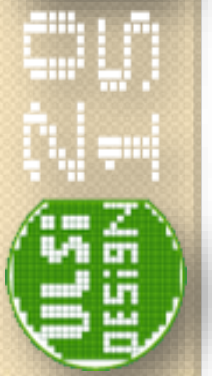
Hardware Security

Debdeep Mukhopadhyay

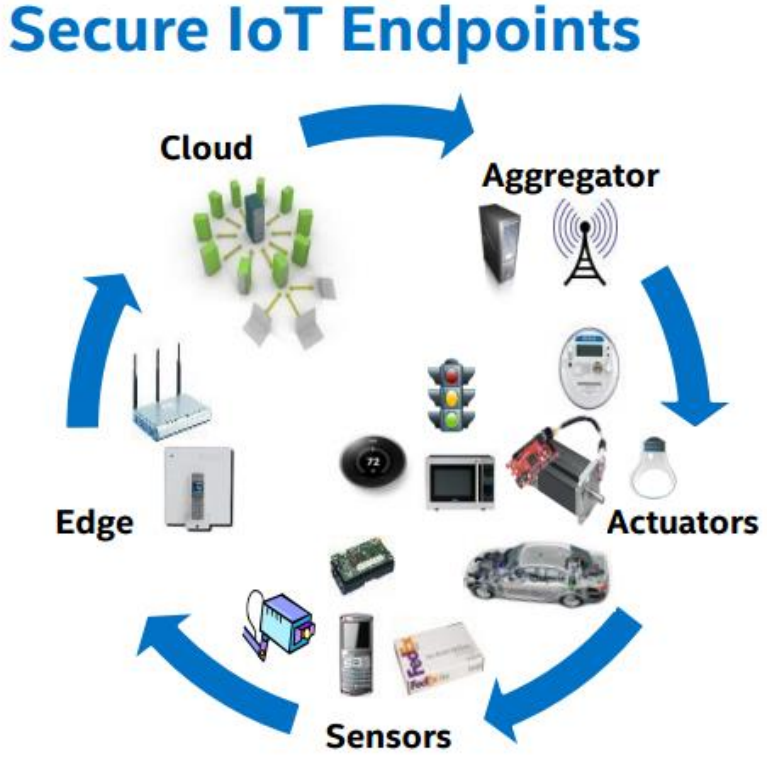
Secured Embedded Architecture Laboratory (SEAL)
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur, West Bengal, INDIA – 721302

E-mail: debdeep.mukhopadhyay@gmail.com

An Overview



Hardware Root of Trust



Endpoints must underpin security
- Hardware Root-of-Trust
- Secure Boot / Secure Update
- Lightweight Trustworthy Execution

Diversity of Endpoints is a huge challenge

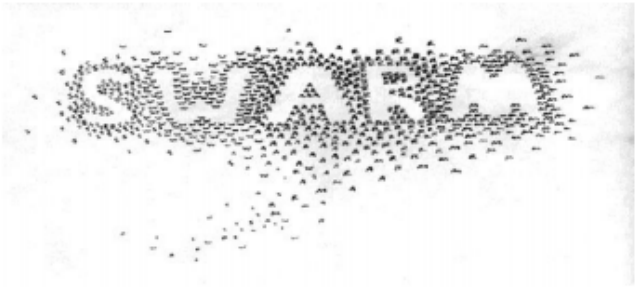
Even Motes may need to establish Trust

Just enough security for each end points

Source: Patrick Koeberl – Security Architect at Intel Labs, Intel Corporation, IDF14

Trustworthy Handling of large Number of Devices

Secure IoT Connections



Many IoT usages will involve ensembles of devices

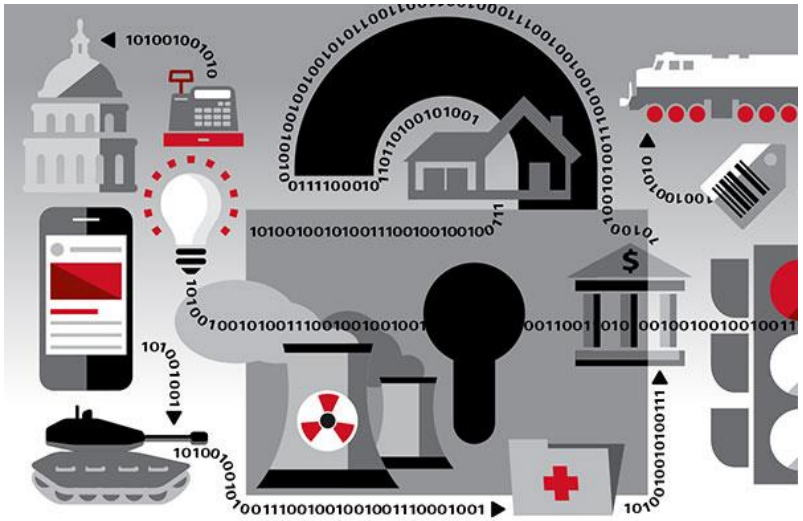
- Secure Device-to-Device Pairing and Communications
- Trustworthy and Flexible Grouping

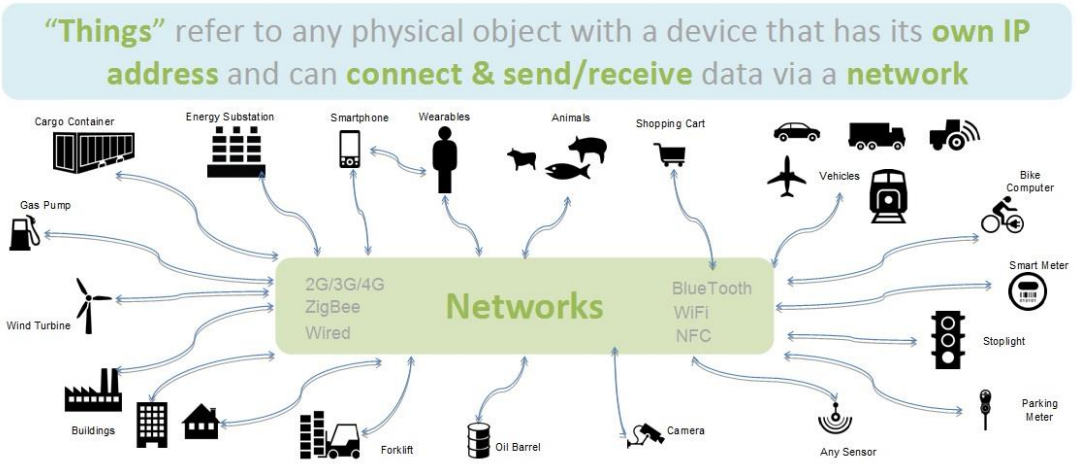
Usability is Key

- Eliminate need for Manual Configuration
- Intuitive for SysAdmin, Seamless for normal users

Source: Patrick Koeberl – Security Architect at Intel Labs, Intel Corporation, IDF14

Trust in Cyber Physical Systems





- 50 Billion Devices to be connected by 2020!
- Devices need to trust the owner and also each other.
- Devices connected through heterogeneous network, and are resource constrained.

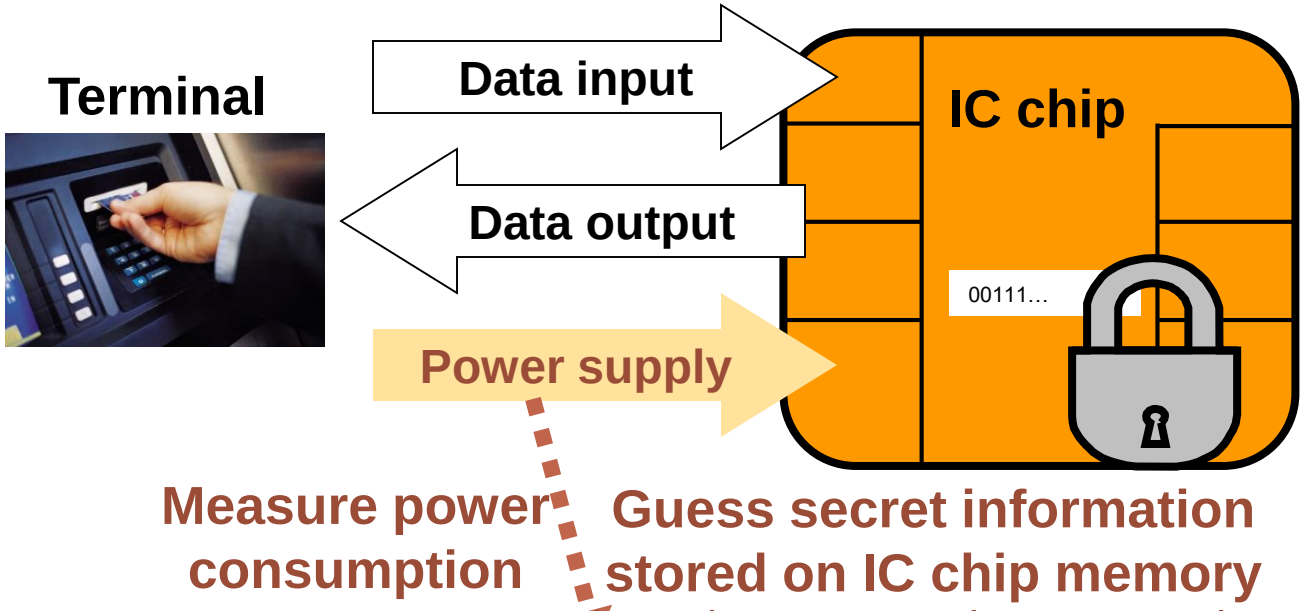
Whom can you Trust?

- **What do we know about the device?**
  - **Is it running the correct software?**
  - **Is it genuine?**
- **We need to guarantee:**
  - **Integrity**
  - **Privacy**
  - **Quality**
- **IoT endpoints operate under resource constraints:**
  - **CPU**
  - **Memory**
  - **Energy**
  - **Communications**
- **Traditional Security features do not scale down!**
  - **The Trusted Computing Base (TCB) must be as small as possible!**

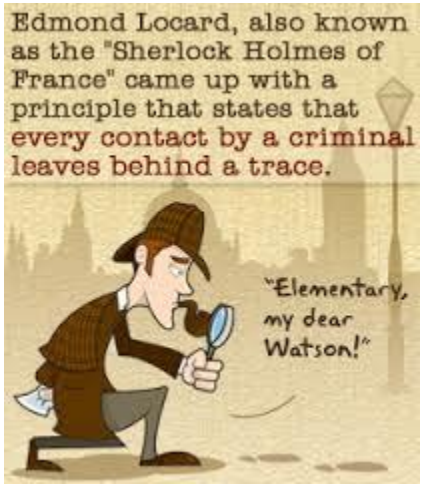
Trust is a major enabler for IoT

Are there more optimal solutions for the hardware root of trust?

Threats from Side Channel Attacks

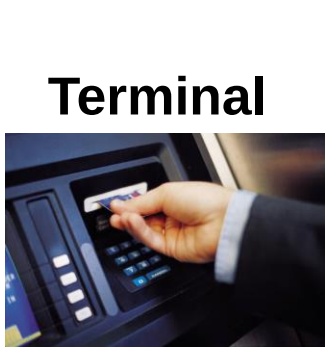


Side Channel Attacks



Power Attacks

Strong cryptographic algorithms are just the beginning!



Terminal

Data input

Data output

Power supply

IC chip

00111…

Measure power consumption

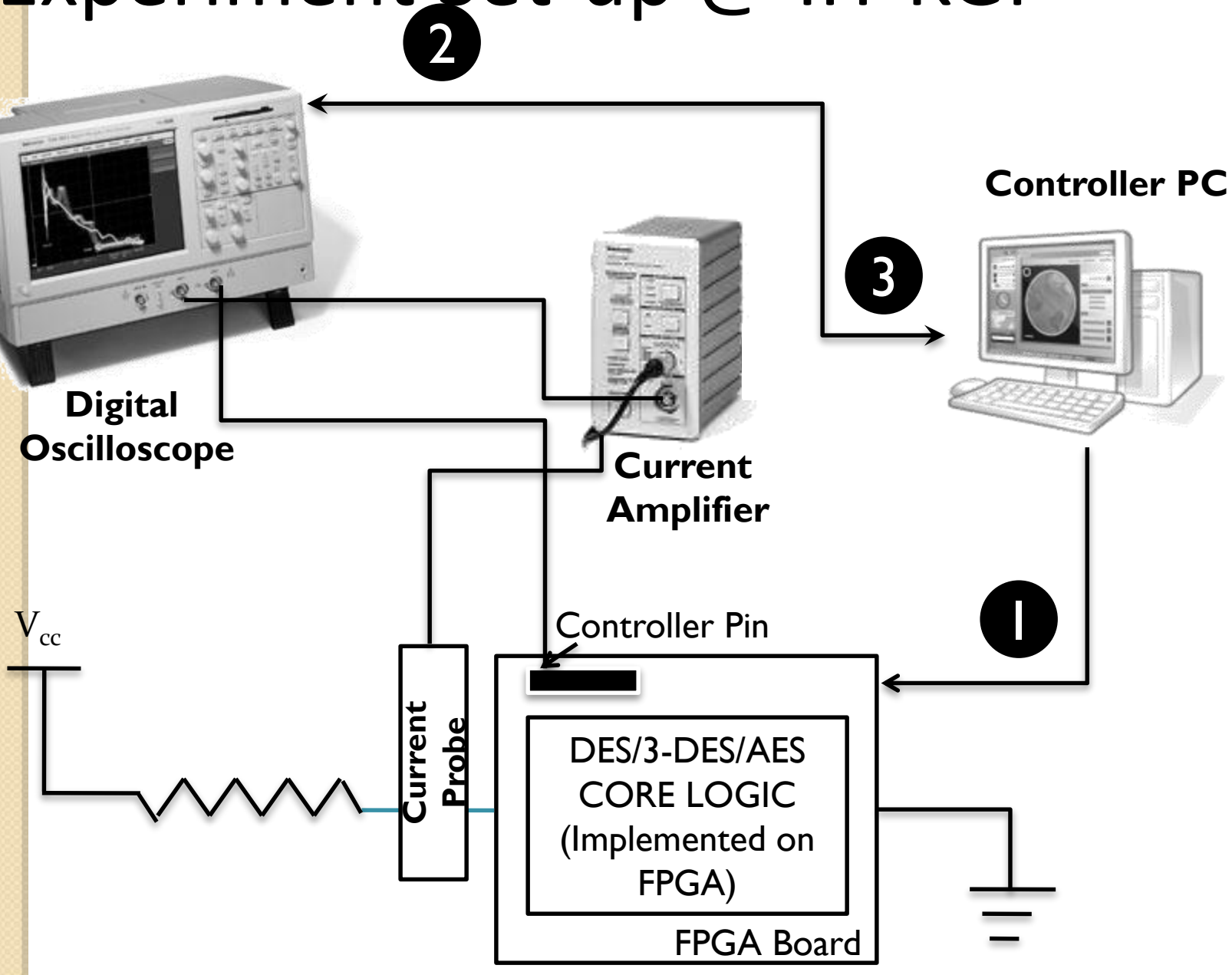
Guess secret information stored on IC chip memory

