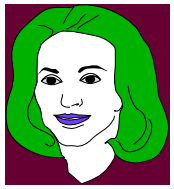
Side Channels in Cryptography

Debdeep Mukhopadhyay
Dept of Computer Sc and Engg
IIT Kharagpur

Topics of Discussion

- What is Side Channel Analysis?
- Power Based SCAs
- Scan Chain Based Attacks
- Fault Based Attacks

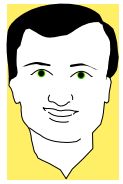
Goals of Conventional Cryptography



Alice

COMUNICATION CHANNEL

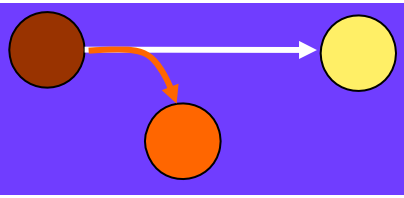
Mallory



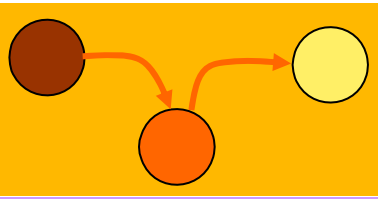
Bob

Policy
- Confidentiality
- Integrity
- Authenticity

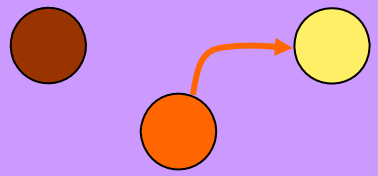
Security Attacks



Interception (Threat) → Confidentiality (Policy) → Encryption (Mechanism)



Modification (Threat) → Integrity (Policy) → Hash (Mechanism)



Fabrication (Threat) → Authenticity (Policy) → MAC (Mechanism)

Threat Model

Alice

Message

E

K_a

Communication Channel

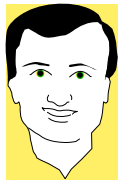
leaked Information

Mallory

D

K_b

Message



Bob

Side Channels in the real world

Through which a cryptographic module leaks information to its environment unintentionally

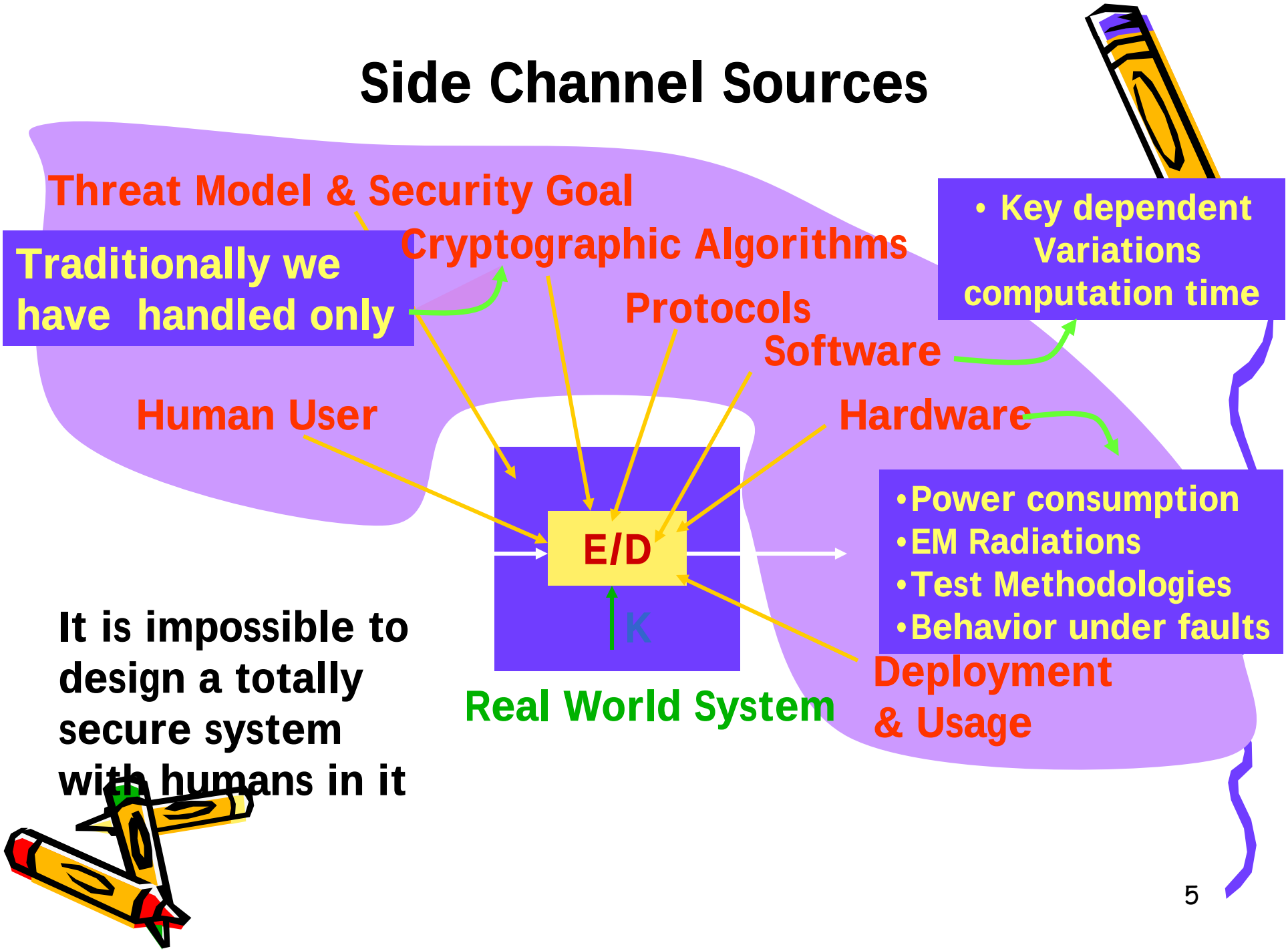
Assumptions

- Only Alice Knows K_a

- Only Bob Knows K_b

- Mallory has access to E, D and the Communication Channel but does not know the decryption key K_b

Side Channel Sources



It is impossible to design a totally secure system with humans in it

Strong cryptographic algorithms are just the beginning!

Terminal

Data input

Data output

Power supply

IC chip

00111…

Measure power consumption

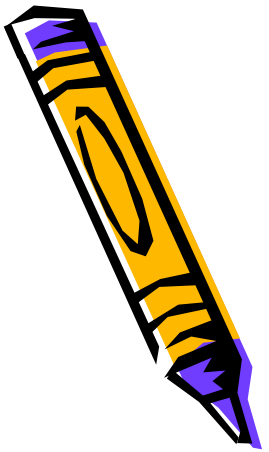
Guess secret information stored on IC chip memory

Power consumption

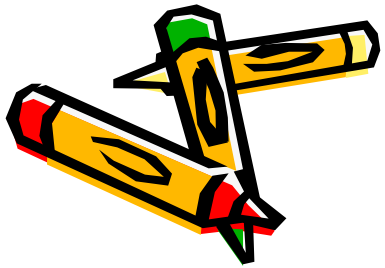
Secret information

0 0 1 1 1

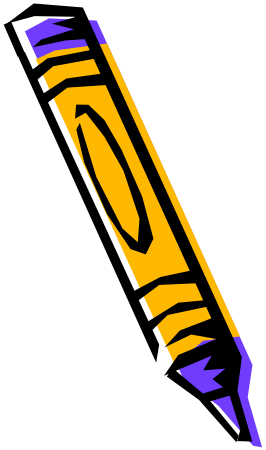
What are Side Channels?



- These are covert channels which leak information which the designers of cryptographic algorithms did not consider.
- Information is leaked because of the implementation:
  - optimization leads to information leakage
  - example: **an if-else statement in a programming language**



Possible Side Channels

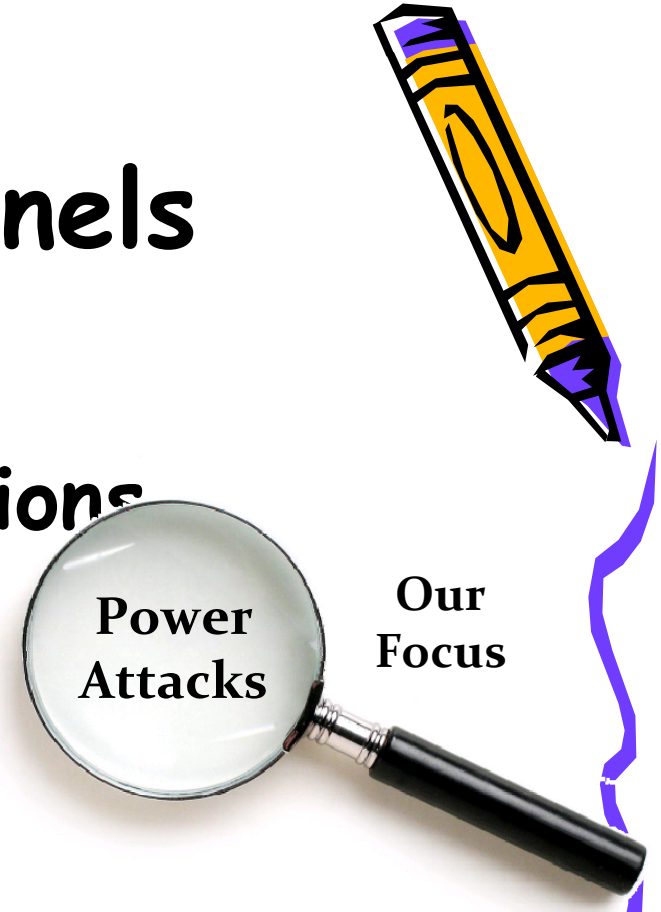


- Power
- Electro-Magnetic radiations
- Timing
  - Cache Timing Attacks
- Faults
- Testability Features in Hardwares

and may be many more…

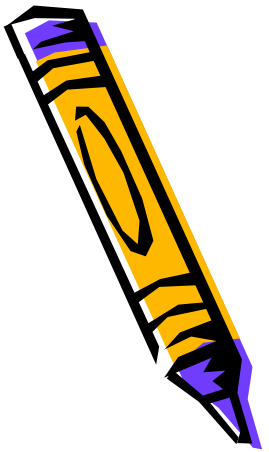
Possible Side Channels

- Power
- Electro-Magnetic radiations
- Timing
  - Cache Timing Attacks
- Faults
- Testability Features in Hardwares



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

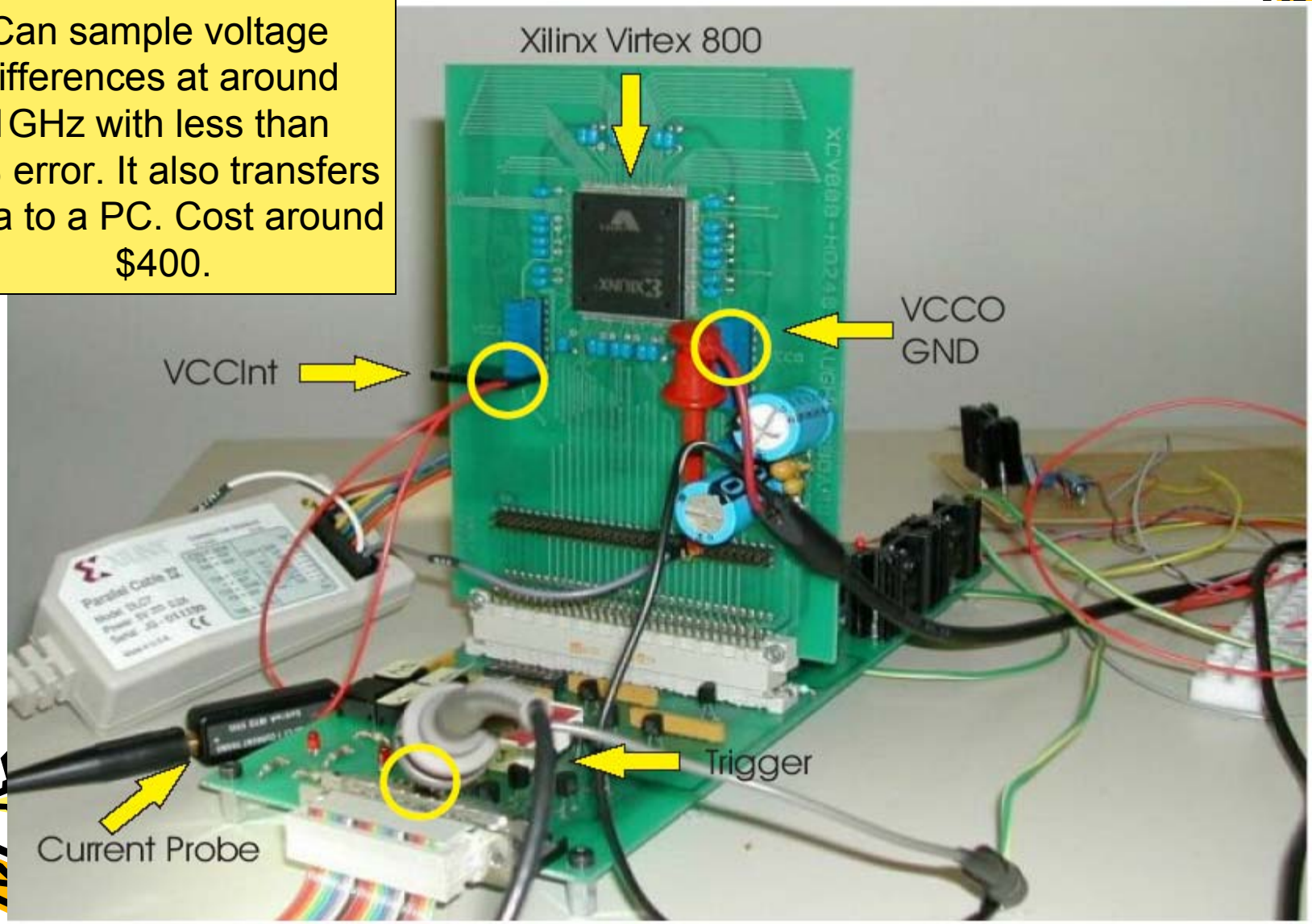
Power Attacks (PA)



- During the last few years 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

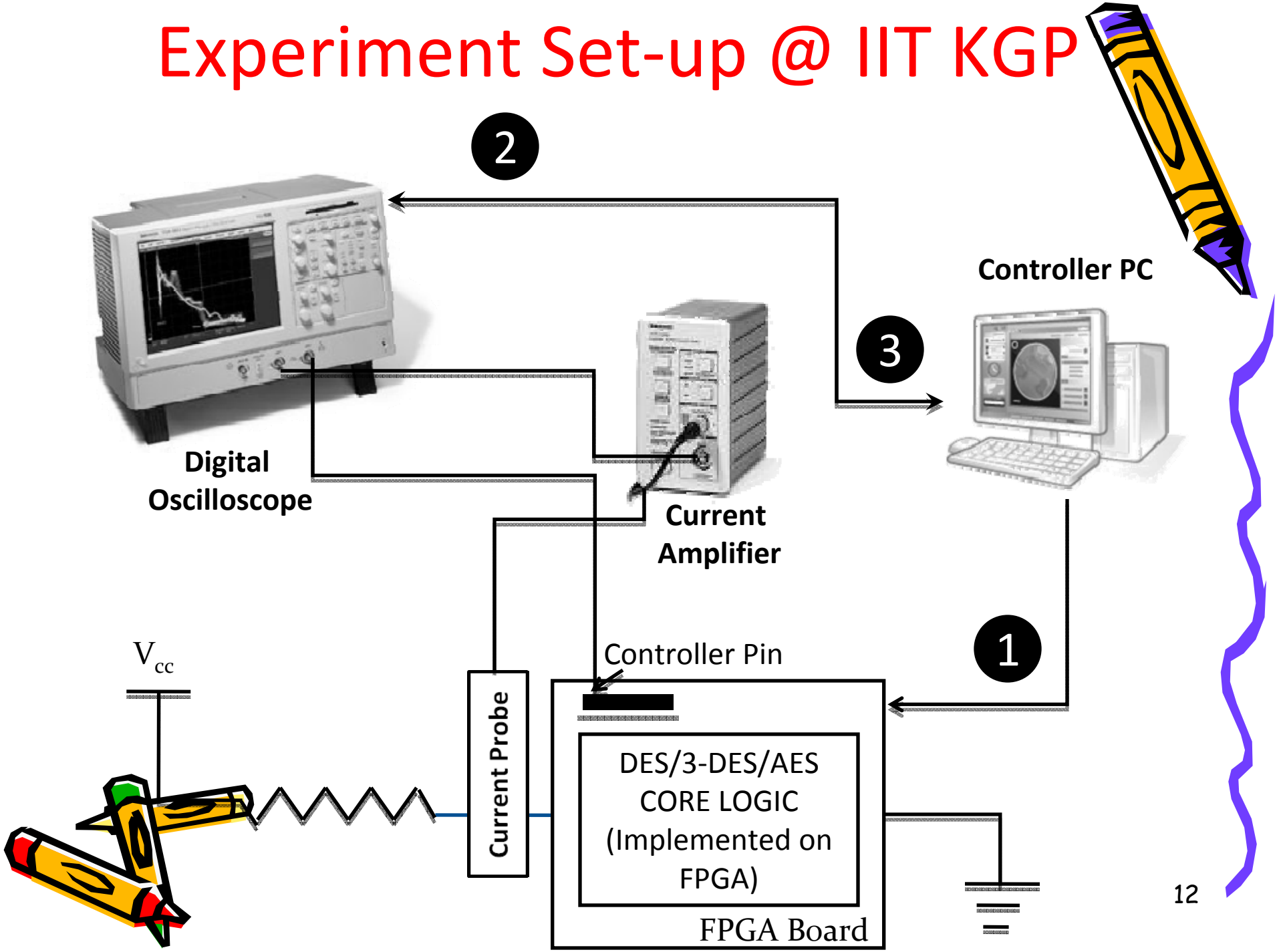
Lab Set Up for Power

Can sample voltage differences at around 1GHz with less than 1% error. It also transfers Data to a PC. Cost around $400.



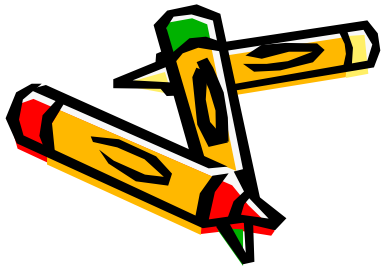
Courtesy: Side-Channel Analysis Lab, *Graz University of Technology*

Experiment Set-up @ IIT KGP

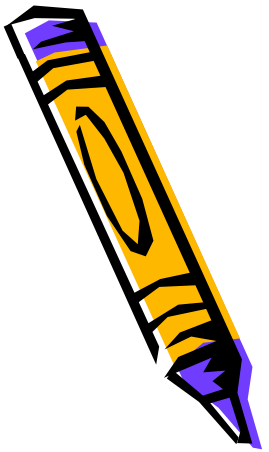


Power Attacks

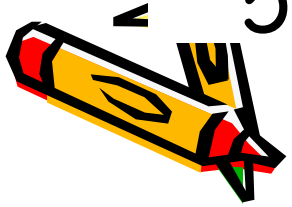
- SPA – Simple Power Analysis attacks
  - Fact exploited - Power consumption at an instant of time is a function of the operation being carried out by the device
- DPA – Differential Power Analysis
  - Fact exploited - Power consumption of the same operation at different instants of time depends on the data being processed.



Simple Power Analysis (SPA)

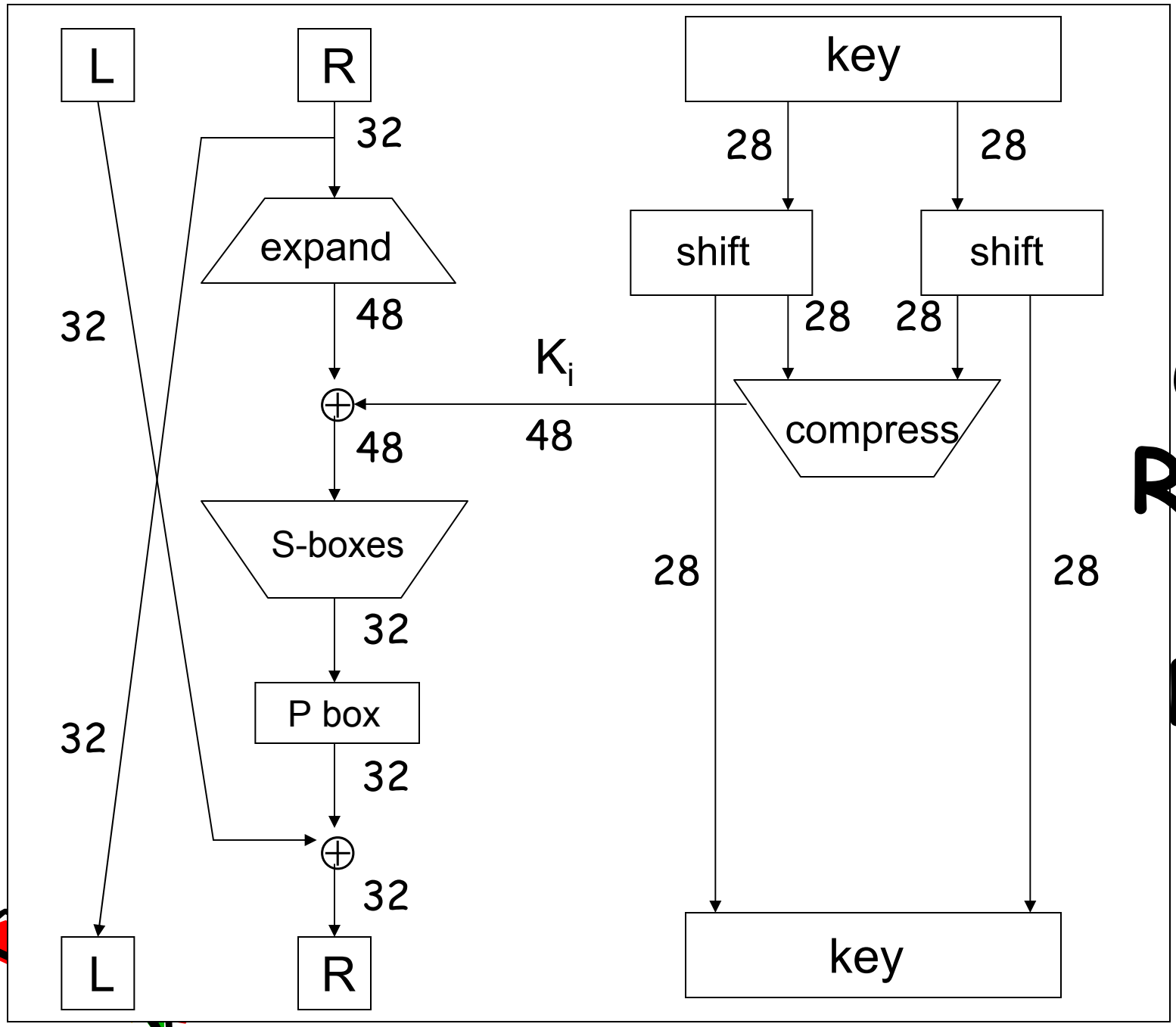


- Directly interprets the power consumption of the device
- Looks for the operations taking place and also the key!
- Trace: A set of power consumptions across a cryptographic process
- 1 millisecond operation sampled at 5MHz yield a trace with 5000 points



