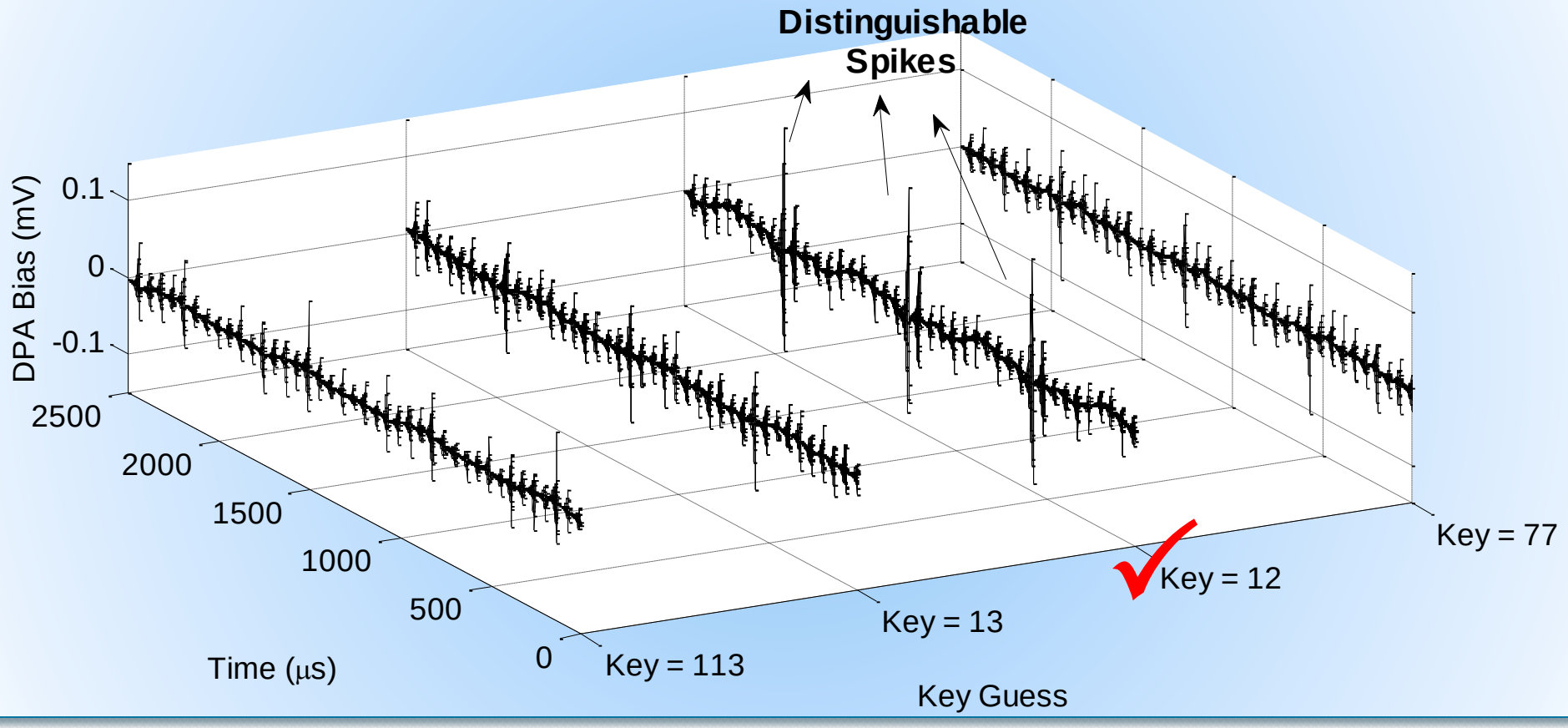


3D Differential Plot

SBOX – 4 **BIT – 2** **TRACE COUNT = 10,000**

Dept. of Computer Science & Engg.
IIT Kharagpur, India

DPA Results - AES



3D Differential Plot

SBOX – 11 **BIT – 8** **TRACE COUNT = 15,000**

Probable Key Differential Power Analysis (PKDPA)

- Dhiman saha, D. Mukhopadhyay, D. Roy chowdhury, PKDPA: An Enhanced Probabilistic Differential Power Attack Methodologies, INDOCRYPT 2011
- Basic idea :
  - Don't look for the key with highest DPA Bias
  - Instead look at a fixed set of keys with high DPA biases
  - Do this for all target bits
  - Then perform a frequency analysis to get the correct key

DPA Vs PKDPA

- Perform Difference of Means(DoM) test
- Find key with highest DoM
- Repeat for all target bits
- If results of all bits be same, then conclude that to be the correct key
- Else repeat with higher # of power traces

- Perform Difference of Means(DoM) test
- Find n keys with high DoMs
- Repeat for all target bits
- Perform frequency analysis
- Key with $f \geq ½$ (# of target bits) = correct key
- Else repeat with higher # of power traces

The Probable Key Matrix

Target Bit →

B1	B2	B3	B4	B5	B6	B7	B8
180	30	101	41	93	197	26	182

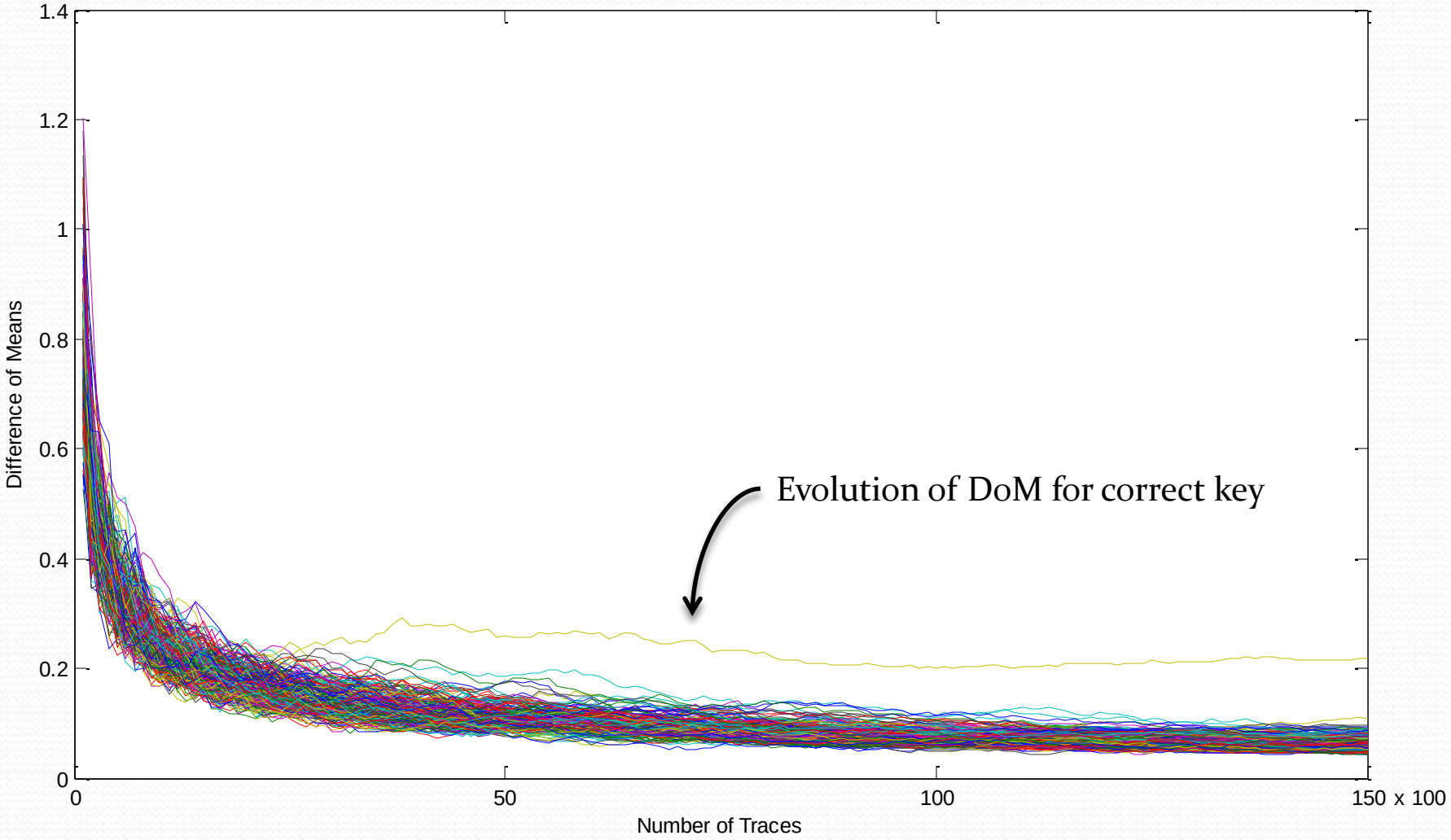
Each column represents a probable key set for the related target bit