Power consumption

Secret information

0	0	1	1	1

Experiment Set-up @ IIT KGP



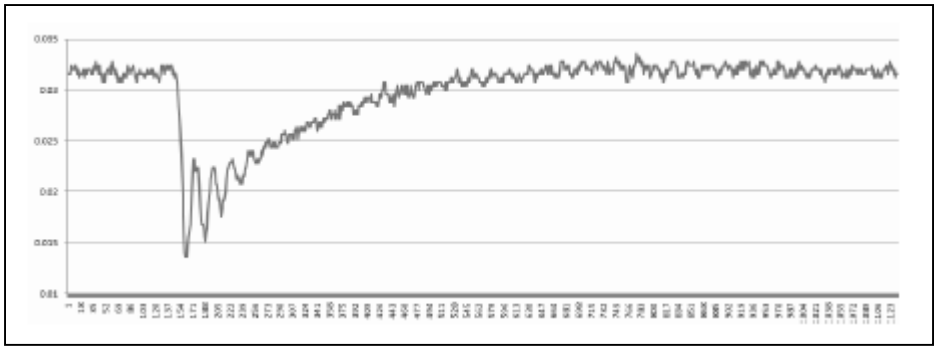
Power Attacks

- 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

A Power Trace



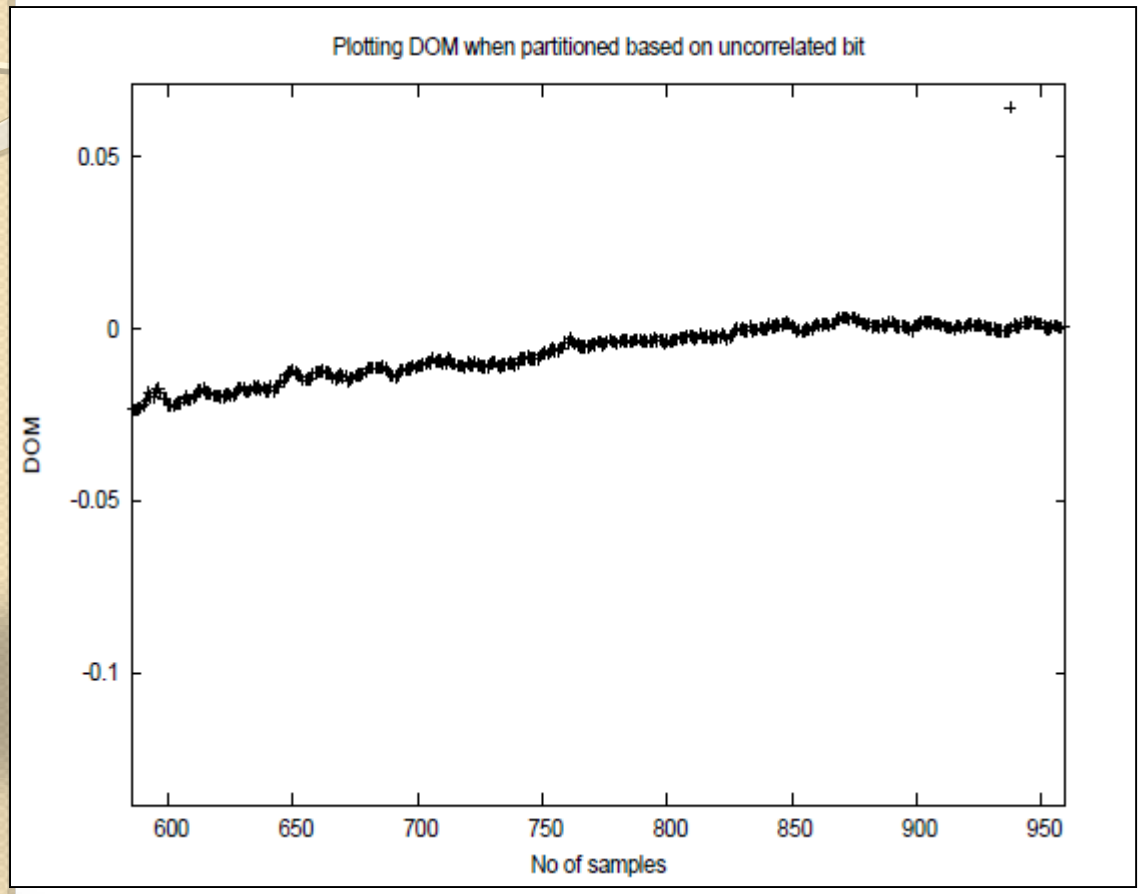
- Power Trace of a round of AES.
- Observe the variation of power values.
- The variations occur because of the operation dependence of power: leads to SPA.
- The variations also occur because of data dependence of power: leads to DPA.

Correlation of Power with bits

s	HW(s)	Target bit (LSB)
0000	0	0
0001	1	1
0010	1	0
0011	2	1
0100	1	0
0101	2	1
0110	2	0
0111	3	1
1000	1	0
1001	2	1
1010	2	0
1011	3	1
1100	2	0
1101	3	1
1110	3	0
1111	4	1

- Assume power leakage follows Hamming Weight.
- Divide the HW(s) into two bins:
  - 0 bin: when LSB is 0
  - 1 bin: when LSB is 1
- Difference-of-Mean (DoM)=20/8-12/8=1

When the partitioning is done wrt. an uncorrelated bit?



- Parititioning done by bits simulated using *rand* function in C.
- Observe the DoM is close to 0, as expected!

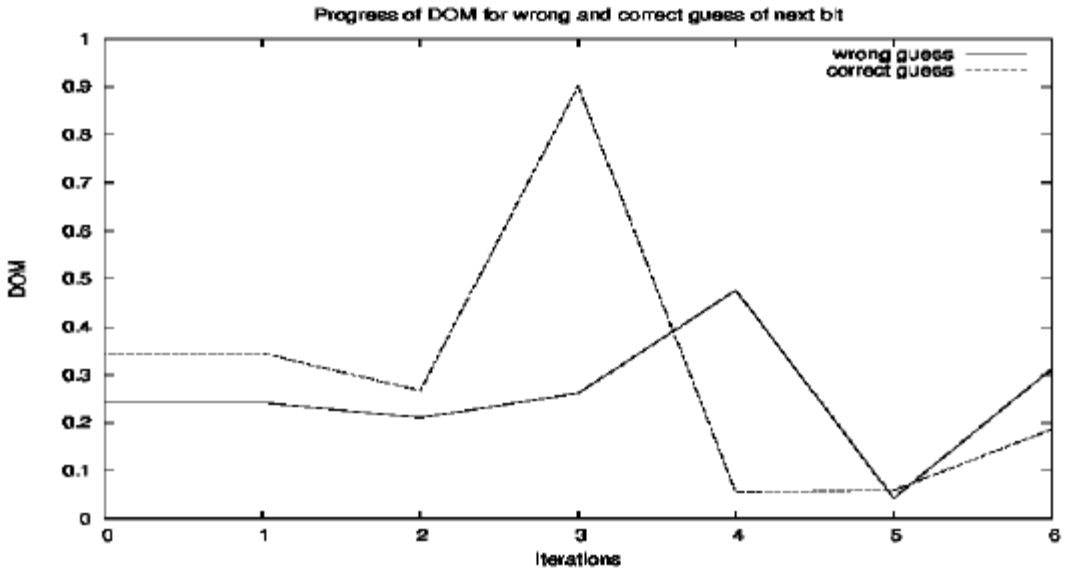
A Toy DPA

- Consider the operation $z=y^x \bmod 256$
- Assume attacker knows first 4 bits of the secret, x.
- Probability of guessing the next bit of x is ½ (with no side channel information).
- Now we assume, the attacker varies y and obtains several power traces.
  - We simulate them through Hamming Weights.

A Toy DPA

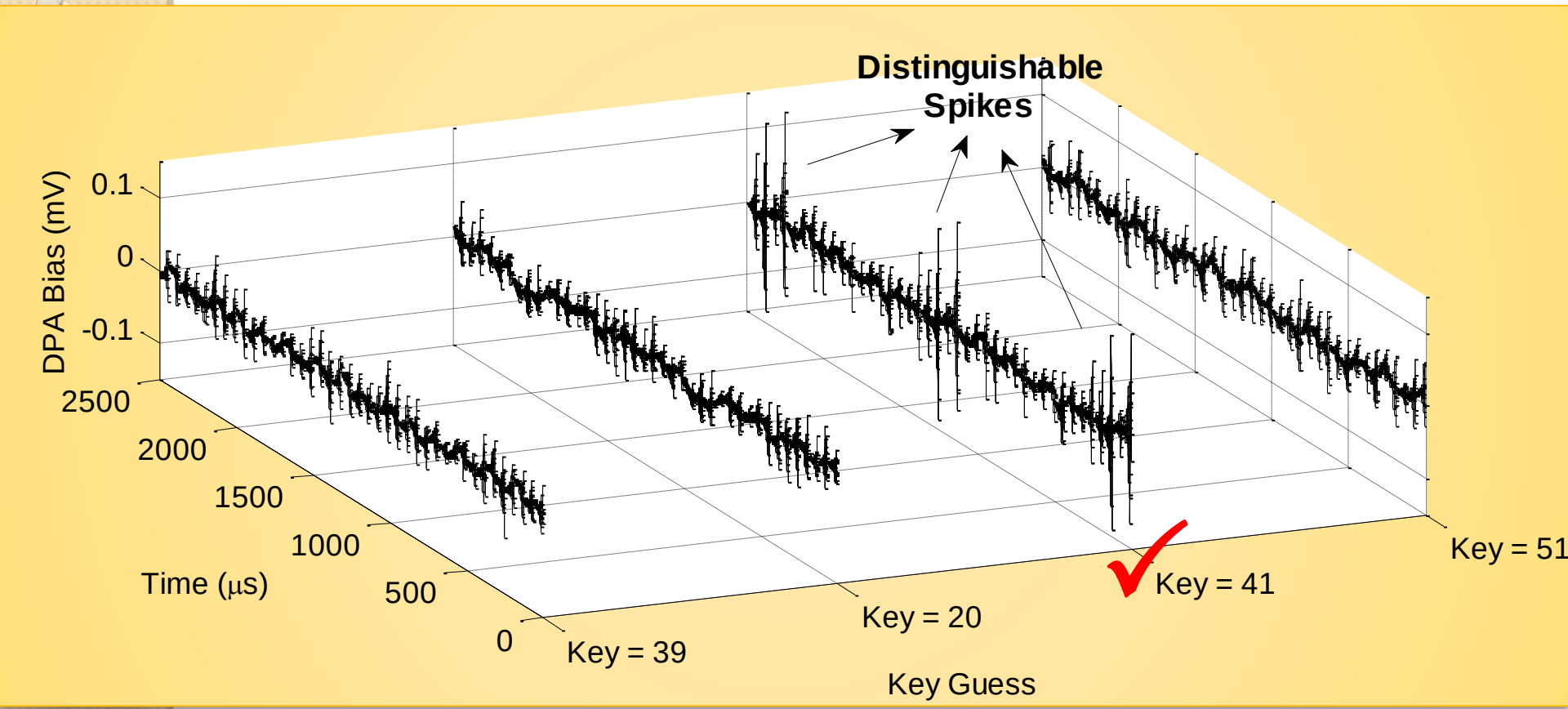
- For a given y, the attacker now guesses the next bit and computes the *probable* s after the 3rd iteration (note that square and multiply starts from bit 6 to bit 0).
- Based on the LSB of y, the corresponding trace is put into the 0 bin or 1 bin.
- For every guess (there are 2 guesses) the DoM is computed and plotted.
- The correct guess is expected to provide large DoM.

A Toy DPA



- Correct Key is 0x8F
- Next bit is thus 1.
- DoM computed after 2^{10} traces.
- Significant difference: 0.9 vs 0.2!

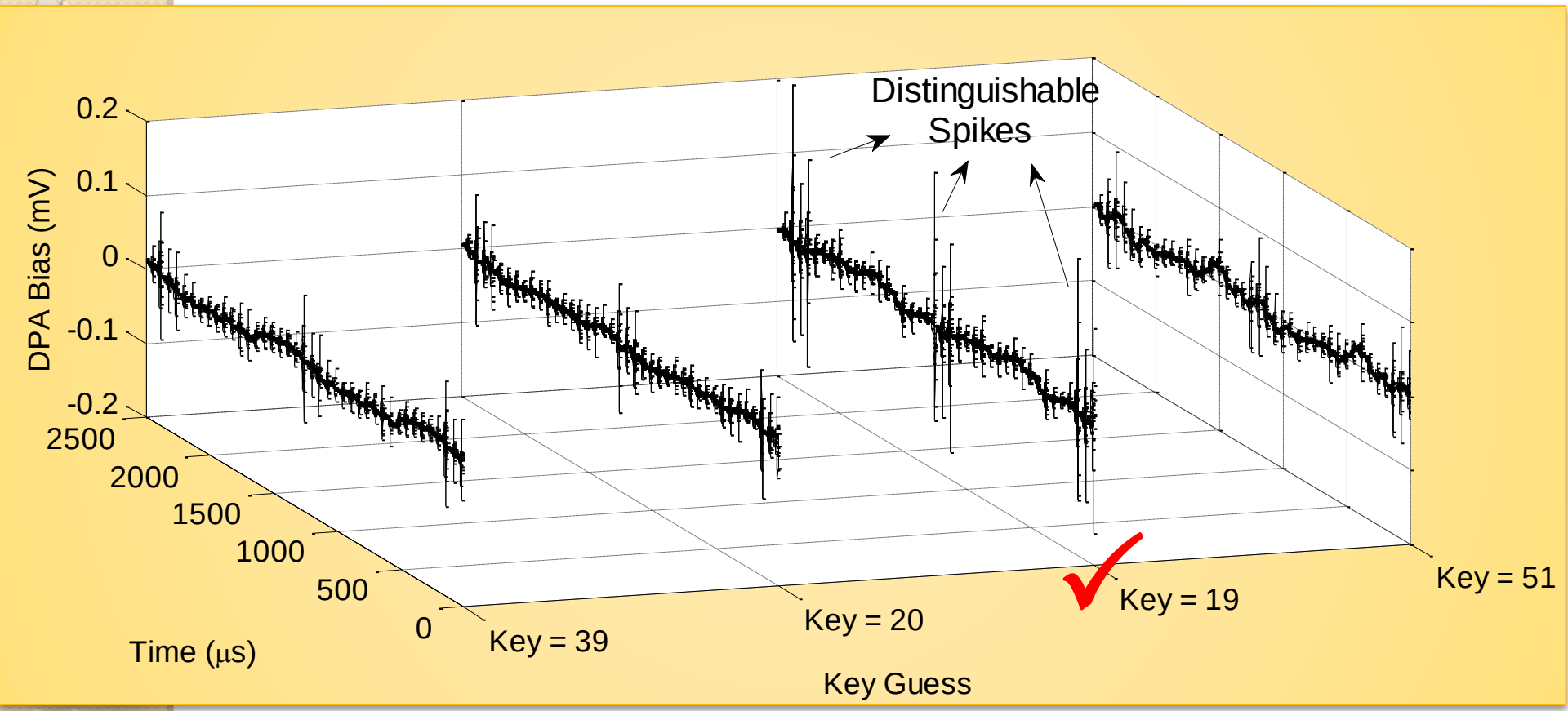
Differential Power Analysis Attacks on DES



