Advance topics in Hardware Security from Theory to Practice 049017

Prof. Avi Mendelson –
avi.mednelson@technion.ac.il

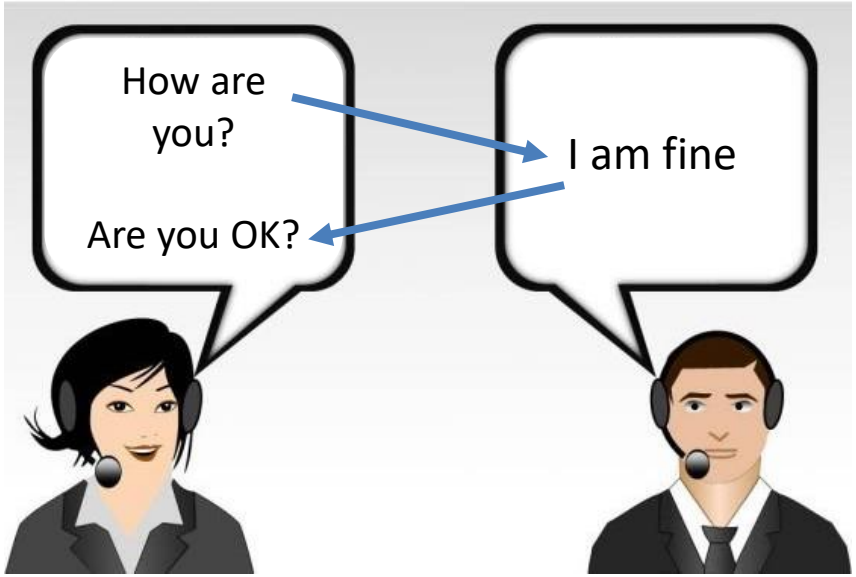
Lecture 6 – Side channel-II.

Agenda

- Rehash
- Covert Channel
- Timing based attacks (AES)
- Power based attacks (DPA)

Side-Channel attack

- Side channel is a technique to reason about a system form “external measurements”
- A side-channel attack is based on information gained from the physical implementation of a system (process), rather than theoretical weaknesses in the algorithms

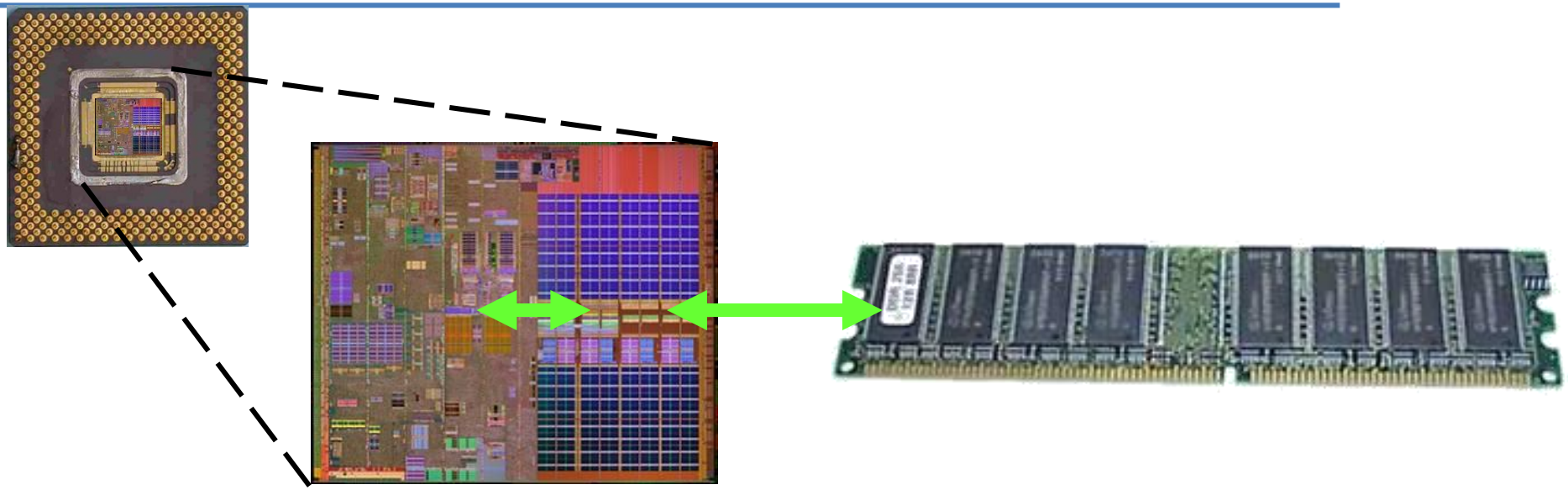


What make her to suspect that something is wrong?

Types of attacks

- White Box – the internal structure of the target is know; e.g., the algorithm, hardware structure
  - SPA – simple power attack
  - Timing attacks
- Black Box – only external observations are known, but no information regarding the internal information can be taken into account
  - DPA – differential power attack
- Grey Box – Some information is known
  - Pattern attack

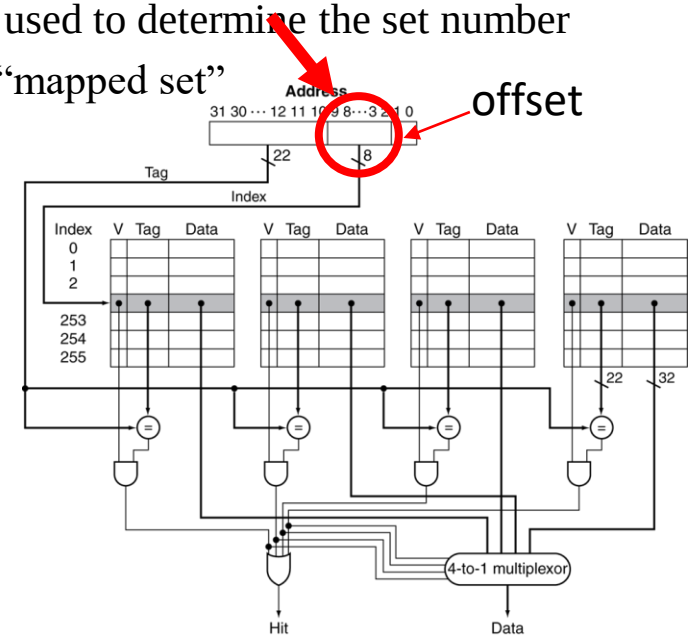
Timing attacks via Cache



	CPU core	cache	Main memory
Annual speed increase:	60% (until recently)		7-9%
Typical latency:		0.3ns	50-300ns ⟶ **timing gap**