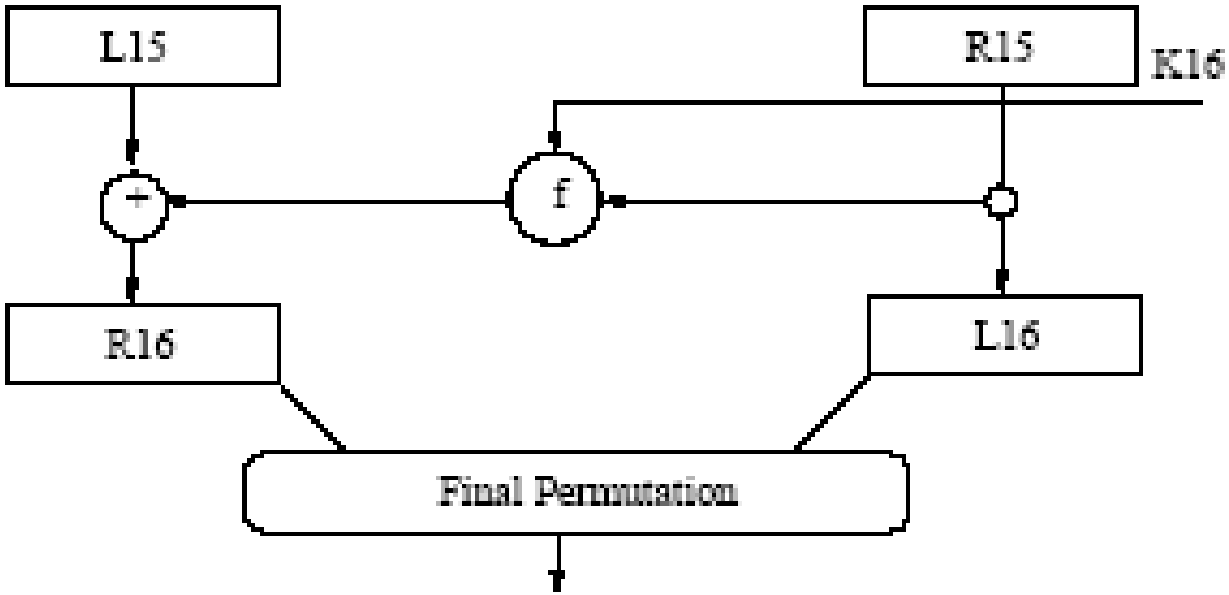
DES Numerology

- DES is a block cipher
- 64 bit block length
- 56 bit key length
- **16 rounds**
- 48 bits of key used each round (subkey)
- Each round is simple (for a block cipher)
- Security depends primarily on "S-boxes"
- Each S-boxes maps 6 bits to 4 bits
- Each S-box has a share of 6 bits of the key

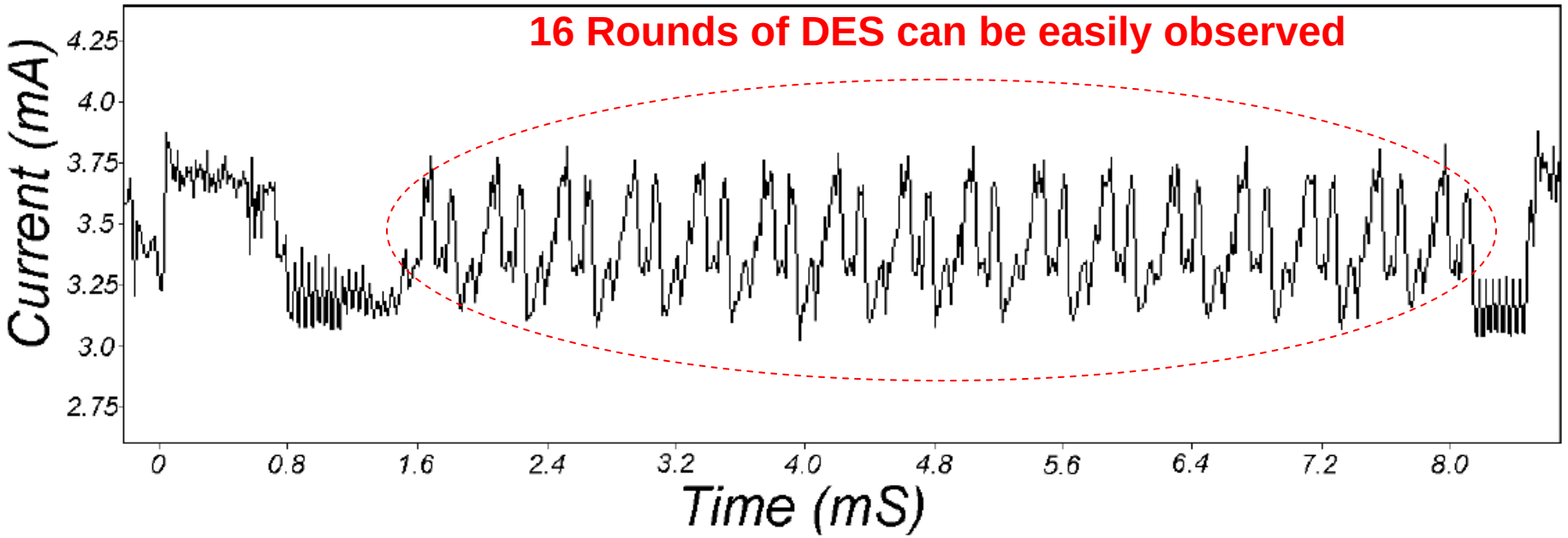


One Round of DES

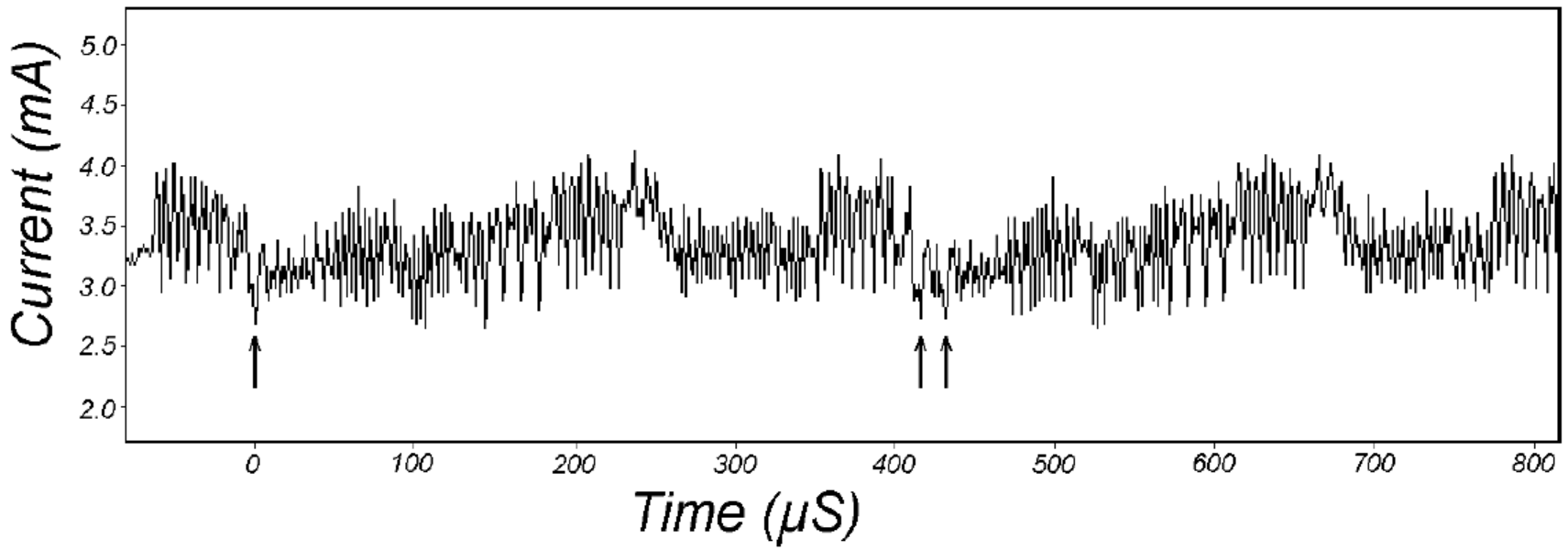
Last Round of DES



Power Traces of DES

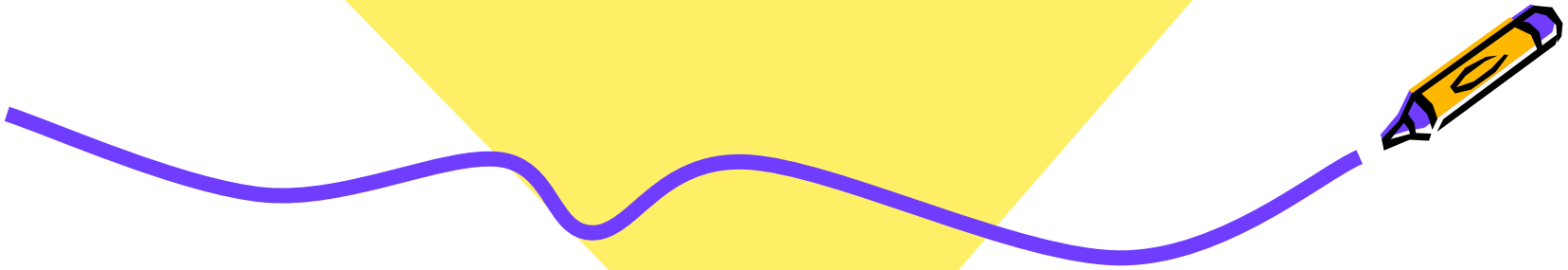


Power Traces for DES



The 28 bit key registers C and D are rotated once in round 2, while twice in round 3. These conditional branches depending on the key bits leak critical information.

Differential Power Analysis (DPA)



DPA Overview

Introduced by P. Kocher and colleagues

More powerful and more difficult to prevent than SPA

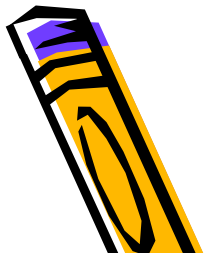
Different power consumption for different state (0 or 1)

Data collection phase and data analysis phase

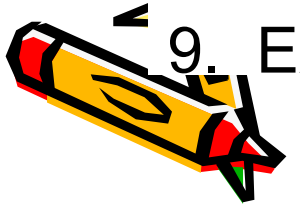
Procedure

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

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 last round S-box first bit output 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



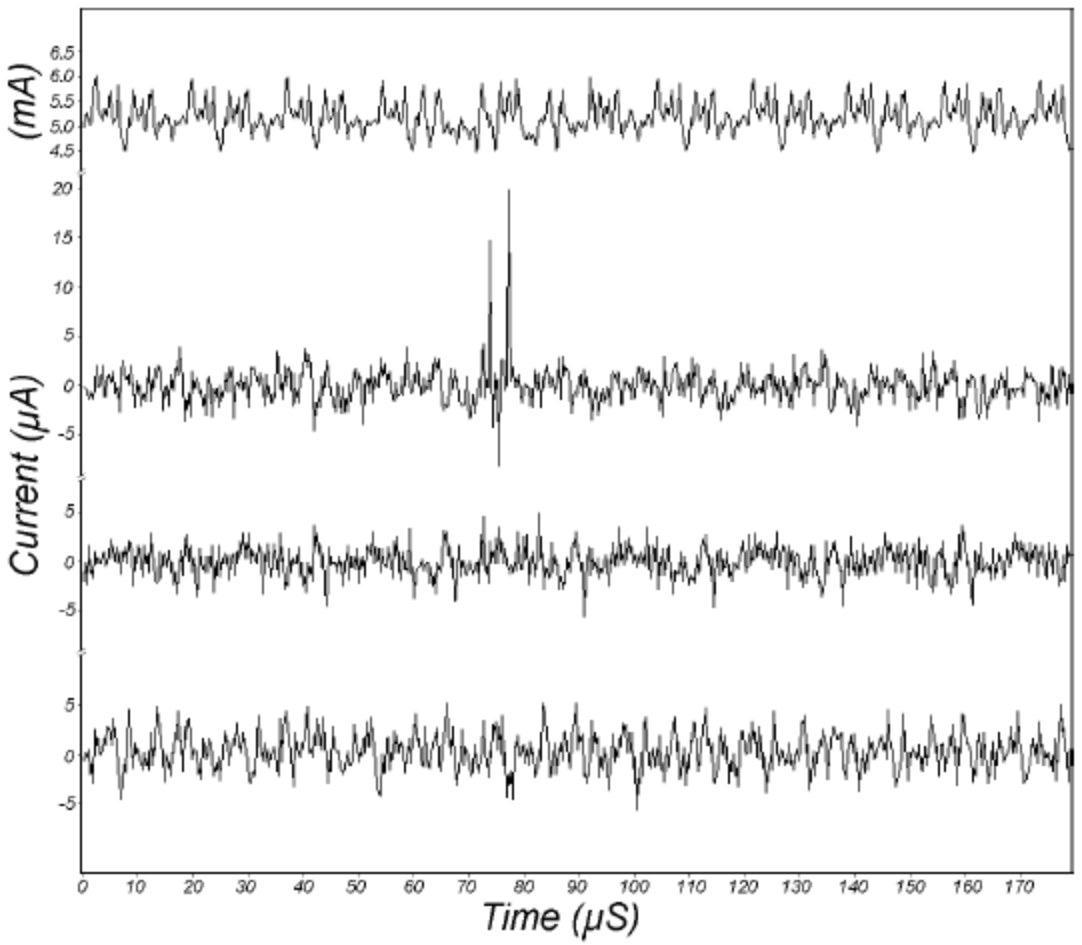
DPA Result Example

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

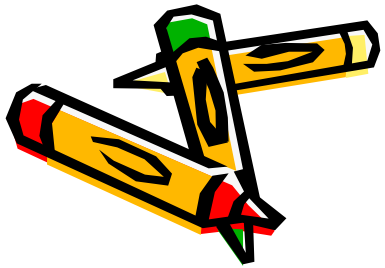


DPA in details

- DPA selection function : $D(C,b,K_s)$ is defined as computing the value of the
  - b^{th} output bit, depending upon
    - C: Ciphertext
    - K_S is the guessed key (6 bits) for the S-Box
- **Note: If K_s is incorrect evaluating D(…) gives the correct bit in half of the cases for each of the ciphertexts.**

DPA in details

- Attacker obtains m encryption operations and capture power traces, $T_{1..m}[1..k]$, with k sample points each.
- An attacker records the m ciphertexts
- No knowledge of the plaintext is required

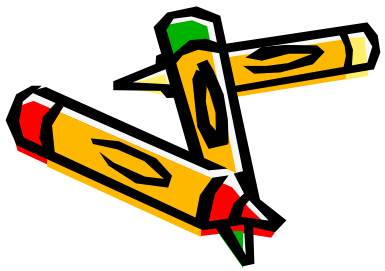


Attacker's Power Board

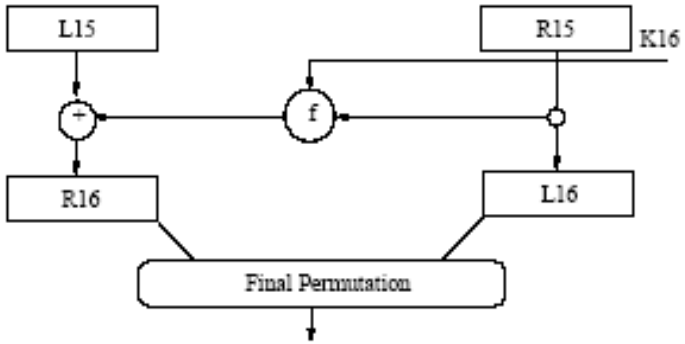
Sample Points

CIPHERTEXTS

T[1][1]	T[1][2]		T[1][k]
T[2][1]	T[2][2]		T[2][k]
		. . .	
T[m][1]	T[m][2]		T[2][k]



The Selection Function D



$$f(R_{15}, K_{16}) = P(S(E(R_{15} \oplus K_{16})))$$

- Attacker knows L16, hence R15
- Attacker knows R16
- Guess K16 (6 bits)
- Compute output of f
- Compute the b^{th} bit of L15
- If K_{16} is wrongly guessed, then the computed value b matches with the correct result half of the time

DPA in details

- Attacker now computes a k-sample differential trace $\Delta_D[1..k]$ by finding the difference between the average of the traces for which D(…) is one and the average for which D(…) is zero.

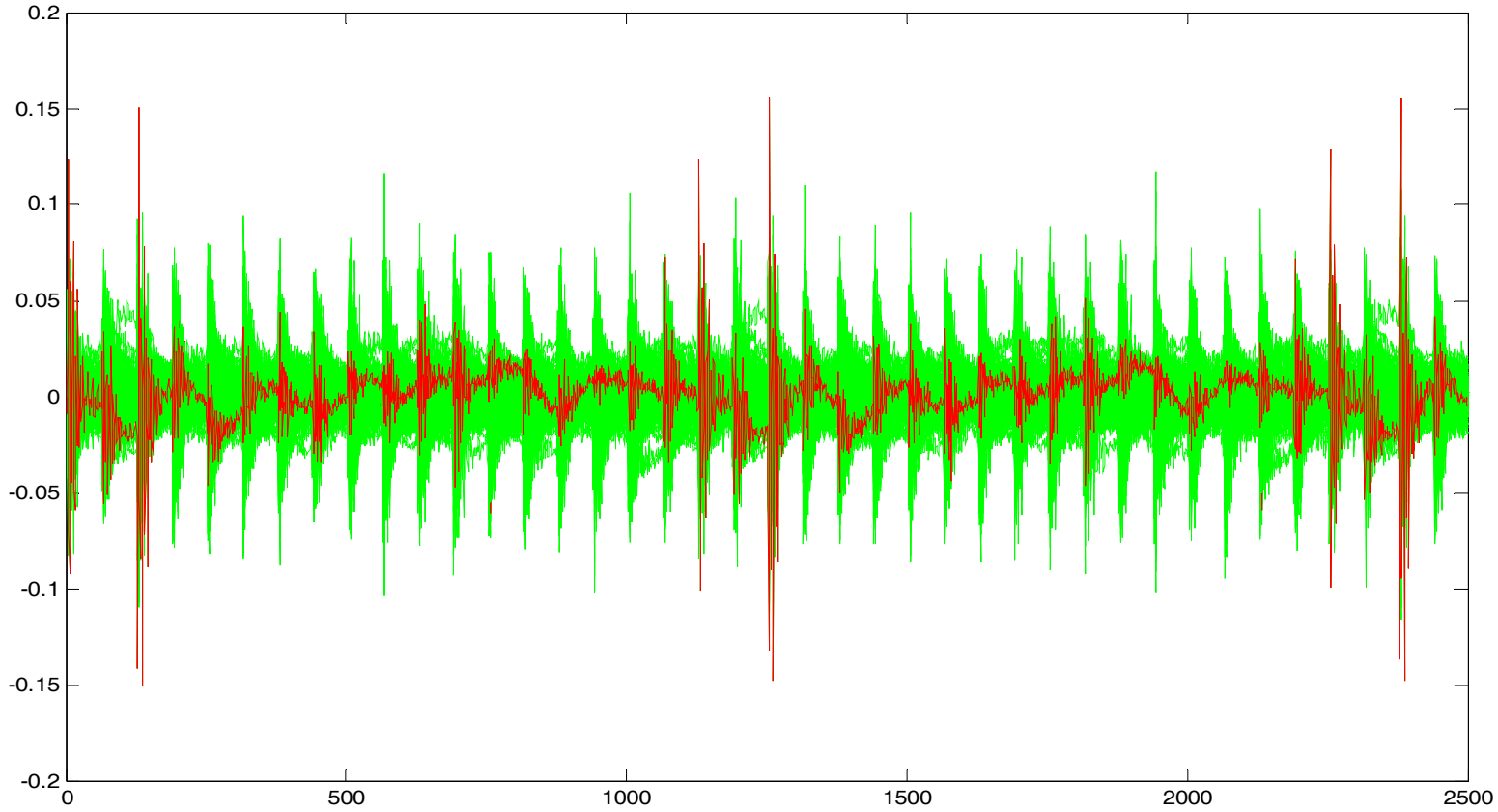
$$\Delta_D = \frac{\sum_{i=1}^{m} D(C_i, b, K_s) T_i[j]}{\sum_{i=1}^{m} D(C_i, b, K_s)} - \frac{\sum_{i=1}^{m} (1 - D(C_i, b, K_s)) T_i[j]}{\sum_{i=1}^{m} (1 - D(C_i, b, K_s))}$$

Principle: If K_s is wrongly guessed, D behaves like a random guess. Thus for a large number of sample points, $\Delta D[1..k]$ tends to zero. But if its correct, the differential will be non-zero and show spikes when D is correlated with the value being processed.