3D Differential Plot

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

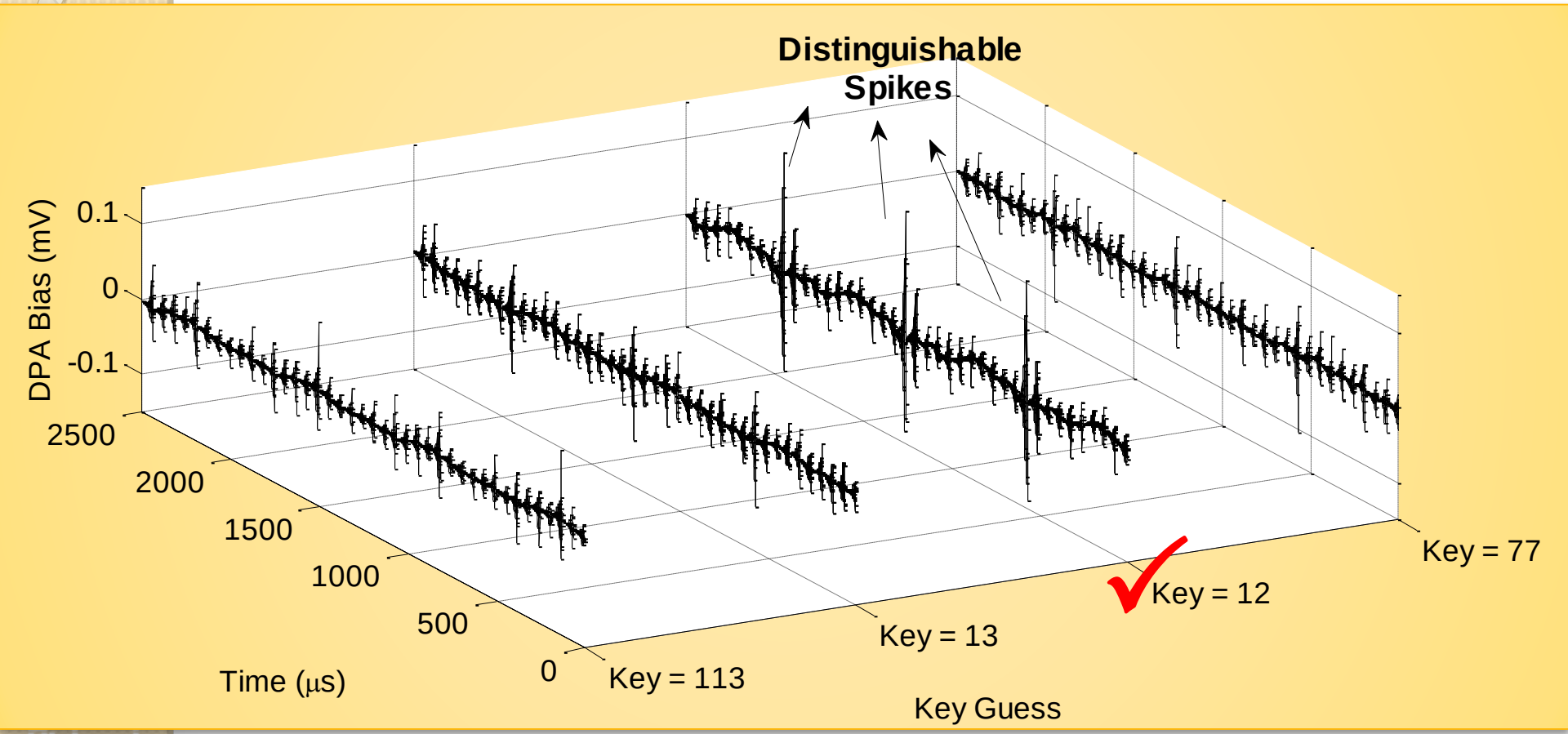
Differential Power Analysis Attacks on 3-DES



3D Differential Plot

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

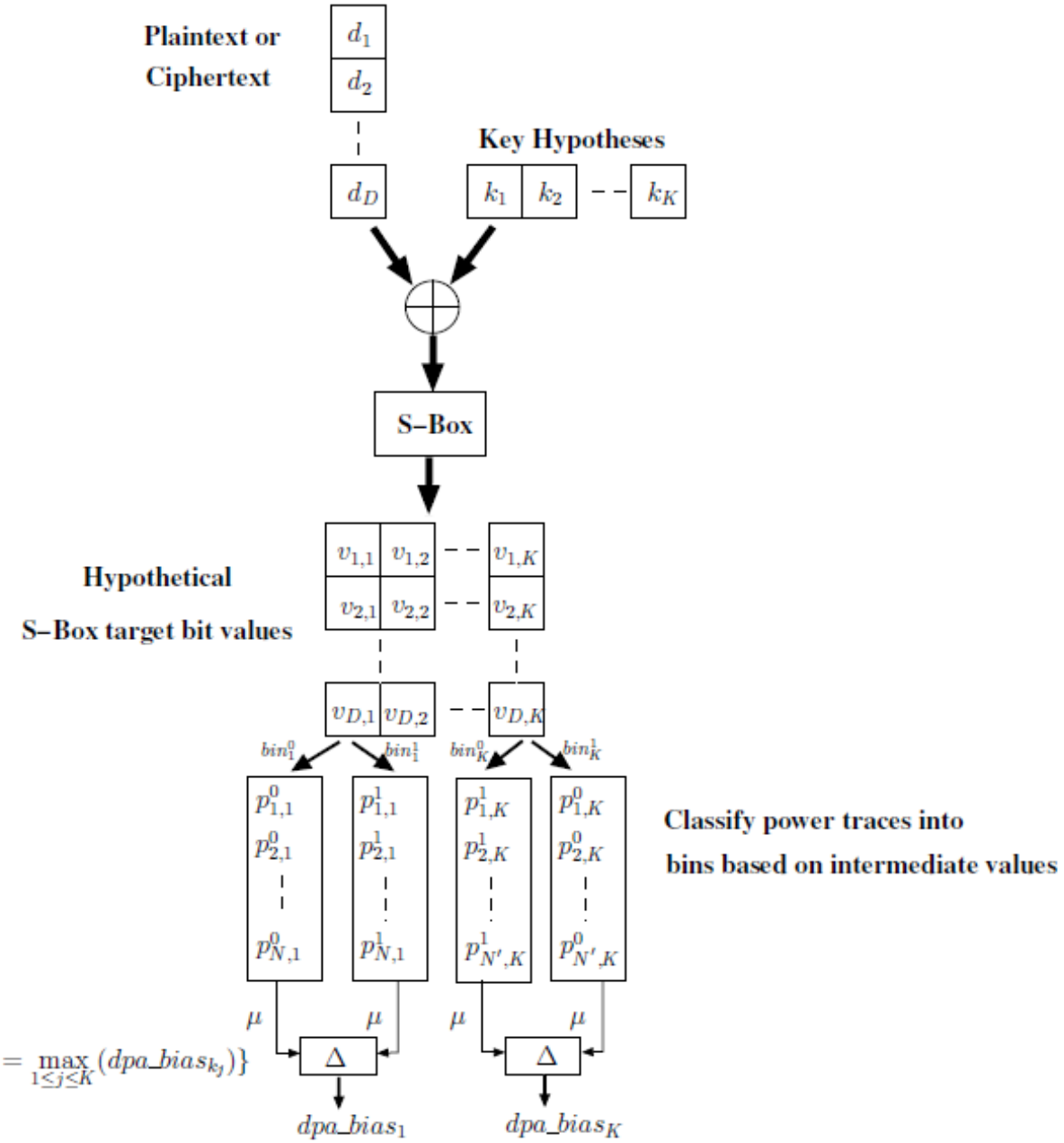
Differential Power Analysis Attacks on AES



3D Differential Plot

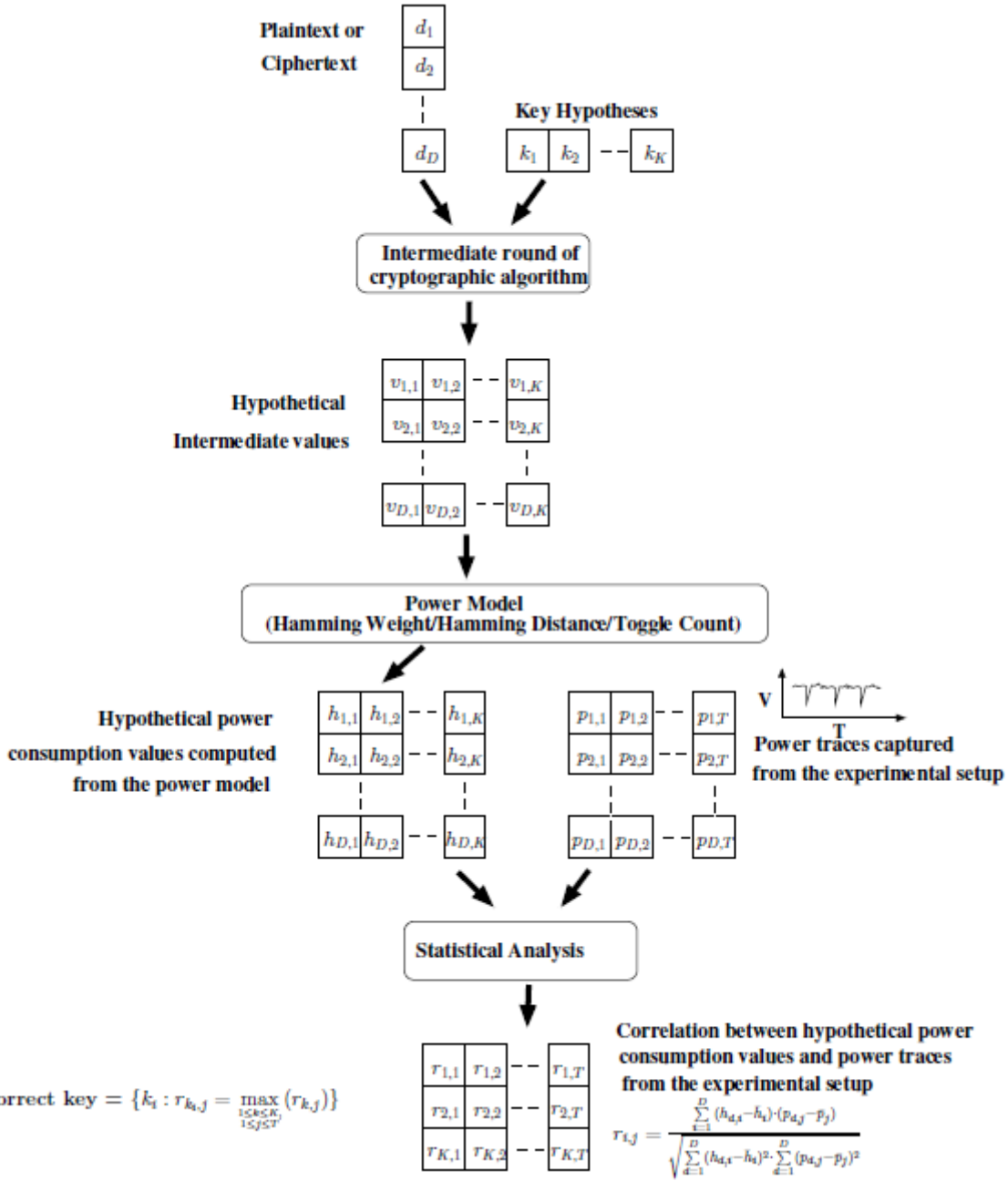
SBOX – 11 **BIT – 8** **TRACE COUNT = 15,000**

Differential Power Analysis: A Summary



$$Correct\ key = \{k_i : dpa\_bias_{k_i} = \max_{1 \le j \le K}(dpa\_bias_{k_j})\}$$

Correlation Power Analysis: An Improved DPA technique



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

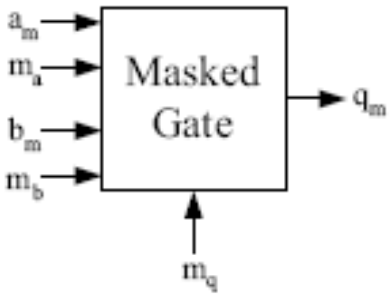
Countering DPA (Second Approach)

- *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

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

$$q_m = (a \cdot b) \oplus m_q$$

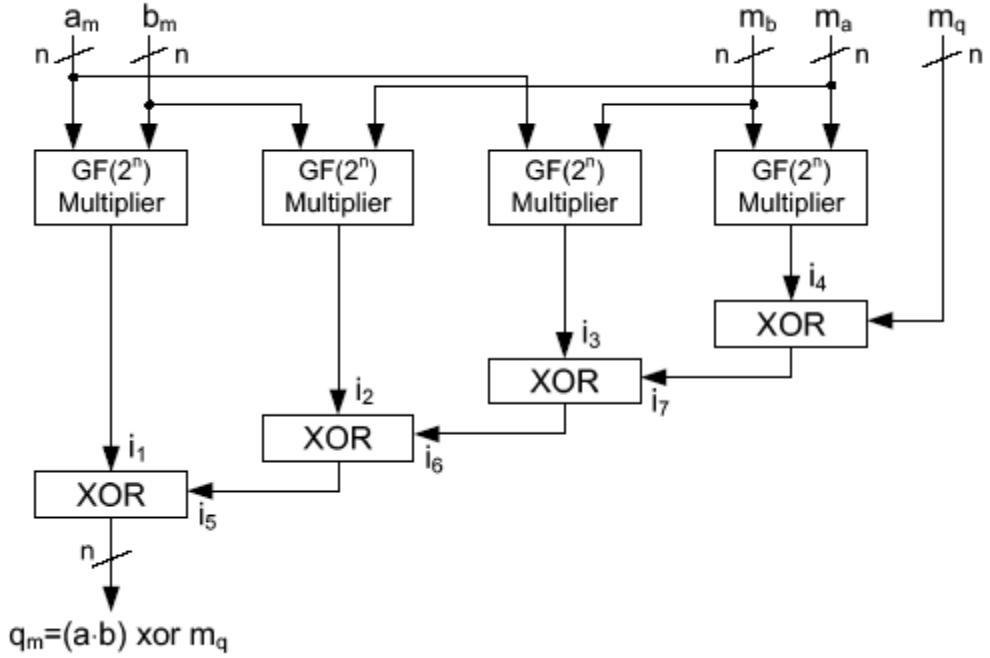
$$= (a_m \oplus m_a).(b_m \oplus m_b) \oplus m_q$$

$$= (((a_m.b_m \oplus b_m.m_a) \oplus (m_b \cdot a_m)) \oplus m_a.m_b) \oplus m_q$$

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.

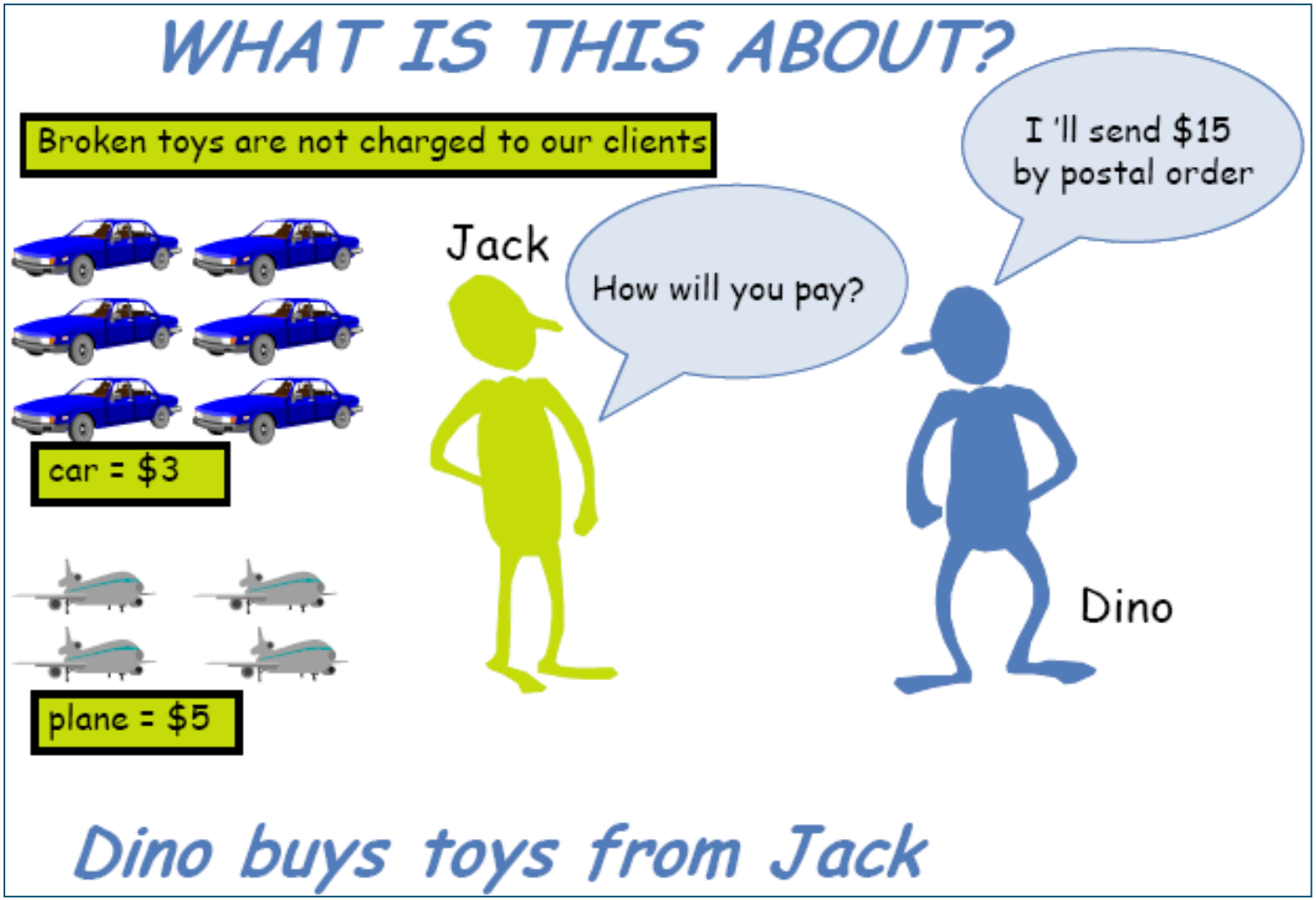
Masked Multiplier



Same Principle may be applied for multiplier circuits.