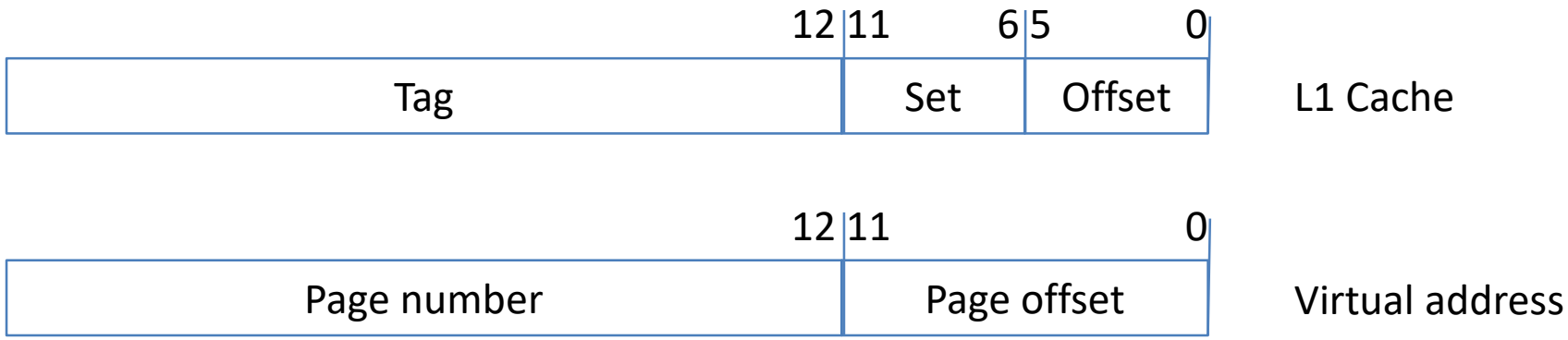
How Cache Works?

- When a memory access instruction is processed, memory cell is searched in the cache first.
- If a cache miss occurs, a full memory block is copied into the appropriate set (S possible sets) into **one** of the W cache lines.
- LRU (least recently used) is usually usedto determine which cache line to replace (if needed). Some systems combine LRU + pseodo random mechanism
- Few more comments:
  1. For L1 cache, a set of bits out of the address are being used to determine the set number
  2. For LLC, an hashing is performed in order to find the “mapped set”



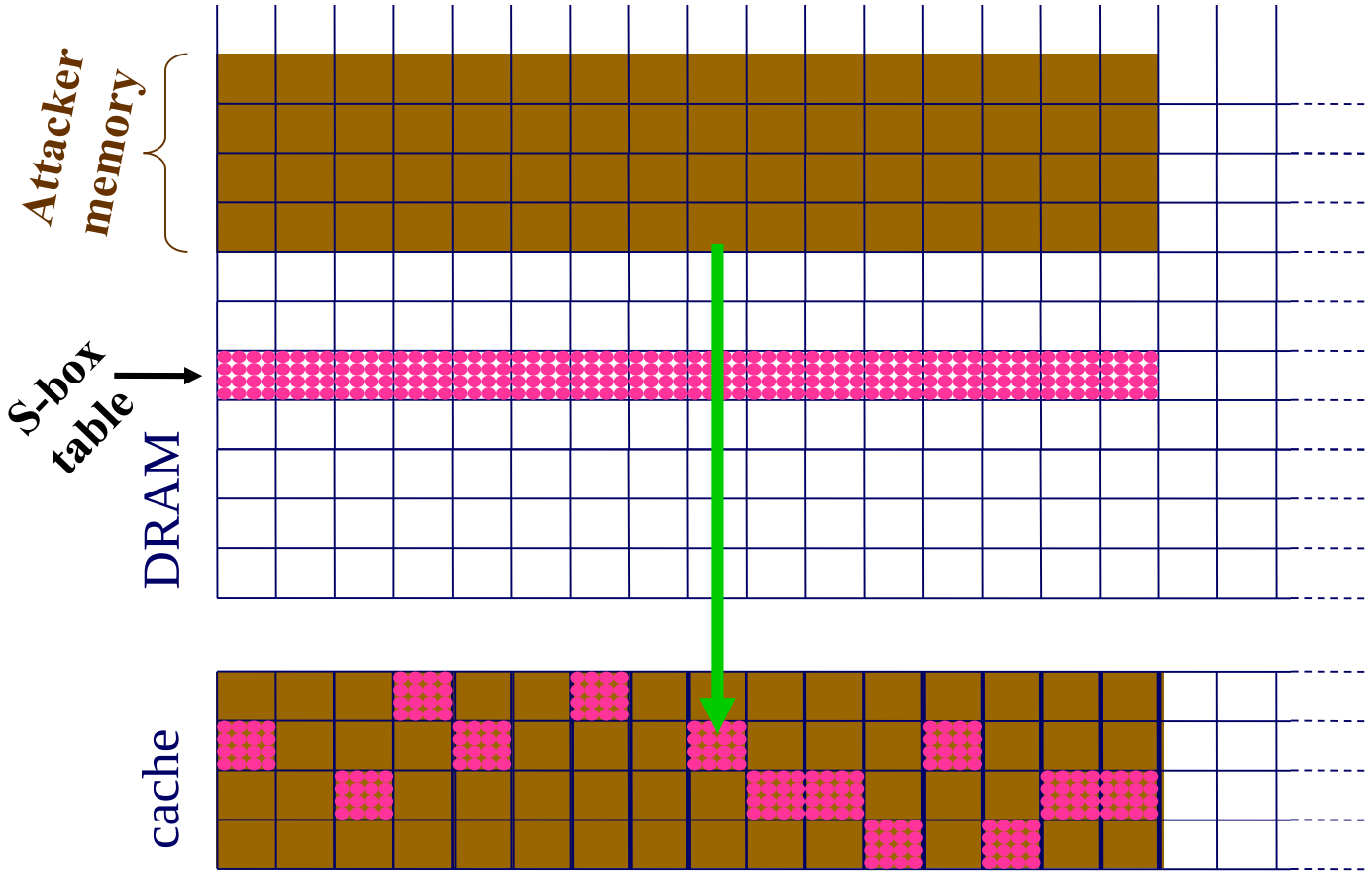
Virtual address space

Cache width (#sets x Cache_block_size) smaller or equal to a page size so mapping doesn't impact by virtual mechanism



Measurement Method 2: Prime + Probe

• Trying to discover the set of memory blocks read by the encryption *a posteriori*, by examining the state of the cache after encryption.

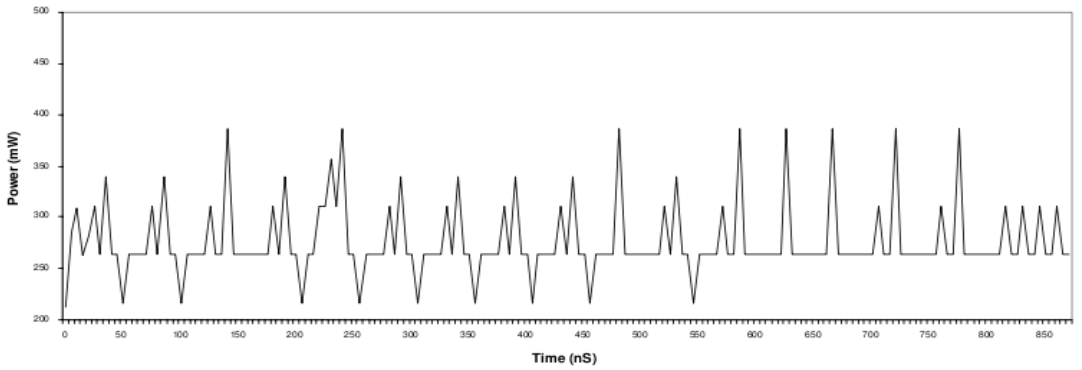


1. Completely evict tables from cache
2. Trigger a single encryption
3. Access attacker memory again and see which cache sets are slow

