Side Channels in Cryptography-Part II

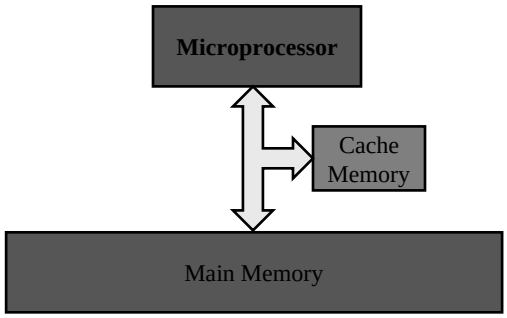
Debdeep Mukhopadhyay
Dept of Computer Sc and Engg
IIT Kharagpur

Topics of Discussion

- Cache Attacks
- Fault Based Attacks
- Information Theoretic modeling of SCAs
- Provable Security in Side Channel Analysis

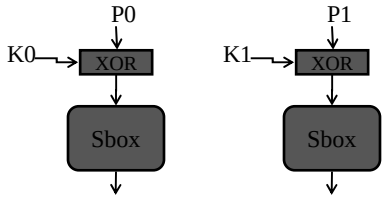
Cache Attacks

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 (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.

5

Classes of Cache Attacks

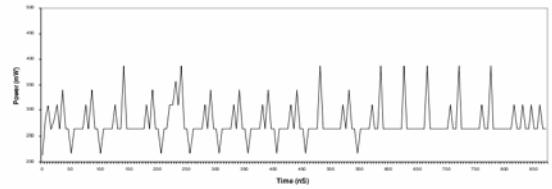
Three ways to identify cache behavior

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

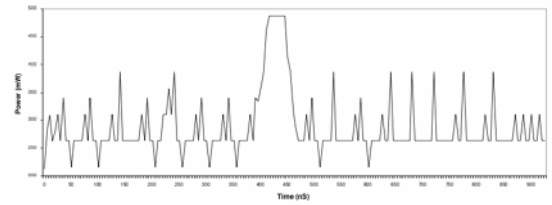
Cache Trace Attacks

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

Without Cache Miss

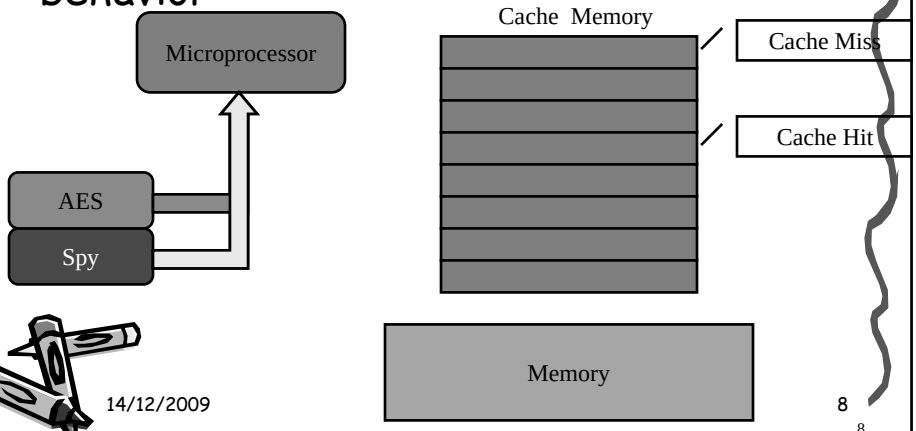


With Cache Miss



Cache Access Attacks : Osvik's Attack

- Uses a spy program to determine cache behavior

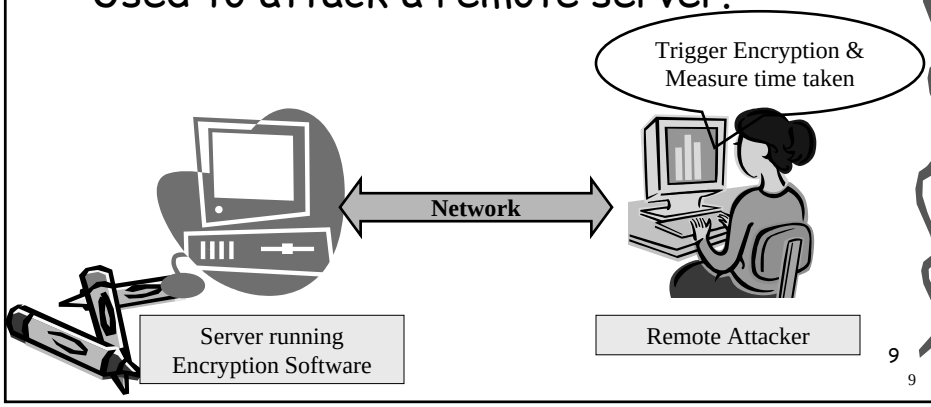


Cache Timing Attack

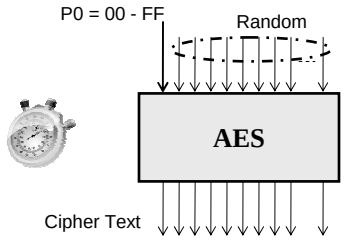
- Based on measuring the total time for encryption.