First row represents keys with highest DoM values i.e., DPA keys

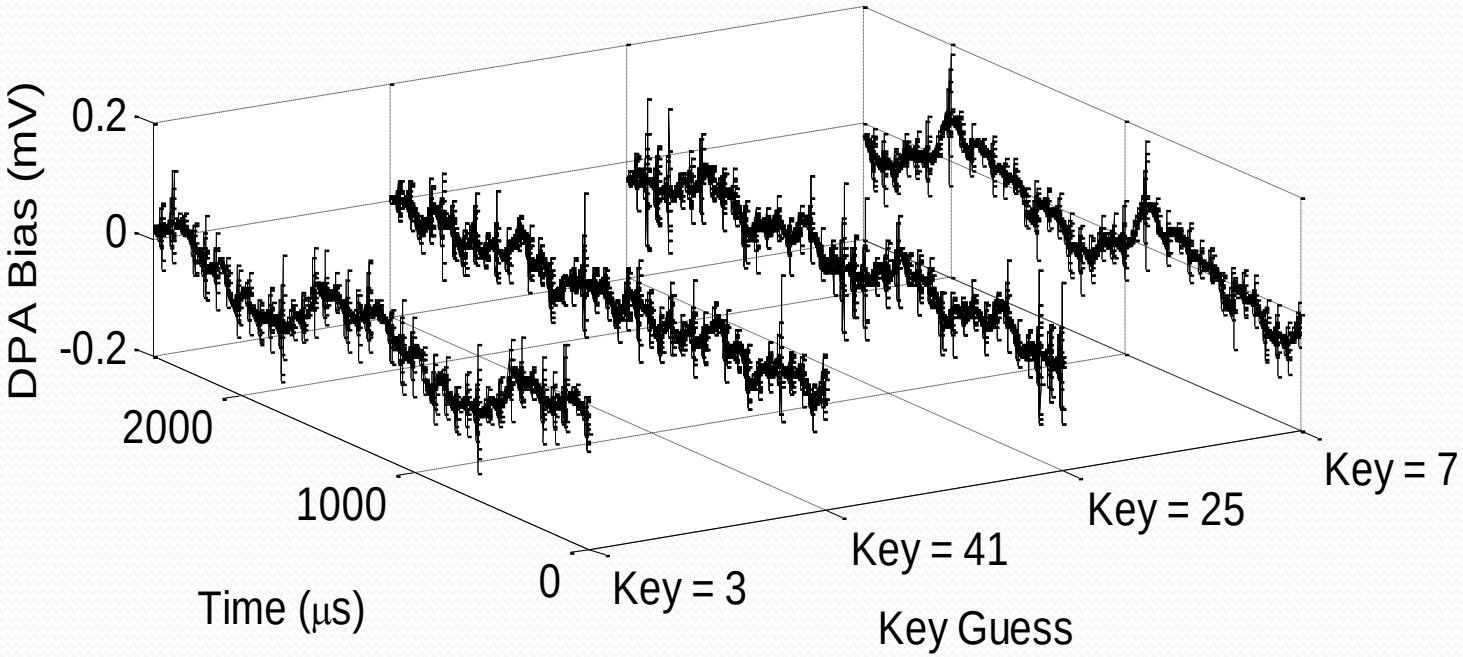
Keys arranged in descending order of DoM for every target bit

Probable Key Matrix for AES
Sbox – 13, Traces 2,500, Window-Size = 10

Probable Key Progression



PKDPA Results - DES



3D Differential Plot

SBOX – 7 BIT – 4 TRACE COUNT = 900

Bit	Key Returned
1	37
2	22
3	34
4	7

DPA Keys

?

Correct Key = 25

Cannot be Distinguished

Classical DPA fails for 900 Traces

PKDPA Results - DES

SBOX – 7

TRACE COUNT = 900

Window-Size = 5

Most Frequent Key = 25

Frequency = 3

B1	B2	B3	B4
37	22	34	7
12	40	31	41
25	42	25	3
46	10	60	9
61	20	44	25

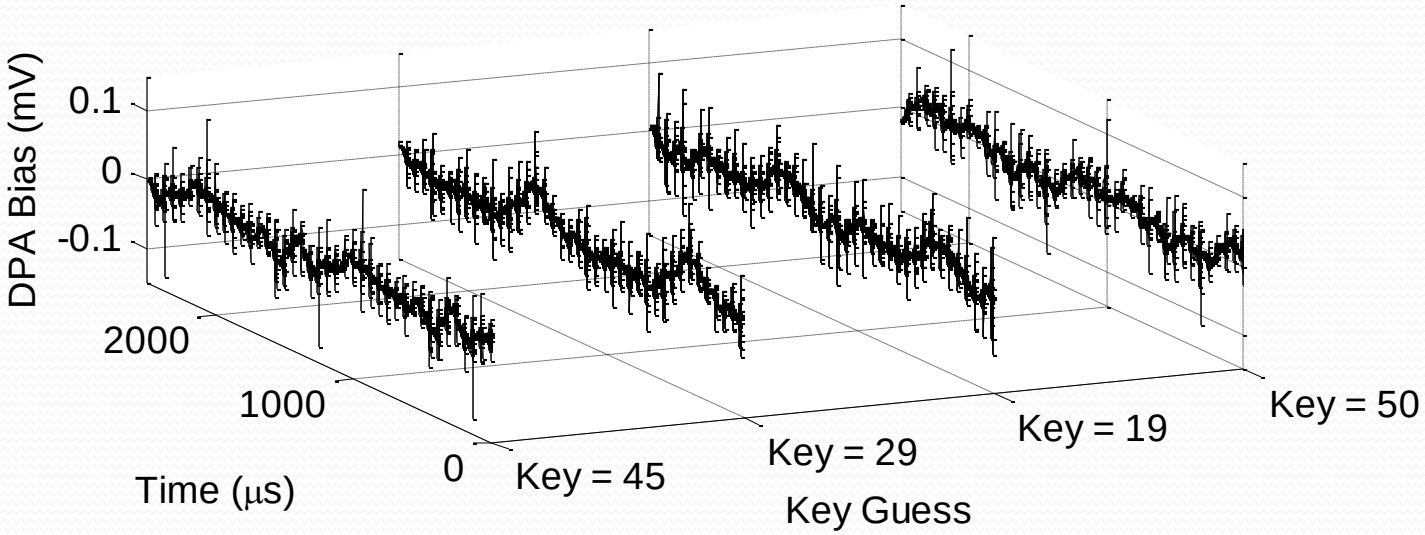
Probable Key Matrix

Correct Key = 25

By PKDPA Principle

PKDPA succeeds for 900 Traces

PKDPA Results - TripleDES



Bit	Key Returned
1	6
2	19
3	45
4	30

DPA Keys

3D Differential Plot

SBOX – 4 **BIT – 3** **TRACE COUNT = 3,200**

?

Correct Key = 19

Cannot be Distinguished

Classical DPA fails for 3,200 Traces

PKDPA Results - TripleDES

SBOX – 4

TRACE COUNT = 3,200

Window-Size = 5

Most Frequent Key = 19

Frequency = 3

B1	B2	B3	B4
6	19	45	30
32	24	22	45
5	20	29	19
38	7	38	25
19	54	50	3

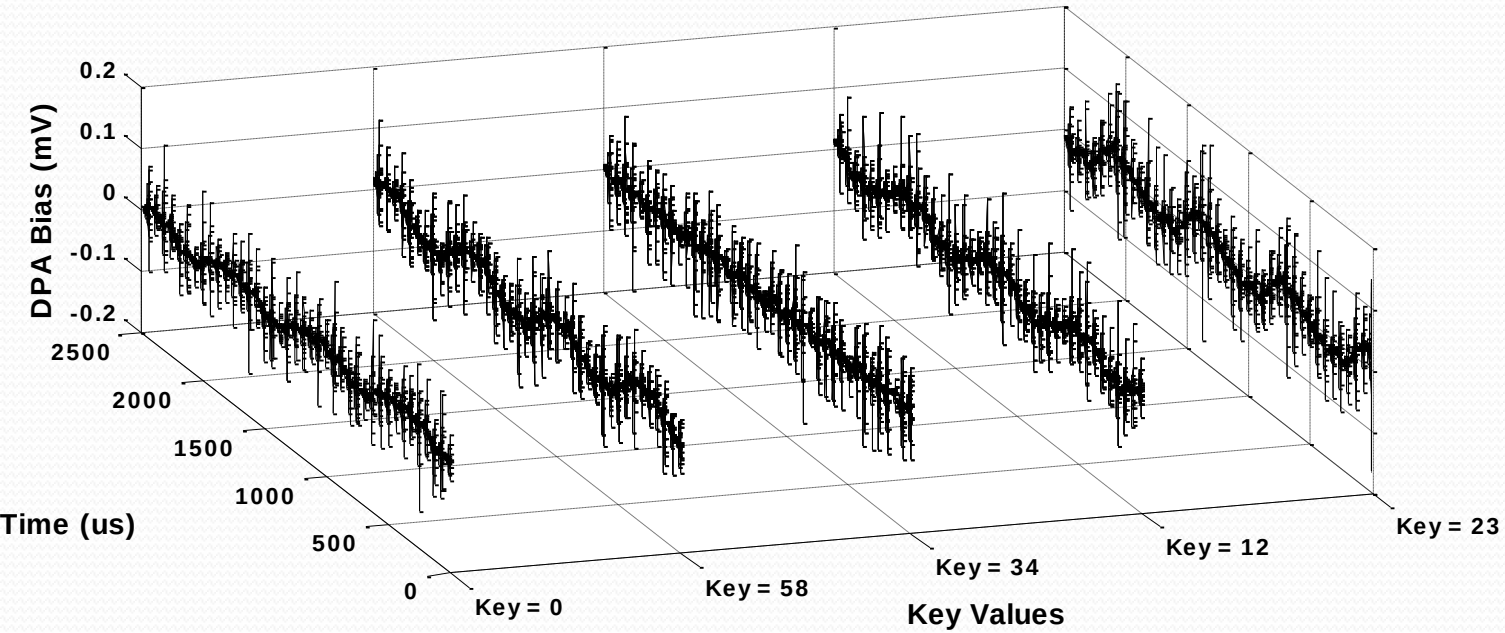
Probable Key Matrix

Correct Key = 19

By PKDPA Principle

PKDPA succeeds for 3,200 Traces

PKDPA Results - AES



3D Differential Plot

SBOX – 15 BIT – 4 TRACE COUNT = 2,500

Bit	Key Returned
1	11
2	12
3	244
4	124
5	86
6	244
7	242
8	147

DPA Keys

?