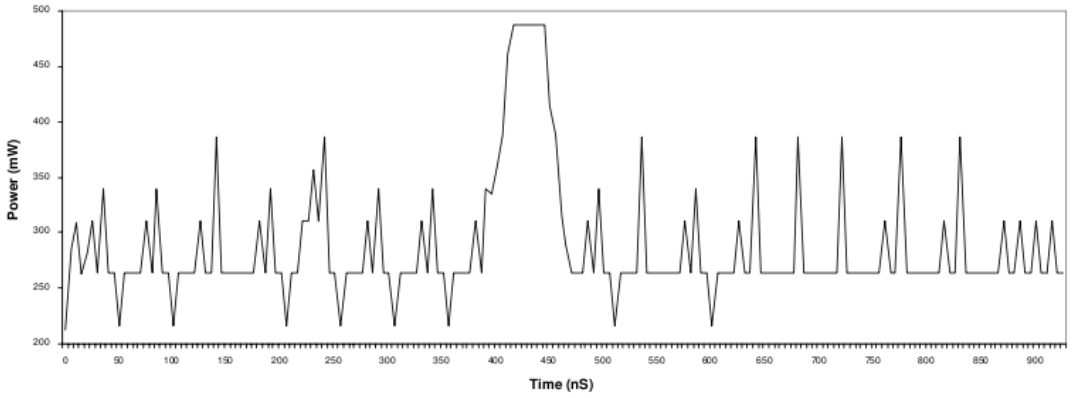
How can we measure time ? Cache Trace Attacks

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

Without Cache Miss

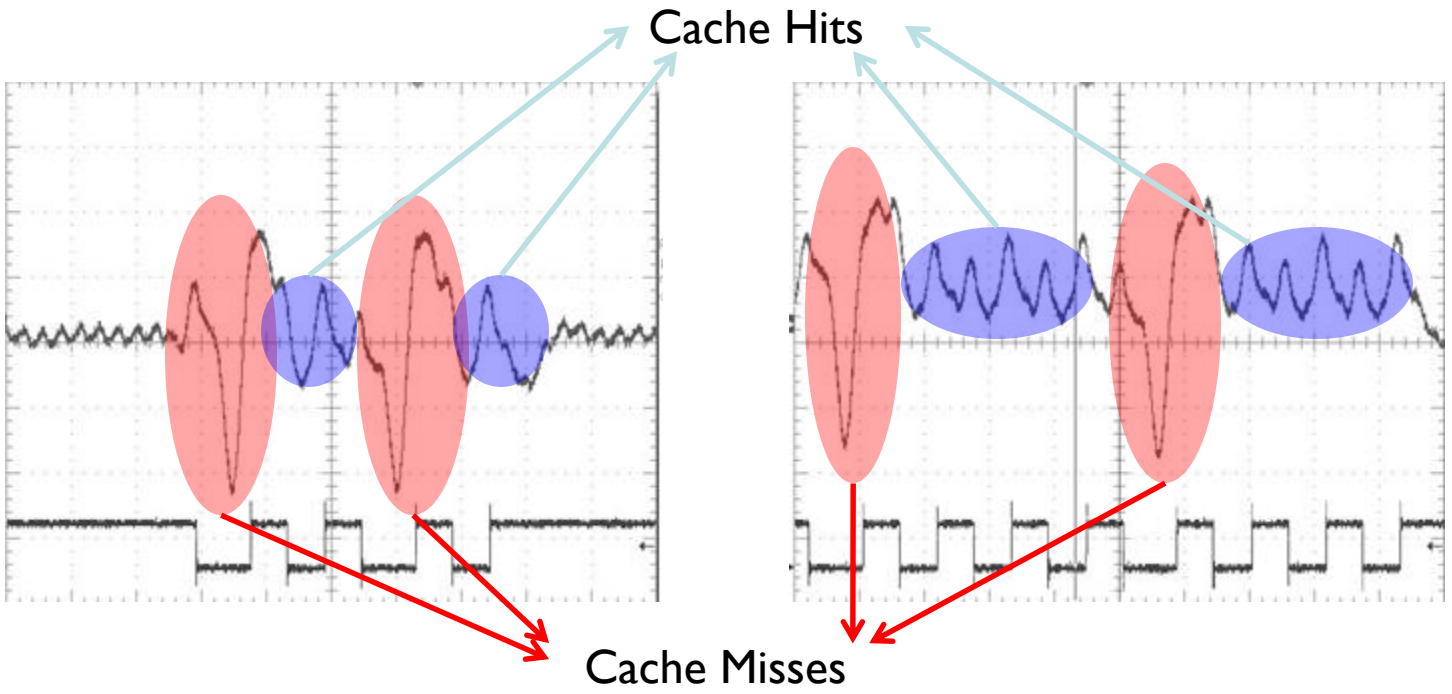


With Cache Miss



CacheLeaks - Power

Cache Memories Leak Information



- Oscilloscope Waveforms
- Power Consumptions of 4 accesses to the S-Box
- But the correspondence is not so obvious for the complete cipher.

Covert Channel

- Covert Channel is a mechanism that aims to allow 2 process that should not communicate to exchange information

- Example
  - 2 processes needs to know if the are running on Multi-threaded core or on 2 different cores
  - They decide to take advantage that L1 is shared only between multithreaded virtual cores
  - Process A fill out L1
  - Process B make sure he never use all the ways in 10 different known sets
  - When A returns, if found “remaining” of his data is ALL these sets (or the majority of them), most likely he share the core with the other thread

Using time-based side channels in AES

- Common implementation of AES, uses 4 lookup tables.
- These tables are precomputed once by the programmer or during system initialization.
- There are 8 such tables, T_0; T_1; T_2; T_3 and $T_0^{(10)}$; $T_1^{(10)}$; $T_2^{(10)}$; $T_3^{(10)}$, each containing 256 4-byte words(*).
- During setup time, the 16 byte secret key $\boldsymbol{k} = (k_0, \ldots k_{15})$ is expanded into 10 round keys $\boldsymbol{K}^{(r)}$ for $r = 1,\ldots, 10$

(*) Joan Daemen, Vincent Rijmen, *The design of Rijndael: AES | The Advanced En*
ISBN, 3-540-42580-2, Springer-Verlag, 2001

Notation the paper is use

- <y> = $\lfloor y/\delta \rfloor$ ➔ *memory block of y*
- if <y> = <z> iff, when used as lookup indices into the same table T_l, they would cause access to the same memory block
- $Q_k(\boldsymbol{p},l,y) = 1$ iff the AES encryption of the plaintext ***p*** using the encryption key ***k*** accesses the memory block of index *y* in T_l at least once throughout the 10 rounds
- $E[M_k(\mathbf{p},l,y)]$ is larger as it predict that $Q_k(\boldsymbol{p},l,y) = 1$ most of the time