DPA in details

- The correct value of K_s can thus be identified from the spikes
- After computing the 48 bits, one can perform brute force attack on the remaining 8 bits in the keying material.
- Note that noise, measurement errors etc have no effect on this method (as they also are uncorrelated to the data being processed--- just like the wrong guess)…

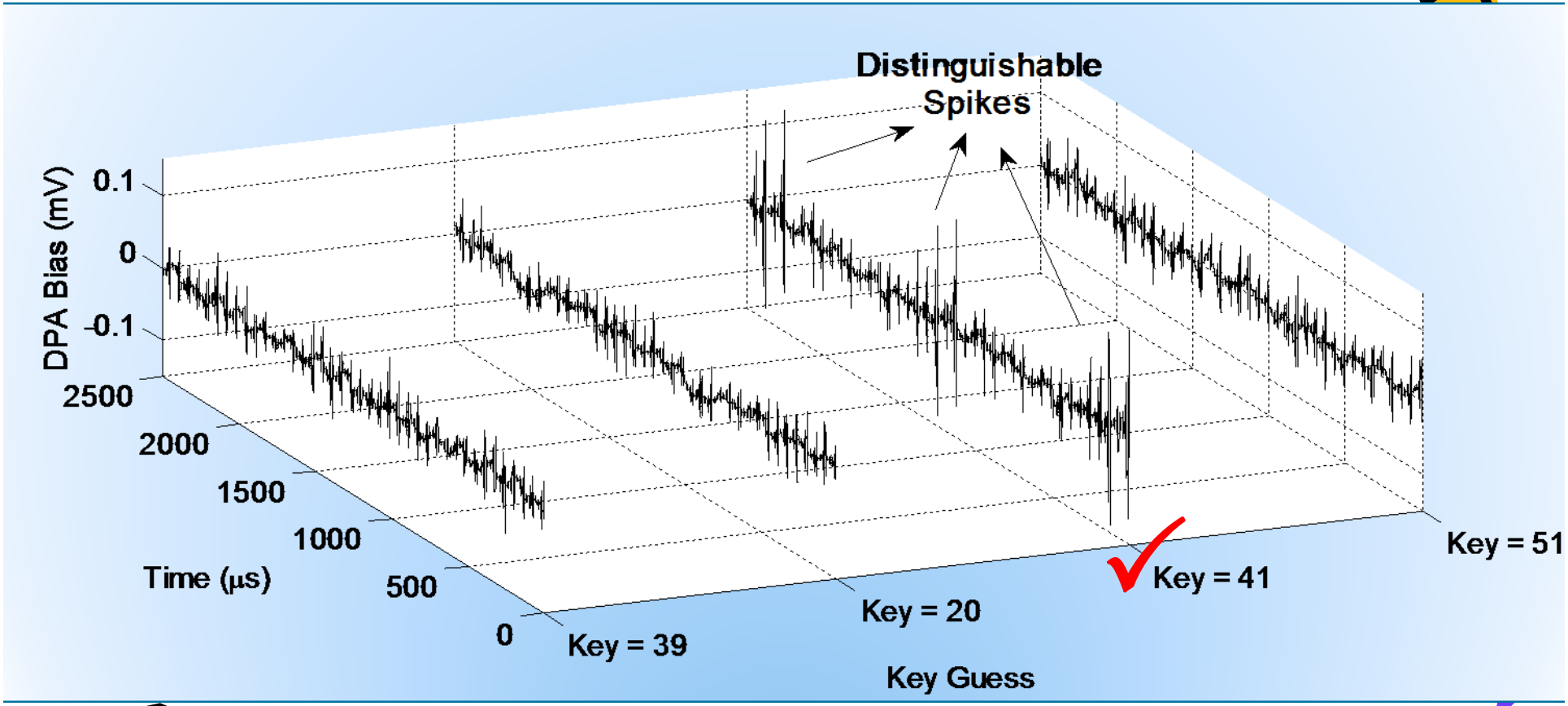
DPA Results - DES



2D Differential Plot

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

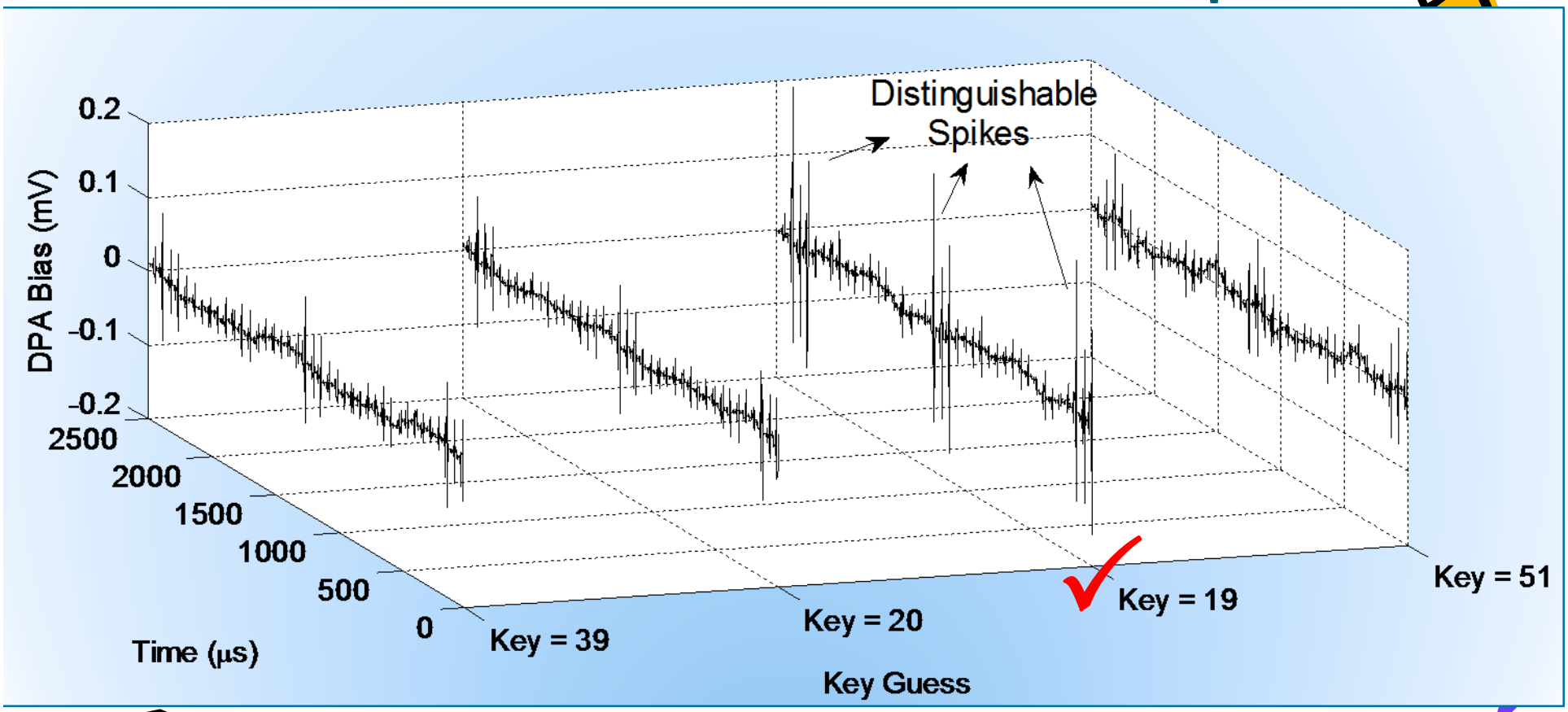
DPA Results - DES



3D Differential Plot

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

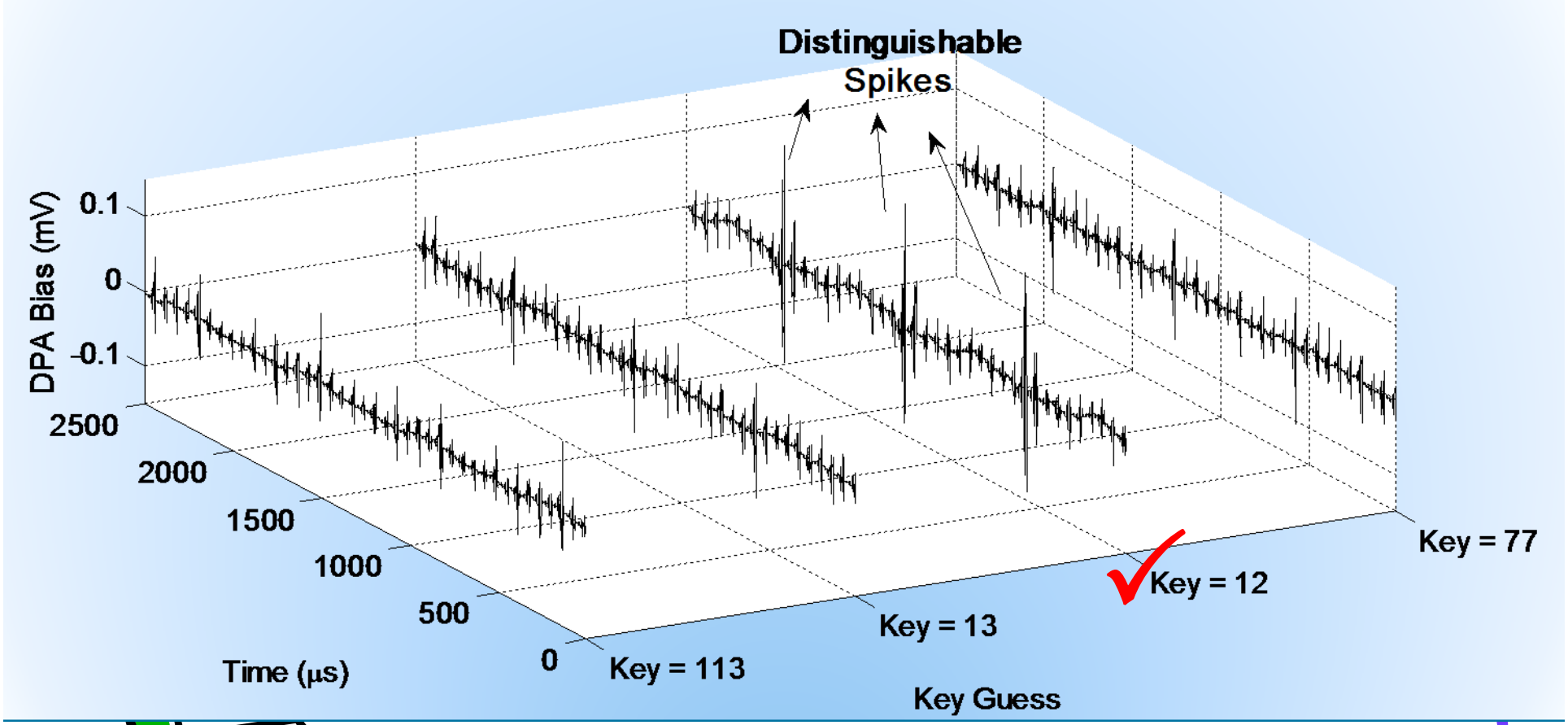
DPA Results - Triple-DES



3D Differential Plot

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

DPA Results - AES



3D Differential Plot

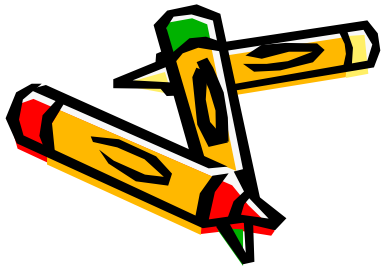
SBOX – 11 **BIT – 8** **TRACE COUNT = 15,000**

Countering DPA

- Two broad *approaches* are taken
  - Make the power consumption of the device independent of the data processed
    - Detached power supplies
    - Logic styles with a data independent power consumption
    - Noise generators
    - Insertion of random delays
  - Methods are costly and not in tune with normal CAD methodologies

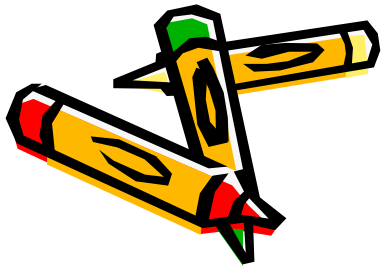
Countering DPA

- *Second Approach* is to randomize the intermediate results
- Based on the principle that the power consumption of the device processing randomized data is uncorrelated to the actual intermediate results
- Masking: Can be applied at the algorithm level or at the gate level

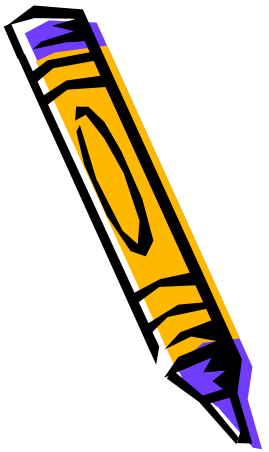


Gate Level Masking

- No wire stores a value that is correlated to an intermediate result of the algorithm.
- Process of converting an unmasked digital circuit to a masked version can be automated



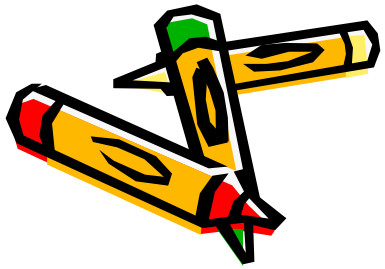
Why are normal gates susceptible to DPA?



a	b	q	Energy
$0 \to 0$	$0 \to 0$	$0 \to 0$	$E_{0\to0}$
$0 \to 0$	$0 \to 1$	$0 \to 0$	$E_{0\to0}$
$0 \to 0$	$1 \to 0$	$0 \to 0$	$E_{0\to0}$
$0 \to 0$	$1 \to 1$	$0 \to 0$	$E_{0\to0}$
$0 \to 1$	$0 \to 0$	$0 \to 0$	$E_{0\to0}$
$0 \to 1$	$0 \to 1$	$0 \to 1$	$E_{0\to1}$
$0 \to 1$	$1 \to 0$	$0 \to 0$	$E_{0\to0}$
$0 \to 1$	$1 \to 1$	$0 \to 1$	$E_{0\to1}$

a	b	q	Energy
$1 \to 0$	$0 \to 0$	$0 \to 0$	$E_{0\to0}$
$1 \to 0$	$0 \to 1$	$0 \to 0$	$E_{0\to0}$
$1 \to 0$	$1 \to 0$	$1 \to 0$	$E_{1\to0}$
$1 \to 0$	$1 \to 1$	$1 \to 0$	$E_{1\to0}$
$1 \to 1$	$0 \to 0$	$0 \to 0$	$E_{0\to0}$
$1 \to 1$	$0 \to 1$	$0 \to 1$	$E_{0\to1}$
$1 \to 1$	$1 \to 0$	$1 \to 0$	$E_{1\to0}$
$1 \to 1$	$1 \to 1$	$1 \to 1$	$E_{1\to1}$

Normal And gate, q= a & b



Why are normal gates susceptible to DPA?

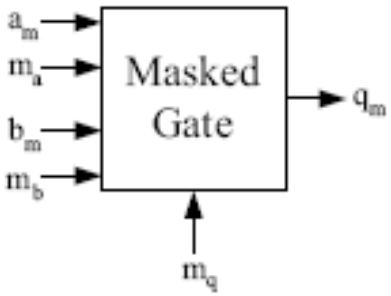
- Attacker measures large number of power traces
- Splits the traces into two groups when q=0 and when q=1 at the end of the clock cycles.
- The expected means are not in general equal, leading to DPA attacks
  (as there are spikes in the differential trace)
- Here, means of the energies of the groups are:

$$E(q=0)=(3E_{1->0}+9E_{0->0})/12;$$
$$E(q=1)=(3E_{0->1}+E_{1->1})/4$$

Since, $E(q=0) \neq E(q=1)$,

Hence, DPA attack is possible

Masked And Gate



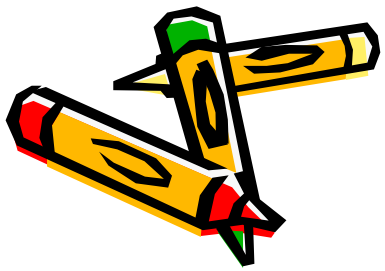
$$a_m = a \oplus m_a$$

$$b_m = b \oplus m_b$$

$$q_m = q \oplus m_q$$

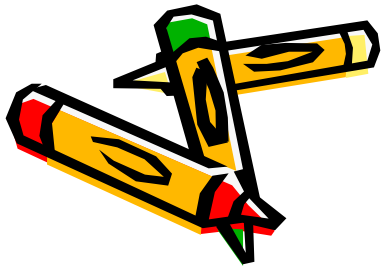
$$q = f(a,b)$$

$$q_m = \hat{f}(a_m, m_a, b_m, m_b, m_q)$$



Masked And Gate

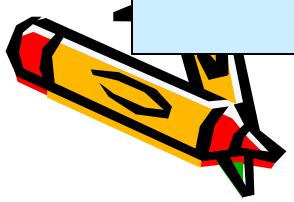
$q_m=(a.b)\text{xor } m_q=(a_m\text{xor } m_a).(b_m \text{ xor } m_b)$
$\text{xor } m_q$
$=(((a_m.b_m \text{ xor } b_m.m_a) \text{ xor } (m_b.a_m)) \text{ xor}$
$m_a.m_b) \text{ xor } m_q$



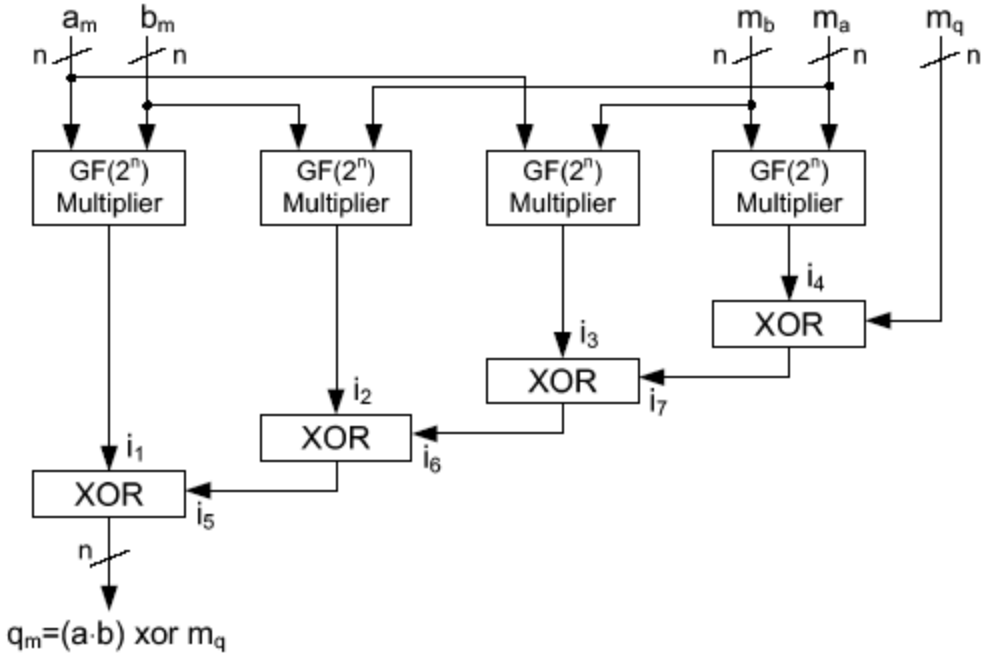
Masked And Gate

- There are 4^5=1024 possible input transmissions that can occur.
- It turns out that the expected value of the energy required for the
- processing of q=0 and q=1 are identical.
- Thus protected against DPA, under the assumption that the CMOS

  gates switch only once in one clock cycles.
- But we know there are glitches, and so the output of gates swing a number of times before reaching a steady state. Hence... the argument continues.



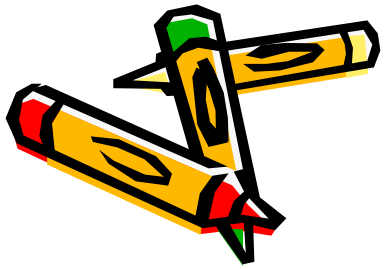
Masked Multiplier



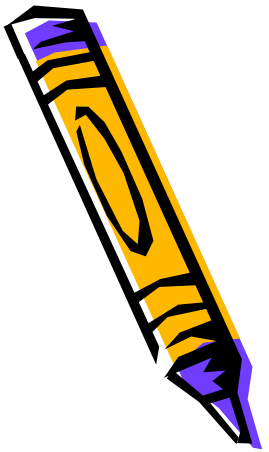
Same Principle may be applied for multiplier circuits.

But Masked Circuits are not safe

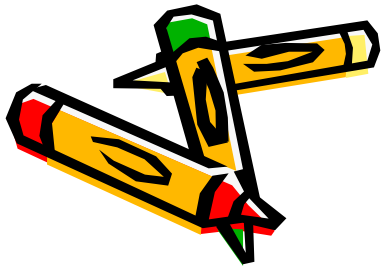
- Transitions, $T(a_m)$, $T(m_a)$, $T(b_m)$, $T(m_b)$ does not leak
- Correlations, $\rho(T(i_j),a)= \rho(T(i_j),b)= \rho(T(i_j),c)=0$, for $j=1$ to 4.
- So xor gates leak information about unmasked values
- Reason is that the xor gates does not change output when both the inputs change value simultaneously or within a small time



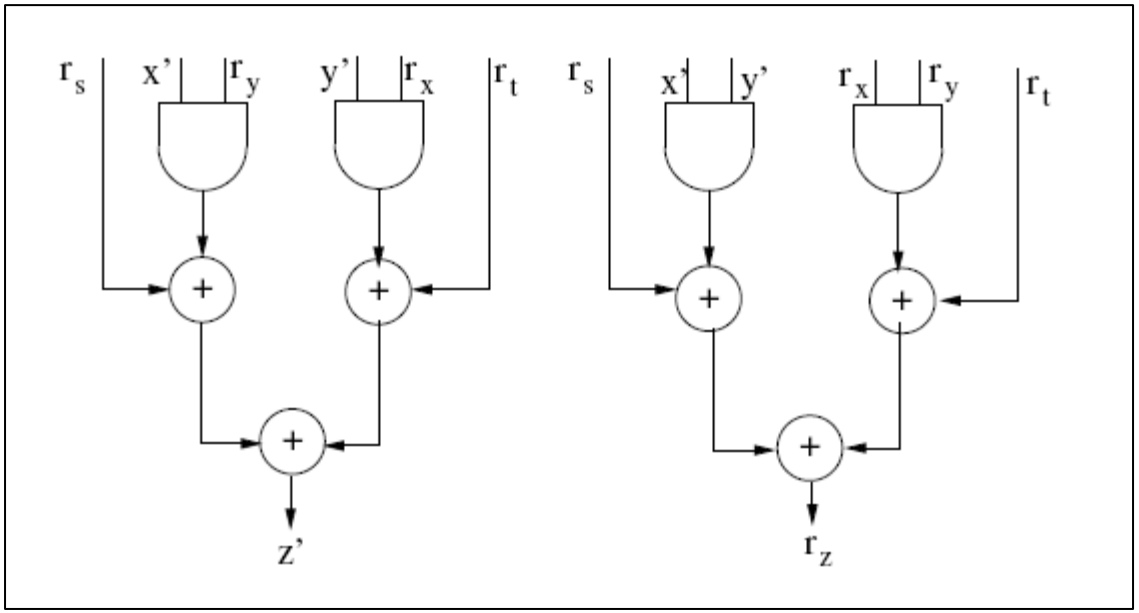
But Masked Circuits are not safe



- Thus the power consumption of the xor gates depend on the time of arrival of the signals i_1 to i_4.
- These time delays are related to the unmasked values
- Thus the masked circuits are still vulnerable to DPA, because of delays in circuits.