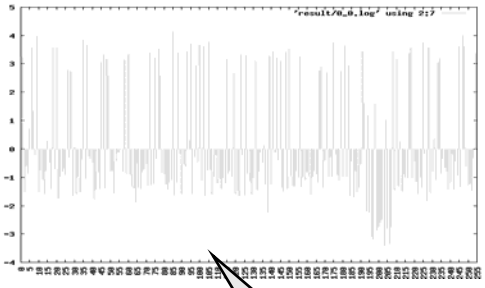
  Execution Time ≈ Nh*Th + Nm*Tm + K

- Used to attack a remote server.



Bernstein's Cache Timing Experiment



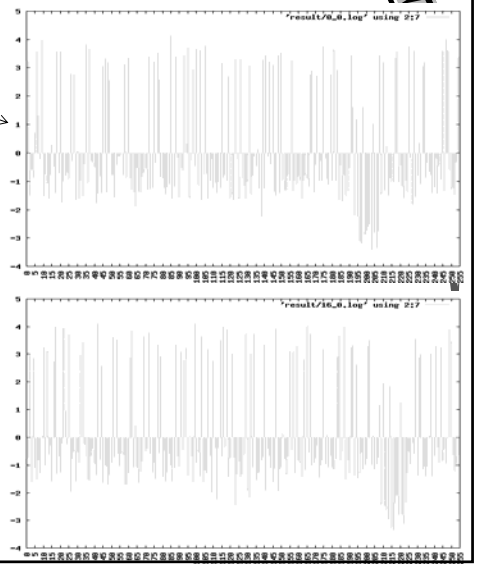


- For P0 = 0 to 255
  - For 2^{15} iterations
    - Vary remaining plaintext by
    - AES Encrypt
    - Determine Time For Encryption

Characteristic of Plaintext Byte P0 (Deviation from mean time)

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

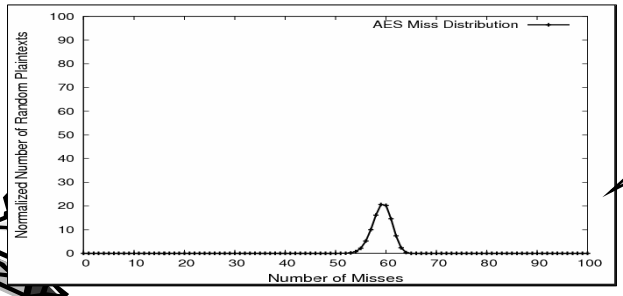
- **Encryption Time(E) ≈ Nh*Th + Nm*Tm + K**

Since, Nt = Nm + Nh

E = (Nt – Nm) Th + Nm*Tm + K

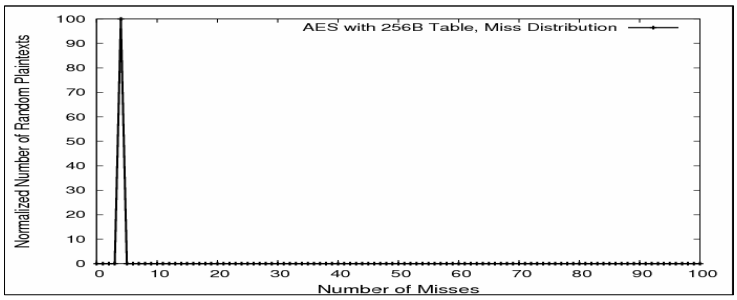
= Nt*Th + (Tm – Th)Nm + K

- **Time(E2) – Time(E1) = aNm**



Cache Attack works because of varying execution time.

AES Execution Time with 256Byte Table

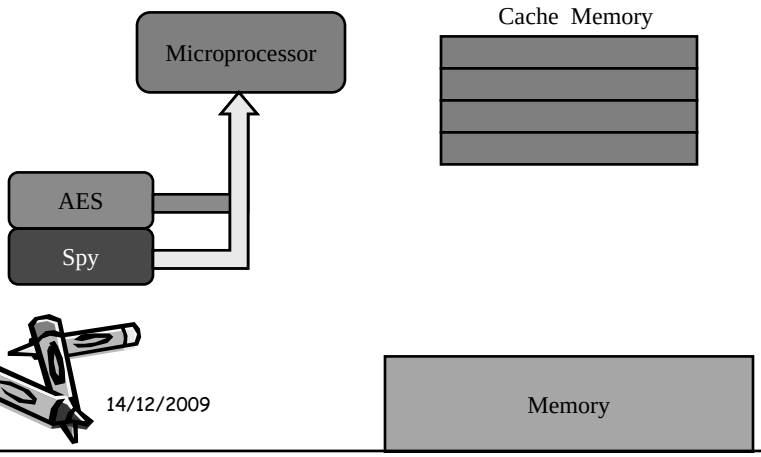


- ΔEncryption Time = aNm
- Nm = TableSize/CacheLineSize
  = 256/64 = 4
  …… Encryption Time is c