First round accesses

T_0	$T_0[P_0 \oplus K_0]$	$T_0[P_4 \oplus K_4]$	$T_0[P_8 \oplus K_8]$	$T_0[P_{12} \oplus K_{12}]$
T_1	$T_1[P_1 \oplus K_1]$	$T_1[P_5 \oplus K_5]$	$T_1[P_9 \oplus K_9]$	$T_1[P_{13} \oplus K_{13}]$
T_2	$T_2[P_2 \oplus K_2]$	$T_2[P_6 \oplus K_6]$	$T_2[P_{10} \oplus K_{10}]$	$T_2[P_{14} \oplus K_{14}]$
T_3	$T_3[P_3 \oplus K_3]$	$T_3[P_7 \oplus K_7]$	$T_3[P_{11} \oplus K_{11}]$	$T_3[P_{15} \oplus K_{15}]$

- AES encryption uses a sequence of 160 table lookups to indices $l1, l2, \ldots, l160$.
- A “cache collision” occurs if two separate lookups li, lj satisfy $\langle li \rangle = \langle lj \rangle$.

Cache-Collision Assumption. *For any pair of lookups i, j, given a large number of random AES encryptions with the same key, the average time when* $\langle li \rangle \neq \langle lj \rangle$ *will be less than the average time when* $\langle li \rangle = \langle lj \rangle$

First Round Attack

- In the first round, bytes x^0_0, x^0_4, x^0_8, x^0_{12} are used as an index to T_0 (we say they creat a "family")
- 3 other families are created for T_1,T_2 and T_3
- A cache collision occurs whenever two bytes x^0_i, x^0_j **in the same family** satisfy $\langle x^0_i \rangle = \langle x^0_j \rangle$
- This should occur when
  $\langle p_i \rangle \oplus \langle k_i \rangle = \langle p_j \rangle \oplus \langle k_j \rangle$, or after rearranging,
  $\langle p_i \rangle \oplus \langle p_j \rangle = \langle k_i \rangle \oplus \langle k_j \rangle$

First Round Attack – cont.

1. For each table T (e.g., T0) find the average access time to table $t[i, j, \langle pi \rangle \oplus \langle pj \rangle]$, $\exists i, j \in$ same table family.
2. If a low average time occurs at $t[i, j, \Delta]$, the algorithm estimates that $\langle ki \rangle \oplus \langle kj \rangle = \Delta$.
3. For each table family, the attacker will eventually have a redundant set of six equations, such as
   $\langle k_0 \rangle \oplus \langle k_4 \rangle = \Delta_1$, $\langle k_0 \rangle \oplus \langle k_8 \rangle = \Delta_2$,
   $\langle k_0 \rangle \oplus \langle k_{12} \rangle = \Delta_3$, $\langle k_4 \rangle \oplus \langle k_8 \rangle = \Delta_4$,
   $\langle k_4 \rangle \oplus \langle k_8 \rangle = \Delta_5$, $\langle k_8 \rangle \oplus \langle k_{12} \rangle = \Delta_6$ for table T_0

First round attack – partial conclusions

- For each combination of bytes within the "family"; e.g., for $T_0 = P_0, P_4, P_8$ and P_{12}, we can calculate all the positions that can cause a collision.
- This will reduce the number of possibilities, to be small enough, so we can guess the options in a "reasonable time".
- The problem is that due to the table sizes, we can use the method just to guess half of the bytes and we need to have an exhaustive search on the rese

AES – Attack on Round>1

- For rounds 1-9, the state is calculated as following

$$
\begin{aligned}
X^{i+1} = \{ & T_0[x_0^i\ ] \oplus T_1[x_5^i\ ] \oplus T_2[x_{10}^i] \oplus T_3[x_{15}^i] \oplus \{k_0^i\ , k_1^i\ , k_2^i\ , k_3^i\ \}, \\
& T_0[x_4^i\ ] \oplus T_1[x_9^i\ ] \oplus T_2[x_{14}^i] \oplus T_3[x_3^i\ ] \oplus \{k_4^i\ , k_5^i\ , k_6^i\ , k_7^i\ \}, \\
& T_0[x_8^i\ ] \oplus T_1[x_{13}^i] \oplus T_2[x_2^i\ ] \oplus T_3[x_7^i\ ] \oplus \{k_8^i\ , k_9^i\ , k_{10}^i, k_{11}^i\}, \\
& T_0[x_{12}^i] \oplus T_1[x_1^i\ ] \oplus T_2[x_6^i\ ] \oplus T_3[x_{11}^i] \oplus \{k_{12}^i, k_{13}^i, k_{14}^i, k_{15}^i\}\}.
\end{aligned}
$$

- Since there is no MixColumns transformation in the last round, it use S-box ‘S’ tables instead of these ‘T’ tables for the last round.