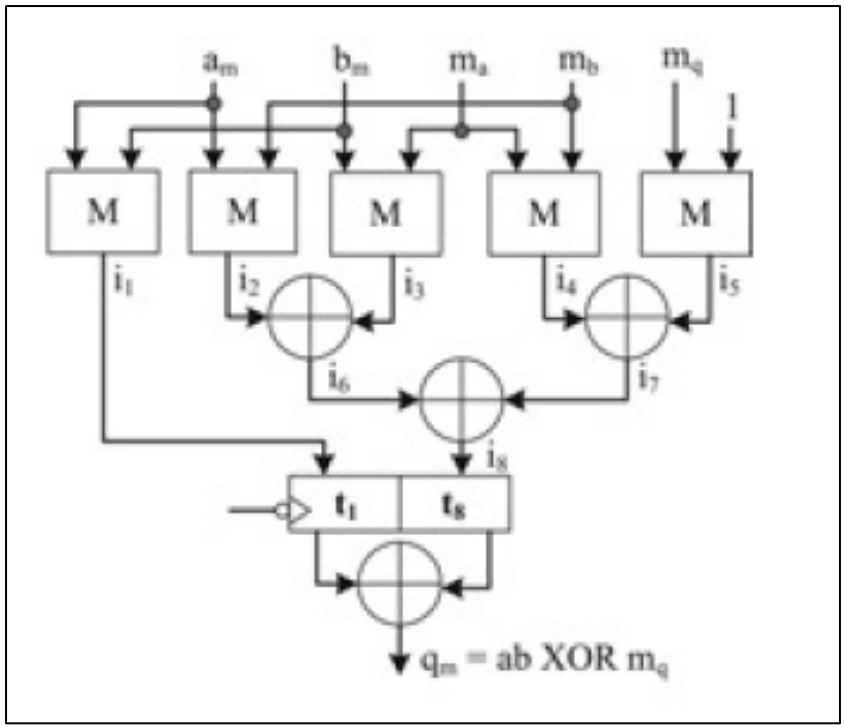


Data dependent glitch free circuit



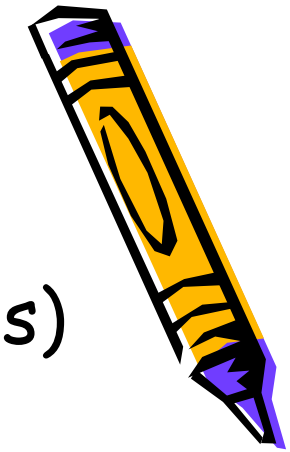
K. Kumar, D. Mukhopadhyay, D.RoyChowdhury, "Design of a Differential Power Analysis Resistant Masked AES S-Box", INDOCRYPT 2007

Pipe-lined AES S-Box

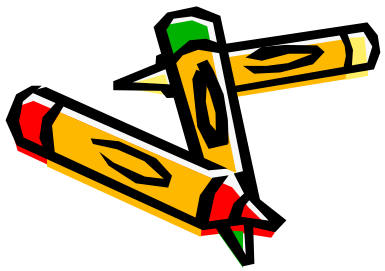


- M. Alam, S.Ghosh, M.J. Mohan, D. Mukhopadhyay, D.R. Chowdhury, I.S.Gupta, "Effect of Glitches against masked AES S-Box Implementation and countermeasures", IET Security, Vol 3(1),, 2009

Stream Ciphers



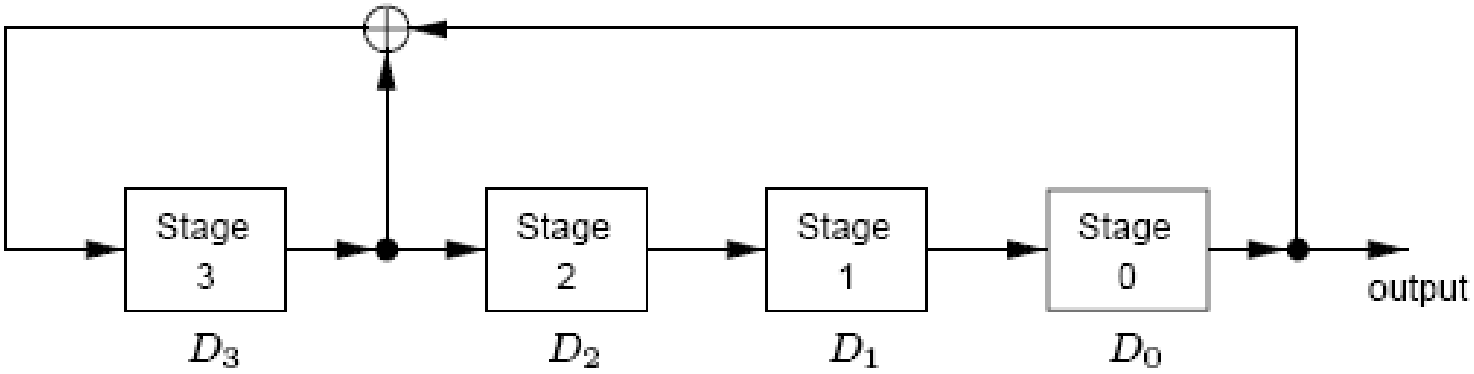
- LFSR (Linear Feedback Shift Registers) are used as building blocks for stream ciphers
- LFSRs are susceptible to power based SCAs
- An n-bit LFSR can be completely determined by making O(n) power measurements.
  - neither the primitive polynomial nor the value of n be known to the attacker.



S. Burman, D. Mukhopadhyay, V. Kamakoti, "LFSR Based Stream Ciphers are Vulnerable to Power Attacks", INDOCRYPT 2007

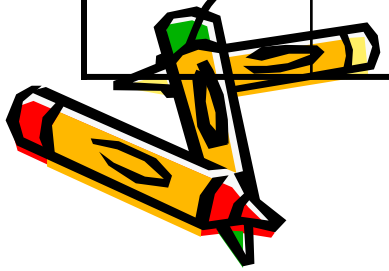
Example

- Consider the LFSR <4,1+D+D^4>



Sequence of the LFSR

t	D_3	D_2	D_1	D_0	HD_t	PD_t
0	0	1	1	0	2	0
1	0	0	1	1	2	1
2	1	0	0	1	3	1
3	0	1	0	0	2	0
4	0	0	1	0	2	0
5	0	0	0	1	2	1
6	1	0	0	0	1	0
7	1	1	0	0	1	0

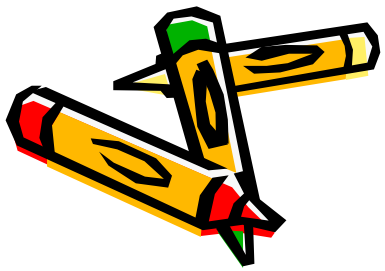


Sequence of the LFSR

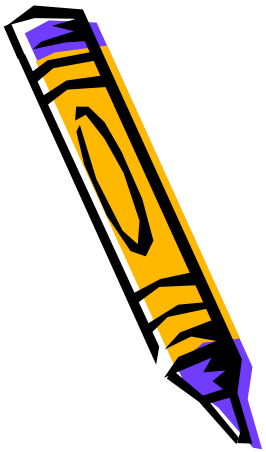
t	D_3	D_2	D_1	D_0	HD_t	PD_t
8	1	1	1	0	1	0
9	1	1	1	1	1	1
10	0	1	1	1	2	1
11	1	0	1	1	3	1
12	0	1	0	1	4	1
13	1	0	1	0	3	0
14	1	1	0	1	3	1
15	0	1	1	0	2	0

The Attack

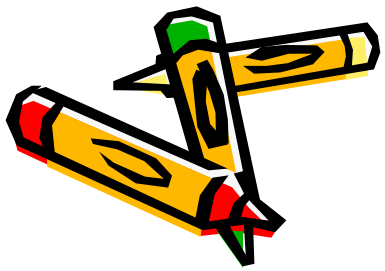
- Step 1: Measure Pow(0) (dynamic power) at t=0
- Step 2: For t=k, k≥1:
  - Measure Pow(k) (dynamic power)
  - PD'_{k-1}=1, if Pow(k-1)≠Pow(k), else 0
  - Input PD'_{k-1} to Berlekamp-Massey (BM). If BM terminates then exit, else repeat Step 2.

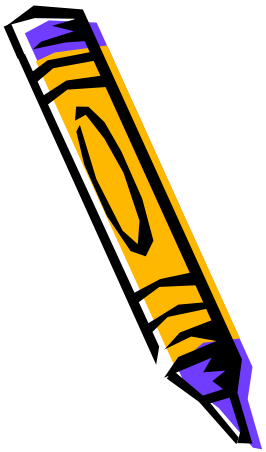


The Attack

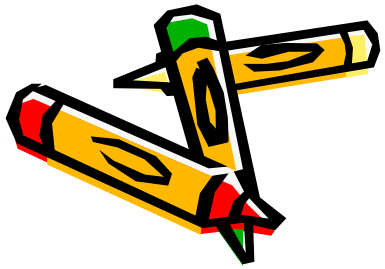


- Result:
  - BM outputs the length of the LFSR, feedback polynomial, F and the connection polynomial realized by F.
  - The initial state can be ascertained by solving a system of linear equations from the previous knowledge.



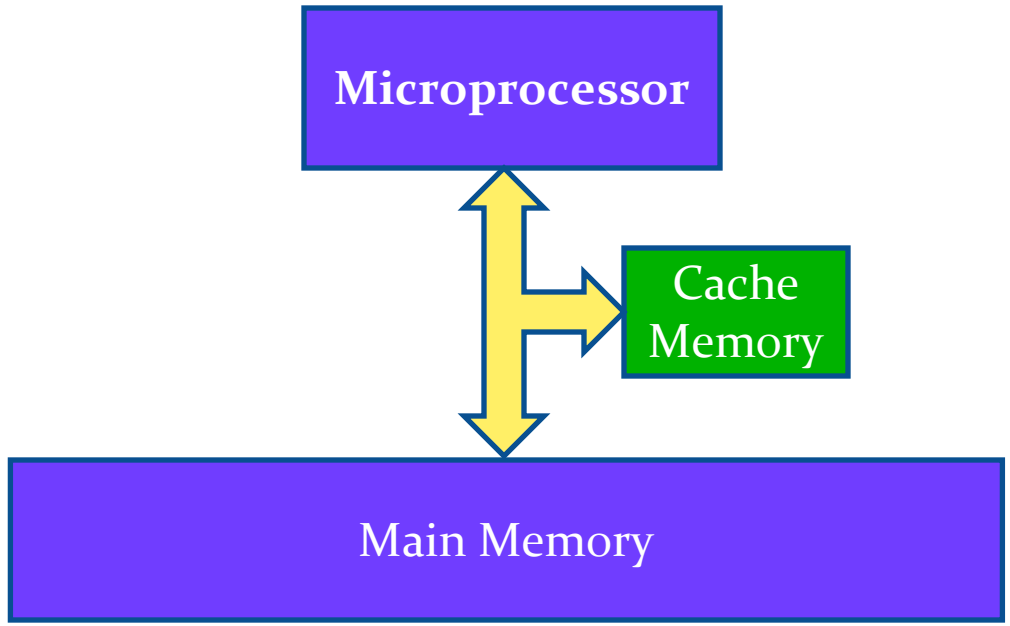


Cache Attacks



53

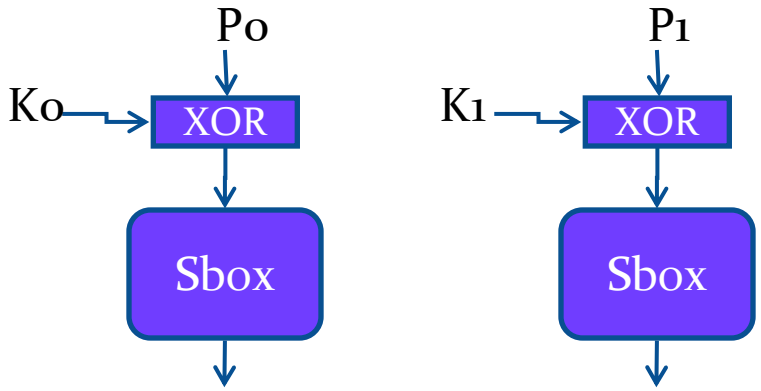
Cache Memory Leaks Information



- If there is a *Cache Hit*
  - Access time is less
  - Power Consumption is less

- If there is a *Cache Miss*
  - Access time is more
  - Power Consumption is more

Cache Attacks : The Principle



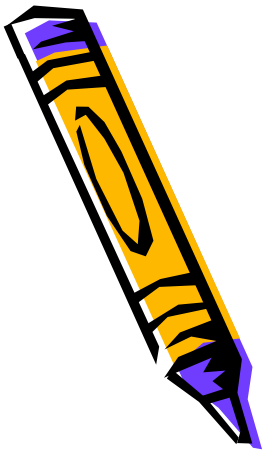
- Power and Time for the (P1 ^ K1) depends on the previous sbox access.
  - If cache hit :
    - (P0 ^ Ko) = (P1 ^ K1)

    =>(K0 ^ K1) = (P0 ^ P1)

    Since we know P0 and P1, we can determine (K0 ^ K1).

    ...but we need to differentiate between a cache hit and miss.

Classes of Cache Attacks



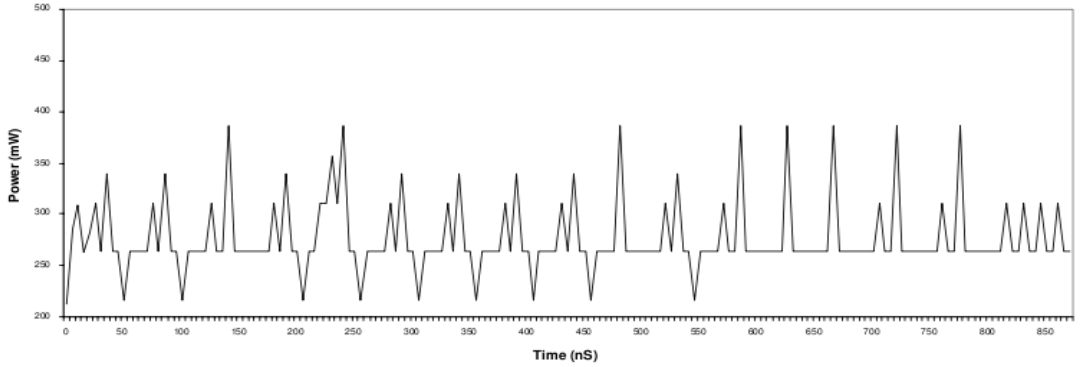
Three ways to identify cache behavior

- Cache Trace Attacks
- Cache Access Attacks
- Cache Timing Attacks

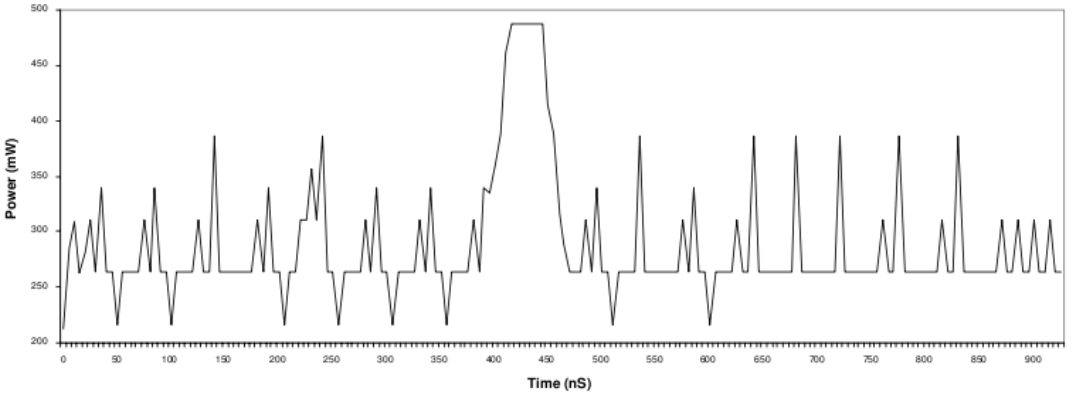
Cache Trace Attacks

- The Power Profile of the system gives away cache behavior.

Without Cache Miss

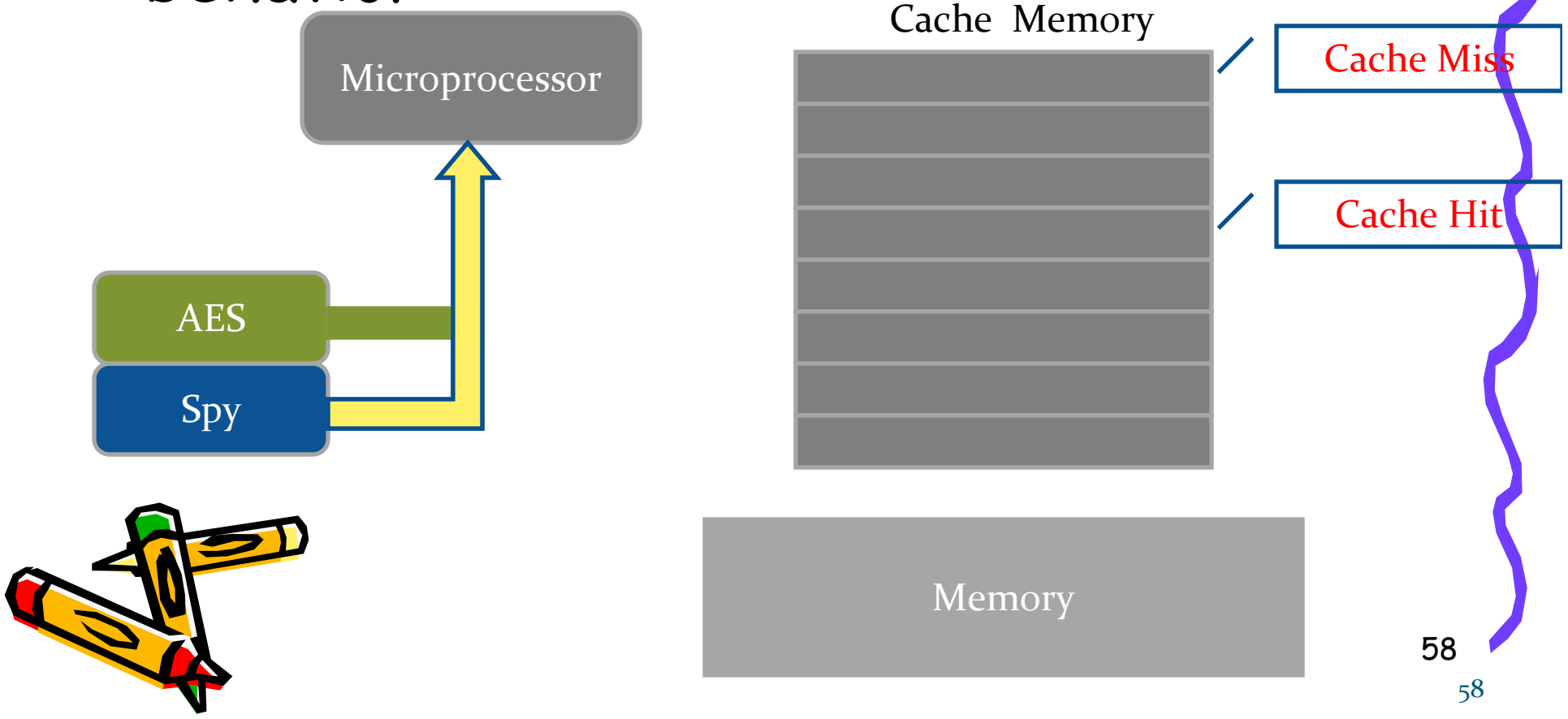


With Cache Miss



Cache Access Attacks : Osvik's Attack

- Uses a spy program to determine cache behavior



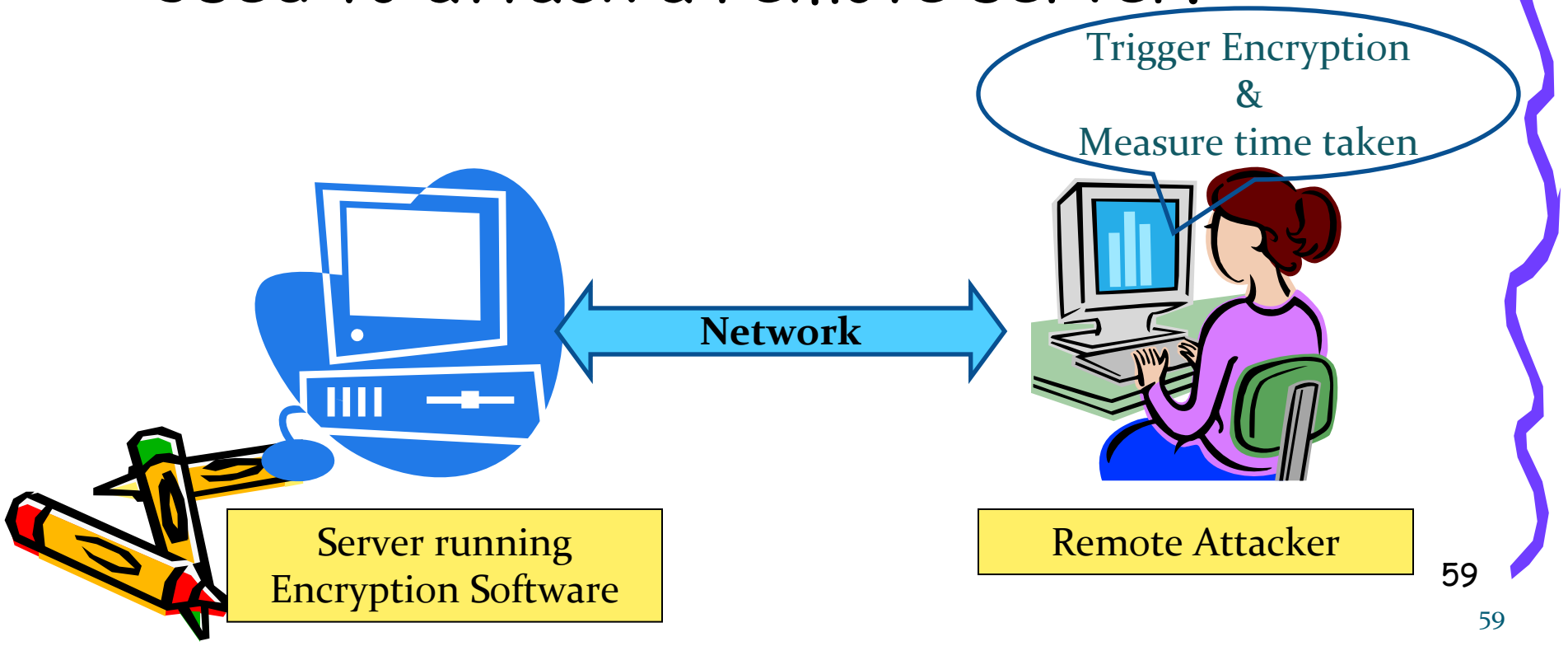
58

Cache Timing Attack

- Based on measuring the total time for encryption.

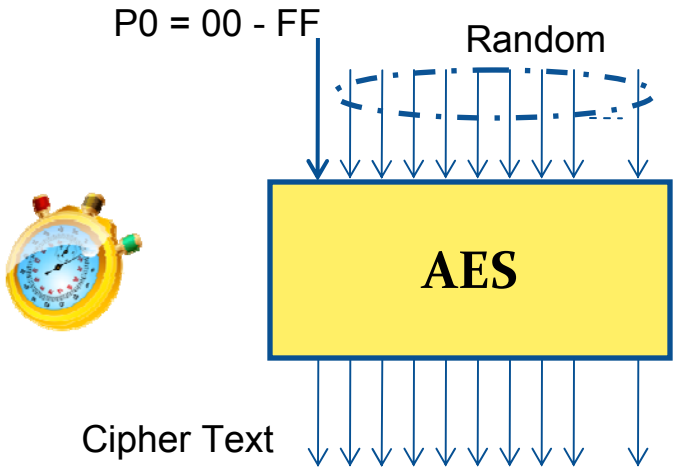
  Execution Time ≈ Nh*Th + Nm*Tm + K

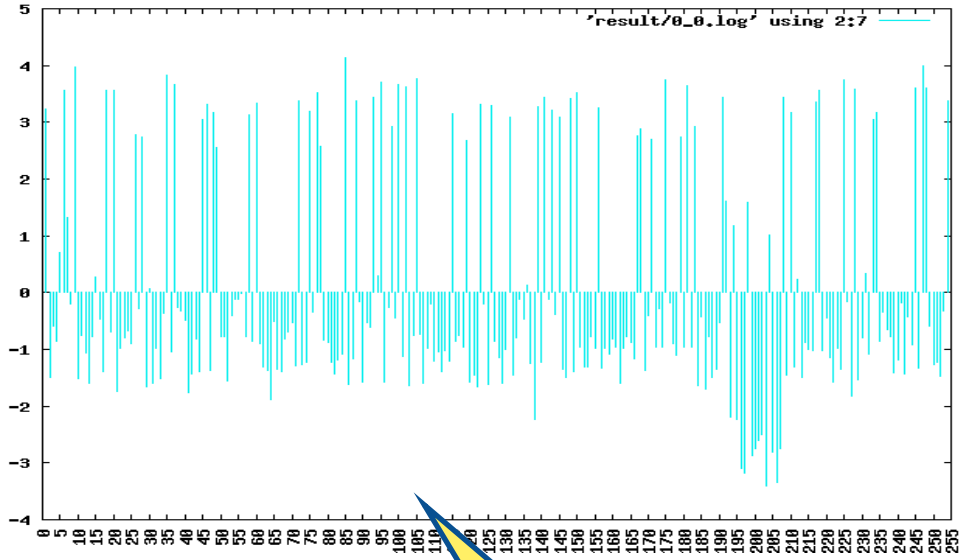
- Used to attack a remote server.