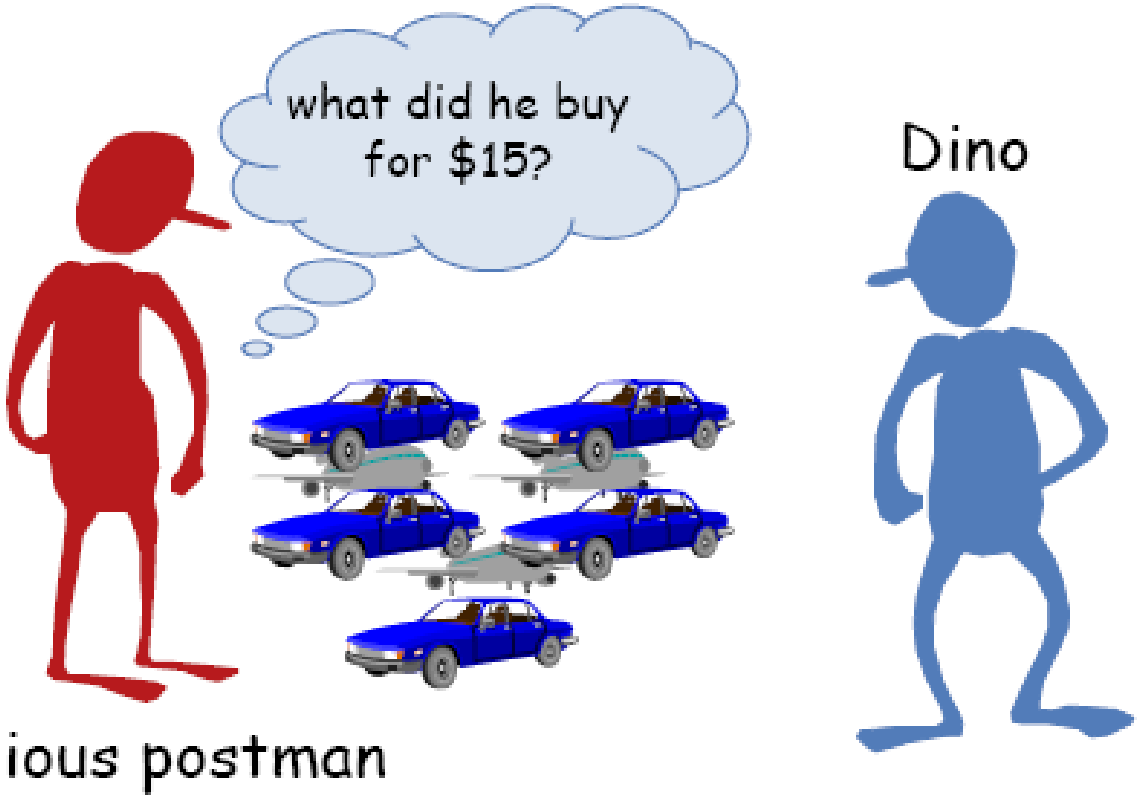
FAULT ATTACKS:

- *"WHEN A SECRET IS REVEALED, IT IS THE FAULT OF THE MAN WHO CONFIDED IT."*



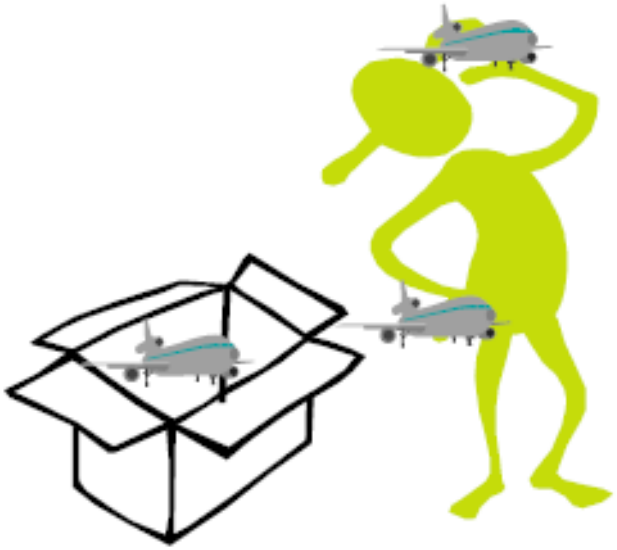
Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

The postman wants to know what Dino bought for $15

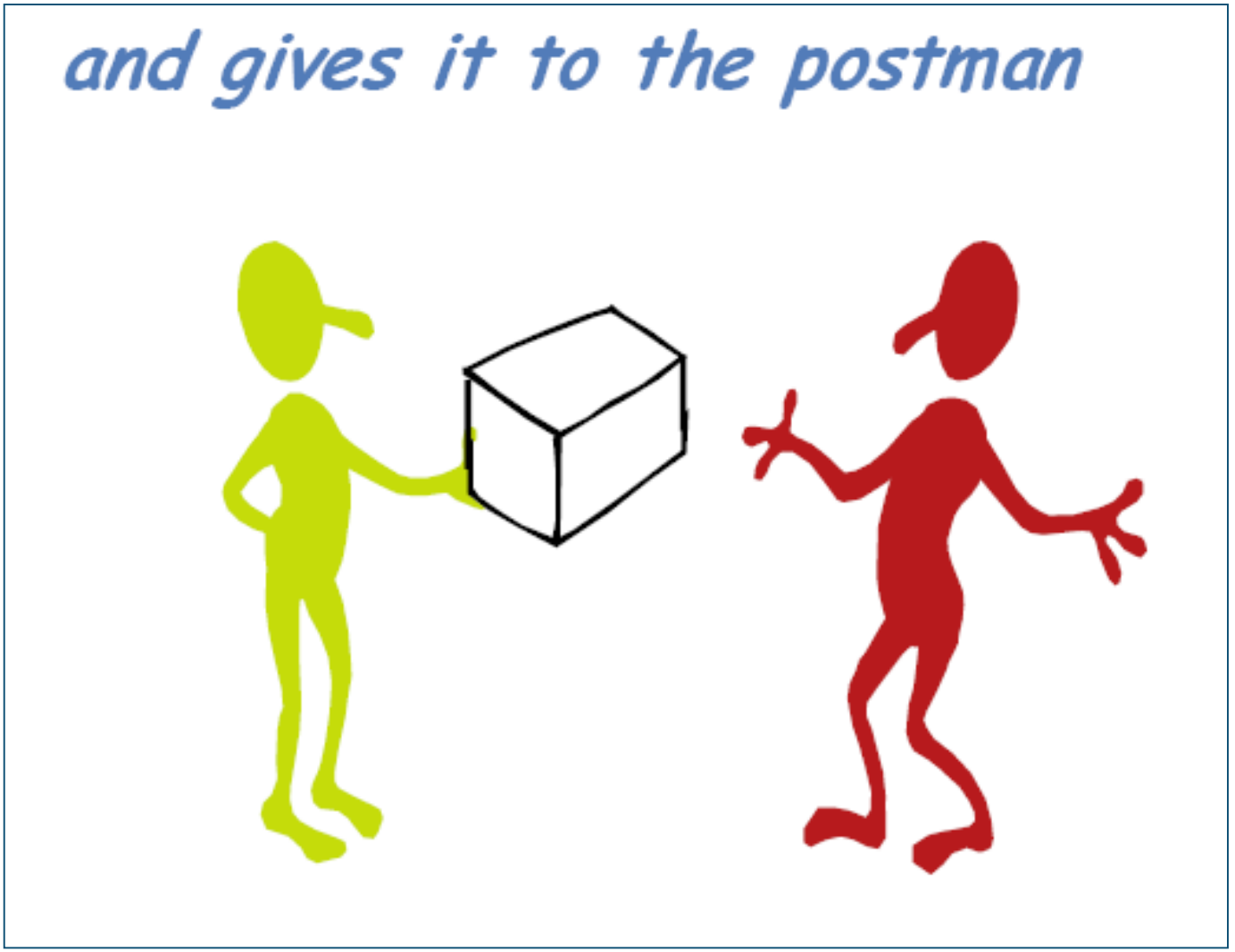


Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

In the meanwhile Jack prepares the DHL

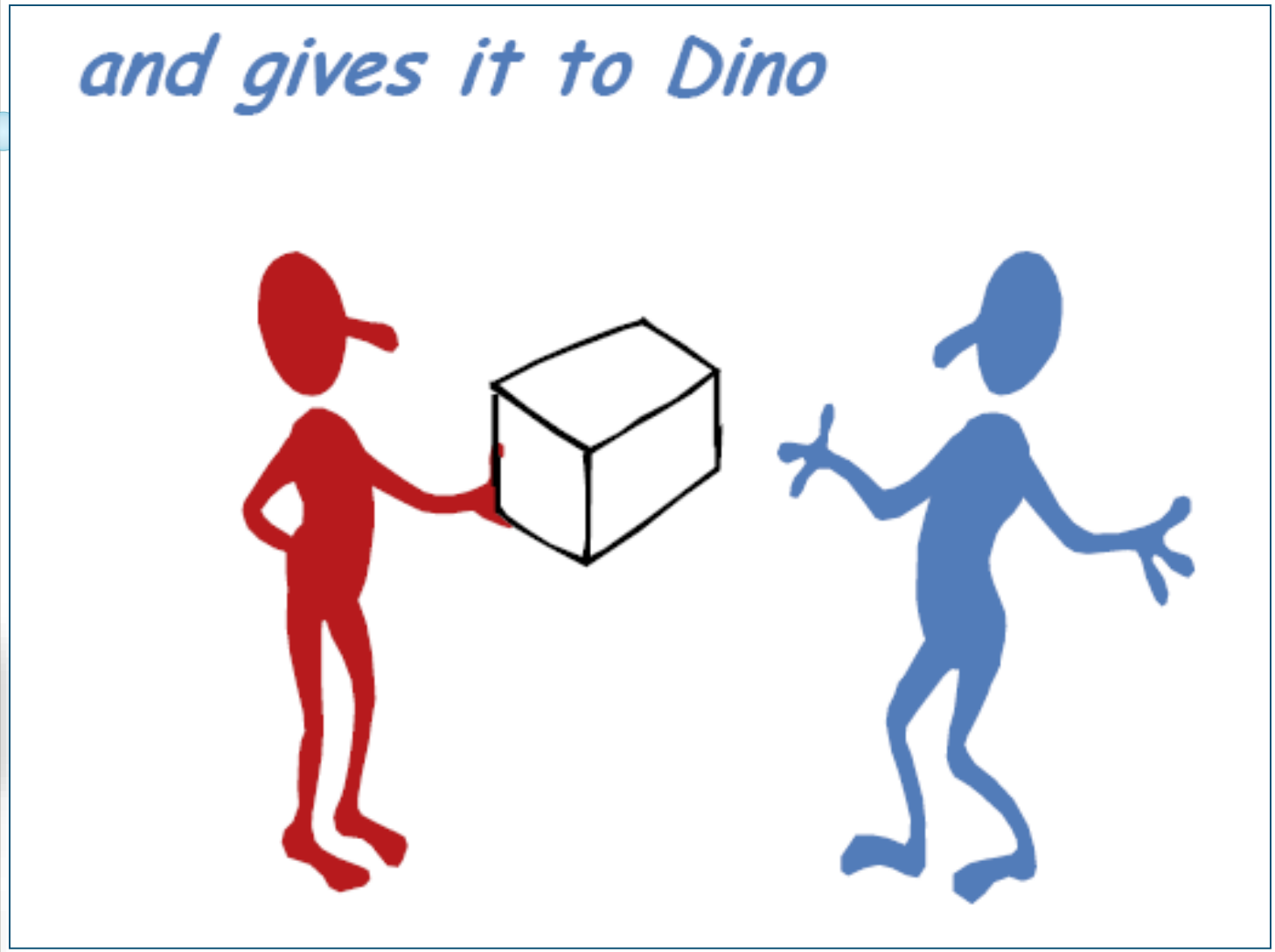


Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006



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

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



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 : RSA Cipher

- A generate keys:
  - Select p,q large prime numbers (at least hundred digits) and denote N=pq
  - Select a small odd integer e relatively prime (only common factor is 1) to $\phi(N) = (p-1)(q-1)$
  - Find integer d so that $de = 1 \bmod \phi(N)$
  - (e,N) – public key; (d,N) – private key
- B wants to send A a message M
  - B encrypts M using A's public key $S = M^e \bmod N$
    - M is restricted to $0 \le M \le N-1$
  - A decrypts using private key d -

$$S^d \bmod N = M^{de} \bmod N = M$$

Fault Attacks

- Another attack – by injecting faults
  - Vary the supply voltage – generate a spike
  - Vary the clock frequency – generate a glitch
  - Overheat the device
  - Expose to intense light – camera flash or precise laser beam
  - Faults injected into a byte or a few bits