Cache Attacks Believed to be not possible

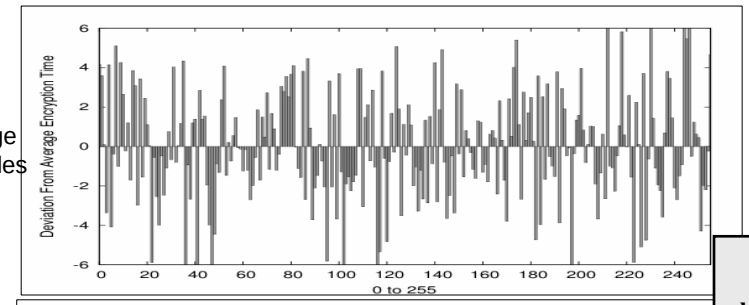
Osvik's Attack on AES-with 256B Table

- Won't Work!!!

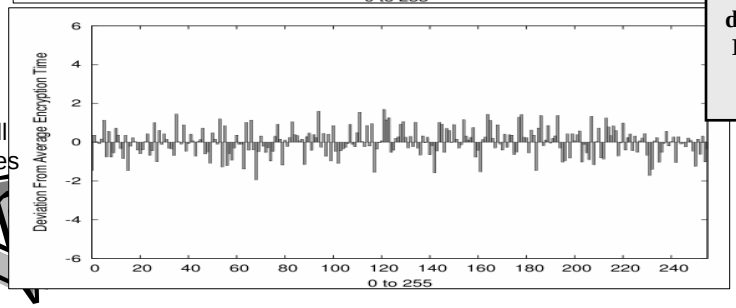


Bernstein's Attack on Small Tables

Large Tables



Small Tables



Lesser deviations in Encryption Time

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

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

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.

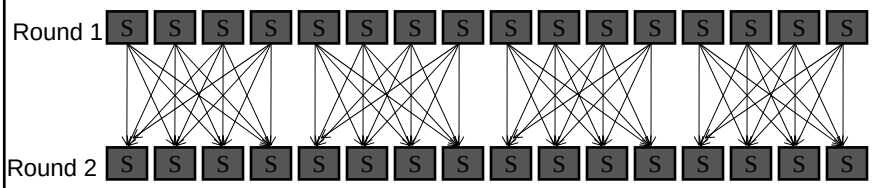
Program

```
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
```


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

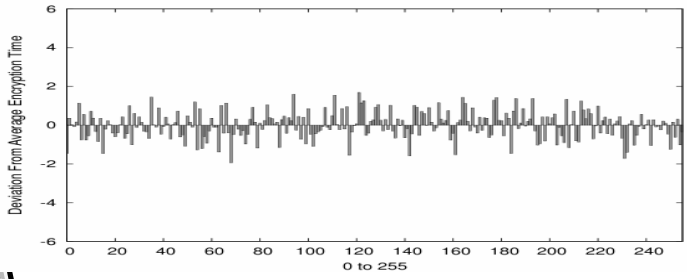
On Modern Caches

- 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.

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

Because,

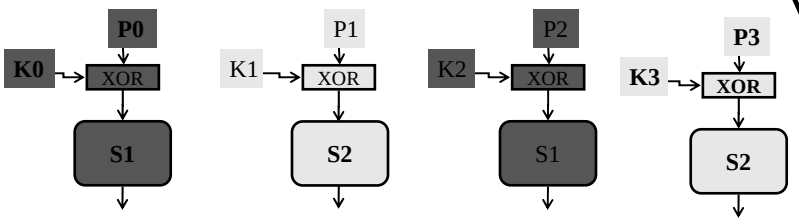
1. the deviation in encryption time is too less.
2. Bernstein's timing measurements



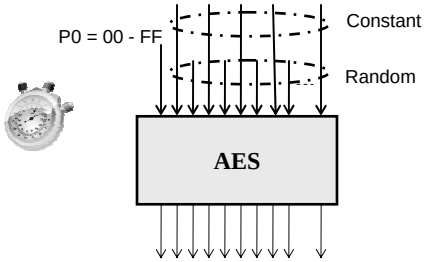
Modification's to Bernstein's Attack(1)

- 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.

Modification's to Bernstein's Attack(2)



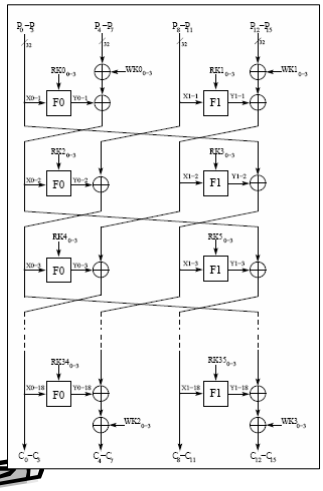
- Vary only those bytes which go to the same sbox as P0.