Correct Key = 12

Classical DPA fails for 2,500 Traces

Cannot be Distinguished

PKDPA Results - AES

SBOX – 15

TRACE COUNT = 2,500

Window-Size = 10

Most Frequent Key = 12

Frequency = 4

B1	B2	B3	B4	B5	B6	B7	B8
11	12	244	124	86	244	242	147
23	86	76	95	64	38	143	107
61	227	58	244	69	67	128	42
133	19	217	12	210	17	44	88
197	161	164	142	127	124	174	137
35	38	69	139	60	103	41	12
22	191	52	117	218	61	36	122
220	164	123	74	193	196	68	125
238	105	12	73	18	82	133	159
178	26	193	147	89	78	202	96

Probable Key Matrix

PKDPA succeeds for 2,500 Traces

Correct Key = 12
By PKDPA Principle

Masked AES

- Most popular counter-measure against DPA attacks is masking
- Using PKDPA attacked AES implementation protected by masking technique by Oswald *et al.* [18]
- The bit model of power analysis was used to attack masked Sbox during the encryption of 13, 000 random plaintexts.
- A randomly generated mask was used for each of these encryptions

A Related Result

- Mangard *et al.* attacked the same masked AES implemented on an ASIC in [19]

chip. It has turned out that the attacks on the unmasked and the masked implementations lead to similar results. DPA attacks using simple power models, such as the Hamming weight or the value of a bit, were in general not successful.

- Power models based on simulation required 30,000 traces to attack the masked design

In this article, we [illegible] possible to mount [illegible] DPA attacks on masked ASIC implementations of AES. The attacks we have presented are based on power models that have been derived from simulations of back-annotated netlists.

However, an attacker usually does not have easy access to the back-annotated netlist of a product. This is why we are currently closely analyzing the characchannel leakage [illegible] Our goal is [illegible]

PKDPA Results – Masked AES

SBOX – 15

TRACE COUNT = 13,000

Window-Size = 15

Most Frequent Key = 12

Frequency = 6

PKDPA is based on "bit" model

B1	B2	B3	B4	B5	B6	B7	B8
111	163	196	12	129	42	130	125
12	215	239	147	12	76	16	0
216	54	62	169	113	218	139	143
94	22	249	200	13	37	98	126
139	181	86	87	244	58	10	2
172	50	113	67	30	22	176	225
160	226	169	97	218	97	116	73
57	194	240	250	183	117	146	178
11	137	76	81	145	230	244	200
110	42	137	201	35	173	158	90
44	162	102	180	182	138	44	83
233	72	176	0	216	227	245	22
237	12	55	142	67	56	12	145
208	253	12	94	93	155	236	13
103	73	19	84	203	191	147	111

Probable Key Matrix

PKDPA succeeds for 13,000 Traces

Correct Key = 12
By PKDPA Principle

A Comparative Study

Cipher	Scheme	# of Traces	Validation	Attack Model / Knowledge Required
DES	[3]	Not reported	Simulation	
	[2]	2048	Micro-controller	
	Classical DPA [21]	10,000	Virtex XCV800	Bit Model
	Classical DPA (Ours)	4,000	Spartan-3 XC3S400	Bit Model
	PKDPA	**900**		
3-DES	Classical DPA(Ours) (No reported results)	10,000	Spartan-3 XC3S400	Bit Model
	PKDPA	**3,200**		
AES	Classical DPA [15]	6,532	ASIC	Hamming Distance Model
	Classical DPA (Ours)	15,000	Spartan-3 XC3S400	Bit Model
	[13]	1,000	Virtex XCV800	Correlation Matrix - Extensive Knowledge about target registers, architecture etc, required
	PKDPA	**2,500**	Spartan-3 XC3S400	Bit Model - Requires little knowledge about target device
Masked AES [18]	DPA [19]	Not possible with 1,000,000 traces	ASIC	Bit Model
		30,000	ASIC	Power model derived from simulation - Requires knowledge of target netlist
	PKDPA	**13,000**	Spartan-3 XC3S400	Bit Model

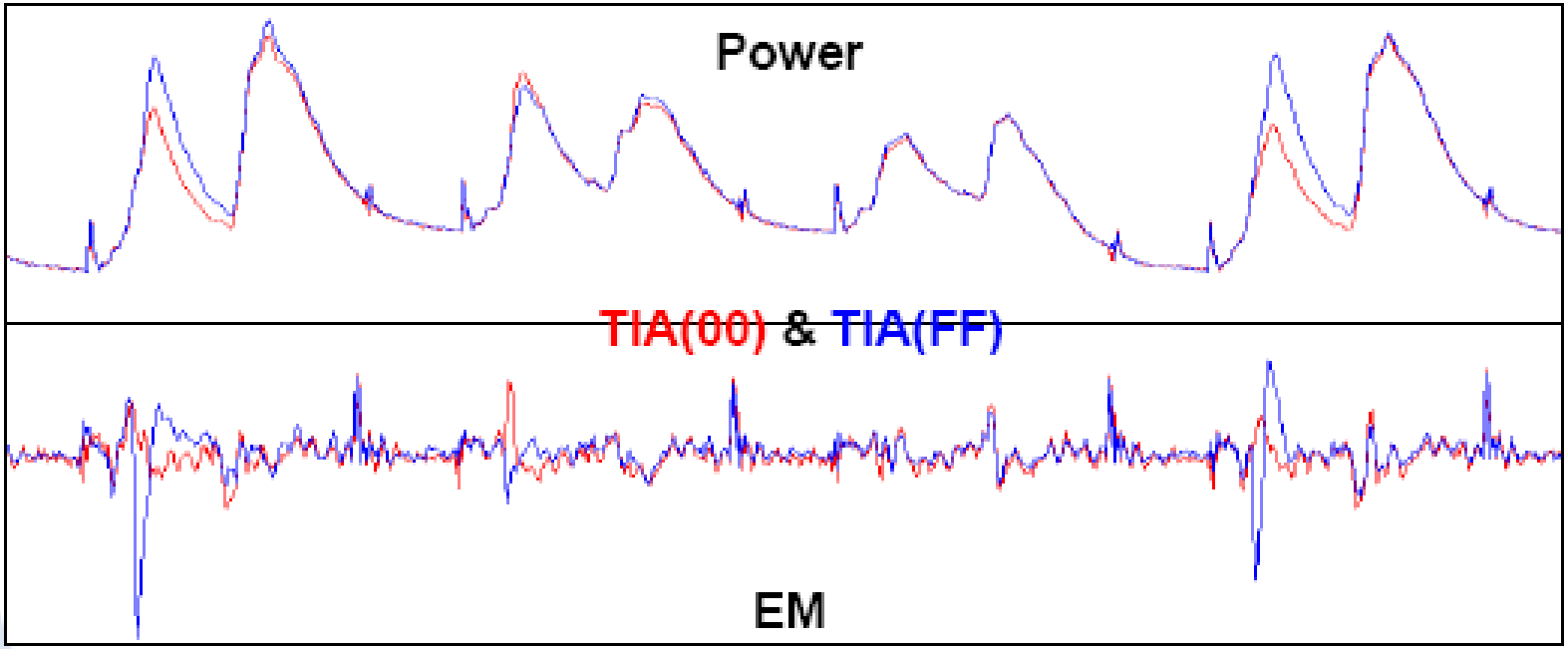
Outline

- Introduction
- Type of Side Channel Attacks
- Power attacks
- **Electromagnetic Radiation attacks**
- Timing attacks
- Fault attacks
- Scan Attack