Second/Last-round attack - cont

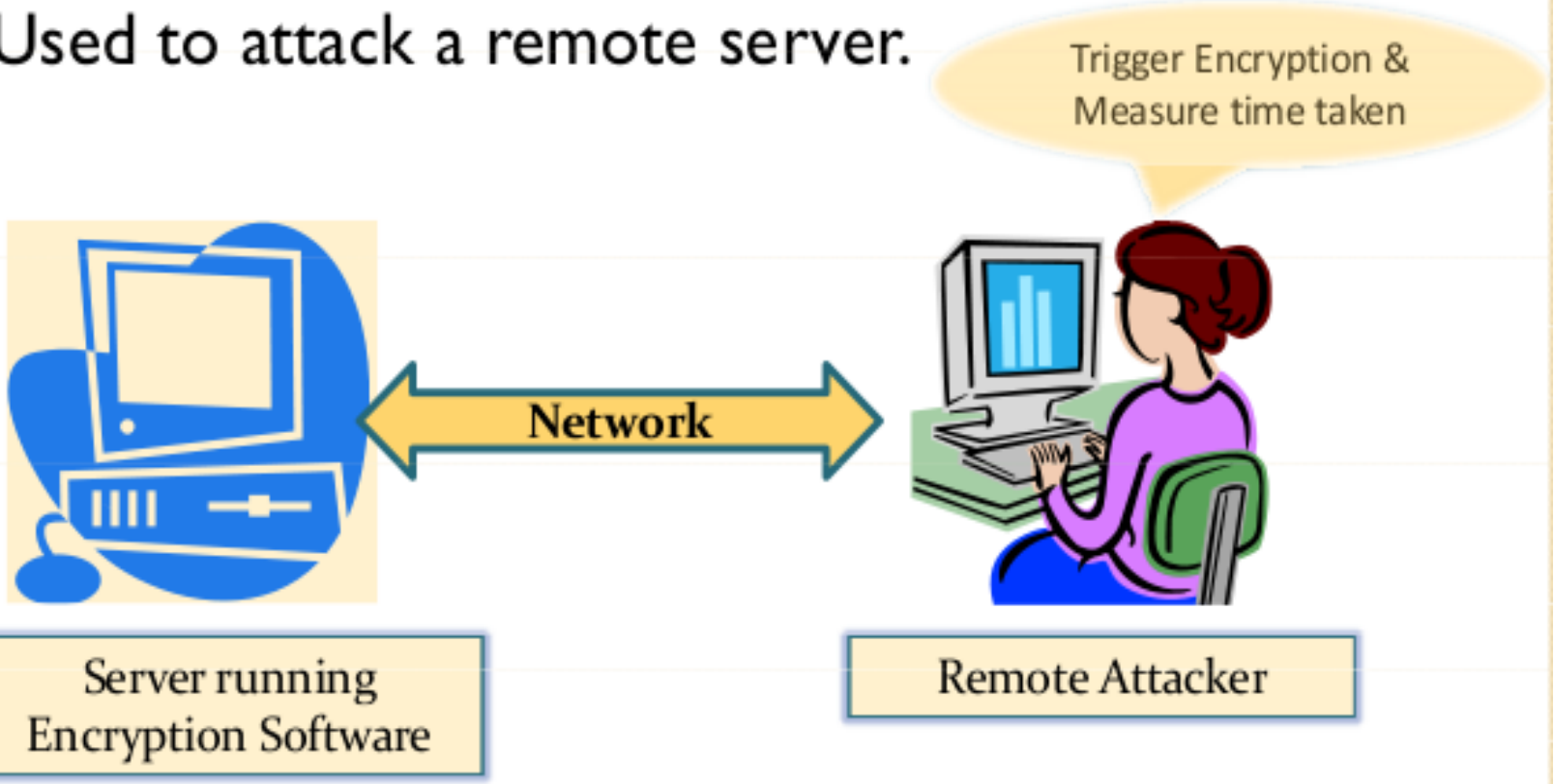
- But, usually we can measure only at the end of the entire encryption.
- Here, we have 36 accesses to Tl (4 for each round and 9 rounds).
- The remining 35 accesses to different bytes will also impact
- so heuristically the probability that a cache set will not be accessed at any round is
  $(1 - \delta/256)^{35} = 0.104 \rightarrow$ 10% assuming $\delta = 16$
- This attack can narrows each key byte down to one of δ possibilities
- In reality, we do not have the luxury of the ideal predicate, and have to deal with measurement score distributions $M_k(\boldsymbol{p}; l; y)$

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.



DPA – Differencial power analsis

Statistical behavior - cont

- In Figure 5 the assumption is that the power consumption when LSB=0 is different than the power consumption when LSB=1
- The graph shows the traces where the LSB=1 (M=16.9, SD=10.7) and where LSB=0 (M=121.9 SD=9.7).
- To aluminate the need for information about the target device, we averaging the measurements over time (offsets) within the traces.
- The basic DPA process examines the difference of these averages at each point in the set of traces

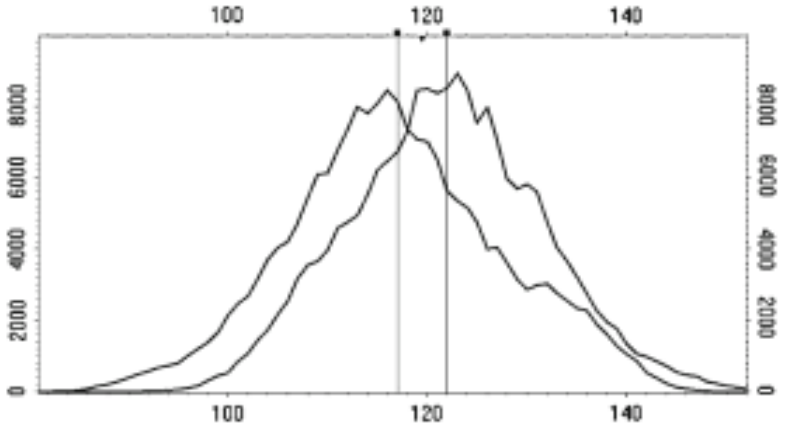


Fig. 5 Distributions of power consumption measurements for traces with the LSB of the output of the first S-box being 1 (left) and 0 (right)

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

Differential Power Analysis -- DPA

- The basic method involves partitioning a set of traces into subsets, then computing the ***difference of the averages*** of these subsets (A.K.A Differences of Means – DoM)
  - If the choice of which trace is assigned to each subset is uncorrelated to the measurements contained in the traces, the difference in the subsets' averages will approach zero as the number of traces increases.
  - Otherwise, the averages will approach a nonzero value. Given enough traces
- extremely tiny correlations can be isolated—no matter how much noise is present in the measurements

DPA – Cont.

Four traces are shown;

1. The average of the traces: LSB = 1
2. The average of the traces: LSB = 0
3. the difference of the top two traces – entire trace
4. The difference of the averages when focused on the “focused area” (*Y* axis scaling increased by a factor of 15.