59

Bernstein's Cache Timing Experiment





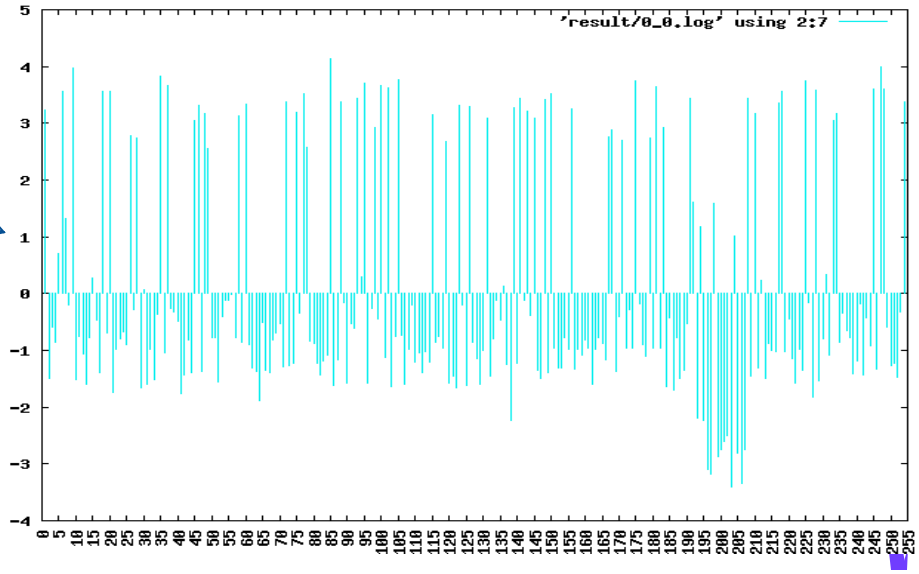
- For P0 = 0 to 255
  - For 2^{15} iterations
    - Vary remaining plaintext by
    - AES Encrypt
    - Determine Time For Encryption

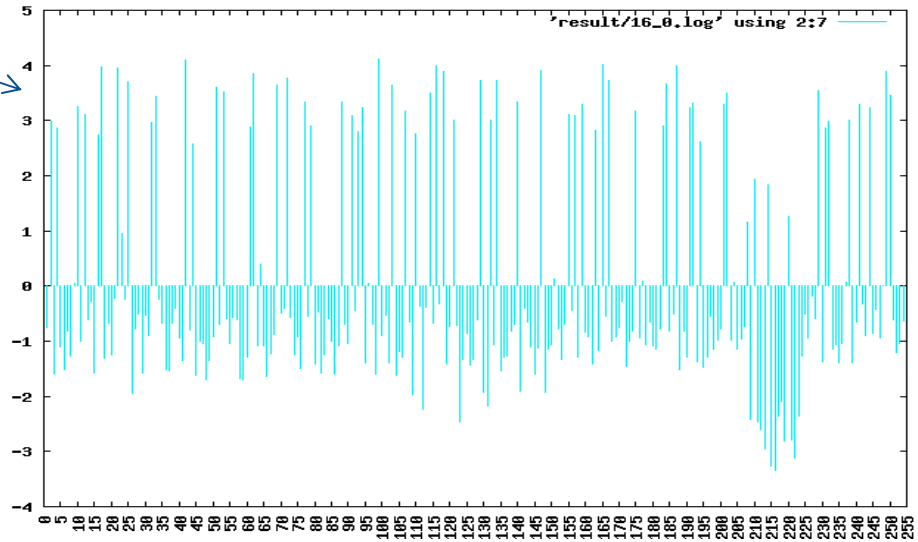
Characteristic of Plaintext Byte Po (Deviation from mean time)

60

Bernstein's Cache Timing Attack

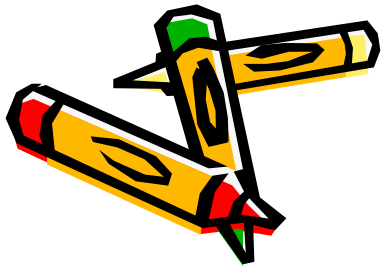
- Put key as all ZEROs and perform experiment
- Repeat experiment with unknown key
- Correlate the two results





Why Cache Attacks Work on AES

- The AES Structure
  - Add initial Key
  - Nine Rounds of
    - Sub Byte
    - Shift Row
    - Mix Column
    - Add Round Key
  - Final Round
    - Sub Byte
    - Shift Row
    - Add Round Key



Substitute 16 bytes from a 256 byte lookup table

Chester Rebeiro, Mainack Mandal, Debdeep Mukhopadhyay, "Pinpointing Cache Timing Attacks on AES", To appear in the 23rd International Conference on VLSI Design and 9th International Conference on Embedded Systems, VLSID 2010, Bangalore, India.

Software Implementations of AES (OpenSSL)

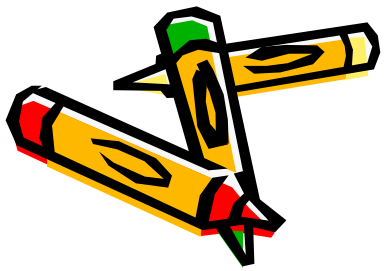
- **Merges all round operations into 4 lookups.**
- **Uses 4 tables T0, T1, T2, T3 , each of size 1024 byte.**
- **The Software Structure is**

Round 1

```
a0 =T0[b0_24] ^ T1[b1_16] ^ T2[b2_8] ^ T4[b3_0] ^ rk[4]
a1 =T0[b1_24] ^ T1[b2_16] ^ T2[b3_8] ^ T4[b0_0] ^ rk[5]
a2 =T0[b2_24] ^ T1[b3_16] ^ T2[b0_8] ^ T4[b1_0] ^ rk[6]
a3 =T0[b3_24] ^ T1[b0_16] ^ T2[b1_8] ^ T4[b2_0] ^ rk[7]
```

Round 2

```
b0 =T0[a0_24] ^ T1[a1_16] ^ T2[a2_8] ^ T4[a3_0] ^ rk[8]
b1 =T0[a1_24] ^ T1[a2_16] ^ T2[a3_8] ^ T4[a0_0] ^ rk[9]
b2 =T0[a2_24] ^ T1[a3_16] ^ T2[a0_8] ^ T4[a1_0] ^ rk[10]
b3 =T0[a3_24] ^ T1[a0_16] ^ T2[a1_8] ^ T4[a2_0] ^ rk[11]
```



AES Execution Time for 4KB Table

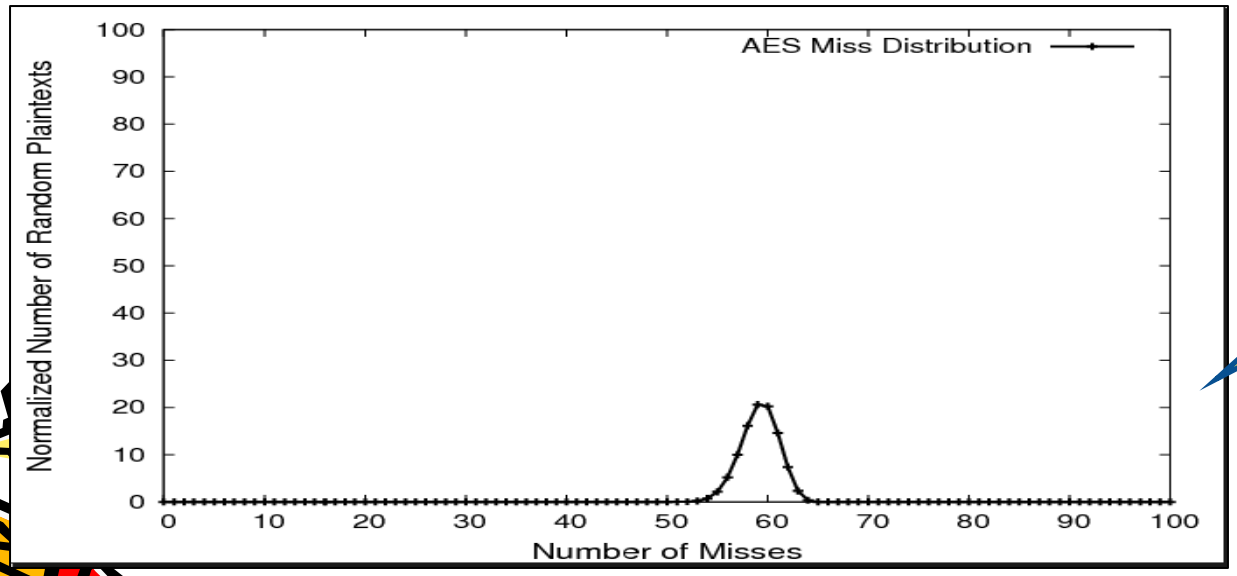
- Encryption Time(E) ≈ Nh*Th + Nm*Tm + K

Since, Nt = Nm + Nh

E = (Nt – Nm) Th + Nm*Tm + K

= Nt*Th + (Tm – Th)Nm + K

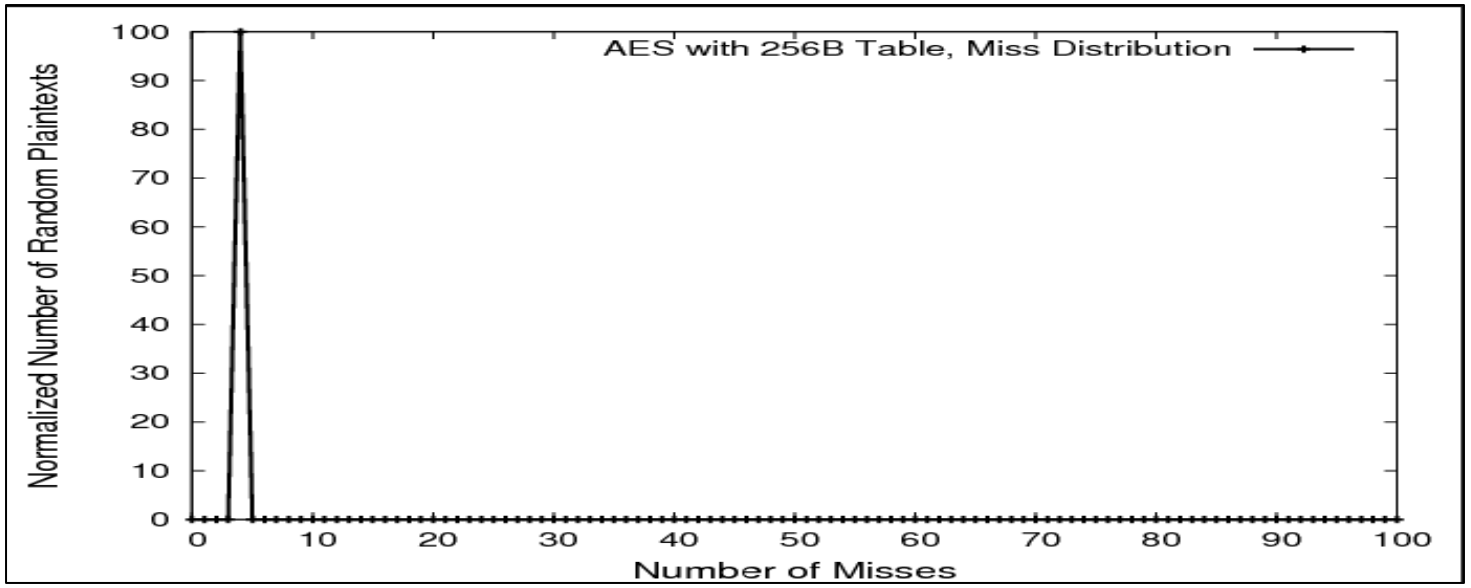
- Time(E2) – Time(E1) = αNm



Cache Attack works because of varying execution time.

64

AES Execution Time with 256Byte Table



- ΔEncryption Time = aNm
- Nm = TableSize/CacheLineSize
  = 256/64 = 4

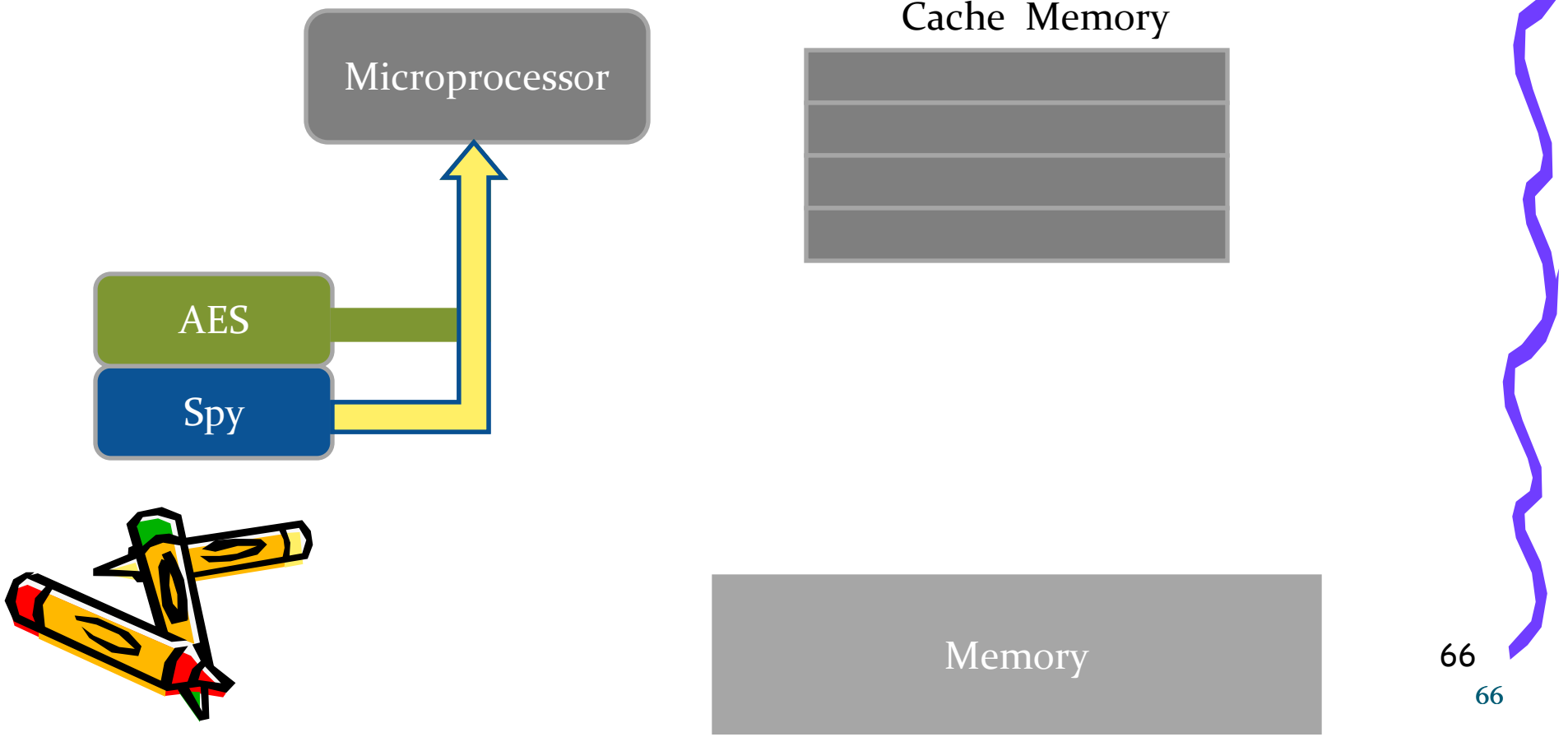
…… Encryption Time is a constant

Cache Attacks Believed to be not possible

65

Osvik's Attack on AES-with 256B Table

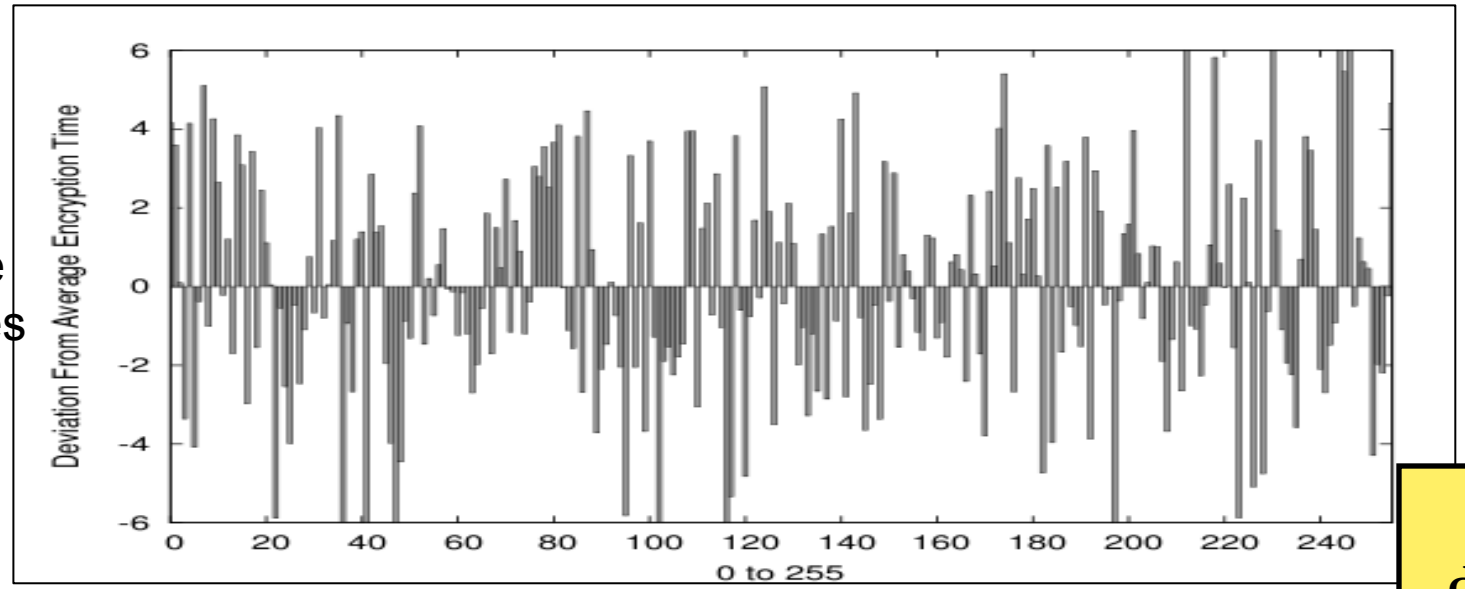
- Won't Work!!!



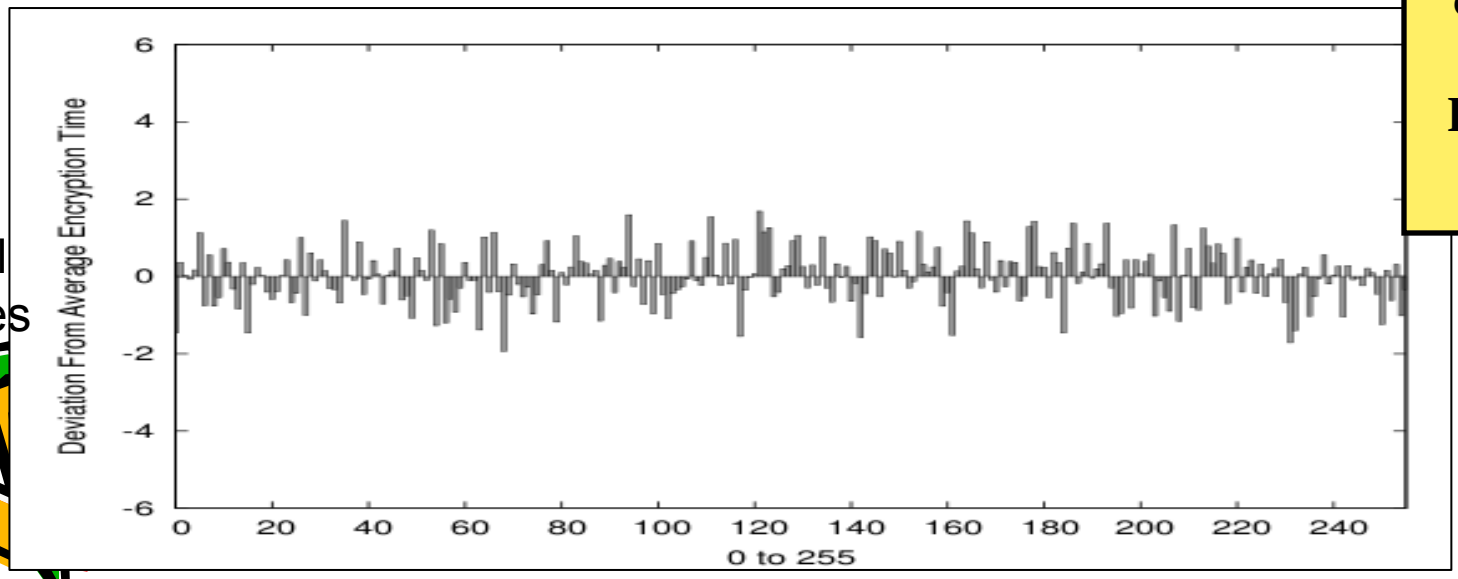
66

Bernstein's Attack on Small Tables

Large Tables



Small Tables



Lesser deviations in Encryption Time

67

Some Opinions on Cache Timing Attacks

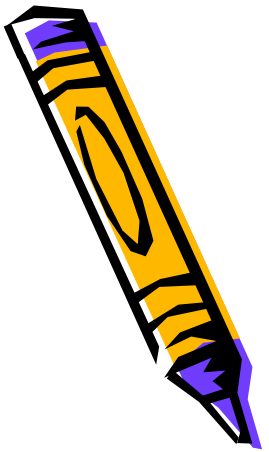
[Kelsey. et.al., 2000, Section 7]

We believe attacks based on cache hit ratio in large Sbox ciphers like Blowfish, CAST and Khufu are possible

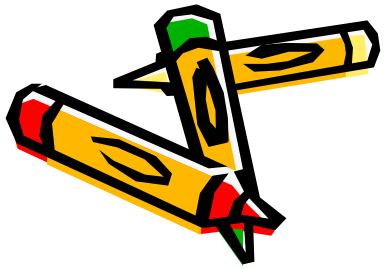
We now show that cache attacks are possible even on ciphers with small sboxes

68

Modern Cache Memory



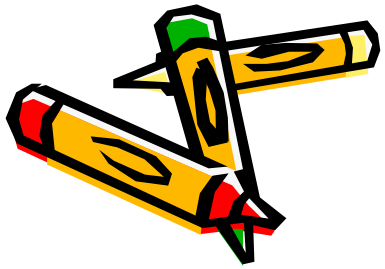
- Processor's aim : Reduce the miss penalty
- Techniques used to achieve this are
  - Speculative Loading
  - Prefetching
  - Out-of-order loading
  - Parallelization
  - Overlapping



69

Speculative Loading & Prefetching

- ***Speculative Loading :*** Loading of data into processor before preceding branches are resolved.
  - ***Effect on Cipher Execution : None****, since branches are generally not used in block ciphers.*
- ***Prefetching :*** data to be loaded into cache when processor detects memory accesses in a sequential order.
  - ***Effect on Cipher Execution : None****, since memory accesses are random in nature.*



Out-of-Order Loading

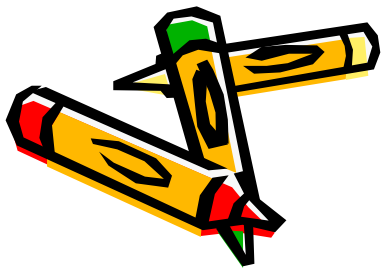
- Data accessed in an order not strictly specified by the program.