Electromagnetic power analysis

EMA signal

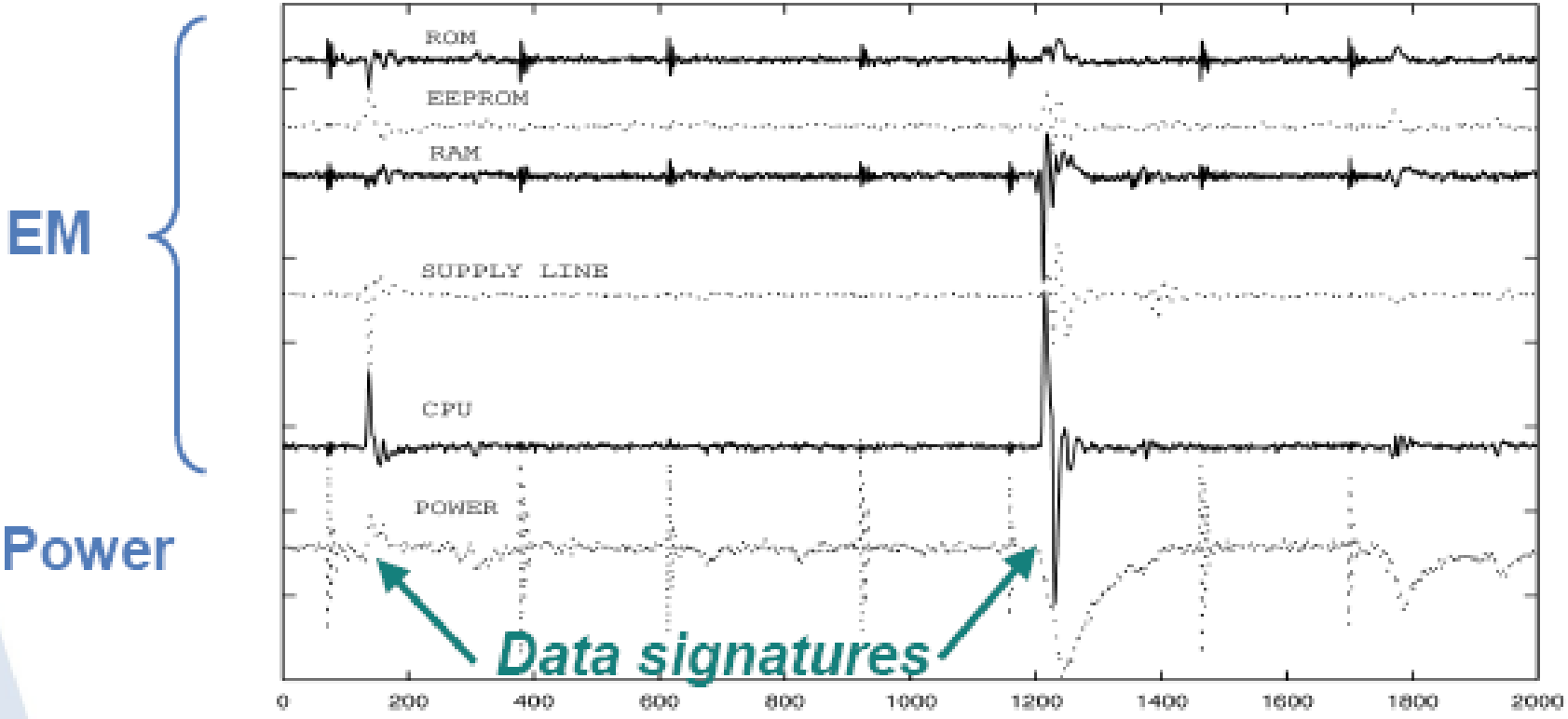
- Raw signals (TIA : transfer into accumulator instruction)
  - Power is less noisy
  - But EM signatures are sharper !



Spatial positioning

- EM signals versus XY probe position

Differential traces between (00h ⊕ 00h) and (FFh ⊕ 00h) picked up at different locations



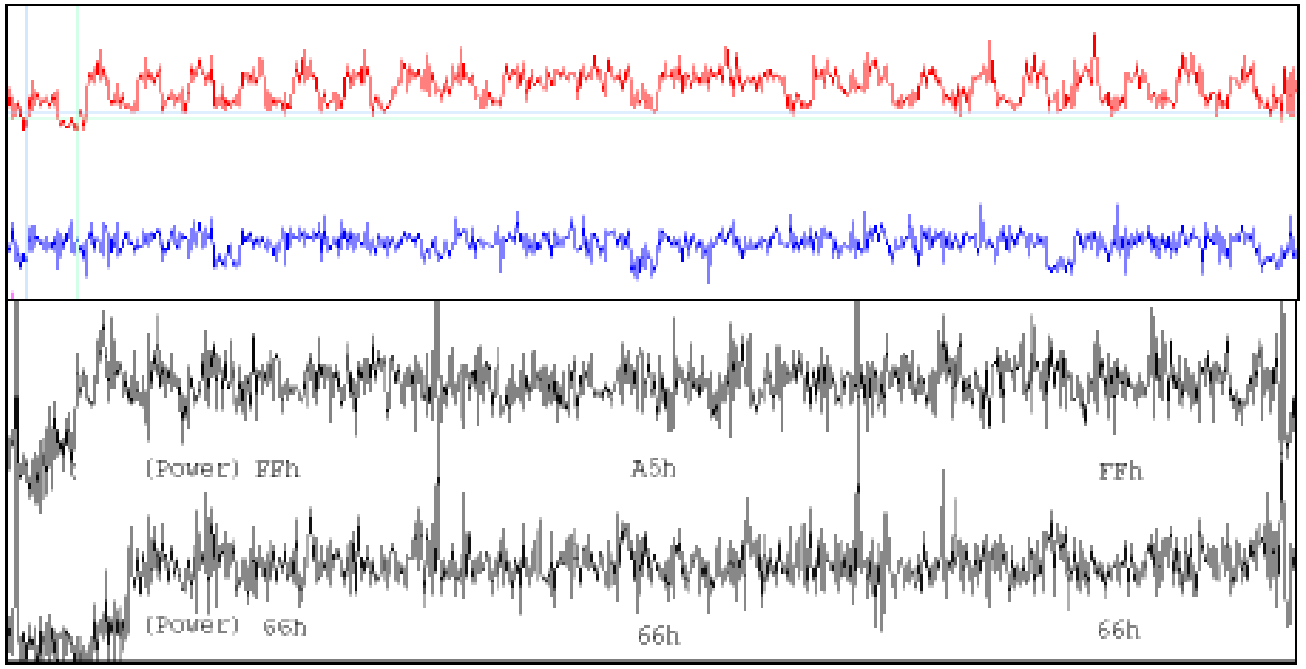
Example: SEMA on RSA

- SEMA/SPA exploit larger scale patterns (single trace)
- Decapsulation (no statistical improvement for S/N)

2 exponentiations involving 3 bytes of the private key : FFA5FFh and 666666h (same message and modulus).

EM patterns :

possible SEMA



Power

(no pattern : no SPA)

Outline

- Introduction
- Type of Side Channel Attacks
- Power attacks
- Electromagnetic Radiation attacks
- Timing attacks
- **Fault attacks**
- Scan Attack

Timing attacks

- Running time of a cryptographic algorithm can be used as an information channel
- The idea was proposed by Kocher, Crypto'96

Outline of Kocher's Attack

- Idea: guess some bits of the exponent and predict how long decryption will take
- If guess is correct, will observe correlation; if incorrect, then prediction will look random
  - This is a signal detection problem, where signal is timing variation due to guessed exponent bits
  - The more bits you already know, the stronger the signal, thus easier to detect (error-correction property)
- Start by guessing a few top bits, look at correlations for each guess, pick the most promising candidate and continue

Outline

- Introduction
- Type of Side Channel Attacks
- Power attacks
- Electromagnetic Radiation attacks
- Timing attacks
- **Fault attacks**
- Scan Attack
- Analysis of Hash Function

Fault Attack

- Fault Attack on AES
- Fault Analysis of eStream winners
  - Grain Family of ciphers
  - MICKEY Family.

Fault injection techniques

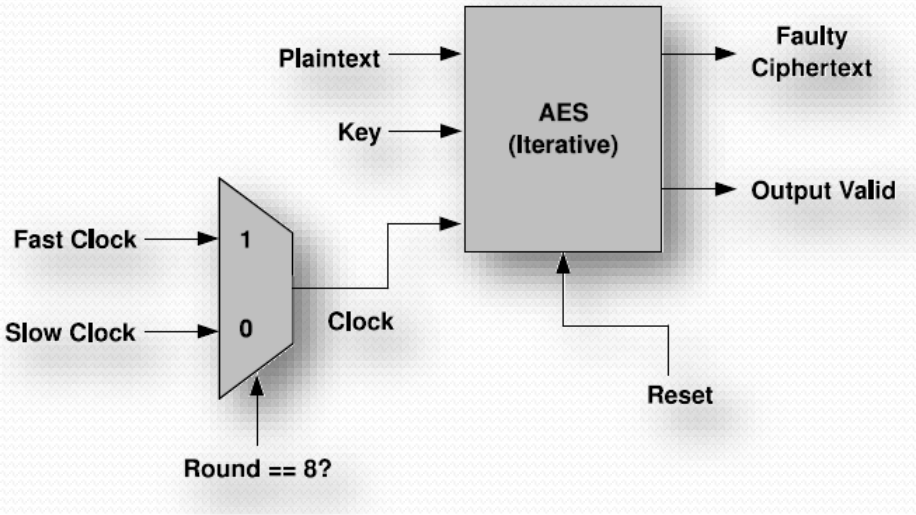
- Transient (provisional) and permanent (destructive) faults
  - Variations to supply voltage
  - Variations in the external clock
  - Temperature
  - White light
  - Laser light
  - X-rays and ion beams
  - Electromagnetic flux

Fault Attack Principle

- Fault attacks are one of the most efficient side channel attacks known till date.
- In this kind of attack, faults are injected during cipher operations.
- The attacker then analyzes the fault free and faulty cipher-texts or key-streams to deduce partial or full value of the secret key.
- The literature shows that both the block ciphers and stream ciphers are vulnerable against fault attack.
- Methods like clock glitch, laser shots have been shown to be a practical way of inducing faults in a crypto-system.