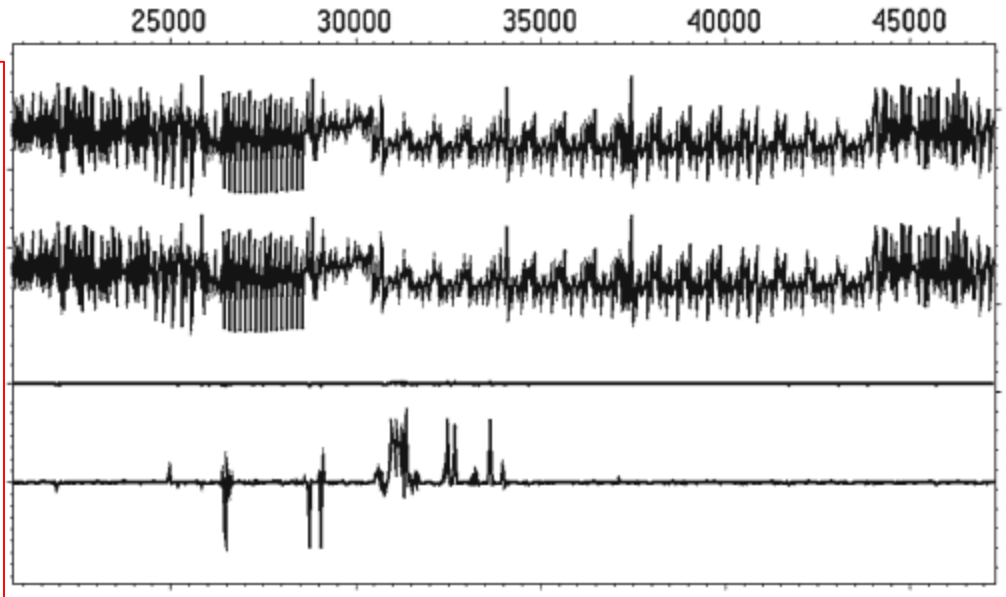


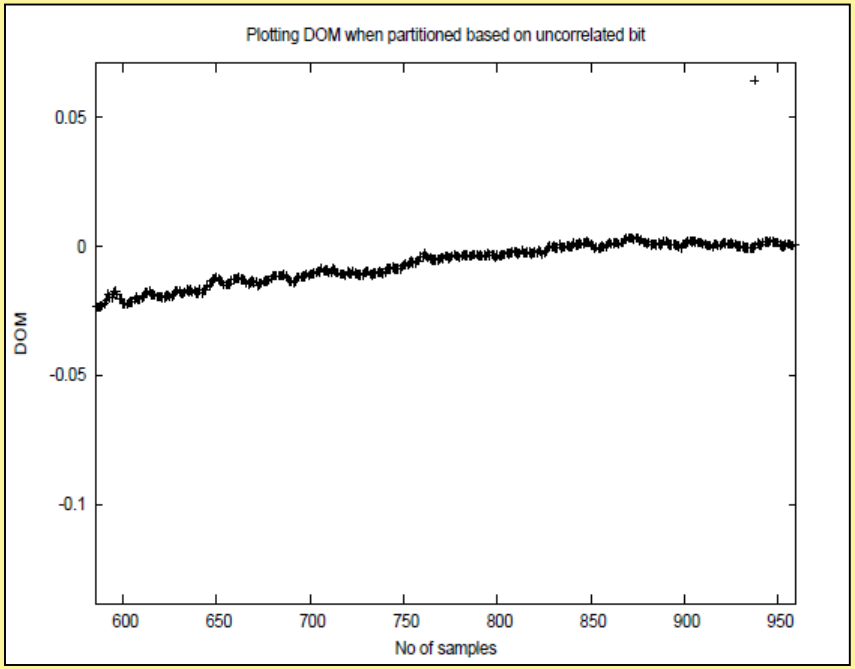
Fig. 6 Typical DPA result showing (from top to bottom) the average of the traces where the LSB of the output of first S-box in round 1 is 1, the average of traces where the LSB is 0, the difference between the top two traces, and the difference with the *Y* axis magnified by a factor of 15

Principle of DPA

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

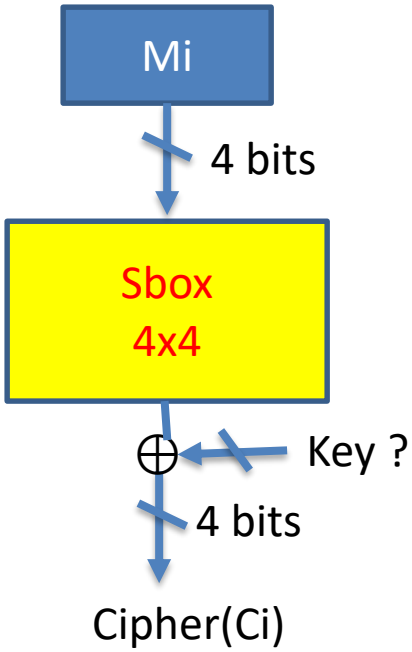
When the partitioning is done wrt. an uncorrelated bit?



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

A Toy DPA – Simple AES like function

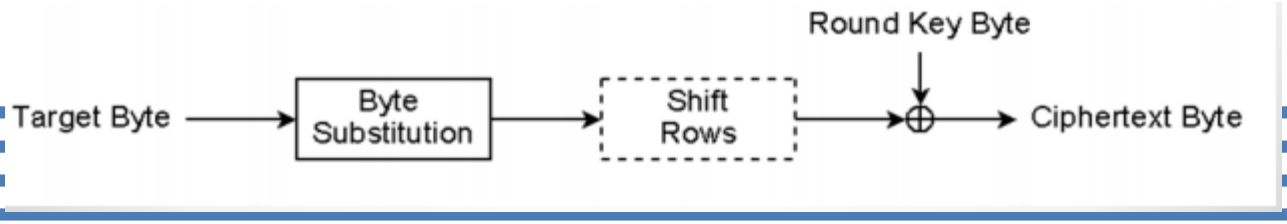
- Suppose we like to find the key, by observing Ci
  - For large number of messages Mi
    1. Guess a key (for each 16 options)
    2. Generate Ci from Mi and Key
    3. Use the inverse function and chosen K to find the Mi
    4. Assume the power proportional to the Hamming Weight (HW) of Mi
    5. Partition the messages with respect too their LSB
    6. The key is the one with the largest DoM



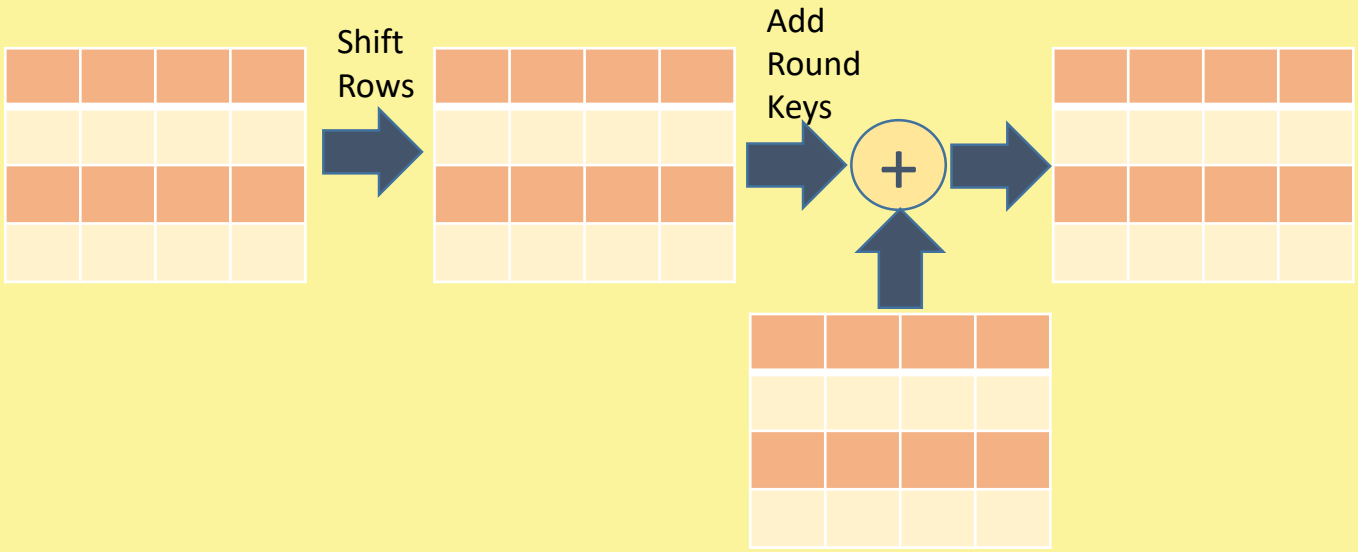
(it assumes that Sbox is served as a random function so w/o correlation is act like a white noise)

Differential Power Analysis on AES

- Select intermediate bit to analyze
  - Target the S-box in final round
    - Since Sub-Bytes operates on each byte independently
  - XORed with final round key value
- Collect power traces and corresponding ciphertext values
- Compute intermediate value
  - Ciphertext value is known
  - Make a guess for key byte
- Partition power traces into 2 sets
  - One set where computed bit is “1” and another where bit is “0”
- Compute average of each set-Compute the difference between the averages
  - If the average depends on the selected bit, and the bit “leaks”, then a correlation will be seen
- Repeat for other 255 key byte guesses using same power measurements