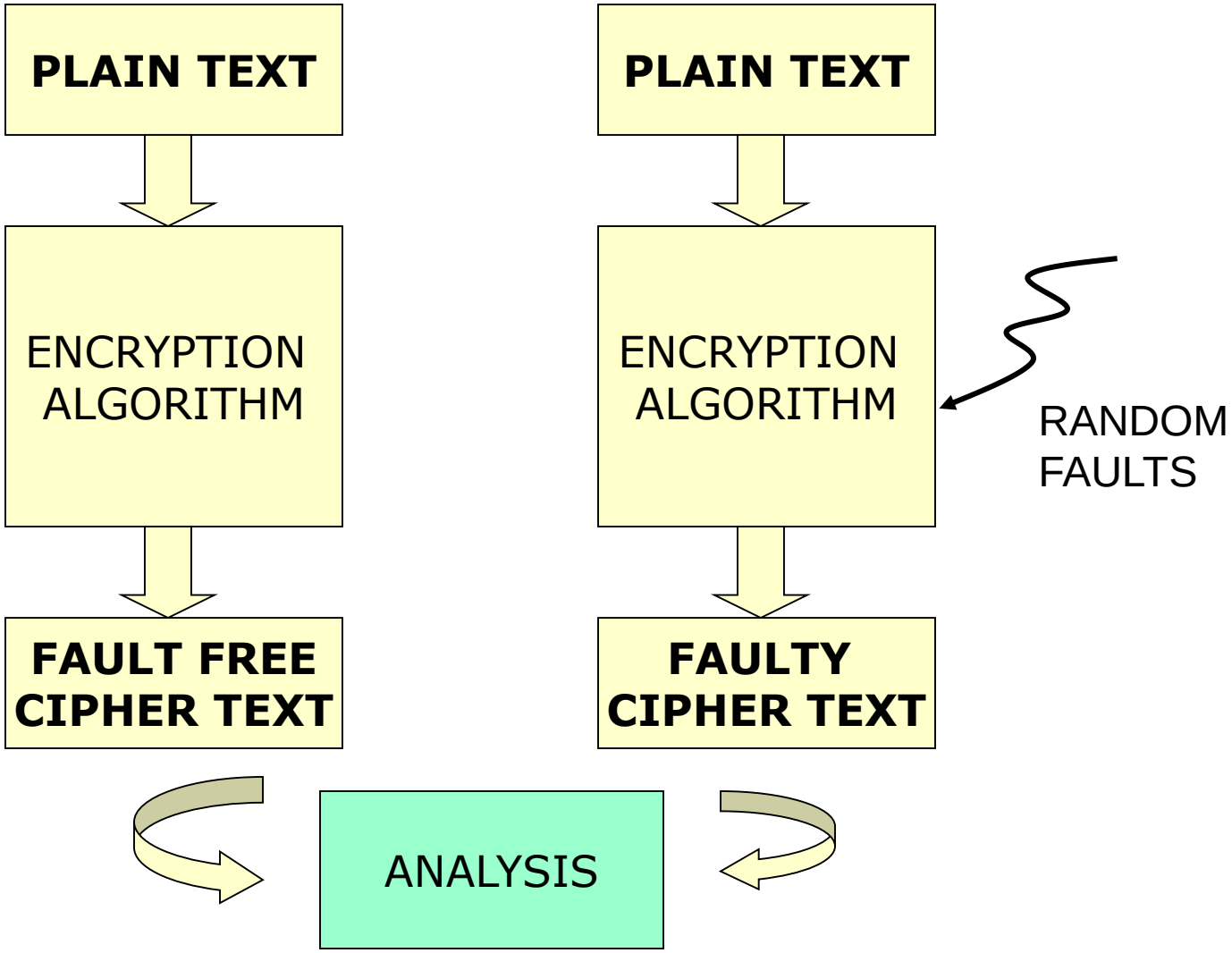
Fault Attacks on RSA

- Only decryption is subject to attacks
- Assume:
  - 1. Attacker can flip a single bit in key d
    2. S and corresponding message M known to attacker
  - Decryption device generates $\hat{M}$ satisfying

$$\frac{\hat{M}}{M} = \frac{S^{2^i \bar{d}_i}}{S^{2^i d_i}} \bmod N$$

  - If $d_i = 0$ then $\hat{M}/M = S^{2^i} \bmod N$
  - If $d_i = 1$ then $\hat{M}/M = 1/S^{2^i} \bmod N$

Fault Attacks on AES



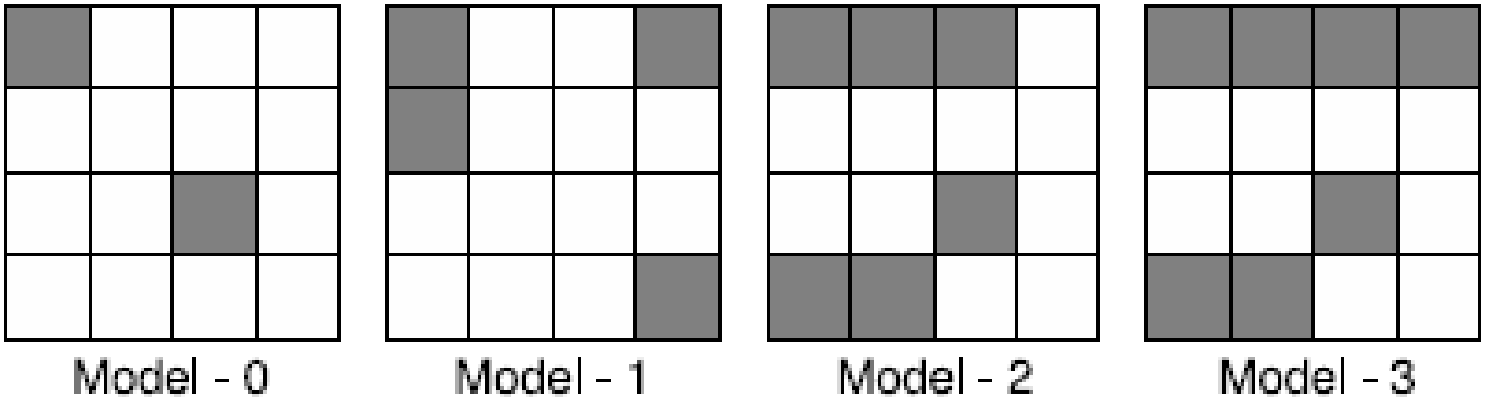
Fault Attacks on RSA

- Assume that the attacker flips randomly a bit in d.

- Example: (e,N)=(7,77), d=43 $d_5 d_4 d_3 d_2 d_1 d_0 = 101011_2$
  - Ciphertext=37 producing M=9 if no fault is injected and $\hat{M} = 67$ if a fault is injected
  - Search for i such that $9 = (67 \cdot 37^{2^i}) \bmod 77$ i=3($d_3 = 1$) since

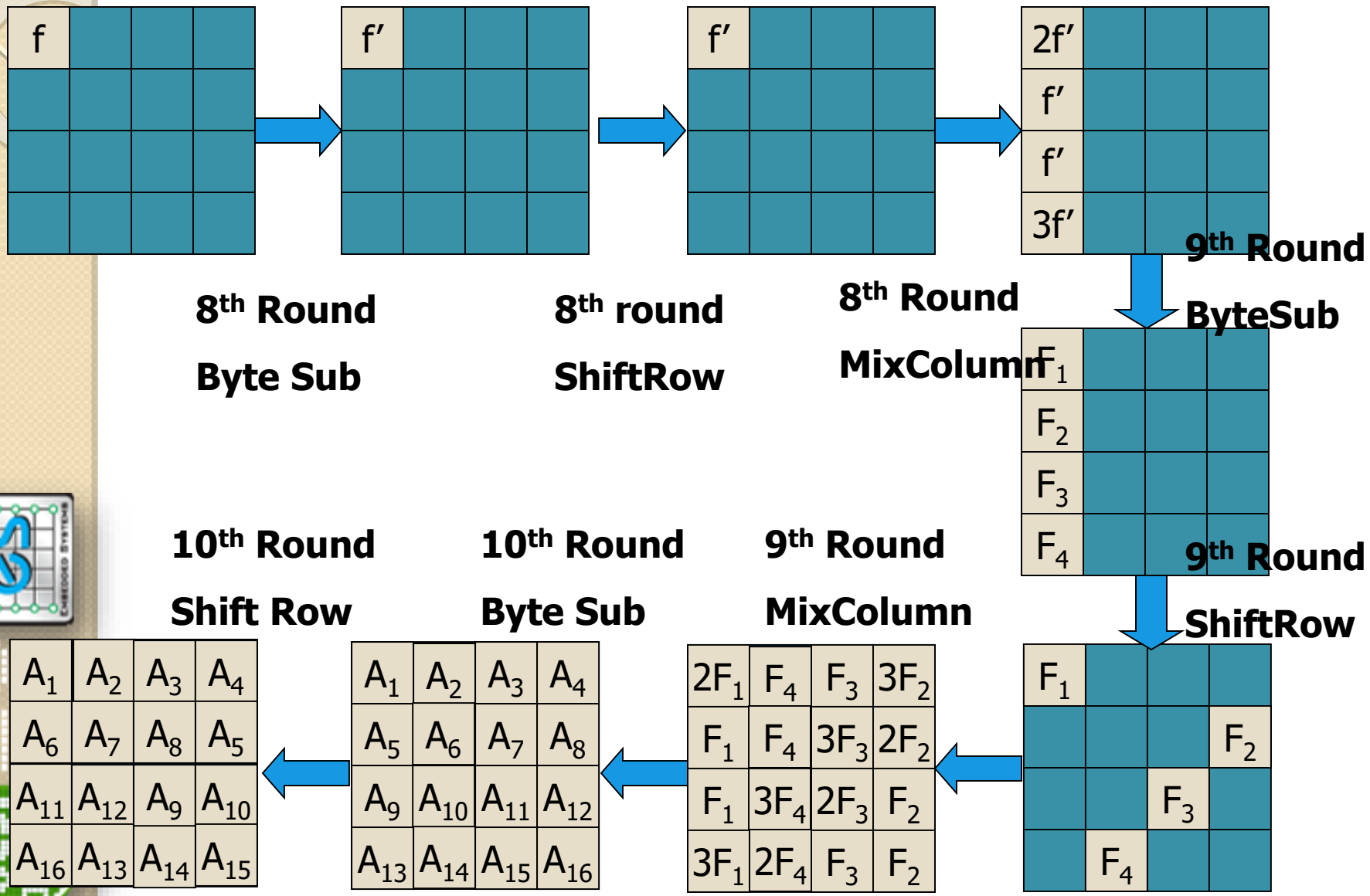
$$(67 \cdot 37^8) \bmod 77 = (67 \cdot 53) \bmod 77 = 9$$

Fault Models



- M0: One Diagonal affected.
- M1: Two Diagonals affected.
- M2: Three Diagonals affected.
- M3: Four Diagonals affected.

Propagation of Fault Induced



Features of the proposed attack

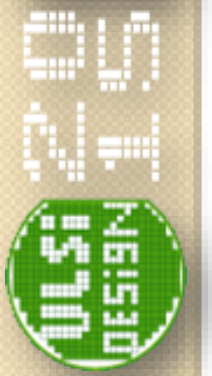
- The attack is the strongest of its class in the literature.
- Needs just one byte random fault induction.
  - Previous best attack reduced key space to 2^{40}.
  - Our attack reduced to 2^{32} (revealed in 400 sec on an Intel Xeon Server with 8 cores).

Debdeep Mukhopadhyay, *An Improved Fault Based Attack of the Advanced Encryption Standard.* AFRICACRYPT 2009: LNCS 5580, pp 421-434

The work was developed as a project from NTT Laboratory, Japan.

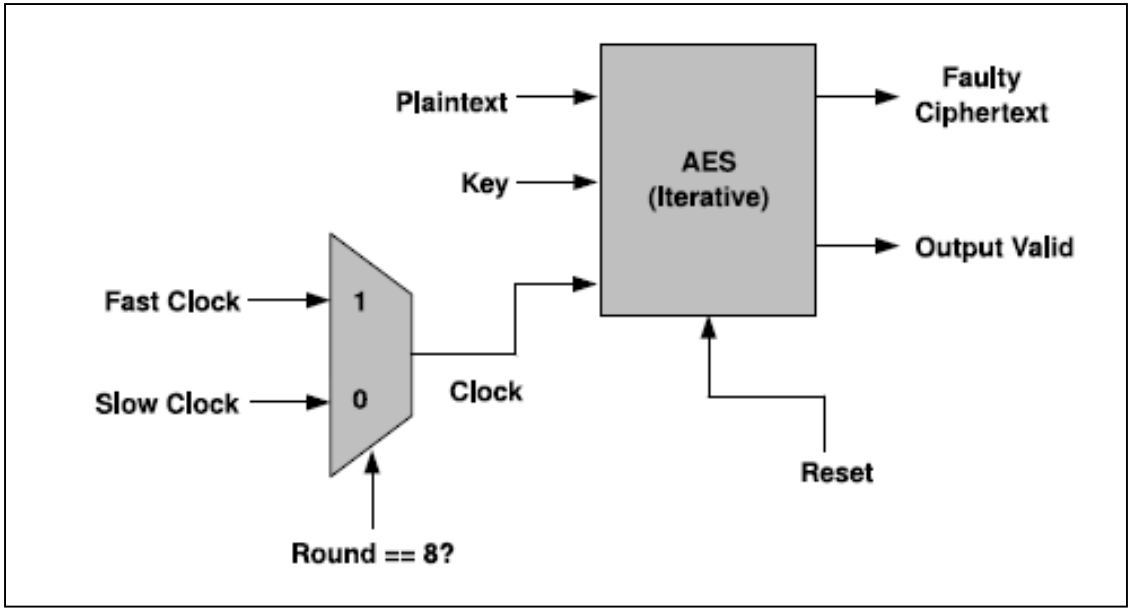
DFA with a single Fault!

- Current research shows that the AES key size can be reduced from 2^{32} to 2^{8} for a single byte fault.
- The small complexity of the attack makes it feasible on real life FPGA implementations of AES using less sophisticated techniques, like clock glitching.



Michael Tunstall and DebdeepMukhopadhyay, Differential Fault Analysis of the Advanced Encryption Standard using a Single Fault, Cryptology ePrint Archive: Report 2009/575

Fault Injection Setup



- **Tools Used:**
  - AES Core Implemented on Xilinx Spartan 3E.
  - Agilent Wavefrom (80 MHz)Generator
  - Xilinx Chipscope Pro Embedded Logic Analyzer.

Experimental Results

Clock Frequency (MHz)	No Fault	Model 0 (M0)	Model 1 (M1)	Model 2 (M2)	Model 3 (M3)
36.0	512	0	0	0	0
36.1	512	0	0	0	0
36.2	512	0	0	0	0
36.3	510	2	0	0	0
36.4	511	1	0	0	0
36.5	508	4	0	0	0
36.6	504	8	0	0	0
36.7	507	5	0	0	0
36.8	490	22	0	0	0
36.9	489	23	0	0	0
37.0	419	79	14	0	0
37.1	448	60	4	0	0
37.2	434	64	13	1	0
37.3	403	94	15	0	0
37.4	408	99	5	0	0
37.5	248	226	38	0	0
37.6	214	205	84	9	0
37.7	128	205	122	57	0
37.8	76	180	133	123	0
37.9	20	122	145	225	0
38.0	158	191	129	34	0
38.1	27	116	185	185	0
38.2	40	127	198	147	0
38.3	26	69	155	257	5
38.4	17	62	137	254	42
38.5	0	20	68	361	63
38.6	0	0	16	319	177
38.7	0	2	20	293	197
38.8	0	1	8	290	213
38.9	0	11	42	368	91
39.0	15	59	107	308	23
39.1	0	2	12	197	301
39.2	0	5	26	339	142
39.3	0	3	11	285	213
39.4	0	0	0	134	378
39.5	0	0	6	138	368
39.6	0	0	0	150	362
39.7	0	0	0	21	491
39.8	0	0	0	18	494
39.9	0	0	0	14	498
40.0	0	0	0	0	512

ATTACK REGION

Dhiman Saha, Debdeep Mukhopadhyay, Dipanwita Roy Chowdhury: **A Diagonal Fault Attack on the Advanced Encryption Standard.** IACR Cryptology ePrint Archive 2009: 581 (2009)

Bypassing the Counter-measures

- Parity-1, Parity-16 (linear codes) and Robust Codes (nonlinear codes) all miss some faults!
- The missed faults also can lie in the DFA-exploitable space.