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



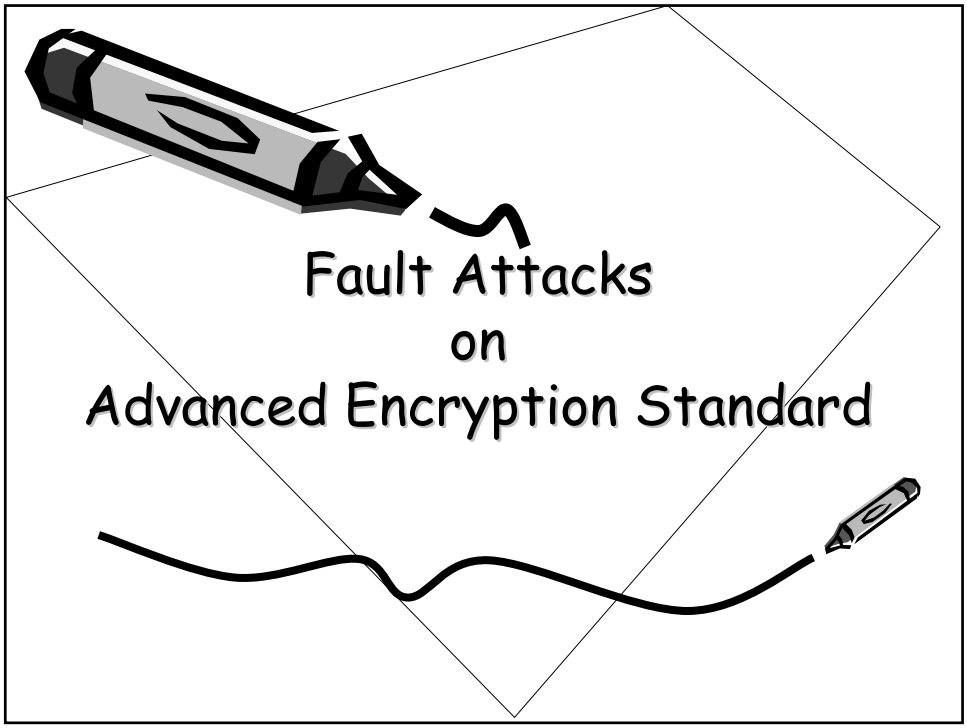
- In around 2^{26} Clefia encryptions the cipher can be shown to break in the face of cache timing attacks
- 3 GHz Intel Core 2 Duo
- 32 kB L1 Cache
- 1 GB RAM
- Linux (Ubuntu 8.04)
- gcc -4.2.4 with O3 optimization.
- Attack Time:
  - First Phase (with known key): 1300 sec
  - Second Phase (with unknown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

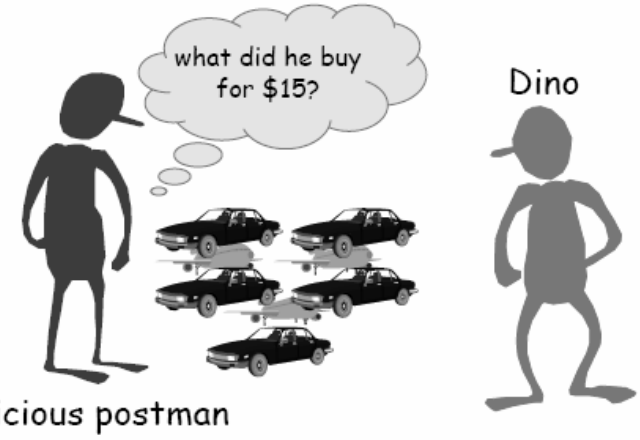
Key Byte	Correct Key	Obtained Correlation results (with correlation value)
$RK0_0$	0a	**0a**(*884.6*), 6b(*469.7*), 5f(*368.3*), 20(*357.3*), ef(*263.7*) ...
$RK0_1$	96	**96**(*1853.4*), 7b(*438.0*), bc(*437.5*), 4a(*366.7*), ee(*361.8*) ...
$RK0_2$	c1	**c1**(*1942.1*), 93(*672.7*), 98(*598.3*), f9(*573.2*), 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



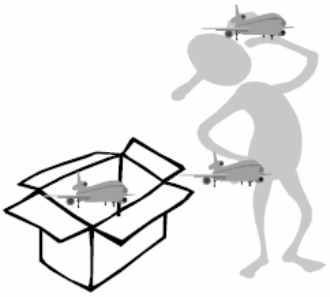
14/12/2009 Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006 32