Program

```
LOAD AX, #1000
LOAD BX, #2000
LOAD CX, #3000
LOAD DX, #4000
```

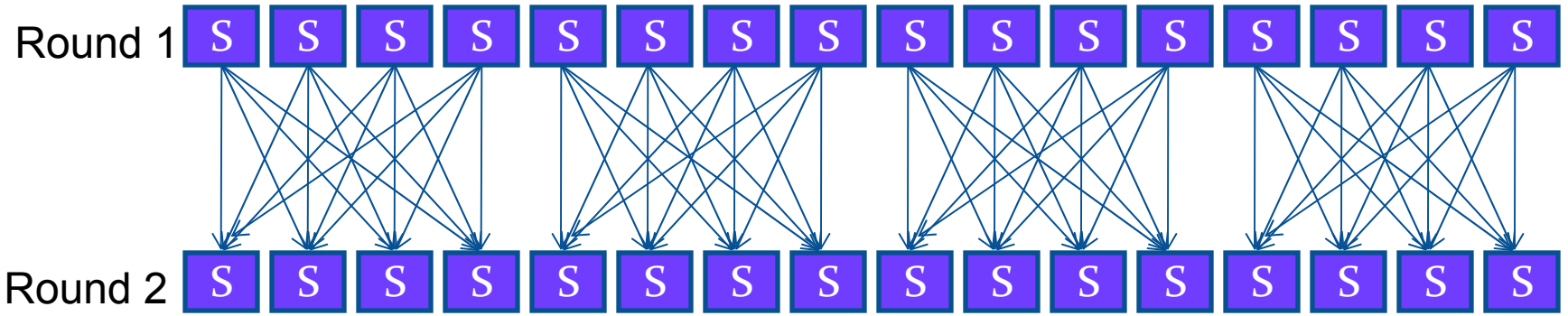
Execution

```
LOAD DX, #4000
LOAD BX, #2000
LOAD AX, #1000
LOAD CX, #3000
```



71

Out-of-Order Loading on a Cipher



Out of order loading works only within a round & not between rounds.

Parallelization & Overlapping

- ***Parallelization*** **:** Two or more cache misses are serviced simultaneously.
- ***Overlapping*** **:** Memory reads are pipelined.

Effect on Cipher Execution :

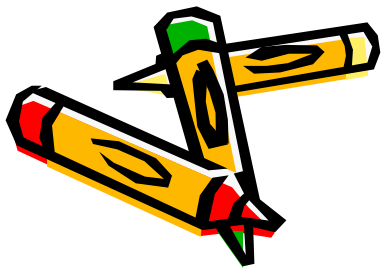
- Consider a 5 round block cipher with 4 misses
- ***Time(E1) < Time(E2)***

E1

2 misses
1 miss
1 miss
0 misses
0 misses

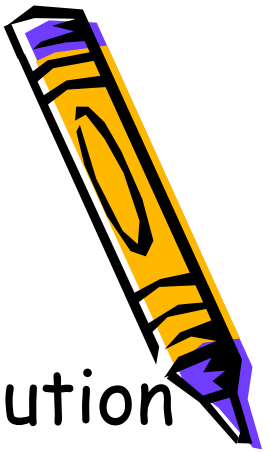
E2

1 miss
1 miss
1 miss
0 misses
1 miss

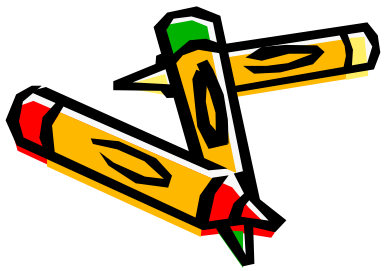


73

On Modern Caches



- Execution Time not only depends on the number of misses, but also on the distribution of misses.
- In fact, our findings show that execution time is inversely proportional to the number of misses in the first round.
- Distribution of misses depends on the key, so, execution time depends on the key.

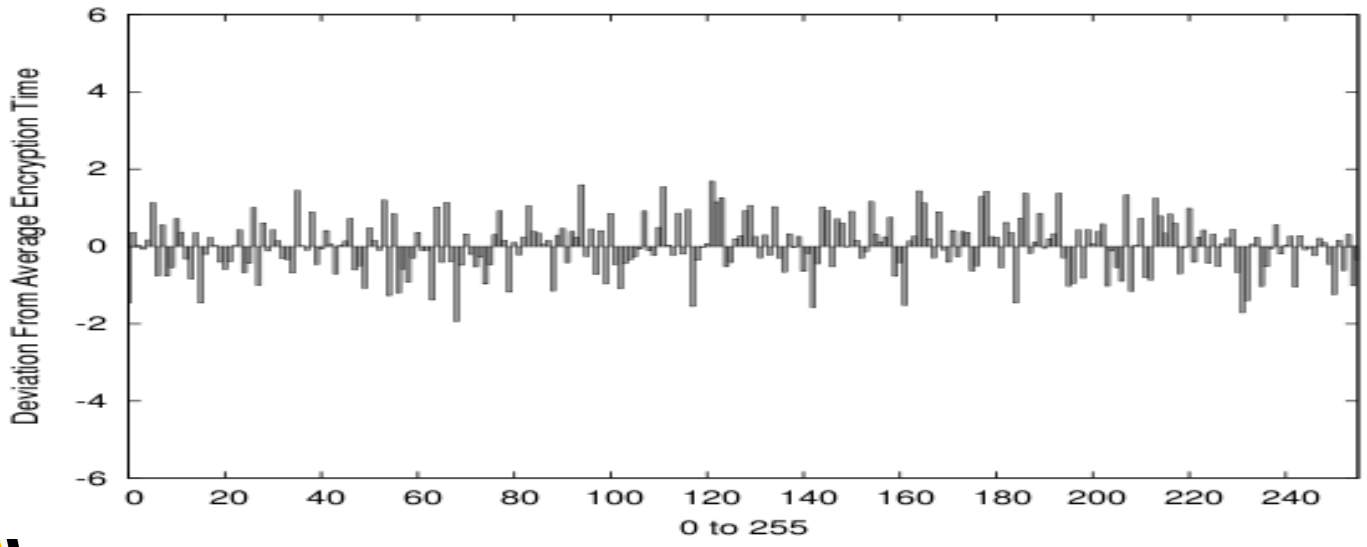


74

But, why doesn't Bernstein's Attack work on small table AES?

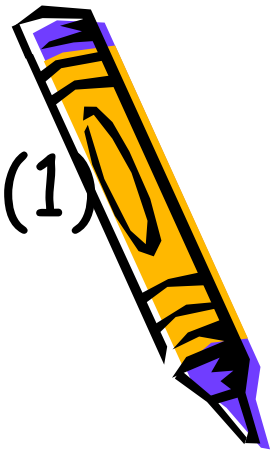
Because,

1. the deviation in encryption time is too less.
2. Bernstein's timing measurements

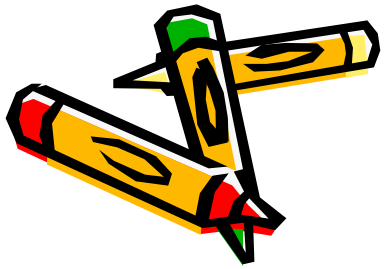


75

Modification's to Bernstein's Attack(1)

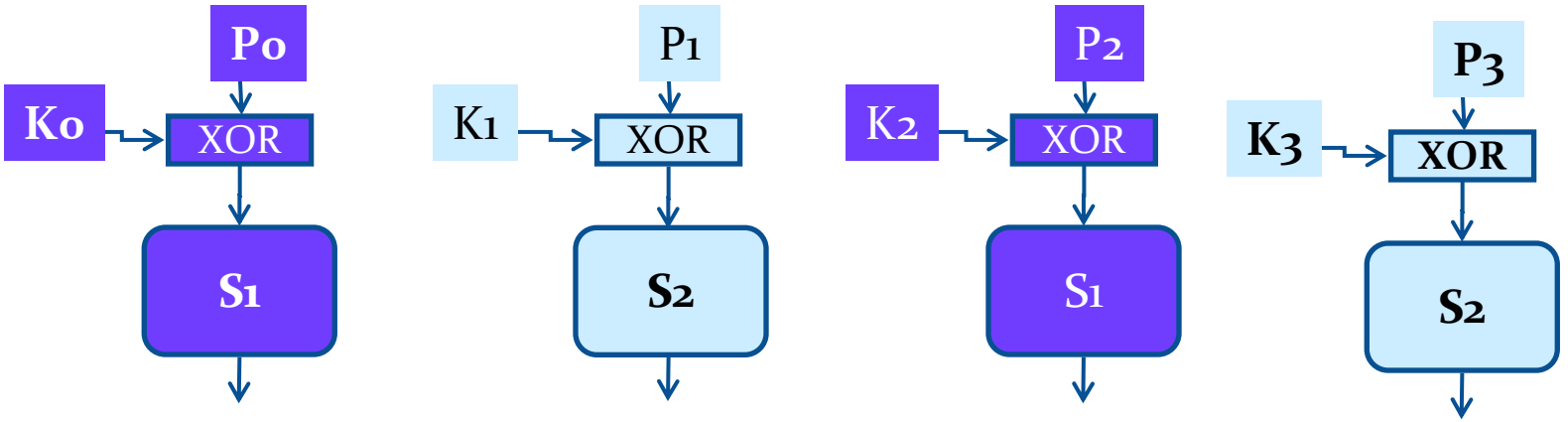


- Improving Timing Measurements
  - Bernstein used 'RDTSC' instruction to measure timing.
    - This is prone to errors due to processor pipeline and out-of-order execution.
  - Instead use 'CPUID' before calling RDTSC
    - This flushes the processor pipeline.
    - Hence all RDTSC instructions done on a fresh pipeline.

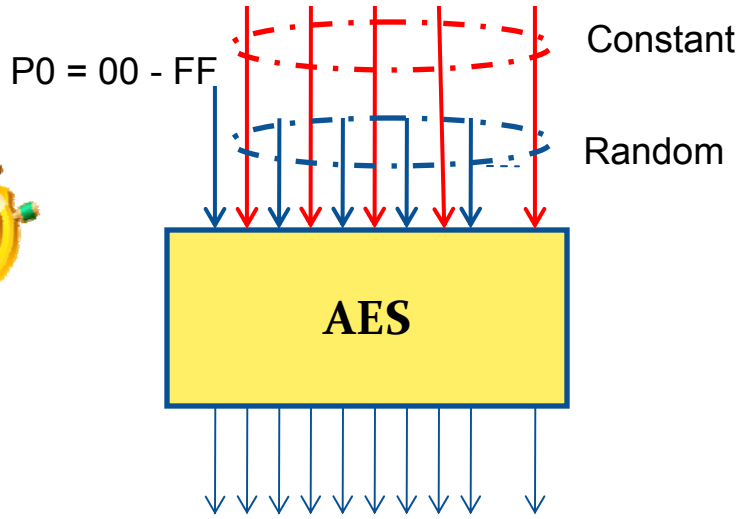


76

Modification's to Bernstein's Attack(2)



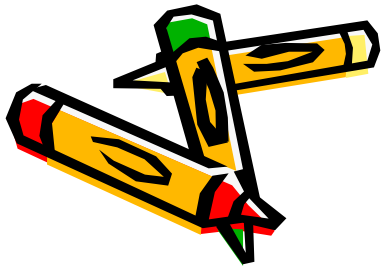
- Vary only those bytes which go to the same sbox as P0.



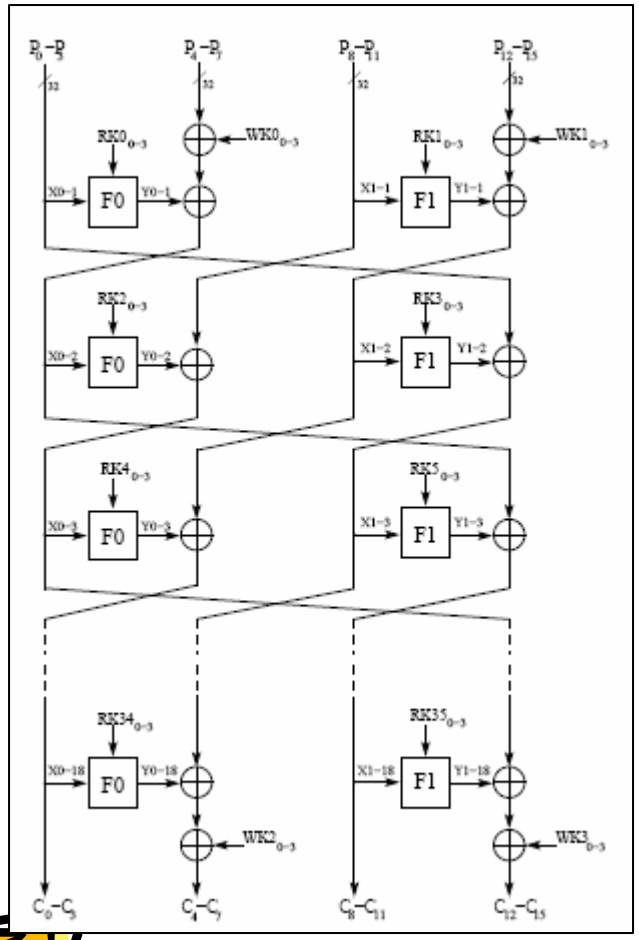
77

Attack of Clefia

- Clefia is a block cipher designed by Sony Corporations
- It has small tables of 256 bytes.
- There are 18 rounds in the cipher
  - each round has 8 look-ups
  - there are 2 S Boxes of 256 bytes
  - Thus there are 4 look-ups per S-box in each round.



Clefia Attack Results



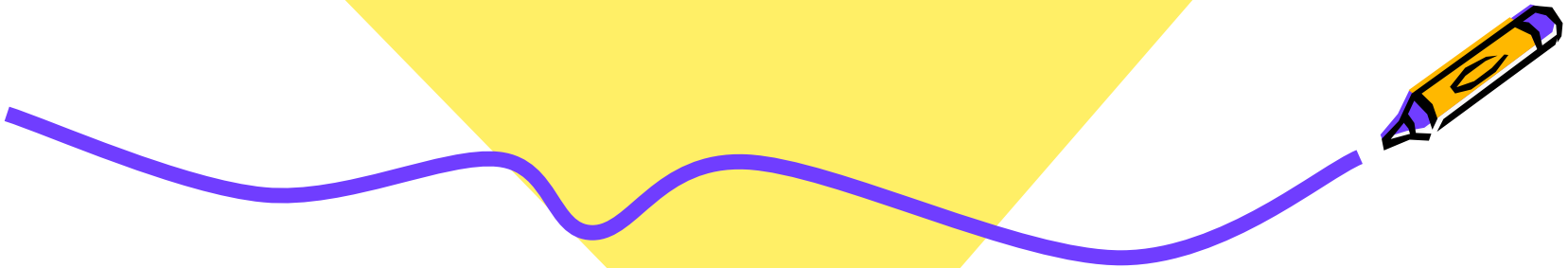
- In around 2^{26} Clefia encryptions the cipher can be shown to break in the face of cache timing attacks
- 3 GHz Intel Core 2 Duo
- 32 kB L1 Cache
- 1 GB RAM
- Linux (Ubuntu 8.04)
- gcc -4.2.4 with O3 optimization.
- Attack Time:
  - First Phase (with known key): 1300 sec
  - Second Phase (with unknown key): 312.5 sec

Chester Rebeiro, Debdeep Mukhopadhyay, Junko Takahashi and Toshinori Fukunaga, "Cache Timing Attacks on Clefia", To appear in Indocrypt 2009.

Correlation Results on CLEFIA

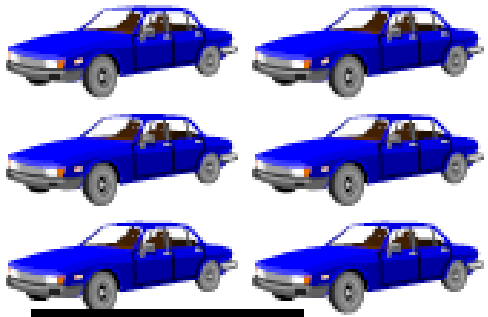
Key Byte	Correct Key	Obtained Correlation results (with correlation value)
$RK0_0$	0a	**0a***(884.6)*, 6b*(469.7)*, 5f*(368.3)*, 20*(357.3)*, ef*(263.7)* ...
$RK0_1$	96	**96***(1853.4)*, 7b*(438.0)*, bc*(437.5)*, 4a*(366.7)*, ee*(361.8)* ...
$RK0_2$	c1	**c1***(1942.1)*, 93*(672.7)*, 98*(598.3)*, f9*(573.2)*, 24*(559.5)* ...
$RK0_3$	68	**68***(1680.3)*, 23*(415.9)*, 9e*(414.1)*, 6e*(398.9)*, 99*(375.9)* ...
$RK1_0$	ac	**ac***(4077.6)*, c1*(853.4)*, 11*(843.5)*, 7c*(650.9)*, 71*(639.2)* ...
$RK1_1$	b0	**b0***(3089.8)*, 73*(740.8)*, 07*(716.7)*, f7*(677.1)*, 01*(658.1)* ...
$RK1_2$	7a	**7a***(5721.0)*, 0a*(1539.1)*, 08*(1230.2)*, 6f*(967.8)*, 05*(931.3)* ...
$RK1_3$	79	**79***(5361.6)*, fb*(1202.0)*, 2b*(1196.0)*, 9a*(1106.6)*, 07*(1007.9)* ...
$RK2_0 \oplus WK0_0$	6e	**6e***(4194.0)*, f9*(1526.2)*, 07*(1491.3)*, 96*(1257.9)*, 2f*(1194.3)* ...
$RK2_1 \oplus WK0_1$	b1	**b1***(4344.0)*, 39*(1197.5)*, 59*(1056.8)*, 63*(980.9)*, f9*(926.9)* ...
$RK2_2 \oplus WK0_2$	9f	**9f***(2662.0)*, d4*(1327.9)*, 68*(1071.1)*, 1b*(1056.2)*, 89*(1000.0)* ...
$RK2_3 \oplus WK0_3$	61	**61***(6840.2)*, 0a*(1783.8)*, 97*(1587.3)*, 8c*(1555.8)*, 87*(1491.4)* ...
$RK3_0 \oplus WK1_0$	c3	**c3***(21042.8)*, 38*(4644.1)*, ea*(4429.9)*, d3*(3999.8)*, 01*(3995.1)* ...
$RK3_1 \oplus WK1_1$	85	**85***(34258.3)*, 7d*(8695.1)*, 83*(8576.9)*, 3a*(8401.3)*, ec*(8318.5)* ...
$RK3_2 \oplus WK1_2$	2c	**2c***(37773.2)*, 3c*(7131.3)*, 28*(6804.1)*, 05*(6263.3)*, b5*(5906.3)* ...
$RK3_3 \oplus WK1_3$	4d	**4d***(37267.7)*, f2*(9903.8)*, 33*(9625.5)*, 24*(8613.2)*, cf*(8595.4)* ...
$RK4_0$	3f	**3f***(1321.7)*, 5e*(535.2)*, 39*(328.4)*, 83*(302.9)*, 04*(276.8)* ...
$RK4_1$	df	**df***(2066.6)*, e6*(510.7)*, 69*(463.6)*, ad*(441.4)*, 5a*(399.3)* ...
$RK4_2$	d7	**d7***(1367.1)*, 09*(331.8)*, b5*(322.7)*, be*(319.7)*, 39*(313.6)* ...
$RK4_3$	5f	**5f***(1530.7)*, cb*(409.6)*, ae*(392.4)*, 1e*(373.3)*, ee*(365.7)* ...
$RK5_0$	66	**66***(5056.0)*, 4e*(938.3)*, 01*(924.7)*, b6*(886.9)*, 05*(870.5)* ...
$RK5_1$	97	**97***(3577.9)*, e4*(795.5)*, 54*(794.1)*, 42*(674.6)*, 4a*(633.2)* ...
$RK5_2$	2d	**2d***(6248.1)*, 5f*(1313.0)*, 5d*(1274.5)*, b3*(1180.1)*, 38*(1134.4)* ...
$RK5_3$	4e	**4e***(6405.4)*, cc*(1363.7)*, 8d*(1173.4)*, ff*(1147.6)*, 1a*(1140.9)* ...

Fault Attacks on Advanced Encryption Standard

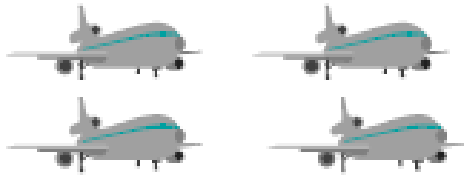


WHAT IS THIS ABOUT?