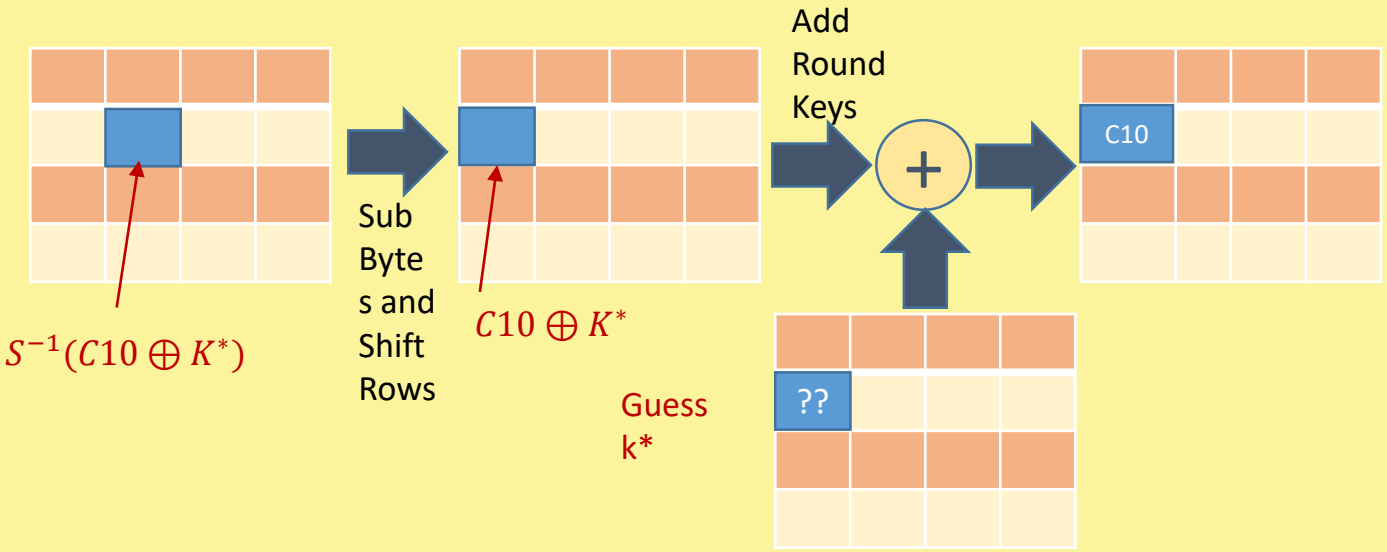


The Selection Function $\mathbb{D}$



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

The Selection Function $\mathbb{D}$



D(C10,b=0,K10)=$S^{-1}(C10 \oplus K^*)|_{(b=0)}$

If the key guess is correct, this matches with the correct value for all ciphertexts collected. However, if wrong it matches roughly half times, assuming sufficiently large number of cipher samples have been collected.

DPA Mathematically

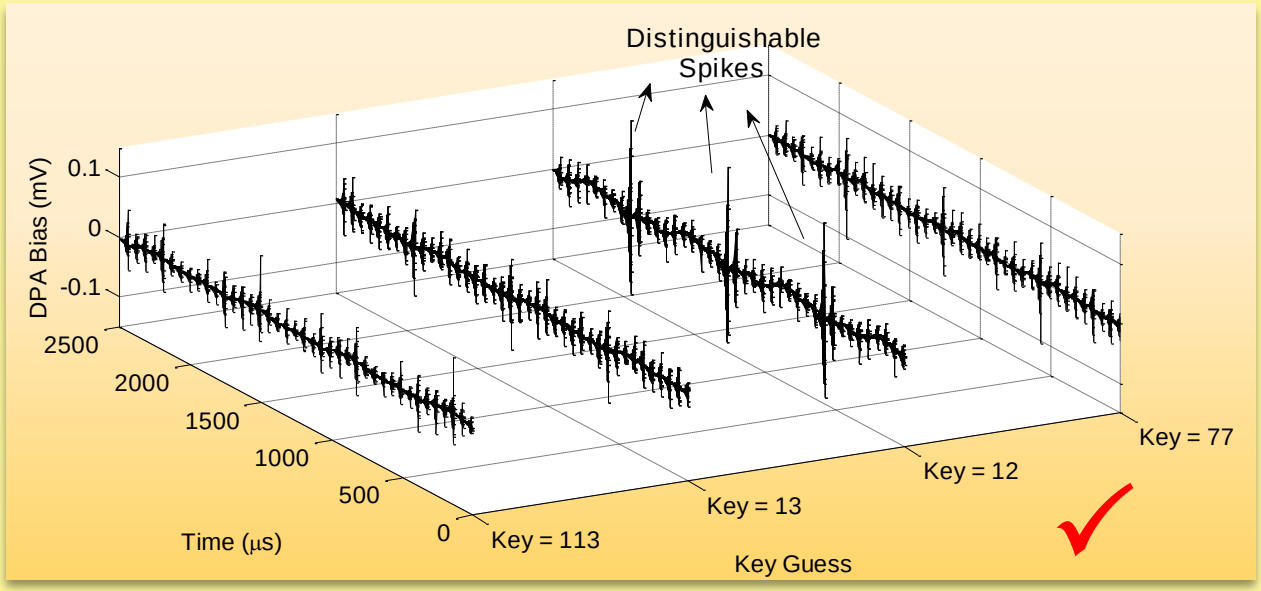
- Attacker now computes a k-sample differential trace $\Delta_D[1..k]$ by finding the difference between the average of the traces for which D(…) is one and the average for which D(…) is zero.

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

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.