The postman wants to know what Dino bought for $15

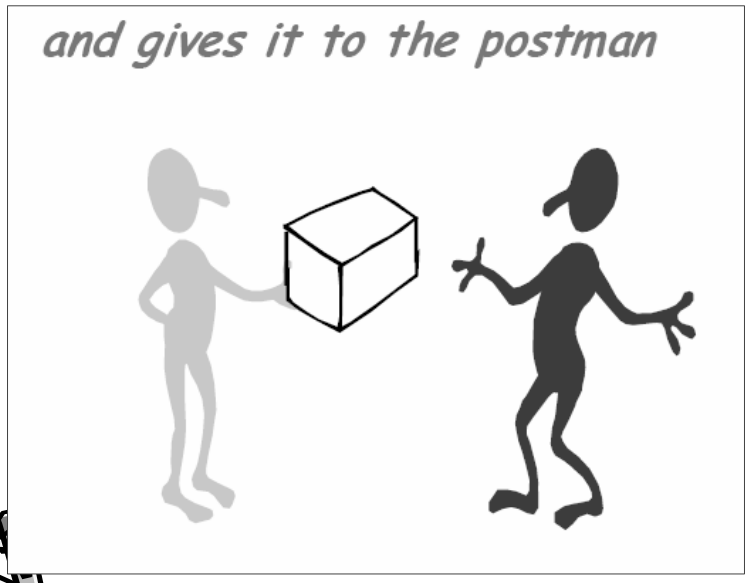


Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006

In the meanwhile Jack prepares the DHL



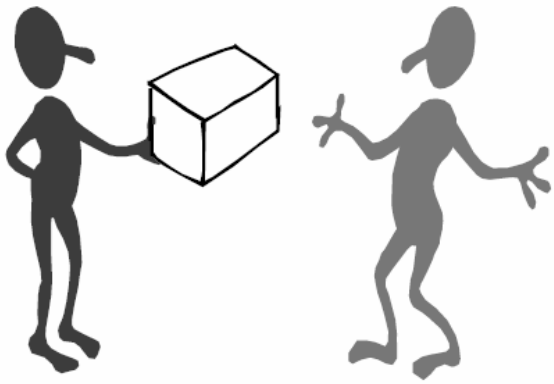
Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006



14/12/2009 Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006 35

14/12/2009 Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006 36

and gives it to Dino



14/12/2009 Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006 37

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!

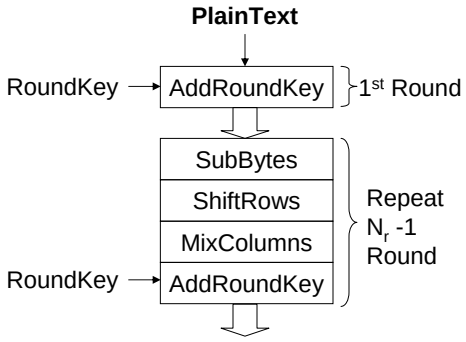
Hardware faults can have various sources: voltage glitches, light beams, laser beams...

14/12/2009 Taken from "The Sorcerer's Apprentice Guide to Fault Attacks", FDTC 2006 38