Broken toys are not charged to our clients



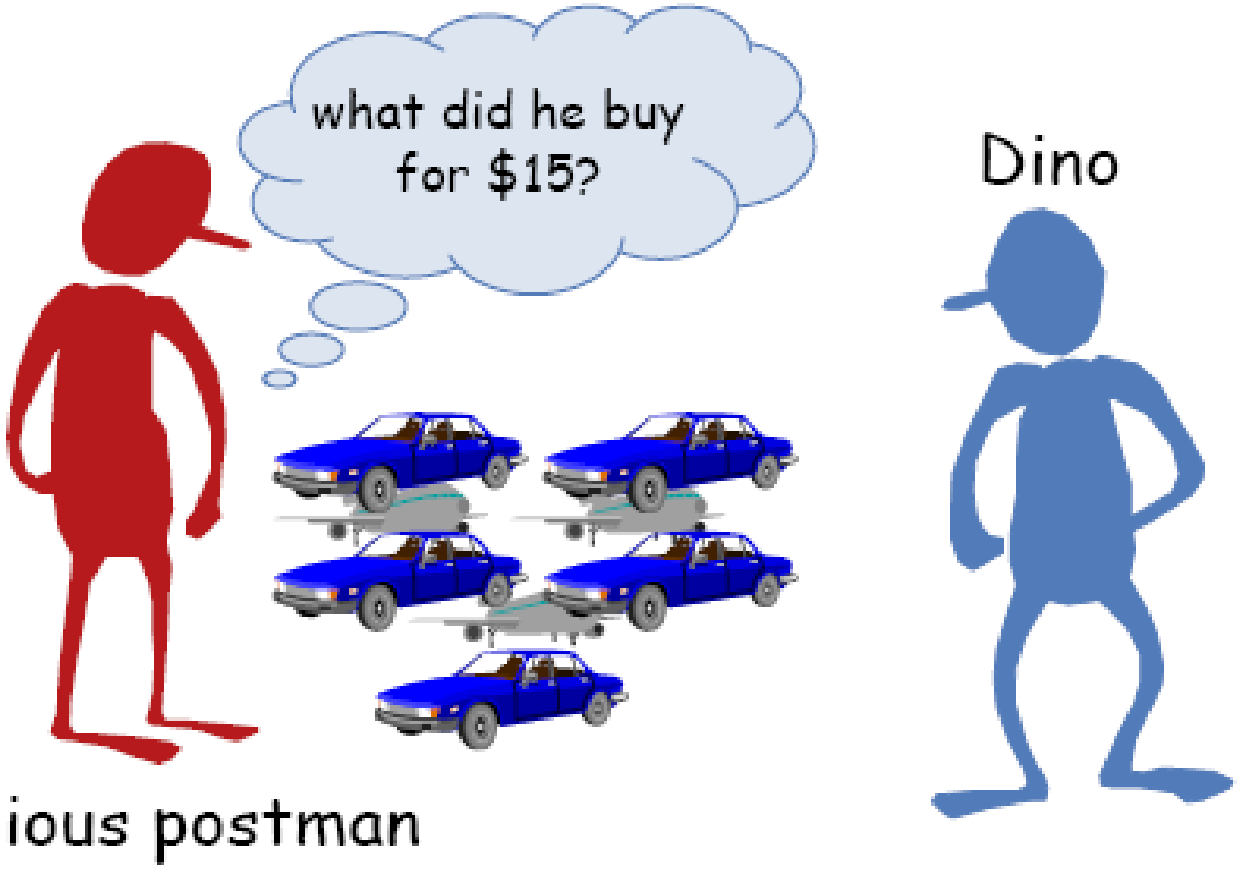
car = $3



plane = $5

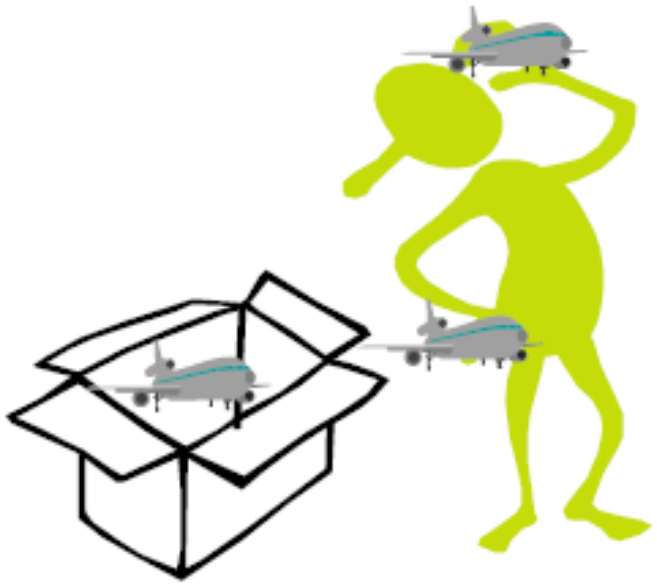
Dino buys toys from Jack

The postman wants to know what Dino bought for $15



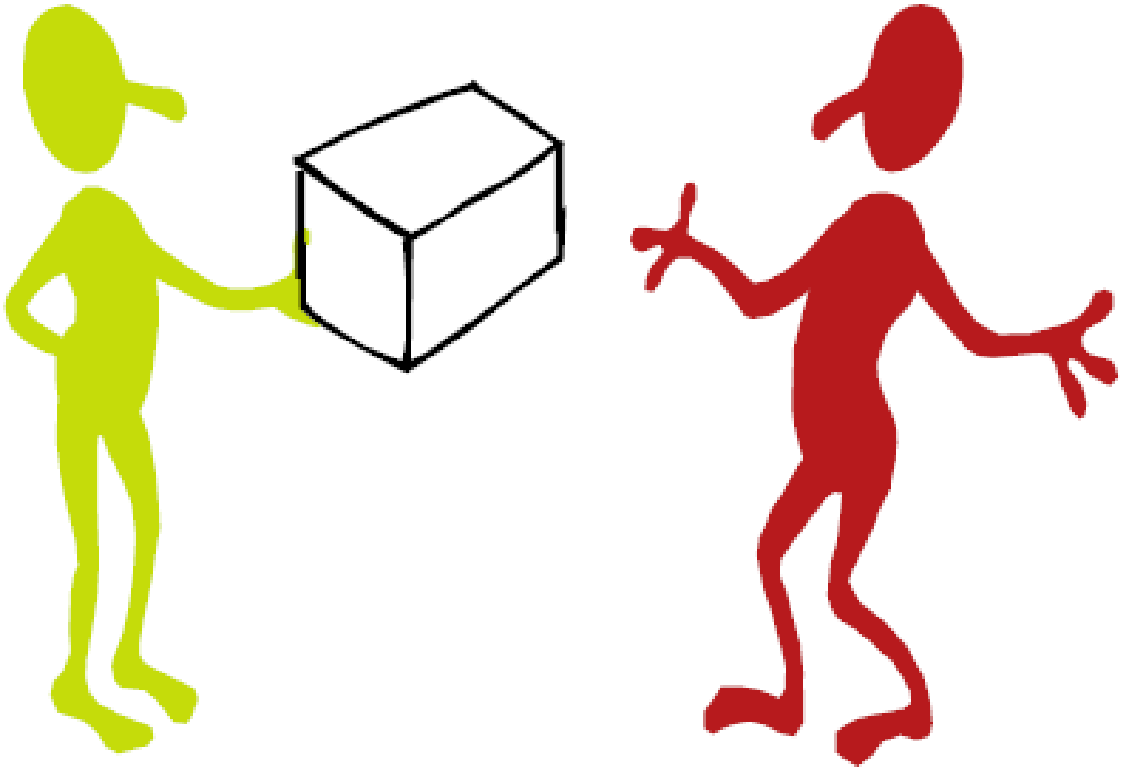
Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

In the meanwhile Jack prepares the DHL



Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

and gives it to the postman

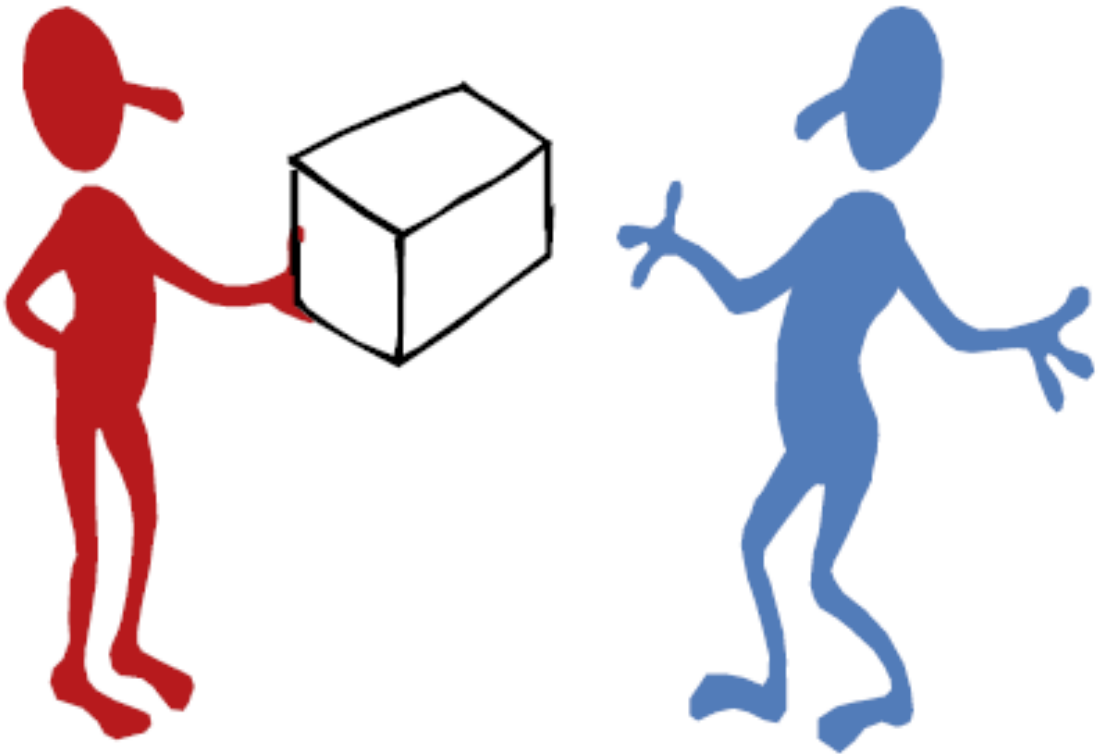


Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

Who kicks it strong enough to break one toy

Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

and gives it to Dino



Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

a week later he monitors Dino's postal order...

= 4 × 3 = $12

= 2 × 5 = $10

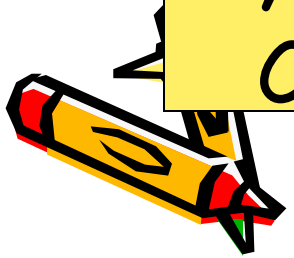
Lesson learned: Fault attacks can also extract secrets from tokens!

Hardware faults can have various sources: voltage glitches, light beams, laser beams...

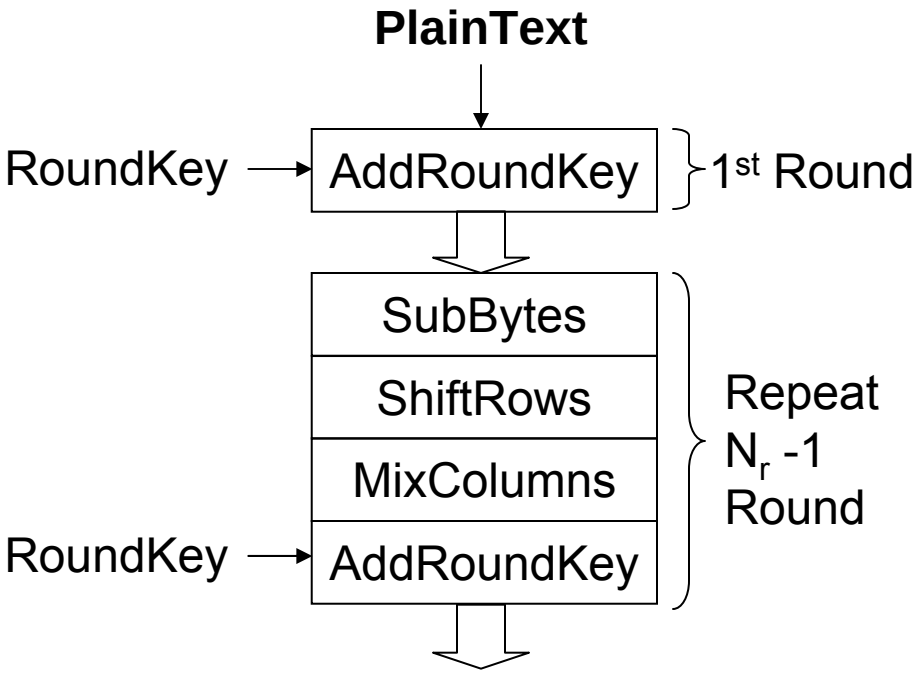
Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

Fault Attacks on Block Ciphers

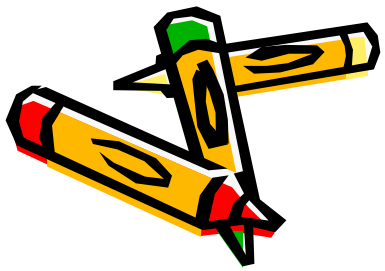
- Attacks based on induction of faults
  - Both accidental and intentional
- First conceived in 1996 by Boneh, Demillo and Lipton
- E. Biham developed Differential Fault Analysis (DFA) attacker DES
- Optical fault induction attacks : Ross Anderson, Cambridge University – CHES 2002

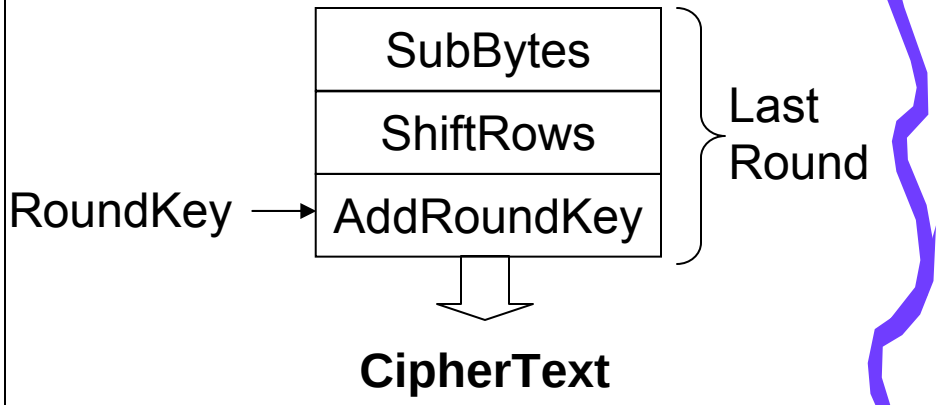


AES Algorithm

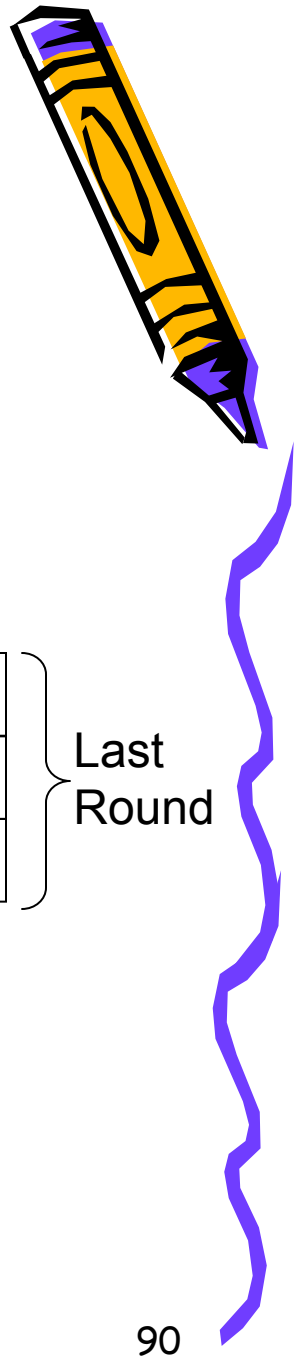


First 9 Rounds





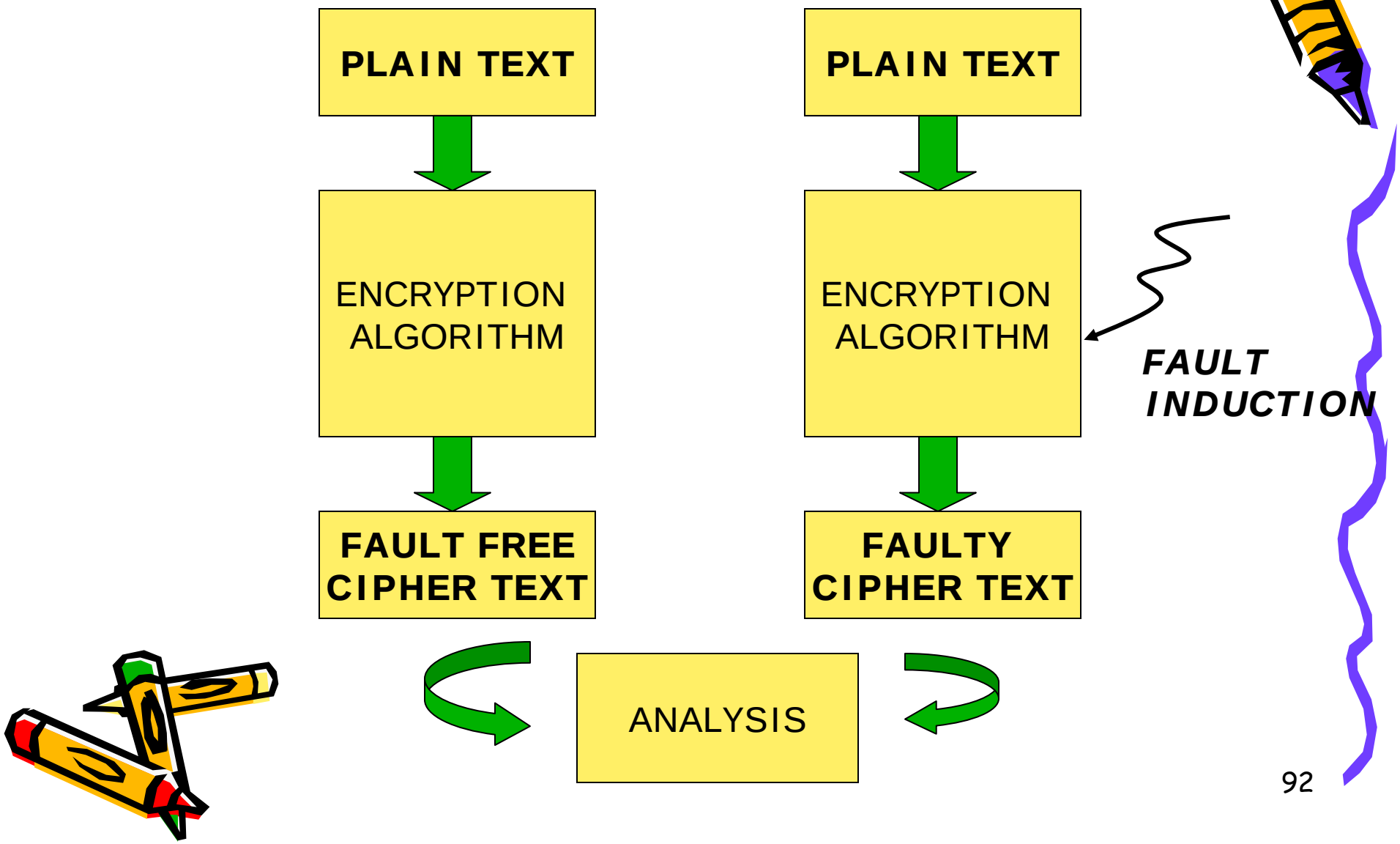
Last Round



Research on Fault Attacks on AES

- Christophe Giraud, 2003: Byte level faults to the input of 9th round, 250 faulty CTs.
- Blomer etal, 2003: 128 to 256 faulty CTs required
- P. Dusart etal : DFA on AES, bytewise fault between 8th and 9th rounds, 40 faulty CTs required
- Piret etal, Between 8th and 9th rounds, 2 faulty CTs required
- Recently fault attacks have been developed by NTT labs, Japan which exploits the key-scheduling algorithm

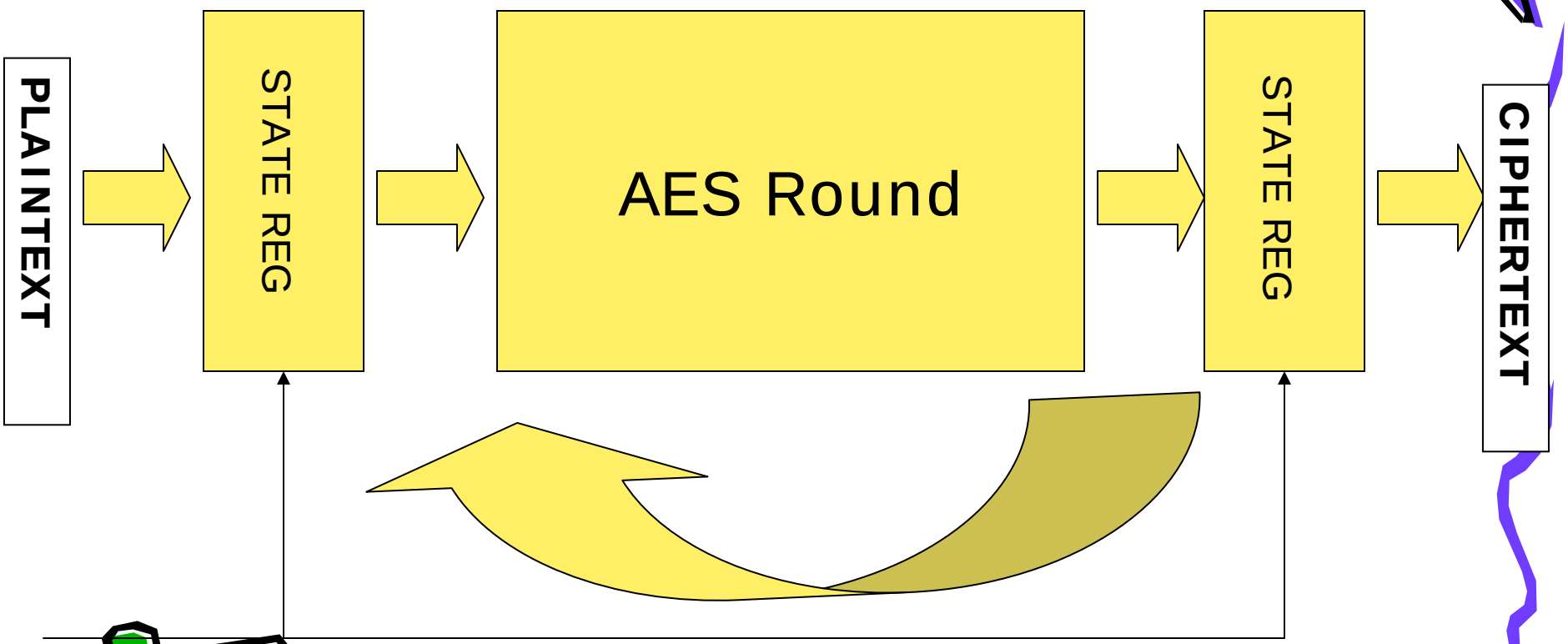
Illustration of a Fault Attack

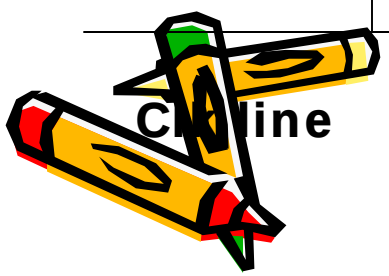


Fault Model Used

- **Single Byte Fault**
  - Attacker induces fault at the input of the 8th round in a single byte
  - Fault value should be non-zero but can be arbitrary
- Relaxing the requirements make the attack more **practical**
  - No knowledge required of the fault value
  - Lesser bytes needed to be faulty
  - Lesser faulty cipher texts required

A Practical Scenario: An Iterated AES Architecture



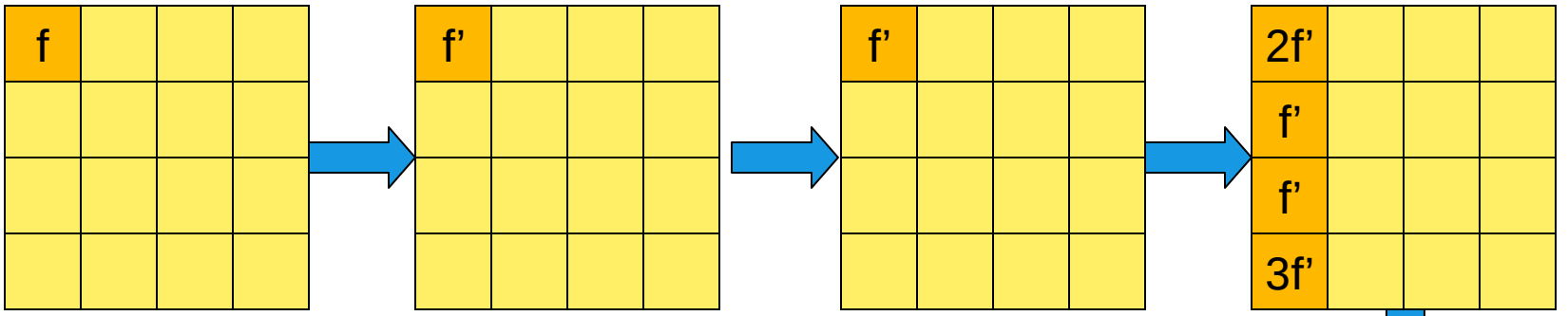


An Attacker can time his attack by counting the number of clock cycles : ***Control on Fault Timing***

Principle of the Attack

- First, consider a single byte arbitrary fault at the input of the 9^{th} round.
- ISB : Inverse Sub Byte
- We develop a filter, which takes as input the faulty and fault free ciphertext.