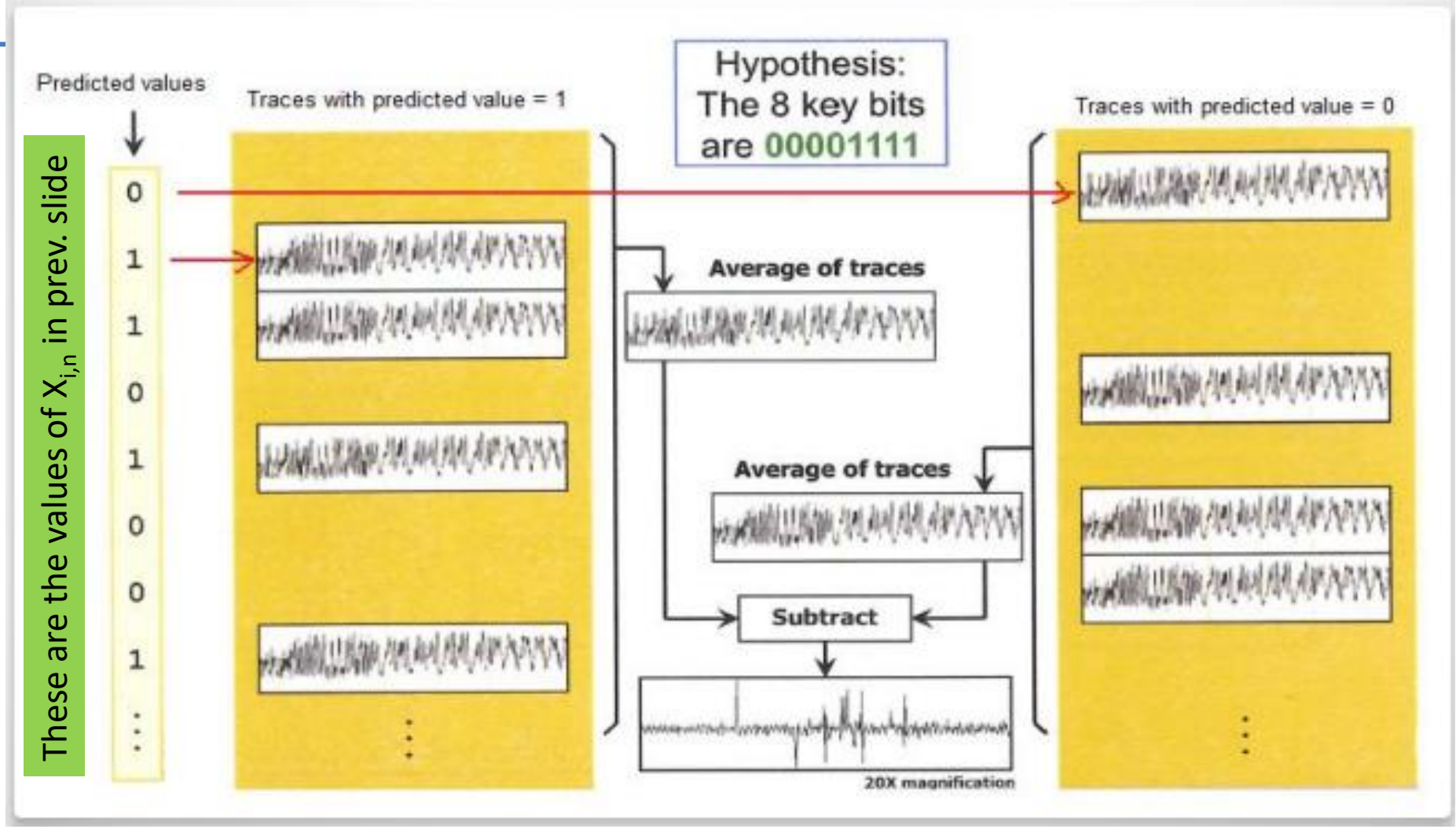
DPA Results - AES

3D Differential Plot



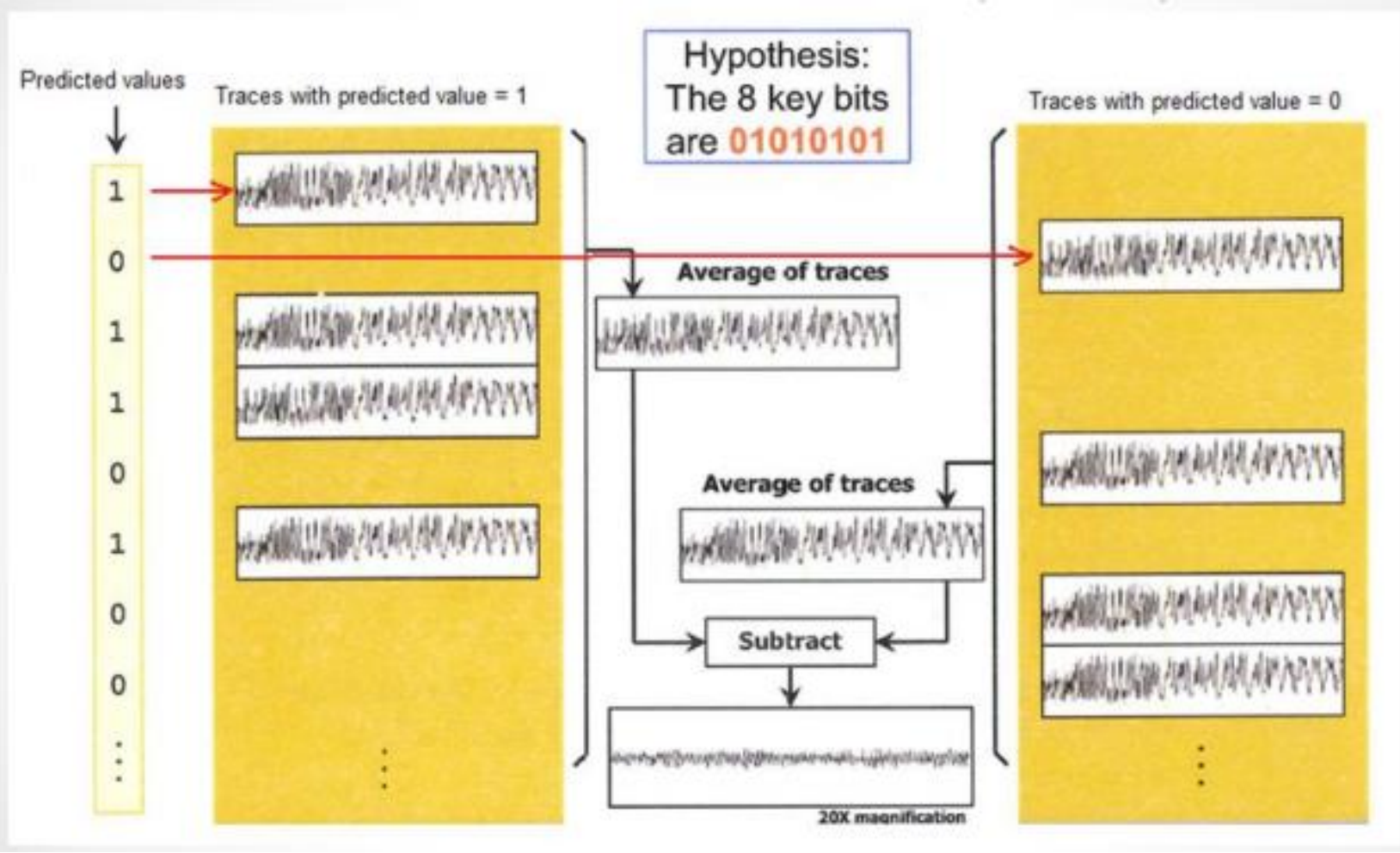
SBOX – 11 **BIT – 8** **TRACE COUNT = 15,000**

DPA Evaluation Process



DPA with correct Key guess

DPA Evaluation Process (cont.)



DPA with incorrect Key guess

Assumption: when wrong key is selected, no correlation between the Average(0) vs. Average(1)

SPA vs. DPA and more

- SPA is using a “white box model” where you expected to fully understand the implementation of the encryption algorithm
- DPA is using a “black box model” where almost no information is needed on the hardware that implement the encryption
- But the system depends on the power model and on the selection mechanism.
- Next class we will discuss “template attacks” amd correlated attacks (CPA) aims to integrate information we learned about the system as part the algorithm

backup