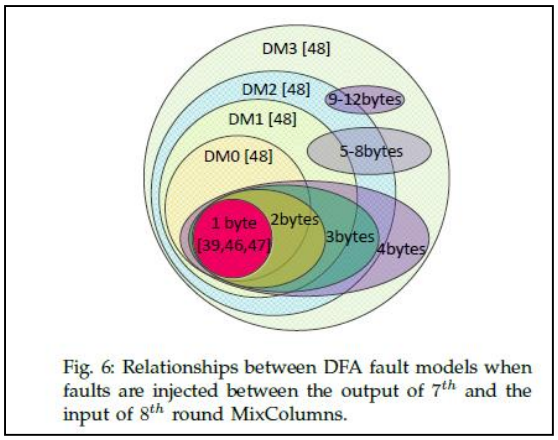
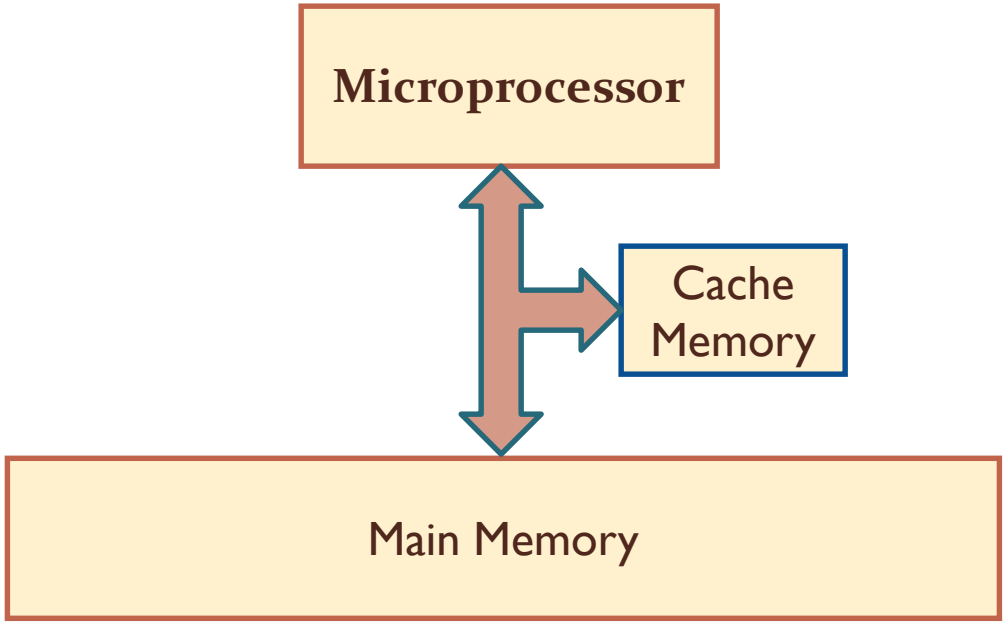


Fig. 6: Relationships between DFA fault models when faults are injected between the output of 7^{th} and the input of 8^{th} round MixColumns.

•An attacker can operate at a region where the faults are missed.
•Probability of faults getting missed is much higher if the fault space is not uniform.
•Can we have fault tolerance techniques which have provable 100% security. w.r.t. all the DFA exploitable faults?
Note that the DFA exploitable space is quite small…

A Quick Look into Cache Attacks

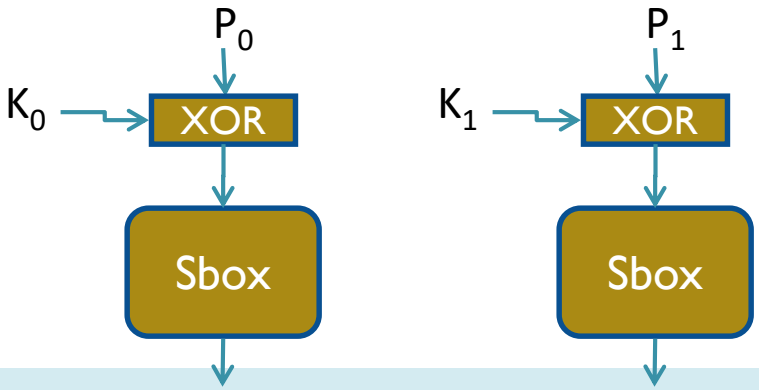
Cache Memory Leaks Information



- 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 second access depends on the previous sbox access.

- If cache hit :

$$P_0 \oplus K_0 = P_1 \oplus K_1$$

$$\Rightarrow K_0 \oplus K_1 = P_0 \oplus P_1$$

Since we know P_0 and P_1, we can determine $K_0 \oplus K_1$

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

Classes of Cache Attacks

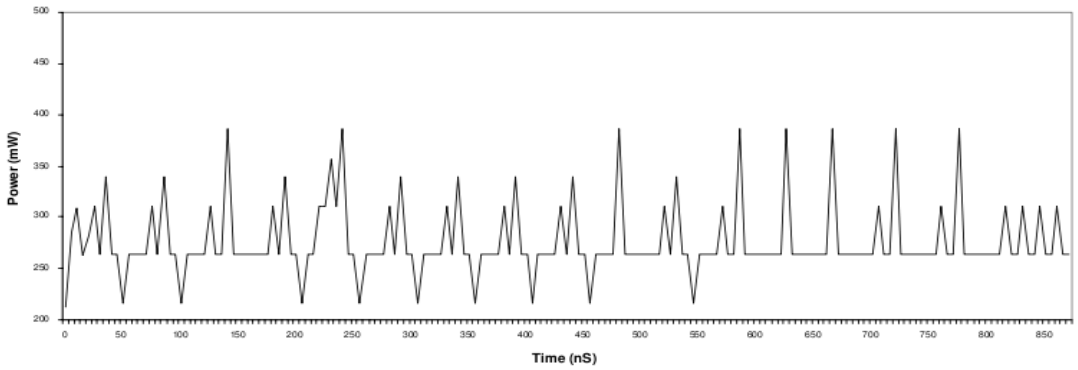
Three ways to identify cache behavior

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

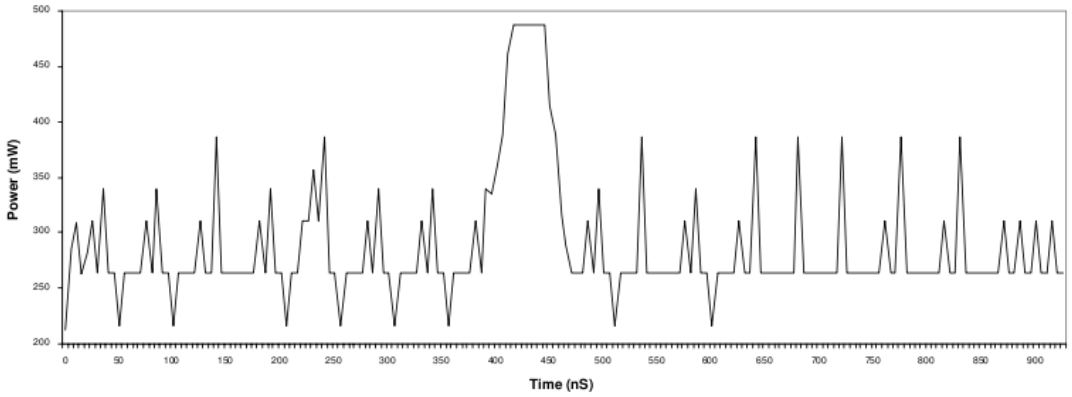
Cache Trace Attacks

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

Without Cache Miss

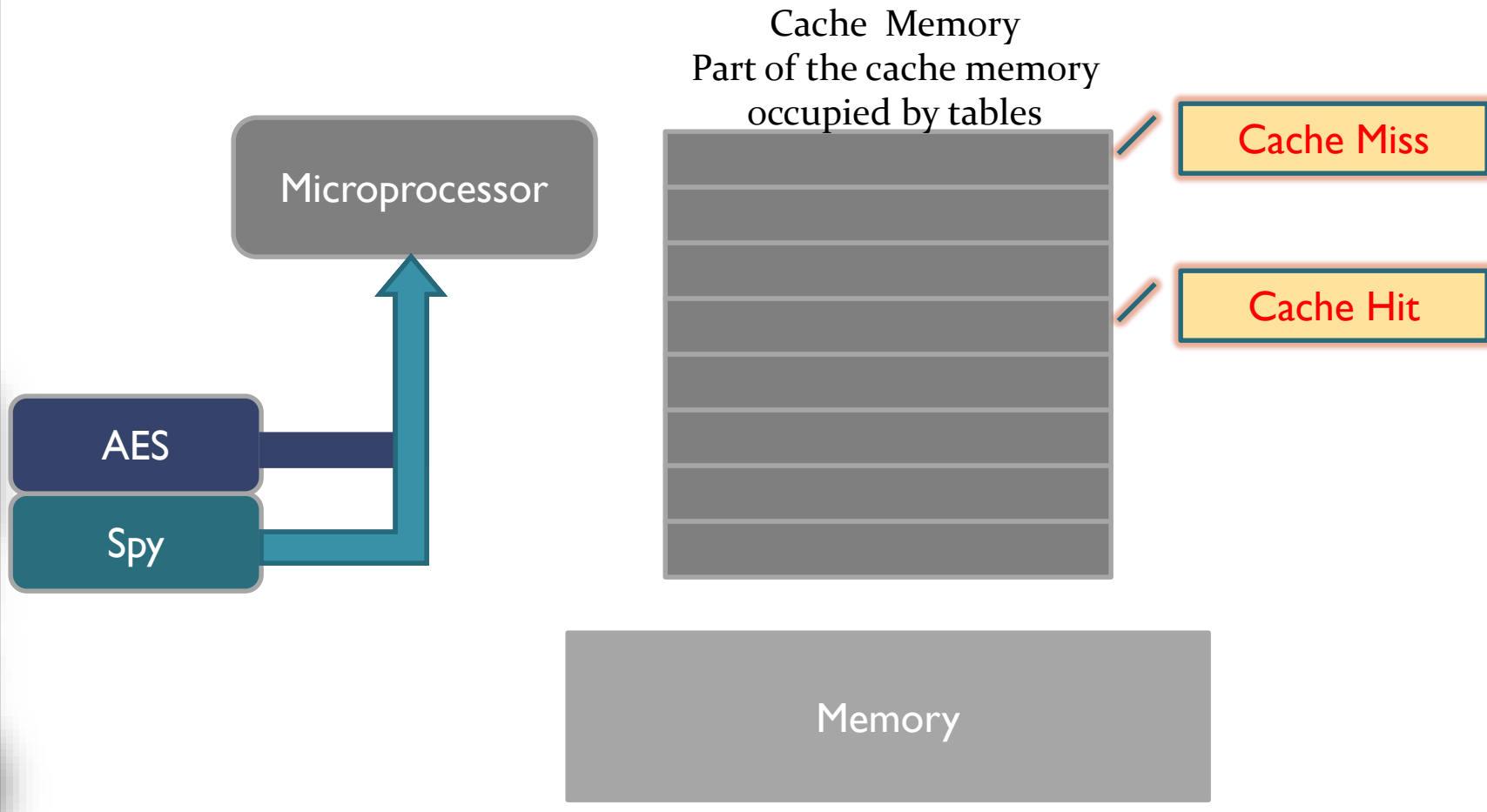


With Cache Miss



Cache Access Attacks : Osvik's Attack

- Uses a spy program to determine cache behavior

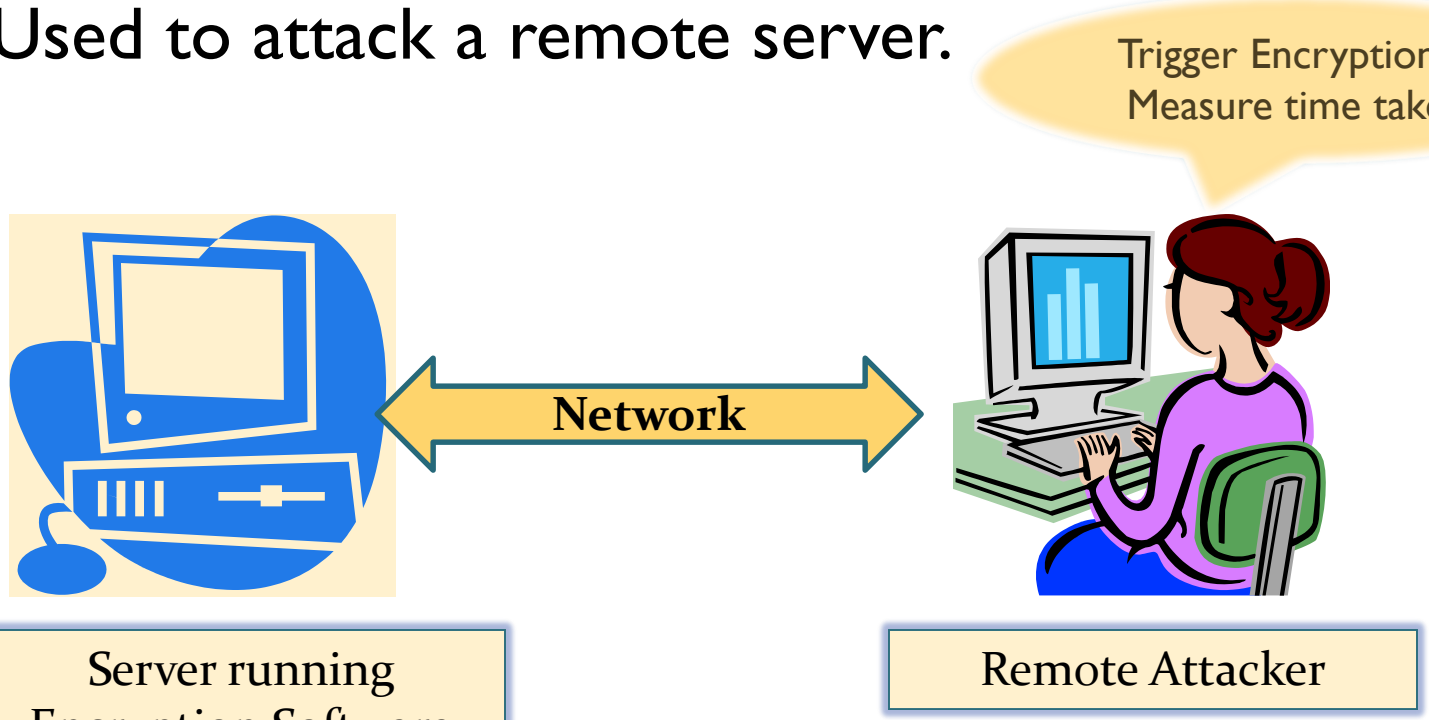


Cache Timing Attack

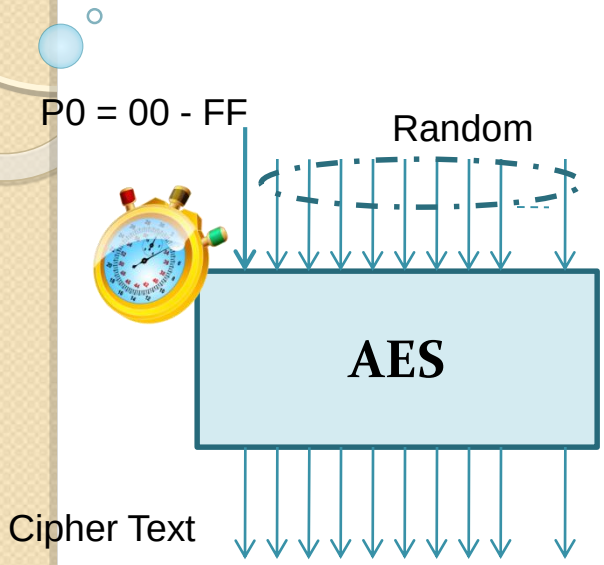
- Based on measuring the total time for encryption.