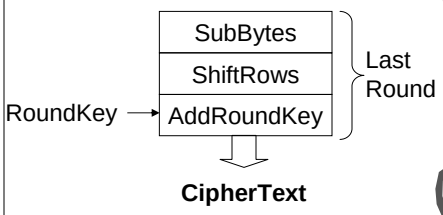
Fault Attacks on Block Ciphers

- 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

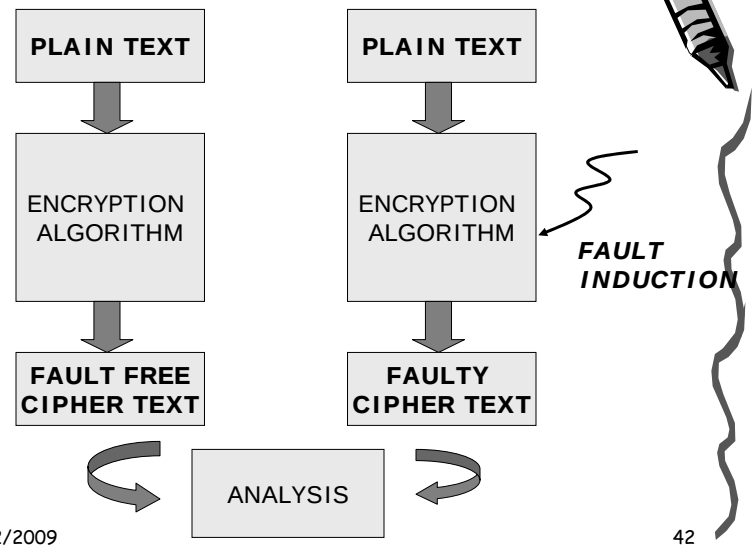


Last Round

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

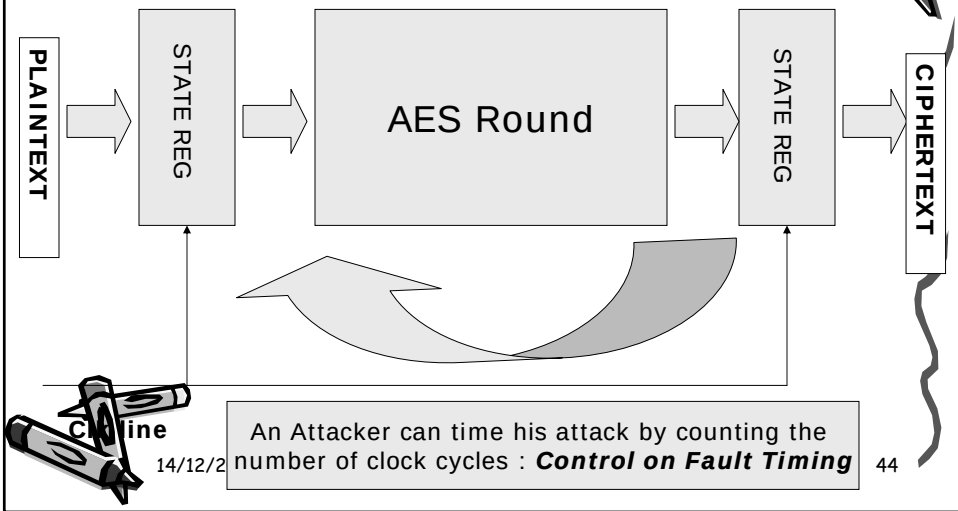
Illustration of a Fault Attack



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

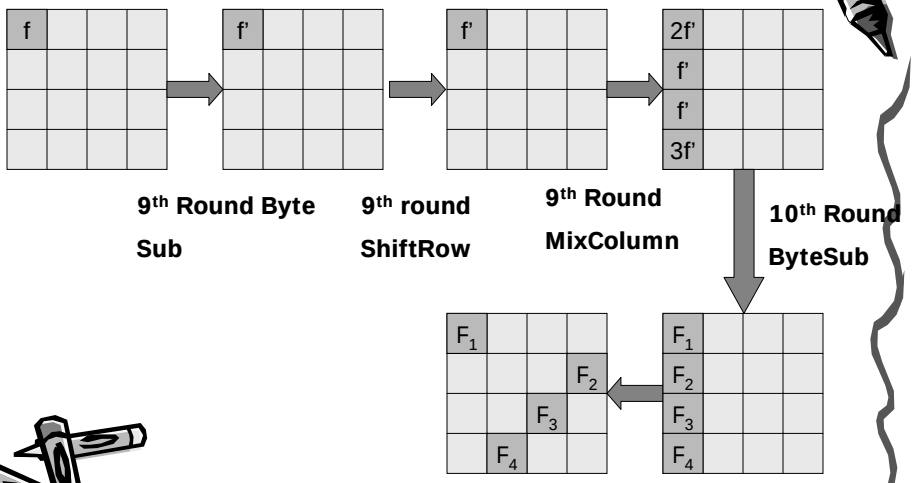


An Attacker can time his attack by counting the number of clock cycles : ***Control on Fault Timing***

Principle of the Attack

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

Propagation of Fault Induced



The patterns gives the following equations:

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

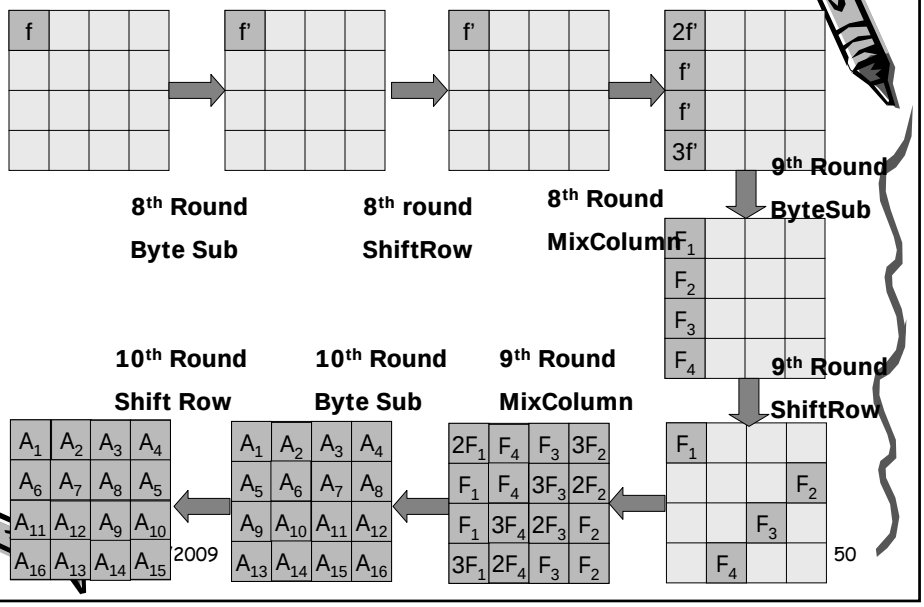
Important Points

- 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:
  $S^{-1}(x)$ ^ $S^{-1}(x$ ^ $a)=\beta$
- 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 7th & 8th round MixColumn	2
This Paper	Disturb 1 byte	Anywhere between 7th round MixColumn and last round input	2