Propagation of Fault Induced

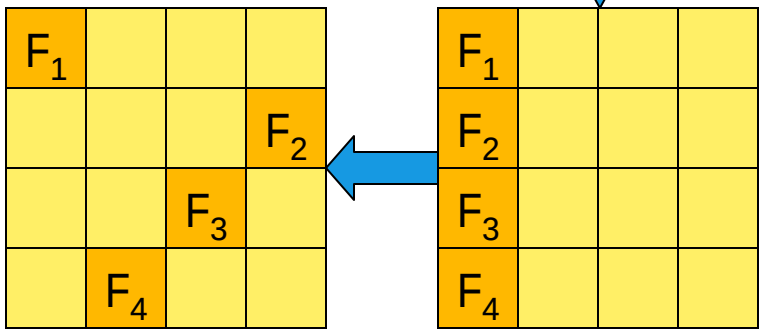


9th Round Byte Sub

9th round ShiftRow

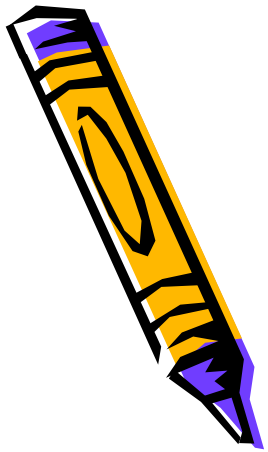
9th Round MixColumn

10th Round ByteSub

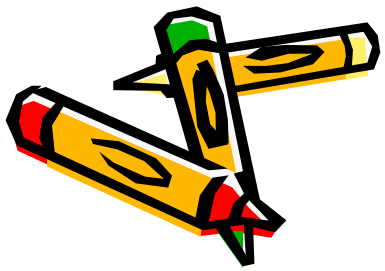


10th Round ShiftRow

The patterns gives the following equations:



- $ISB(x_1+K_1)+ISB(x_1+F_1+K_1)=$
  $2[ISB(x_2+K_2)+ISB(x_2+F_2+K_2)]$
- $ISB(x_2+K_2)+ISB(x_2+F_2+K_2)=$
  $ISB(x_3+K_3)+ISB(x_3+F_3+K_3)$
- $ISB(x_4+K_4)+ISB(x_4+F_4+K_4)=$
  $3[ISB(x_2+K_2)+ISB(x_2+F_2+K_2)]$



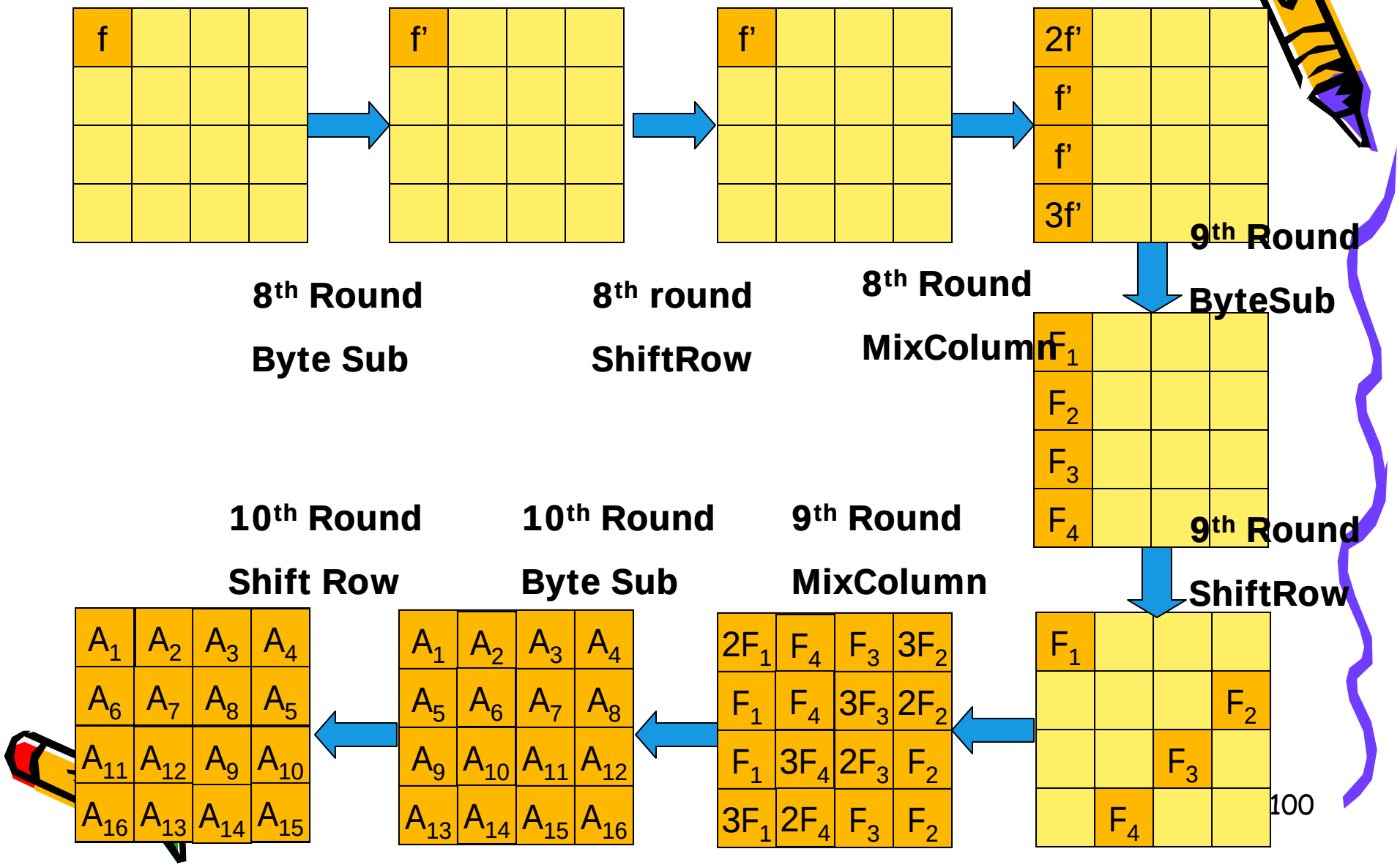
Important Points

- No dependency on the fault value.
- Finds out the key using two faulty encryptions with a probability of around 0.99
- Rest of the cases a third faulty cipher text is needed
- Time Complexity is 2^{16}.
- One byte fault reveals 4 key bytes.
  - To obtain the entire key, 4 faulty cipher texts needed.

When the fault is induced at the 8th Round…

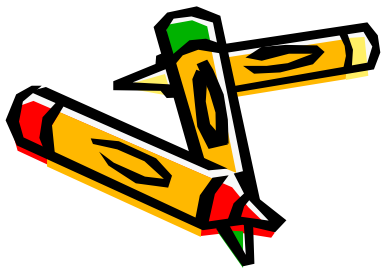
- Fault is induced at the input of 8th round
- A one byte disturbance creates a 4 byte fault at the input of the 9th round
- Let us trace the disturbance through the last 3 rounds
- Equations of similar nature…

Propagation of Fault Induced



The patterns gives the following equations:

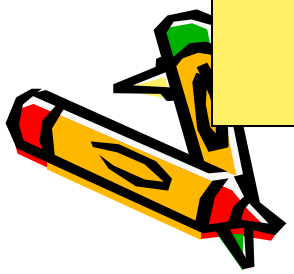
- $ISB(x_1+K_{00})+ISB(x_1+A_1+K_{00})=$
  $2[ISB(x_8+K_{13})+ISB(x_8+F_2+K_2)]$
- $ISB(x_8+K_{13})+ISB(x_8+A_5+K_{00})=$
  $ISB(x_{11}+K_{22})+ISB(x_{11}+A_9+K_{22})$
- $ISB(x_{14}+K_{31})+ISB(x_{14}+A_{13}+K_{31})=$
  $3[ISB(x_8+K_{13})+ISB(x_8+A_5+K_{13})]$



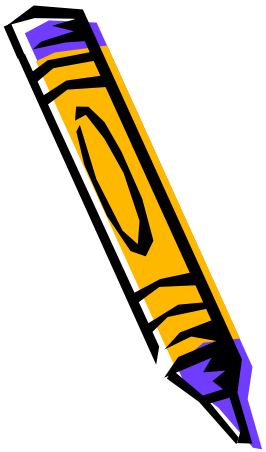
For the other key bytes…

Similar equations are derived for the other key bytes

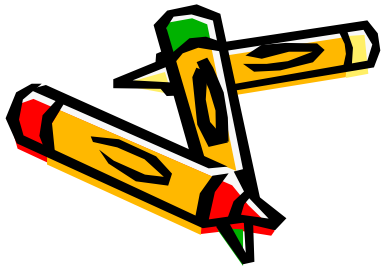
For all the equations the worst case complexity is around 2^8 to 2^9.
Two faulty cipher text pairs reveal the exact key with a high probability.



Can the attack work with one faulty cipher text?



- With one faulty cipher text:
  - Number of possible values per 4 bytes of the key is around 2^8.
  - There are 2^{32} possible candidates for 128 bits of the AES key.
  - Brute force key is thus possible!



Why 2^{32}?

$$
\begin{aligned}
2\,\delta_1 &= S^{-1}(x_1 \oplus k_1) \oplus S^{-1}(x'_1 \oplus k_1) \\
\delta_1 &= S^{-1}(x_8 \oplus k_8) \oplus S^{-1}(x'_8 \oplus k_8) \\
\delta_1 &= S^{-1}(x_{11} \oplus k_{11}) \oplus S^{-1}(x'_{11} \oplus k_{11}) \\
3\,\delta_1 &= S^{-1}(x_{14} \oplus k_{14}) \oplus S^{-1}(x'_{14} \oplus k_{14})
\end{aligned}
$$

- There are 4 solutions to the equation:
  $S^{-1}(x) \wedge S^{-1}(x \wedge a)=\beta$
- Thus there are 4 values of k_1, k_8, k_{11}, k_{14} each which satisfies the equations.
  - thus the size of the list is 2^{16}
  - but there are repetitions in the list
  - the maximum size of the list for which there is no repetition with a high probability follows from Birthday Paradox. It is thus 2^8
  - thus the total size of AES key is 2^{32}

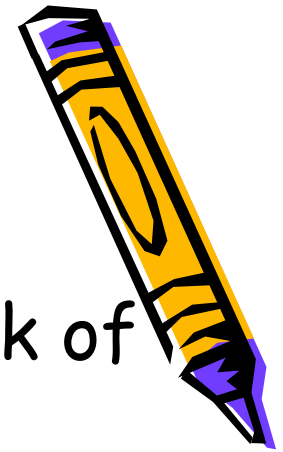
Comparison of Existing Fault Attacks

Reference	Fault Model	Fault Loc.	#Faulty CT
Blomer	Force 1 bit to 0	Chosen	128
Giraud	Switch 1 bit	Any bit of chosen bytes	50
Giraud	Disturb 1 byte	Anywhere among 4 bytes	250
Dusart	Disturb 1 byte	Anywhere between last 2 MixColumn	40
Piret	Disturb 1 byte	Anywhere between 7^{th} & 8^{th} round MixColumn	2
This Paper	Disturb 1 byte	Anywhere between 7^{th} round MixColumn and last round input	2

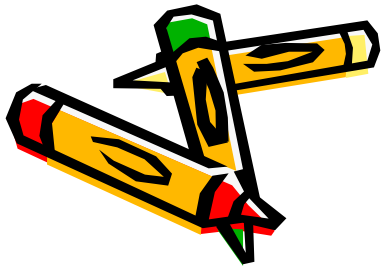
Comparison with existing fault attacks exploiting key scheduling

Reference	No. of fault injection points	No. of faulty encryptions	Brute force search
Takahashi (NTT Lab)	1	2	2^{18}
	2	4	2^{16}
	3	7	0
Takahashi (NTT Lab)	1	2	2^{40}
	3	7	0
Our Attack	1	2	0
	1	**1**	**2^{32}**

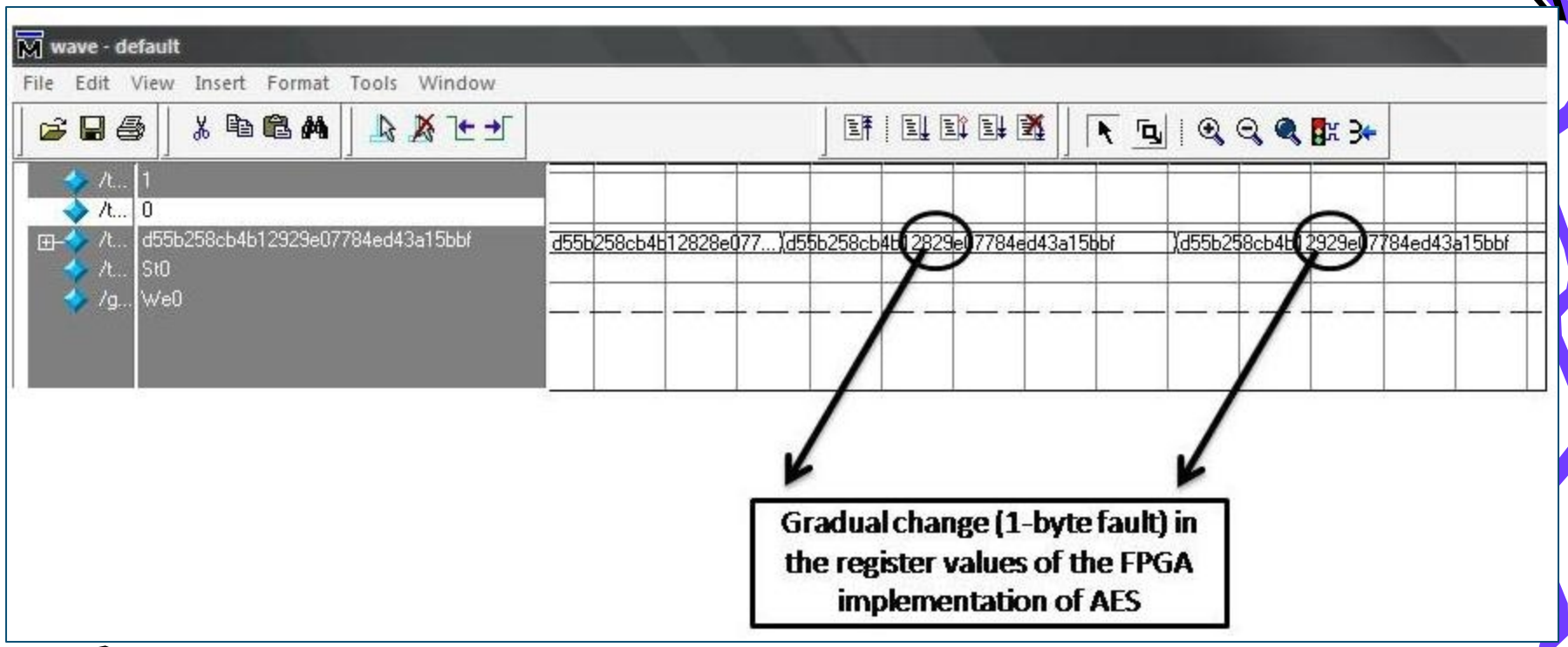
Conclusions



- The work proposes the strongest attack of this class (till date) in literature
  - Using a single byte fault at the input of the 8th round, AES can be broken with a brute force search of:
    - 2^{32} if the byte location is known and
    - 2^{36} if the byte location is unknown.
- Currently, we are working to mount the attack on real life FPGA implementations of AES using less sophisticated techniques, like clock glitching.



Effect of clock glitches…

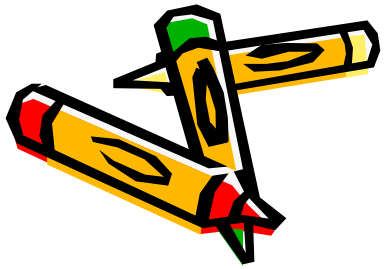


Future Scopes of Research

- Improving the fault attack.
- Performing the fault attack on FPGA by clock glitching.
- Developing counter-measures against Fault attacks:
  - Error correcting Codes may be useful.
- Evaluating whether the fault attack counter-measures reveal other side channel information

Conclusion

- Side Channel Analysis is an extremely important topic
- We have studied some of the well known side channels in cryptography.
- Need for the development of formalisms:
  - Information Theory
- Side Channel Analysis is now not a topic for only engineers: theoreticians are also worried!



Moral of the Story

References

- Stefan Mangard, Elisabeth Oswald, and Thomas Popp, **The DPA Book:"Power Analysis Attacks - Revealing the Secrets of Smart Cards", Springer, 2006**

- Paul C. Kocher,"***Timing Attacks on Implementations of Diffie—Hellman***", *RSA, DSS, and Other Systems, Crypto 96*

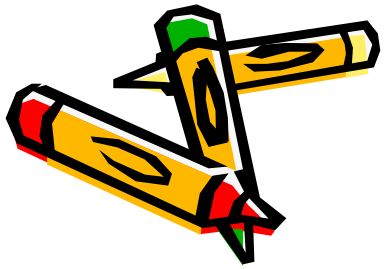
- Paul C. Kocher, Joshua Jaffe, Benjamin Jun, "***Differential Power Analysis***", *CRYPTO '99, 1999*

References

- Jean-Sébastian Coron and Louis Goubin, "***On Boolean and Arithmetic Masking against Differential Power Analysis***", CHES 2000

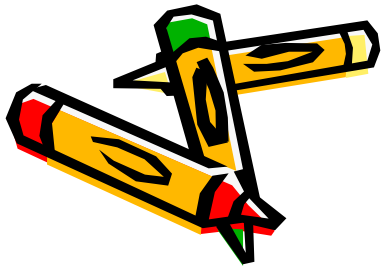
- Stefan Mangard and Thomas Popp and Berndt M. Gammel, "***Side-Channel Leakage of Masked CMOS Gates***", CT-RSA 2005

- Stefan Mangard and Thomas Popp and Berndt M. Gammel, "***Side-Channel Leakage of Masked CMOS Gates.***", CT-RSA 2005



References

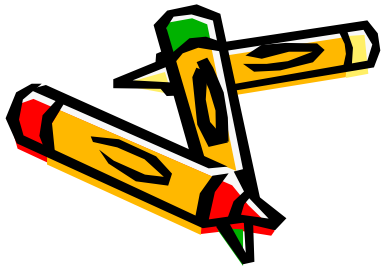
- Stefan Mangard and Norbert Pramstaller and Elisabeth Oswald, "**Successfully Attacking Masked AES Hardware Implementations**", CHES 2005
- Stefan Mangard and Kai Schramm, "**Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations**", CHES 2006
- M. Alam, S. Ghosh, M.J. Mohan, D. Mukhopadhyay, D.R. Chowdhury, and I.S. Gupta, **"Effect of glitches against masked AES S-box implementation and countermeasure"**, IET Information Security, 3(1), 2009



References

- K. Kumar, D. Mukhopadhyay and D. RoyChowdhury, **"Design of a Differential Power Analysis Resistant AES S-Box"**, INDOCRYPT 2007, pp 373-383.

- S. Burman, D. Mukhopadhyay and V. Kamakoti, **"LFSR Based Stream Ciphers are vulnerable to Power Attacks"**, INDOCRYPT 2007, pp 384-392.

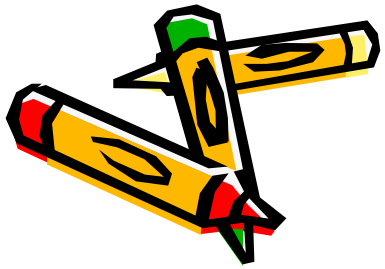


References

- Dag Arne Osvik and Adi Shamir and Eran Tromer, **"Cache attacks and Countermeasures: the Case of AES"**, Cryptology ePrint Archive, 2005

- Daniel J. Bernstein, ***"Cache-timing attacks on AES"***, 2005, http://cr.yp.to/antiforgery/

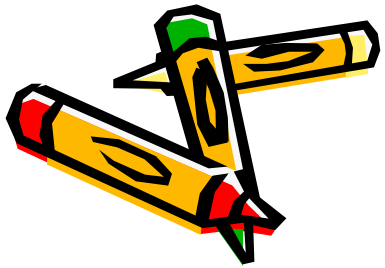
- **Chester Rebeiro and Debdeep Mukhopadhyay, "Pinpointing Cache Timing Attacks on AES", To appear in VLSI 2010**



References

- Chester Rebeiro, Debdeep Mukhopadhyay, Junko Takahashi and Toshinori Fukunaga, **"Cache Timing Attacks on Clefia"**, Indocrypt 2009.

- AES and Combined Encryption/ Authenti-cation Modes, http://gladman.plushost.co.uk/oldsite/AES/index.php



References

- Joan Daemen, Vincent Rijmen: **The Design of Rijndael:** AES - The Advanced Encryption Standard Springer 2002

- Gilles Piret and Jean-Jacques Quisquater **A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD**, CHES 2003

- Dan Boneh, Richard A. DeMillo and Richard J. Lipton: **On the importance of checking cryptographic protocols for faults, Journal of Cryptology 2001.**

References

- Eli Biham, Adi Shamir, "**Differential Fault Analysis of Secret Key Cryptosystems**", Crypto 1997.

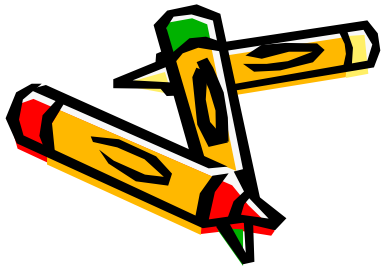
- Oliver Kömmerling and Markus Kuhn "**Design Principles for Tamper-Resistant Smartcard Processors**", USENIX 1999.

- Sergei P. Skorobogatov and Ross J. Anderson "**Optical Fault Induction Attacks**", CHES 2002.

References

- Bar-El, H. Choukri, H. Naccache, D. Tunstall, M. Whelan, C. , **"The Sorcerer's Apprentice Guide to Fault Attacks"**, FDTC 2006

- Debdeep Mukhopadhyay, **"An Improved Fault Based Attack of the Advanced Encryption Standard"**, Africacrypt 2009



References

- Debdeep Mukhopadhyay, **"A New Fault Attack on the Advanced Encryption Standard Hardware"**, ECCTD 2009, Antalya, Turkey (*Invited Paper*).