Fault Induction by Clock Glitch

- Non – invasive
- Strategy : Switch system clock
  - Determine max clock frequency
  - Use a faster clock to violate critical path
- Results in setup/hold violation = fault
- Easy/economical to mount
- Can be used to control
  # of faults
- Has been used to inject
  fault in AES



Fault Injection by Scan -Chain

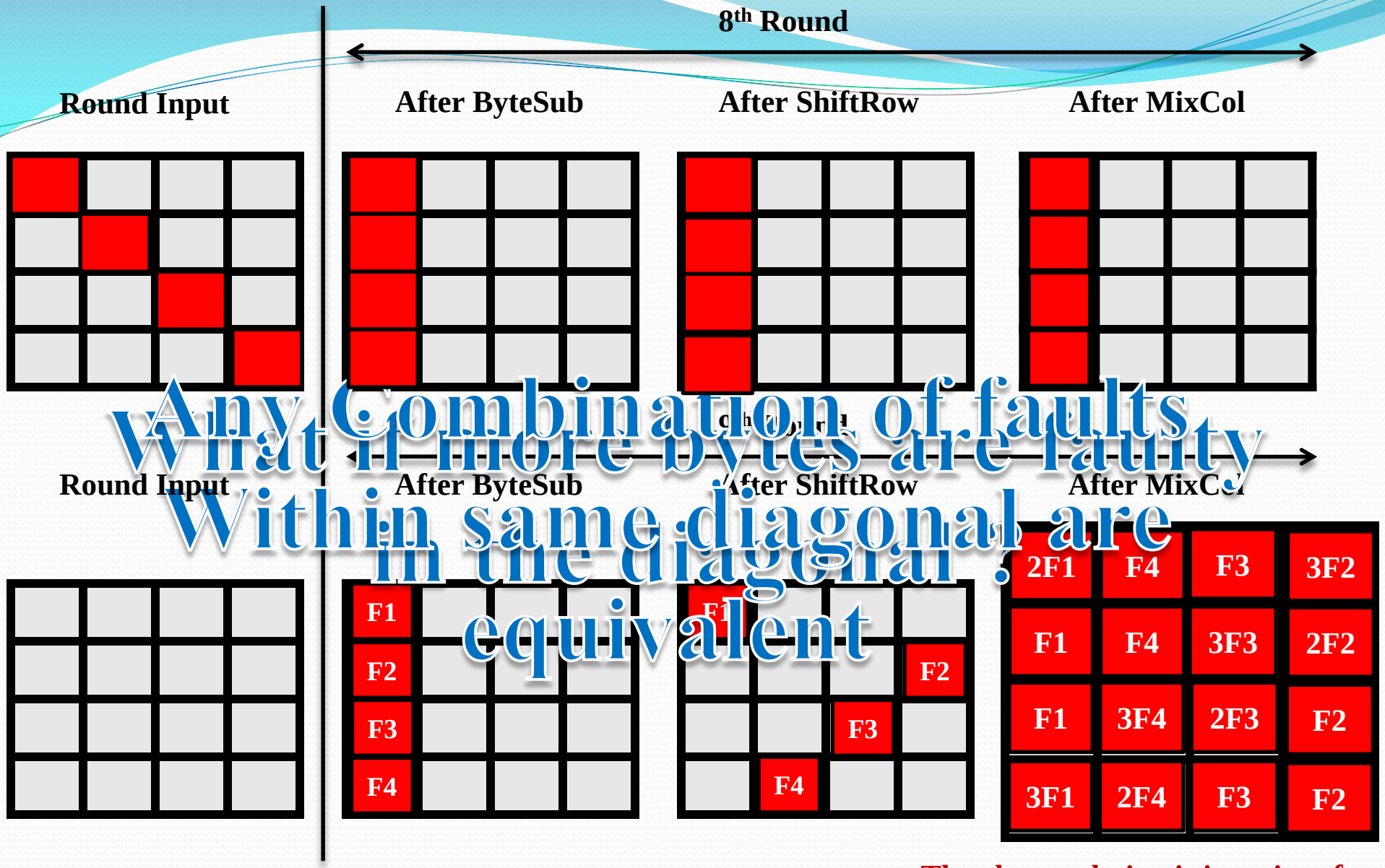
- Non – invasive
- Strategy : Use scan-chain to insert fault
  - Scan out intermediate state in test-mode
  - Flip-bits in scan-in pattern
  - Run in normal mode = fault in positions of bit-flips
- Easy/economical to mount
- Highest form of controllability over fault
- May not work if some special scan-chain protection schemes are in place
- Often protection schemes are avoided to reduce overhead
  - Very useful fault injection in such cases

Fault Attack on AES

Our Attack

Dhiman Saha, D. Mukhopadhyay and D. Roy Chowdhury, A Diagonal Fault Attack on the Advanced Encryption Standard,

IACR Cryptology ePrint Archive, 2009



What if more bytes are faulty

Any Combination of faults Within same diagonal are equivalent

The above relation is invariant for any fault injected within a diagonal

Fault Propagation if Diagonal D0 is faulty

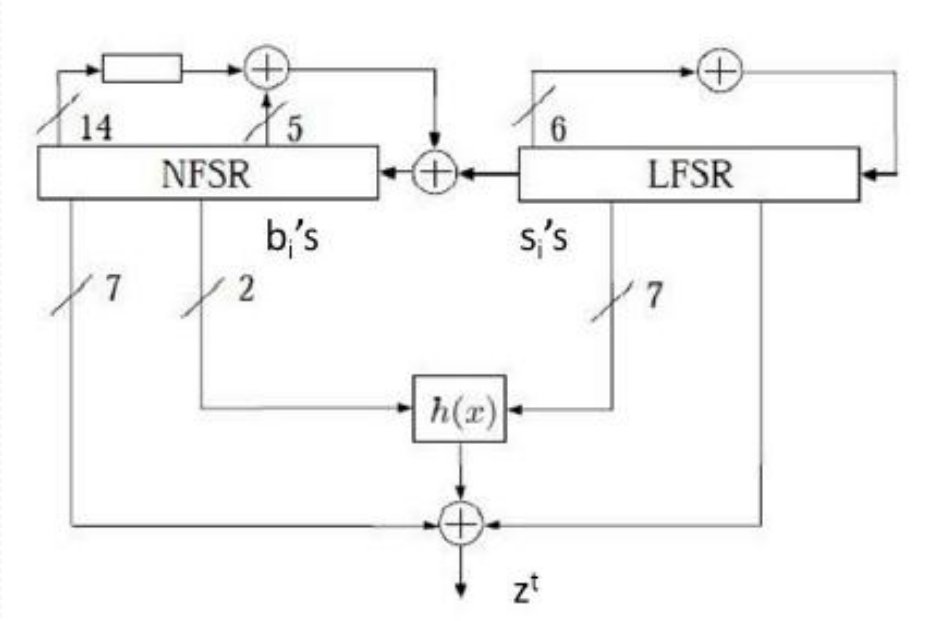
Fault Analysis of Grain

Attacks on Grain

- Grain-128 proposed by Martin Hell et al. is one of the three hardware based ciphers enlisted in the eStream portfolio.
- Cryptanalysis based on dynamic cube attack and one fault based attack (LFSR as a fault target) are the only known weaknesses of Grain-128.
- We inject faults in the NFSR and show that secret key of the cipher can be recovered.
- Complexity of our attack is better than the previous one.

Specification of Grain-128

- Grain-128 stream cipher consists of three main building blocks, namely, an NFSR, an LFSR and a nonlinear filter function, h(x).