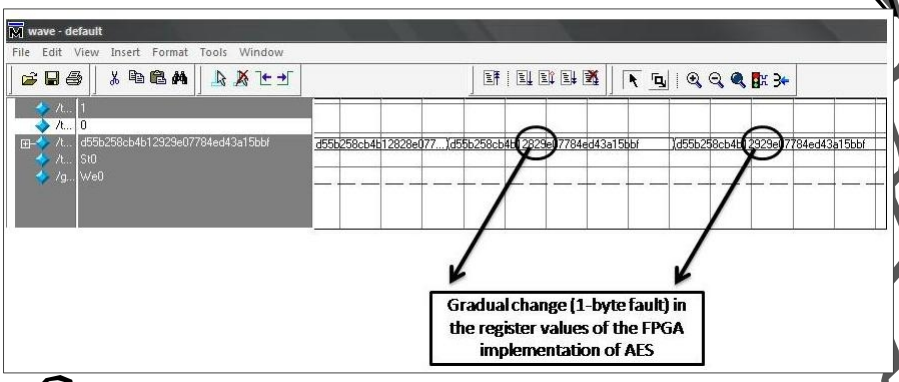
Comparison with existing fault attacks exploiting key scheduling

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

More information in:

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

Models of SCA

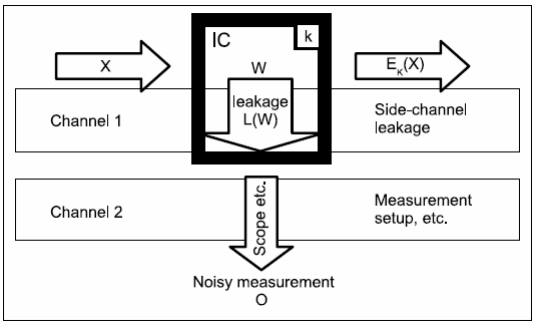
- Mutual Information (MI) Analysis:
  - Let X and Y be random variables in the discrete space *X* and *Y*, with probability distributions P_X and P_Y.
  - Then the Mutual Information between X and Y is:

$$I(X:Y)=H(X)-H(X|Y)$$

If X and Y are independent: I(X:Y)=0
If X and Y are dependent: I(X:Y)=H(X)

SCAs using MI

- Channels for Information Leakage:



Formalizing the Leakage

- Attacker obtains q>0 measurement traces, $o_{xi}(t)$, for q input values, xi being processed by $E_k(t)$.
- Leakage Function:
  - W→L : Information on words is leaked in L
  - L→O : Information on L is observed through the observable

The MI based formalism

- Adversary guesses a key value, k and computes $f_k(x)$, which is an intermediate result like S-Box($x \oplus k$)
- Usually, W is some bits of $f_k(x)$, say 3 MSB.
  - thus for each guesses key k, there are 8 bins: $L_k = \{L_0,\ldots,L_7\}_k$
  - Divide the observables for each inputs among this bins.

Probability Distribution

- **Compute the distributions:**

$$\mathrm{P}_{O|L_i} = \frac{|\{o_{x_j} = o \mid x_j \in L_i\}}{|L_i|}$$

$$\mathrm{P}_O = \frac{|\{o_{x_j} = o\}|}{q}$$

Entropy and Mutual Information

- These distributions lead to the following entropies:

  $H(O|L_i)$ and $H(O)$
- Define Mutual Information,

  $\mathbf{I(L_K;O)=H(O)-H(O|L_i)}$
- Note that these computations are under assumption of a portion of the key.

The MI based Distinguisher

- We repeat this process for all possible key values.
- **The attack strategy is to return the key K*, which maximizes the value of $I(L_k;O)$**
- Also find out the time intervals when the intermediate results are computed.

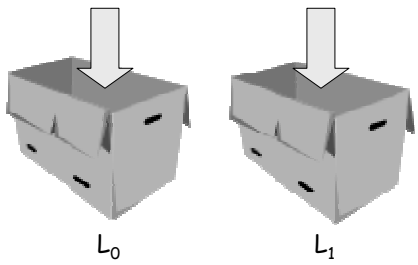
An Example

- Target: AES-128
- 8 bit microcontroller (AT90S8515)
- Observable: Voltage drop across a 50 ohm resistor in the ground line
- Sampled at t=1, …, 1800
- Plaintexts are varied at random
  - the experiment focuses on the first key byte.
  - q=1000 power traces are gathered, $O_{xi}(t)$, i=1 to q.

An Example (contd.)

- Compute $f_k(x)=\text{Sbox}(x \oplus k)$
- The leakage function, L is say the rth bit, if r=0 it is the LSB.