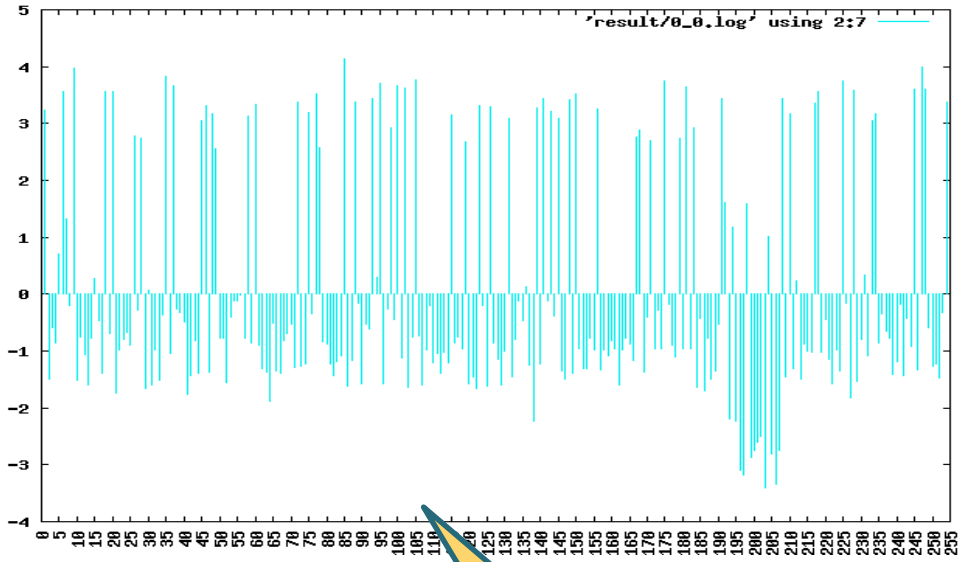
$$Execution\ Time \approx N_h * T_h + N_m * T_m + K$$

- Used to attack a remote server.



Bernstein's Cache Timing Experiment





For $P_0 = 0$ to 255

For 2^{15} *iterations*

Vary the remaining plain texts randomly

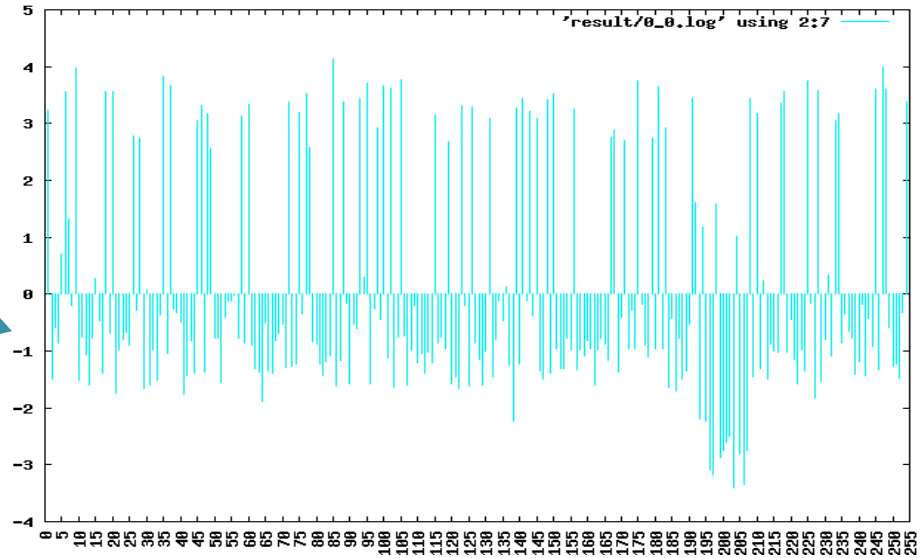
Invoke AES Encryption

Obtain Time for Encryption

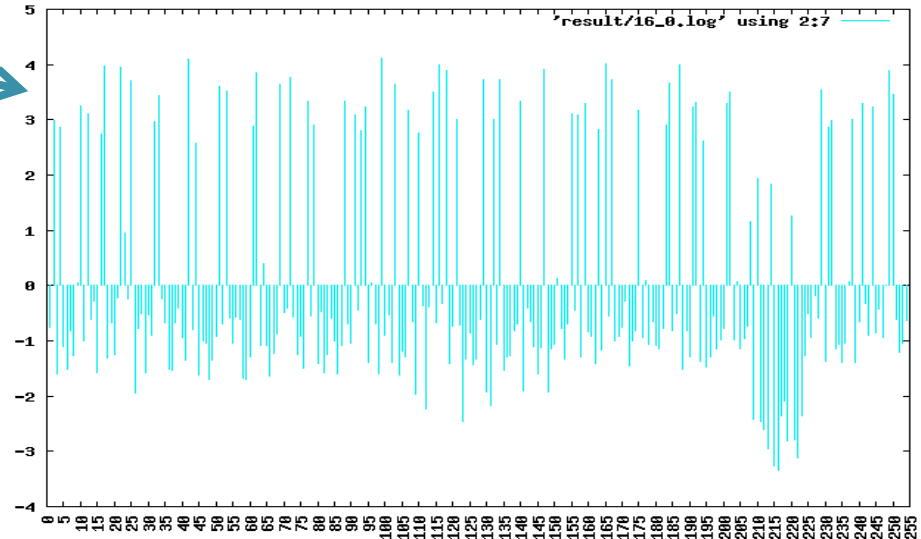
Characteristic of Plaintext Byte P_0 (Deviation from mean time)

Bernstein's Cache Timing Attack

Put key to 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**
    - **Byte Substitution**
    - **Shift Row**
    - **Mix Column**
    - **Add Round Key**
  - **Final Round**
    - **Byte Substitution**
    - **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", VLSID 2010,

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**

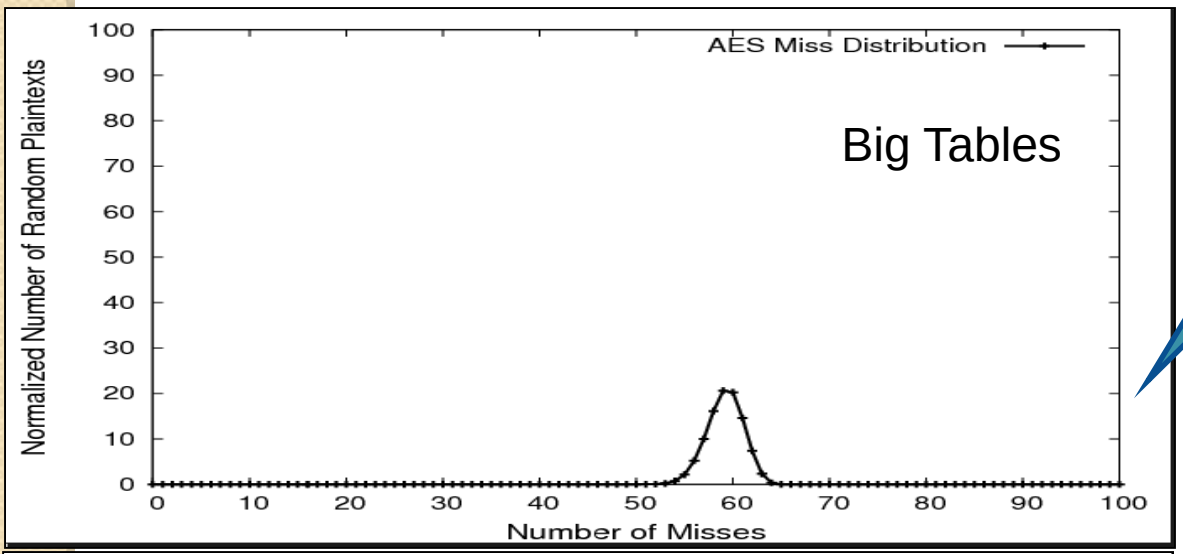
Round 1

```
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]
```