Specification of Grain-128

- NFSR bits are $b_i, b_{i+1}, \ldots, b_{i+127}$ and LFSR bits are $s_i, s_{i+1}, \ldots, s_{i+127}$.
- The update function of the LFSR is given by,

$$s_{i+128} = s_i + s_{i+7} + s_{i+38} + s_{i+70} + s_{i+81} + s_{i+96}$$

- The NFSR is updated by,

$$b_{i+128} = s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} + b_{i+3}b_{i+67} + b_{i+11}b_{i+13} + b_{i+17}b_{i+18} + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + b_{i+68}b_{i+84}$$

- The nonlinear filter function h(.) is given by,

$$h = b_{i+12}s_{i+8} + s_{i+13}s_{i+20} + b_{i+95}s_{i+42} + s_{i+60}s_{i+79} + b_{i+12}b_{i+95}s_{i+95}$$

Specification of Grain-128

- The output bit z is defined as,

$$z_i = b_{i+2} + b_{i+15} + b_{i+36} + b_{i+45} + b_{i+64} + b_{i+73} + b_{i+89} + h + s_{i+93}$$

- An initialization phase is carried out before the cipher generates any keystream.
- During initialization the cipher is run for 256 rounds without producing any keystreams and the output bit is fed back to both LFSR and NFSR.
- The 128 bit key, $k = (k_0, k_1, \ldots, k_{127})$ and the 96 bit initialization vector $IV = (IV_0, IV_1, \ldots, IV_{95})$ are loaded in the NFSR and the LFSR respectively as, $b_i = k_i$; $0 <= i <= 127$ and $s_i = IV_i$; $0 <= i <= 95$, rest of the LFSR bits, $(s_{96}, s_{97}, \ldots, s_{127})$ are loaded with 1.

Fault Analysis Model

- The attacker is able to induce faults at random positions of the NFSR of the Grain-128 implementation (hardware or software).
- A fault is a bit-flip in the internal register.
- The fault affects exactly one bit of the NFSR at any cycle of operation.
- A fault to an NFSR bit can be reproduced at any cycle of operation, once, it is created.
- The attacker is able to determine and control the cycles of operation of the implementation, i.e., the timing of the implementation is under control of the attacker.
- The attacker can reset the implementation to its original state.
- The attacker can run the implementation with different IV, without changing the key.

Overview of the Attack

- The following steps are carried out in the attack:
  - Determining Fault Location in NFSR
  - Pre-computation of Fault Traces
  - Determining NFSR bits
  - Determining LFSR bits
  - Inversion of states to obtain key

Complexity of Our Attack

Our Attack:

1. Sandip Karmakar and Dipanwita Roy Chowdhury, **Fault analysis of Grain-128 by targeting NFSR**, AFRICACRYPT '11, 298–315, 2011.
2. Sandip Karmakar, Dipanwita Roy Chowdhury: **Fault Analysis of Grain Family of Stream Ciphers.** IACR Cryptology ePrint Archive 2014: 261 (2014)

- Experimentally it is seen that, 56 faults on an average are required to obtain full internal NFSR bits of the cipher.
- Maximum 256 faults are needed to obtain full LFSR state of the cipher.
- The time complexity of the attack is $O(2^{21})$.

Complexity

Ciphers	*Grain v1*	*Grain-128*	*Grain-128a*
Online Cost	*(#Faults, Cost)*	*(#Faults, Cost)*	*(#Faults, Cost)*
Location	(0,11)	(0,14)	(0,14)
Pre-computation	$(0, 2^{12})$	$(0, 2^{15})$	$(0, 2^{15})$
NFSR	$(70, 2^{10})$	(56, 128)	(128, 128)
LFSR	$(80, < 2^{21})$	$(256, 2^{21})$	$(256, 2^{21})$
Inversion	$(0, r(T))$	$(0, r(T))$	$(0, r(T))$
Total	$(150, < 2^{21})$	$(312, 2^{21})$	$(384, 2^{21})$

Comparison

Paper	Cipher	Fault Model	Faults
[4]	Grain-128	Similar	1587
[7]	Grain-128a	Similar	1831
Current	Grain v1	-	150
Current	Grain-128	-	312
Current	Grain-128a	-	384

4. Alexandre Berzati et. al. Fault Analysis of Grain-128. Hardware Oriented Security and Trust, IEEE International Workshop on, 0:7-14, 2009.
7. Itai Dinur and Adi Shamir. Breaking Grain-128 with Dynamic Cube Attacks. Cryptology ePrint Archive: Report 2010/570.

Recently the following work reports an attack on Grain ciphers with 10 faults or less, but it uses rekeying and requirements of keystreams are higher.

Santanu Sarkar et. al. , Differential Fault Attack Against Grain Family with very few Faults and Minimal Assumptions, Cryptology ePrint Archive , 2013/494. http://eprint.iacr.org.

Differential Fault Analysis of MICKEY

Attacks on MICKEY

- MICKEY-128 2.0 is one of the three hardware based ciphers enlisted in the eStream portfolio.
- Two fault based attacks are the only known weaknesses of MICKEY.
- [16] targets Mickey-128 and breaks the system with 640 faults with a similar fault model as used here.
- While [17] targets Mickey v1 and breaks the system with $2^{16.7}$ faults, with a different fault model than used here.
- We break Mickey-128 2.0, the strongest version of the family with only 480 faults and few minutes of computation.

Specification of Mickey-128 2.0

- Mickey-128 2.0 consists of two non-linear registers, R and S of 160 bits each, i.e., R0, … R159 and S0, … S159.
- A key of 128 bits and an IV up to 128 bits is used in the cipher.
- The key, IV is initialized using algorithm INIT() described later.
- The key stream is generated following procedure KeyStream(), which in turn clocks the registers and generates key stream per cycle following equation, z=R0+S0.
- A detailed description may be found in eStream .

Fault Analysis Model

- The attacker is able to induce faults at random positions of the R or S registers of the Mickey-128 2.0 implementation (hardware or software).
- The fault affects exactly one bit of register at any cycle of operation.
- A fault to a register bit can be reproduced at any cycle of operation, once, it is created.
- The attacker is able to determine and control the cycles of operation of the implementation, i.e., the timing of the implementation is under control of the attacker.
- The attacker can reset the implementation to its original state.
- The attacker can run the implementation with different IV, without changing the key.

Overview of the Attack

- The following steps are carried out in the attack:
  - Determining Fault Location
  - Determining R bits
  - Determining S bits

Determining the R Bits

- Number of faults required:

Register Bit	Required Values	Known	Unknown	#Faults
$r_0(i)$	-	-	-	1
$r_{159}(i)$	$r_0(i+1)$, $r_0(i)$, $s_{54}(i)+r_{106}(i)$	$r_0(i)$	$r_0(i+1)$, $s_{54}(i)+r_{106}(i)$	2
$r_{158}(i)$	$r_{159}(i+1)$, $r_{159}(i)$, $s_{54}(i)+r_{106}(i)$	$r_{159}(i)$, $r_{159}(i)$, $s_{54}(i)+r_{106}(i)$	$r_{159}(i+1)$	2
$r_k(i)$	$r_{k+1}(i+1)$, $r_{k+1}(i)$, $r_{159}(i)$, $s_{54}(i)+r_{106}(i)$	$r_{k+1}(i)$, $r_{159}(i)$, $s_{54}(i)+r_{106}(i)$	$r_{k+1}(i+1)$	2 $(k=157,\ldots,1)$

Determining S Bits

- Number of faults required:

Register Bit	Required Values	Known	Unknown	#Faults
$s_0(i)$	-	-	-	-
$s_{159}(i)$	-	-	-	1
$s_{158}(i)$	$s_{159}(i+1)$, $s_{159}(i)$, $(s_{106}(i-1)+r_{53}(i-1))$	$s_{159}(i+1)$, $s_{159}(i)$	$(s_{106}(i-1)$ $+r_{53}(i-1))$	1
$s_k(i)$	$s_{k+2}(i+1)$, $s_{k+1}(i)$, $s_k(i)$, $s_{159}(i)$, $(s_{106}(i-1)+r_{53}(i-1))$	$s_{k+1}(i)$, $s_k(i)$, $s_{159}(i)$, $(s_{106}(i-1)+r_{53}(i-1))$	$s_{k+2}(i+1)$	1 $(k=157,\ldots,1)$

Complexity of Our Attack

Our Attack:

1. Sandip Karmakar, Dipanwita Roy Chowdhury, **Differential Fault Analysis of MICKEY-128 2.0**, FDTC 2013,Santa Barbara, CA, USA, August, 2013
2. Sandip Karmakar, Dipanwita Roy Chowdhury: **Differential Fault Analysis of MICKEY Family of Stream Ciphers.** IACR Cryptology ePrint Archive 2014: 262 (2014)

- 480 faults are needed to obtain full state of the cipher. 320 faults are required to determine R register and 160 faults are required to determine S register.
- 480 pairs of faulty/fault-free keystreams are required to break the system.
- The time complexity of the attack in computation is negligible.