Distribute the x values into the two bins

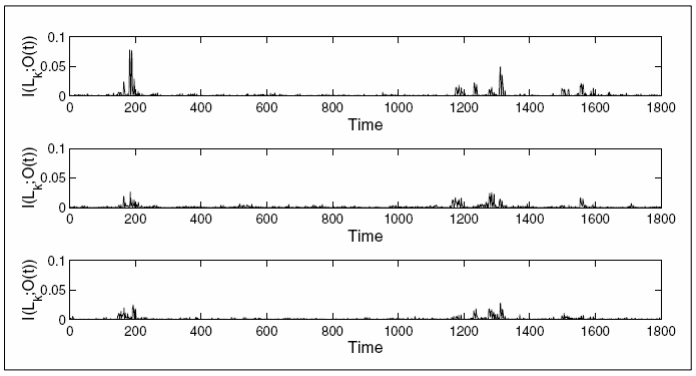


$$P_{O|L_i} = \frac{|\{o_{x_j} = o \mid x_j \in L_i\}|}{|L_i|}$$

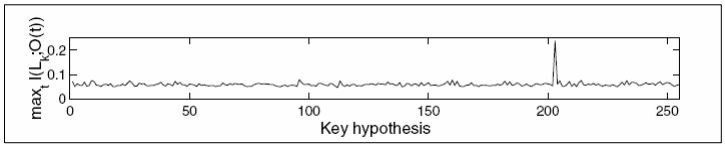
$$P_O = \frac{|\{o_{x_j} = o\}|}{q}$$

Compute, $I(L_k;O)$

Mutual Information Plots for r=0, 1 and 2



Obtaining the actual key



- The MI values are plotted wrt. the key
- The correct key is expected to exhibit larger MI value.

Formal Approach to Leakage Resilient Cryptography

"Stefan Dziembowski and Krzysztof Pietrzak", Leakage-Resilient Cryptography, FOCs, 2009

--attempts to solve the problem from of designing cryptosystems whose mathematical descriptions alone certifies their security against *all possible* forms of side channel attacks!

The Stream Cipher Ingredients

- Three variables A, B (secret) and K_0.
- Three loop variables at the input of the jth round:

  M_0^{j-1}, M_1^{j-1}, O^{j-1}
- Initialization: $M_0^0=A$, $M_1^{j-1}=B$, $O^{j-1}=K_0$

The Adversary Model

- **"Only Computations leak information"**: Micali and Reyzin, "Physically Observable Cryptography, TCC, pp. 278-296, 2004.
- S (Stream Cipher) only accesses $\tau_j = M_{j \bmod 2}^{j-1}$ in the jth round and the output O^{j-1}.

The Adversary Model

- The adversary Q can adaptively choose a function f_j with range $\{0,1\}^\lambda$, where $\lambda << |\text{secret key}|$
- The adversary gets K_j and $\Lambda_j = f_j(\tau_j)$
- The adversary views $View_j = [K_0, \ldots, K_j, \Lambda_1, \ldots, \Lambda_j]$

Measure of Security

- $S(A,B,K_0) \rightarrow Q$: random experiment where an adversary Q attacks S initialized with a key A, B, K_0
- $view(S(A,B,K_0) \rightarrow Q)$: view of Q at the end of the attack

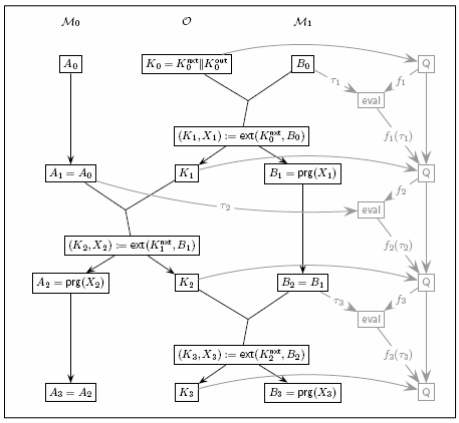
$$p_{rand} = \Pr_{A,B,K_0}[D_{ind}(view(S(A,B,K_0) \rightarrow Q, U_k) = 1]$$

$$p_{real} = \Pr_{A,B,K_0}[D_{ind}(view(S(A,B,K_0) \rightarrow Q, K_j) = 1]$$

$$AdvInd(D_{ind}, Q, S, j) = | p_{real} - p_{random} | < \varepsilon$$

A leakage resilient stream cipher

- **Uses a Pseudorandom generator (PRSG)**
- **The amount of leakage is at least <u>logarithmic</u> of the internal state**



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

- 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"**, To appear 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*).
- Benedikt Gierlichs, Lejla Batina, Pim Tuyls and Bart Preneel, "**Mutual Information Analysis - A Generic Side-Channel Distinguisher"**, CHES 2008

References

- Stefan Dziembowski and Krzysztof Pietrzak, "**Leakage-Resilient Cryptography"**, FOCs, 2009
- "Only Computations leak information": Micali and Reyzin, **"Physically Observable Cryptography"**, TCC 2004.