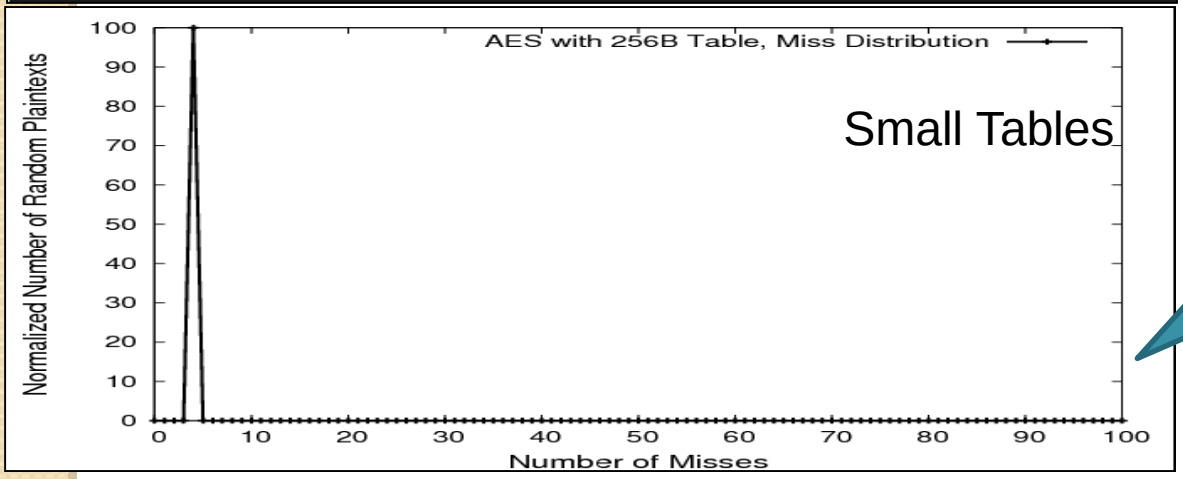
Round 2

```
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 big and small Tables



Cache Attack works because of varying execution time.

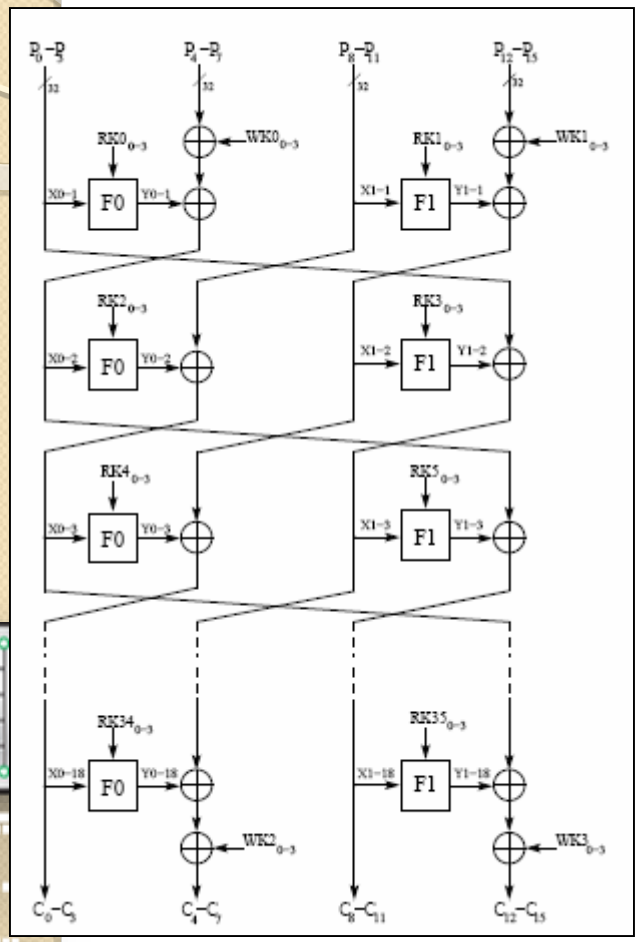


Cache Attacks Believed to be not possible

Attack of Clefia: A Block cipher with small tables

- **Clefia** is a block cipher designed by **Sony Corporations**
- It has **small tables** of 256 bytes.
- It was designed specifically to be protected against cache timing attacks.
  - **Our project from NTT Lab, was to analyze cache timing attacks on a Clefia code written by Sony itself..**

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", Indocrypt 2009.

The work was developed as a project from NTT Laboratory, Japan.

The attack is mentioned in website of SONY.

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)* ...

Can Side Channels be Eliminated *by design*?

DRECON:
DPA Resistance by Construction

Shivam Bhasin and Sylvain Guilley
Télécom ParisTech / Secure-IC S.A.S., France

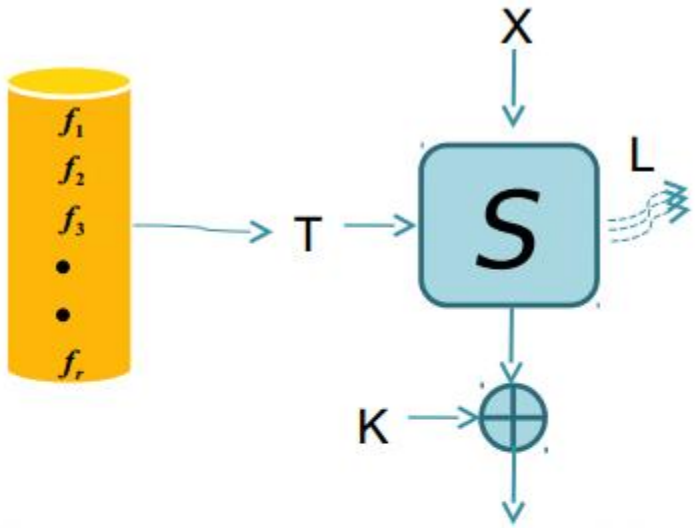
Suvadeep Hajra, Gaurav Bajaj, Sahil Sharma, and Debdeep Mukhopadhyay
Indian Institute of Technology Kharagpur, India

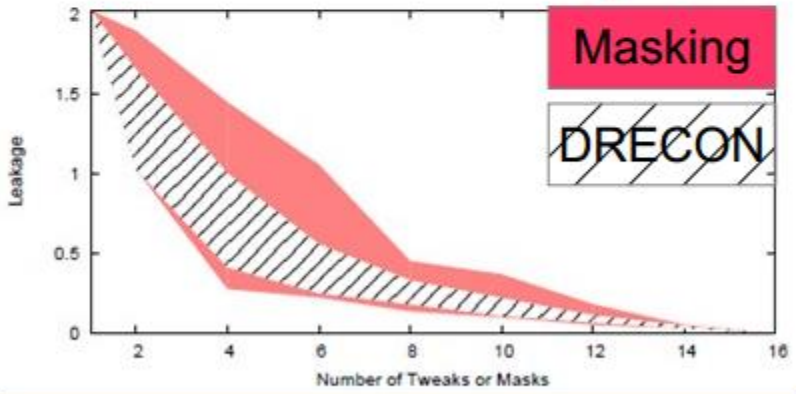
Chester Rebeiro
Columbia University, USA

AFRICACRYPT 2014, Marrakech, Morocco

The new s-box scheme

- Pool of ‘r’ cryptographically strong s-boxes
- At every encryption randomly choose **t(tweak)**
- Tweak is exclusively shared between sender and receiver

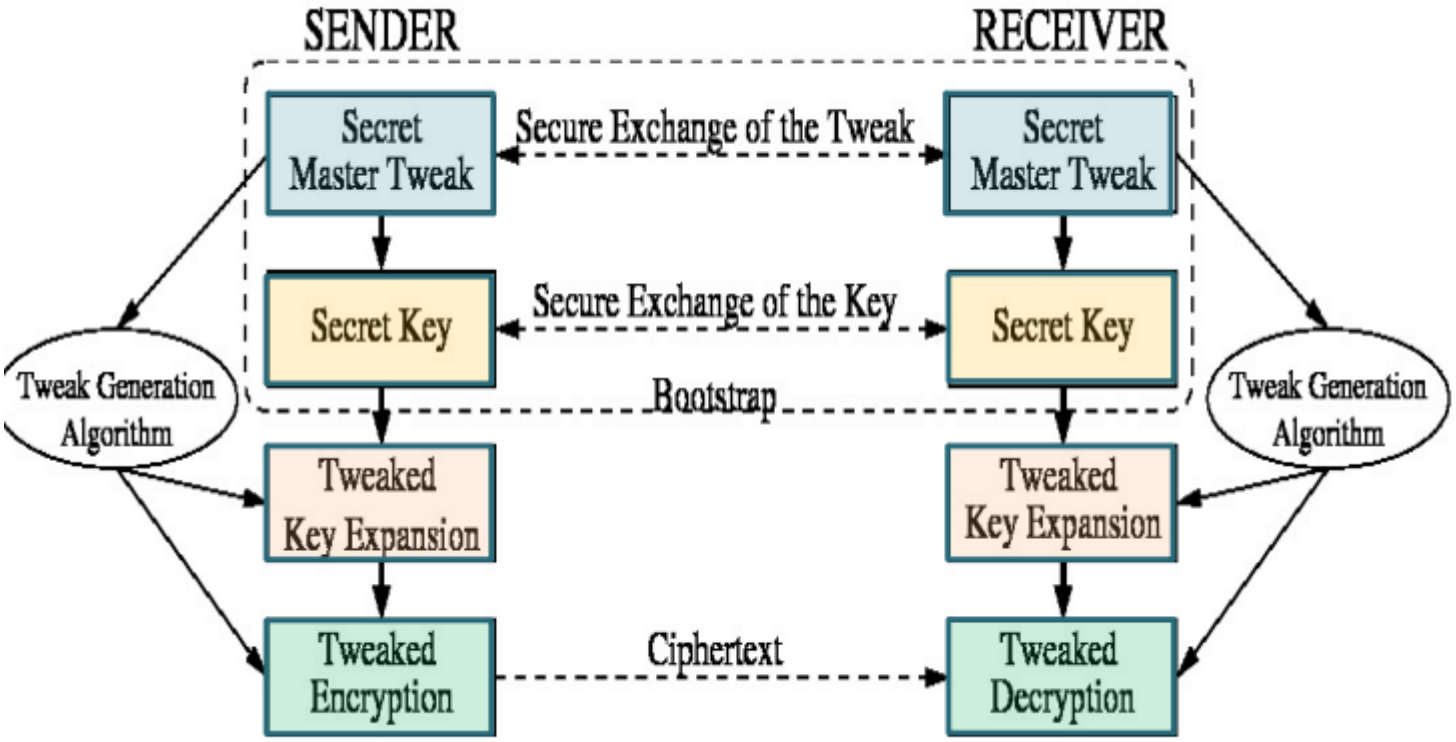




Comparing first order Hamming weigh leakage for masking and DRECON for a random 4 x 4 bijective mapping

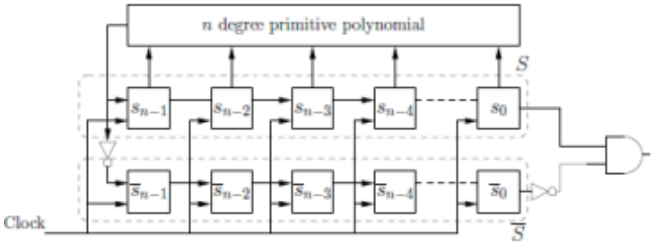
Information theoretically equivalent to first order masking

Application of DRECON

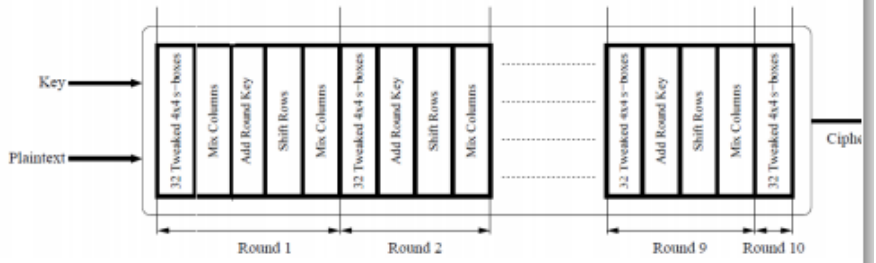


Example of DRECON-AES (4x4)

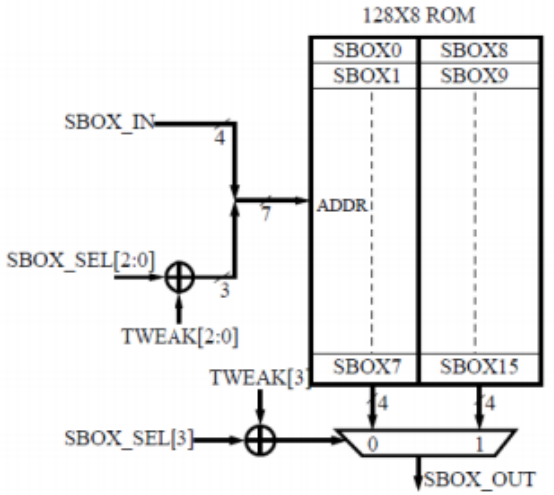
Tweak Generation Algorithm



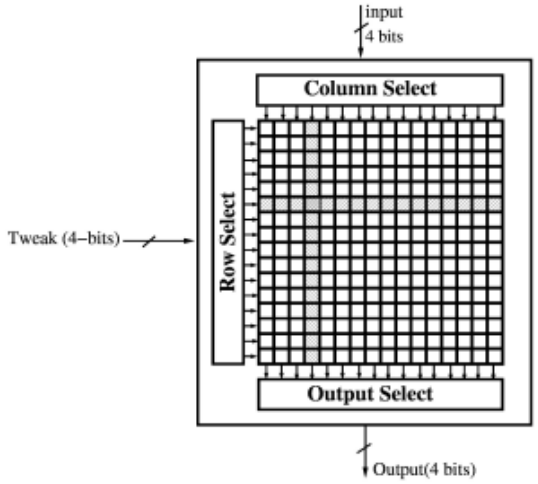
The modified AES Algorithm



Software Implementation



Hardware Implementation



Resources Requirement

- In FPGA

Implementation	Slices	LUTs	Registers	Clock Cycles	Clock Period (ns)
4×4 AES[1]	1120	3472	1270	11	11.14
Masked 4×4 AES	3427	10589	1765	11	23.83
4×4 DRECON-AES	1379	3868	1583	11	10.3

4×4 AES is an implementation of the AES-128 algorithm with the 8×8 bit sbox replaced by a pair of 4×4 cryptographically strong sboxes.

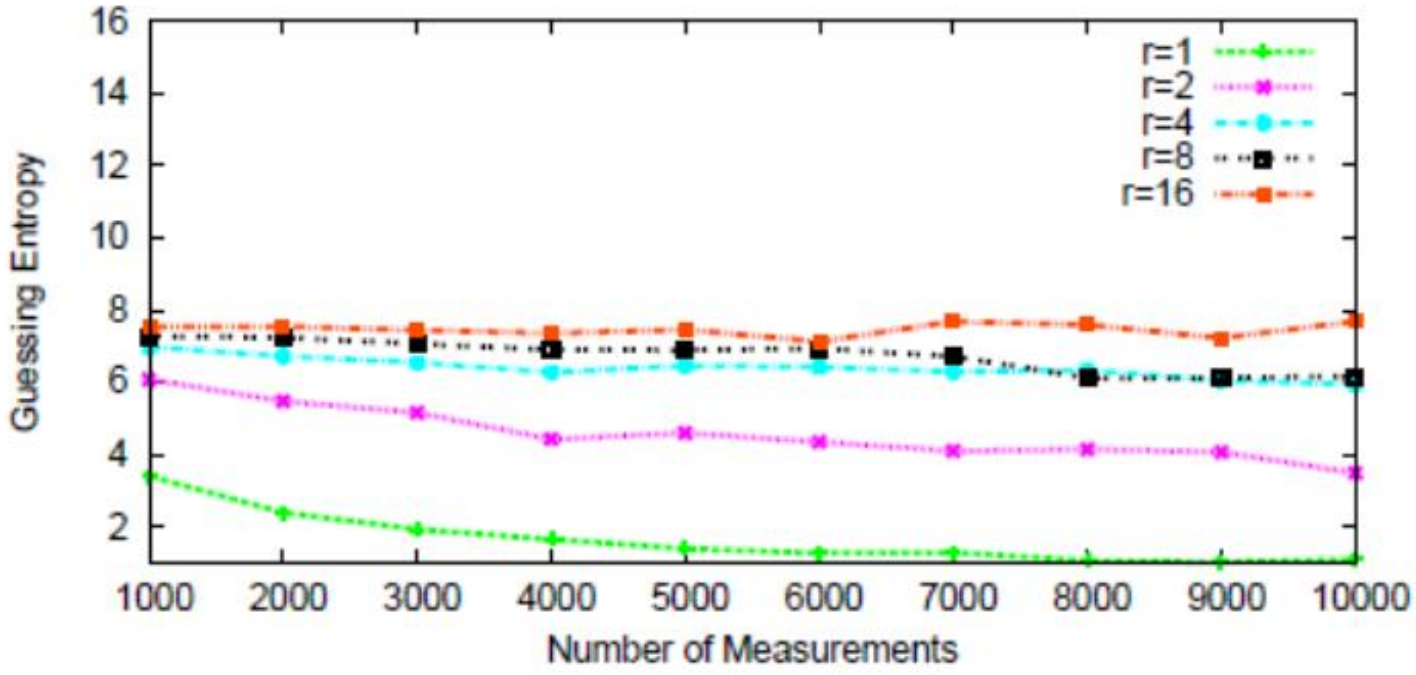
Implementation	Slices	LUTs	Registers	Clock Period (ns)
Masked AES[1]	3948	13278	1592	14.955
8×8 DRECON-AES	1355	3716	1568	10.789

CC : 16*11

[1]The implementation exclude the PRNG used to generate mask.

Masking : double datapath, mask mgmt, mask refresh... are expensive

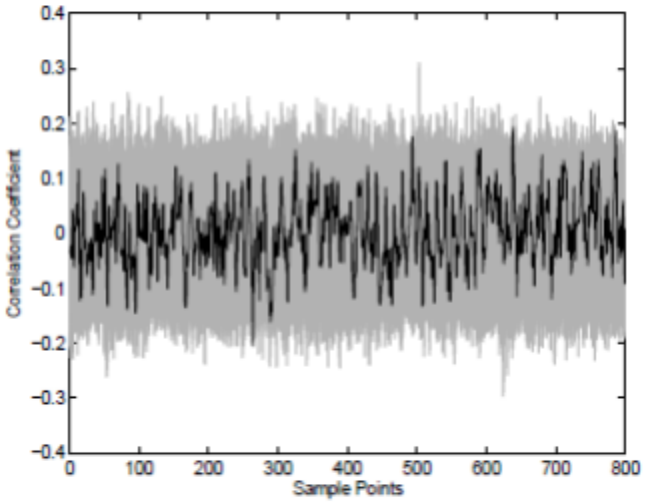
Guessing Entropy of an Attack on DRECON-AES



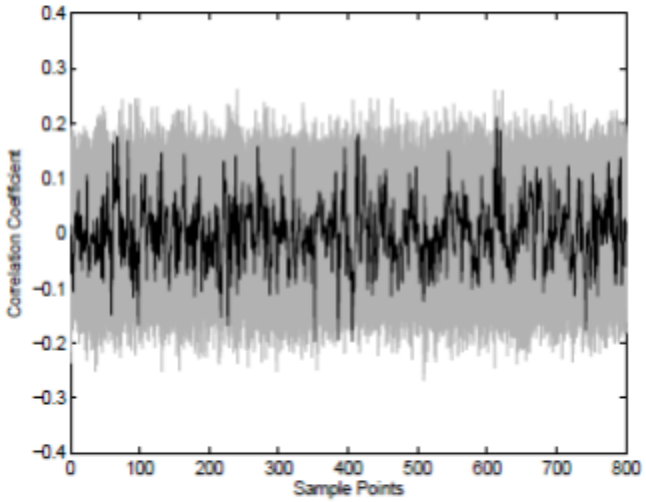
r is the number of s-boxes in the pool

Correlation-collision

Tests some imperfections of the masking
=> e.g., used by Moradi & Mischke at CHES 2012 against some masking schemes



(a) 1O correlation-collision attack



(b) 2O correlation-collision attack

Fig. 7. Results of correlation-collision attack on 8×8 DRECON-AES using about 1 million traces. The correlation coefficients for wrong key differences are shown in grey and for the correct key difference (in this case $k_1 \oplus k_2 = 108$) in black.

Is that the end?

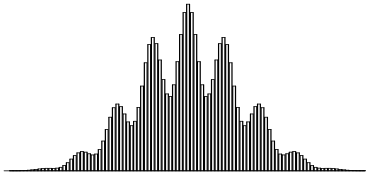
-- Side Channels with Process Scaling

Mathieu Renauld, François-Xavier Standaert, Nicolas Veyrat-Charvillon, Dina Kamel, Denis Flandre: **A Formal Study of Power Variability Issues and Side-Channel Attacks for Nanoscale Devices**. EUROCRYPT 2011: 109-128

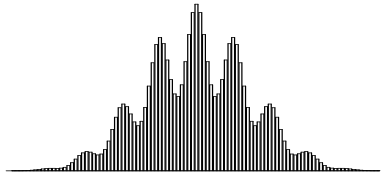
Process Scaling and Side Channels

- Process variation is a central issue in deep-submicron technology.
- Research shows that the leakage models change: they become non-linear!
- In the absence of variations, one can develop a power model, but due to variations one need to infer correct leakage for each chip.
- Also, countermeasures assume independent leakage
  - But below 65nm, there would be cross-talk!

Machine Learning as a Classifier

k_1 ~ 

⋮ ⋮

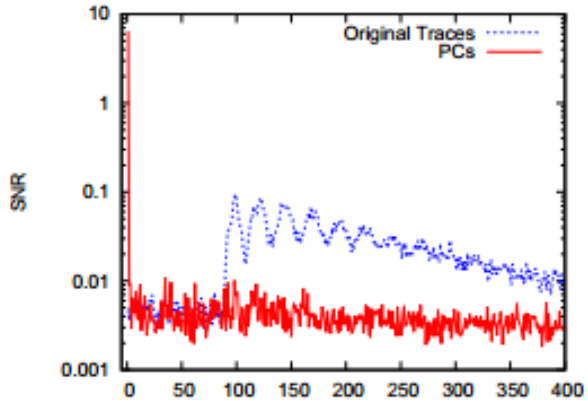
$k_{|K|}$ ~ 

 ⇨ k?

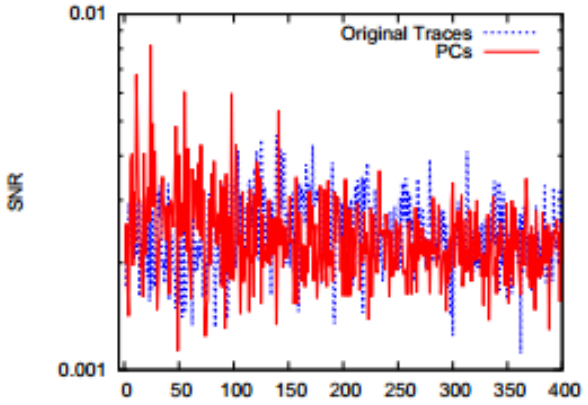
Given a trace, find the key k used for the encryption.

Develop Suitable Metrics to Certify Chips

- $L_t = \Phi_t(S_k) + N_t$
- SNR: $Var(E[L_t|S_k])/Var(L_t - E[L_t|S_k])$



(a) Low noise scenario.



(b) High noise scenario.

- Certification of true side channel security of chips require sophisticated tools!

Thank You for Your Attention!