Comparison

Attack	*Version*	*Fault Model*	*State Size*	*#Faults*	*#Keystream*
[16]	MICKEY-128	Similar	256	640	960
[17]	MICKEY 2.0	Different	200	$2^{16.7}$	-
This Work	MICKEY-128 2.0	-	320	480	960
This Work	MICKEY v1	-	80	240	480
This Work	MICKEY-128	-	256	384	768
This Work	MICKEY 2.0	-	200	300	600

This Work– Sandip Karmakar and Dipanwta Roy Chowdhury, Differential Fault Attack of MICKEY 128 2.0 IEEE FDTC 2013, Santa Barbara, CA, USA, 2013

16. ZHANG Peng,HU Yupu,ZHANG Tao. Fault attack of MICKEY-128. 2013Electronic Science and Technology 2011,24(6) 80-, 2011

17. Subhadeep Banik and Subhamoy Maitra. A Differential Fault Attack on MICKEY 2.0. Cryptology ePrint Archive, Report 2013/029, 2013

Outline

- Introduction
- Type of Side Channel Attacks
- Power attacks
- Electromagnetic Radiation attacks
- Timing attacks
- Fault attacks
- **Scan Attack**

Scan Based Side Channel Attack against eStream Winners

Our Attack

- D. Mukhopadhyay, S. Banerjee, D. RoyChowdhury, and B. B. Bhattacharya. **Cryptoscan: A secured scan chain architecture**. ATS 2005: Proceedings of the 14th Asian Test Symposium on Asian Test Symposium, Washington, DC, USA, pages 348-353, 2005.

- G. Sengar, D. Mukhopadhyay, and D. Roy Chowhury, **Secured flipped scan-chain model for crypto-architecture**. IEEE Trans. on CAD of Integrated Circuits and Systems, 26(11):2080{2084, 2007.

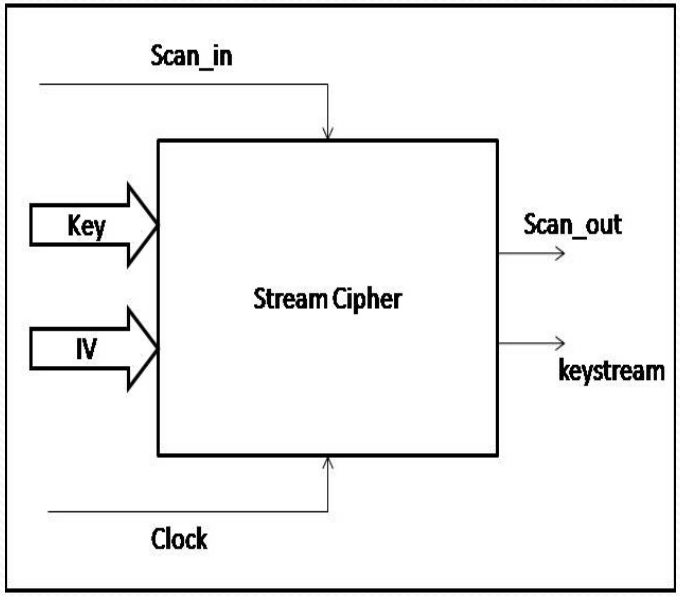
- M Agrawal, S Karmakar, D. Saha, and D. Mukhopadhayay. **Scan based side channel attacks on stream ciphers and their counter-measures**., Progress in Cryptology - INDOCRYPT 2008, 5365/2008:226{238, 2008.
- Sandip Karmakar, Dipanwita Roy Chowdhury: **A Generic Scan Attack on Hardware based eStream Winners.** IACR Cryptology ePrint Archive 2014

- Sandip Karmakar, Dipanwita Roy Chowdhury: **Scan-based side channel attack on stream ciphers and its prevention.** Journal of Cryptogr Eng DOI 10.1007/s13389-017-0178-1, 2017

Principle of Scan Attack

- DFT feature scan-chain though is indispensable for testing circuits, opens a side channel for cryptanalysis of ciphers.
- Proposed a generic scan attack on stream ciphers
- Demonstrated the attack on eStream hardware based winners, Trivium, MICKEY-128 2.0 and Grain-128

Scan Attack Model



- Adversary can input key and IV into the system
- Adversary can scan_in desired input
- After normal mode of operation for few cycles, adversary can scan out internal state through scan_out line

CASE STUDY - TRIVIUM

- m = 108 cycles can distinguish among states
- Key = 0 and IV =0 are set and the system is run in both simulation and online phases
- Therefore at this complexity internal state of Trivium is known
- Cipher operation being invertible we can get back the internal key of the cipher
- Complexity of the attack is, O(ts *108 + to *108 + 288log288), where ts is per clock simulation time, to is total scan out time after a cycle of operation

CASE STUDY – GRAIN-128

- Key =1 and IV = 1 is set
- After m = 167 the uniqueness of states is obtained
- Therefore at this complexity internal state of Grain-128 is known
- Cipher operation being invertible we can get back the internal key of the cipher
- Complexity of the attack is, O(ts *167 + to *167 + 256log256), where ts is per clock simulation time, to is total scan out time after a cycle of operation

CASE STUDY – MICKEY-128 2.0

- Key =1 and IV = 1 is set
- After m = 18 the uniqueness of states is obtained
- Therefore at this complexity of reset internal state of MICKEY-128 2.0 is known
- Cipher operation is not invertible so from known current state we can get back future keystreams
- Complexity of the attack is, O(ts *18 + to *18 + 320log320), where ts is per clock simulation time, to is total scan out time after a cycle of operation

References

1. Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential Power Analysis," in *Proceedings of Advances in Cryptology - CRYPTO 99, LNCS Volume 1666*. 1999, pp. 388–397, Springer-Verlag.
2. Régis Bevan and Erik Knudsen, "Ways to Enhance Differential Power Analysis," in *Proceedings of Information Security and Cryptology (ICISC 2002), LNCS Volume 2587*. 2002, pp. 327–342, Springer-Verlag.
3. Giacomo Boracchi and Luca Breveglieri, "A Study on the Efficiency of Differential Power Analysis on AES S-Box," *Technical Report*, vol. n 2007.17, January 25, 2007.
4. National Institute of Standards and Technology, "Data Encryption Standard," in *Federal Information Processing Standard 46-2*, http://www.itl.nist.gov/fipspubs/fip46-2.htm.
5. Thomas S. Messerges, Ezzat A. Dabbish, and Robert H. Sloan, "Examining Smart-Card Security under the Threat of Power Analysis Attacks," *IEEE Trans. Comput.*, vol. 51, no. 5, pp. 541–552, 2002.
6. Pengyuan Yu and Patrick Schaumont, "Secure FPGA circuits using controlled placement and routing," in *Proceedings of International Conference on Hardware Software Codesign (CODES+ISSS 2007)*. 2007, pp. 45–50, ACM.
7. Eric Brier, Christophe Clavier, and Francis Olivier, "Correlation Power Analysis with a Leakage Model," in *CHES*, 2004, pp. 16–29.
8. Thanh-Ha Le, Jessy Clédière, Cécile Canovas, Bruno Robisson, Christine Servière, and Jean-Louis Lacoume, "A Proposition for Correlation Power Analysis Enhancement," in *CHES*, 2006, pp. 174–186.
9. Toshihiro Katashita, Akashi Satoh, Takeshi Sugawara, Naofumi Homma, and Takafumi Aoki, "Enhanced Correlation Power Analysis Using Key Screening Technique," in *RECONFIG '08: Proceedings of the 2008 International Conference on Reconfigurable Computing and FPGAs*, Washington, DC, USA, 2008, pp. 403–408, IEEE Computer Society.
10. Huiyun Li, Keke Wu, Bo Peng, Yiwei Zhang, Xinjian Zheng, and Fengqi Yu, "Enhanced Correlation Power Analysis Attack on Smart Card," *Young Computer Scientists, International Conference for*, vol. 0, pp. 2143–2148, 2008.
11. Franois-Xavier Standaert, Loic van Oldeneel tot Oldenzeel, David Samyde, and Jean-Jacques Quisquater, "Differential Power Analysis of FPGAs : How Practical is the Attack?," in *Proceedings of Field-Programmable Logic and Applications (FPL 2003), LNCS Volume 2778*. 2003, pp. 701–711, Springer-Verlag.
12. Siddika Berna rs, Elisabeth Oswald, and Bart Preneel, "Power-analysis attacks on an FPGA first experimental results," in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems (CHES 2003), LNCS Volume 2779*. 2003, pp. 35–50, Springer-Verlag.

References

13. François-Xavier Standaert, Siddika Berna Örs, and Bart Preneel, "Power Analysis of an FPGA: Implementation of Rijndael: Is Pipelining a DPA Countermeasure?," in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), LNCS Volume 3156*. 2004, pp. 30–44, Springer-Verlag.
14. Franois-Xavier Standaert, Franois Mace, Eric Peeters, and Jean-Jacques Quisquater, "Updates on the Security of FPGAs Against Power Analysis Attacks," in *Proceedings of Reconfigurable Computing: Architectures and Applications, LNCS Volume 3985*. 2006, pp. 335–346, Springer-Verlag.
15. Stefan Mangard, Elisabeth Oswald, and Thomas Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
16. National Institute of Standards and Technology, "Triple-DES," in *Federal Information Processing Standard 46-3*, http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.
17. National Institute of Standards and Technology, "Advanced Encryption Standard," in *Federal Information Processing Standard 197*, http://csrc.nist.gov/archive/aes/frn-fips197.pdf.
18. Maria Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen, "A Side-Channel Analysis Resistant Description of the AES S-box," in *Proceedings of Fast Software Encryption (FSE 2005), LNCS Volume 3557*. 2005, pp. 413–423, Springer-Verlag.
19. Stefan Mangard, Norbert Pramstaller, and Maria Elisabeth Oswald, "Successfully Attacking Masked AES Hardware Implementations," in *Proceedings of Cryptographic Hardware and Embedded Systems (CHES 2005), LNCS Volume 3659*. 2005, pp. 157–171, Springer-Verlag.
20. Torben Hagerup and C. Rüb, "A guided tour of Chernoff bounds," in *Information Processing Letters, Volume 33, Issue 6*. 1990, pp. 305–308, Elsevier North-Holland, Inc.
21. Franois-Xavier Standaert, Siddika Berna Ors, Jean-Jacques Quisquater, and Bart Preneel, "Power Analysis Attacks against FPGA Implementations of the DES," in *Field Programmable Logic and Application (FPL 2004), LNCS Volume 3203*. 2004, pp. 84–94, Springer-Verlag.
22. Ccile Canovas and Jessy Cldire, "What do S-boxes Say in Differential Side Channel Attacks? Cryptology ePrint Archive," Tech. Rep., 2005.
23. James Alexander Muir, "Techniques of Side Channel Cryptanalysis," Tech. Rep., University of Waterloo, http://cs.smu.ca/ jamuir/papers/mmthesis-side-channel.pdf.

Referrences

- The eStream project. "http://www.ecrypt.eu.org/stream/".
- Mukesh Agrawal, Sandip Karmakar, Dhiman Saha, and Debdeep Mukhopadhayay., Scan Based Side Channel Attacks on Stream Ciphers and their Counter-measures., Progress in Cryptology - INDOCRYPT 2008, 5365/2008:226-238, 2008.
- Steve Babbage, Christophe De Canniere, Anne Canteaut, Carlos Cid,, Henri Gilbert, Thomas Johansson, Matthew Parker, Bart Preneel,, Vincent Rijmen, and Matthew Robshaw. The eStream Portfolio. "http://www.ecrypt.eu.org/stream/portfolio.pdf".
- Alexandre Berzati, Cecile Canovas, Guilhem Castagnos, Blandine Debraize, Louis Goubin, Aline Gouget, Pascal Paillier, and Stephanie Salgado. Fault Analysis of Grain-128. Hardware-Oriented Security and Trust, IEEE International Workshop on, 0:7-14, 2009.
- E. Biham and A. Shamir. Differential Fault Analysis of Secret Key Cryptosystems. Crypto 97, 1294 of LNCS:513-525, 1997.
- Johannes Blomer and Jean-Pierre Seifert. Fault Based Cryptanalysis of the Advanced Encryption Standard. Financial Cryptography, 2742:162-181, 2003.

Referrences

- Itai Dinur and Adi Shamir. Breaking Grain-128 with Dynamic Cube Attacks. Cryptology ePrint Archive: Report 2010/570.
- Martin Hell, Thomas Johansson, and Willi Meier. A Stream Cipher Proposal: Grain-128. eSTREAM, ECRYPT Stream Cipher Project, 2006.
- Jonathan J. Hoch and Adi Shamir. Fault Analysis of Stream Ciphers. CHES 2004, 3156/2004 of LNCS:1-20, 2004.
- Michal Hojsk and Bohuslav Rudolf. Differential Fault Analysis of Trivium. FAST SOFTWARE ENCRYPTION, 5086/2008 of LNCS:158-172, 2008.
- Aleksandar Kircanski and Amr M. Youssef. Differential Fault Analysis of Rabbit. SELECTED AREAS IN CRYPTOGRAPHY, 5867/2009:197-214, 2009.
- Sergei P. Skorobogatov. Optically Enhanced Position-locked Power Analysis. CHES 2006, 4249 of LNCS:61-75, 2006.
- Sergei P. Skorobogatov and Ross J. Anderson. Optical Fault Induction Attacks. CHES 2002, 2523 of LNCS:2-12, 2002.
- Thomas Wollinger and Christof Paar, **How Secure Are FPGAs in Cryptographic Applications? (Long Version)**
- Dhiman Saha, Debdeep Mukhopadhyay and Dipanwita Roy Chowdhury, "A Diagonal Fault Attack on the Advanced Encryption Standard", http://eprint.iacr.org/2009
- Fault Analysis of Grain 128 by Targetting NFSR, Sandip Karmakar, Dipanwita Roy Chowdhury, Africacrypt 2011, LNCS 6737, pp 298-315.

References

- Itai Dinur and Adi Shamir. Breaking Grain-128 with Dynamic Cube Attacks., Cryptology ePrint Archive: Report 2010/570.
- Martin Hell, Thomas Johansson, and Willi Meier. A Stream Cipher Proposal:, Grain-128. eSTREAM, ECRYPT Stream Cipher Project, 2006.
- Jonathan J. Hoch and Adi Shamir. Fault Analysis of Stream Ciphers. CHES 2004, 3156/2004 of LNCS:1-20, 2004.
- Michal Hojsk and Bohuslav Rudolf. Differential Fault Analysis of Trivium. FAST SOFTWARE ENCRYPTION, 5086/2008 of LNCS:158-172, 2008.
- Aleksandar Kircanski and Amr M. Youssef. Differential Fault Analysis of Rabbit. SELECTED AREAS IN CRYPTOGRAPHY, 5867/2009:197-214, 2009.
- Sergei P. Skorobogatov. Optically Enhanced Position-locked Power Analysis. CHES, 2006, 4249 of LNCS:61-75, 2006.
- Sergei P. Skorobogatov and Ross J. Anderson. Optical Fault Induction Attacks, CHES 2002, 2523 of LNCS:2-12, 2002.

References

- The eStream project. ”http://www.ecrypt.eu.org/stream/”.
- Mukesh Agrawal, Sandip Karmakar, Dhiman Saha, and Debdeep Mukhopadhayay. Scan Based Side Channel Attacks on Stream Ciphers and their Counter-measures. Progress in Cryptology - INDOCRYPT 2008, 5365/2008:226–238 , 2008.
- Steve Babbage, Christophe De Canniere, Anne Canteaut, Carlos Cid, Henri Gilbert, Thomas Johansson, Matthew Parker, Bart Preneel, Vincent Rijmen, and Matthew Robshaw. The eStream Portfolio.”http: //www.ecrypt.eu.org/stream/portfolio.pdf”.
- Alexandre Berzati, Cecile Canovas, Guilhem Castagnos,Blandine Debraize, Louis Goubin, Aline Gouget, Pascal Paillier, and Stephanie Salgado. Fault Analysis of Grain-128.Hardware-Oriented Security and Trust, IEEE InternationalWorkshop on, 0:7–14 , 2009.
- E. Biham and A. Shamir. Differential Fault Analysis of Secret Key Cryptosystems. Crypto 97, 1294 of LNCS:513–525,1997.
- Johannes Blomer and Jean-Pierre Seifert. Fault Based Cryptanalysis of the Advanced Encryption Standard. Financial Cryptography, 2742:162–181 , 2003.
- Itai Dinur and Adi Shamir. Breaking Grain-128 with DynamicCube Attacks. FSE 2011: 167–187.
- Martin Hell, Thomas Johansson, and Willi Meier. A Stream Cipher Proposal: Grain-128. eSTREAM, ECRYPT StreamCipher Project, 2006.

References

- Jonathan J. Hoch and Adi Shamir. Fault Analysis of Stream Ciphers. CHES 2004, 3156/2004 of LNCS:1–20 , 2004.
- Michal Hojsk and Bohuslav Rudolf. Differential Fault Analysis of Trivium. FAST SOFTWARE ENCRYPTION, 5086/2008 of LNCS:158–172 , 2008.
- Aleksandar Kircanski and Amr M. Youssef. Differential Fault Analysis of Rabbit. SELECTED AREAS IN CRYPTOGRAPHY,5867/2009:197–214 , 2009.
- Sandip Karmakar and Dipanwita Roy Chowdhury. Fault analysis of Grain-128 by targeting NFSR. AFRICACRYPT'11, 298–315, 2011.
- Sergei P. Skorobogatov. Optically Enhanced Position locked Power Analysis. CHES 2006, 4249 of LNCS:61–75 , 2006.
- Sergei P. Skorobogatov and Ross J. Anderson. Optical Fault Induction Attacks. CHES 2002, 2523 of LNCS:2–12 , 2002.
- Steve Babbage and Matthew Dodd. The stream cipher MICKEY 2.0. eSTREAM, ECRYPT Stream Cipher Project,2006
- ZHANG Peng,HU Yupu,ZHANG Tao. Fault attack of MICKEY-128. Electronic Science and Technology 2011,24(6) 80-, 2011
- S Banik and S Maitra. A Differential Fault Attack on MICKEY 2.0. Cryptology ePrint Archive, Report 2013/029, 2013
- *S. Kuila, D. Saha, M. Pal, and D. Roy Chowdhury,* Practical Distinguishers Against 6-Round Keccak-f Exploiting Self-Symmetry, AFRICACRYPT May 2014.