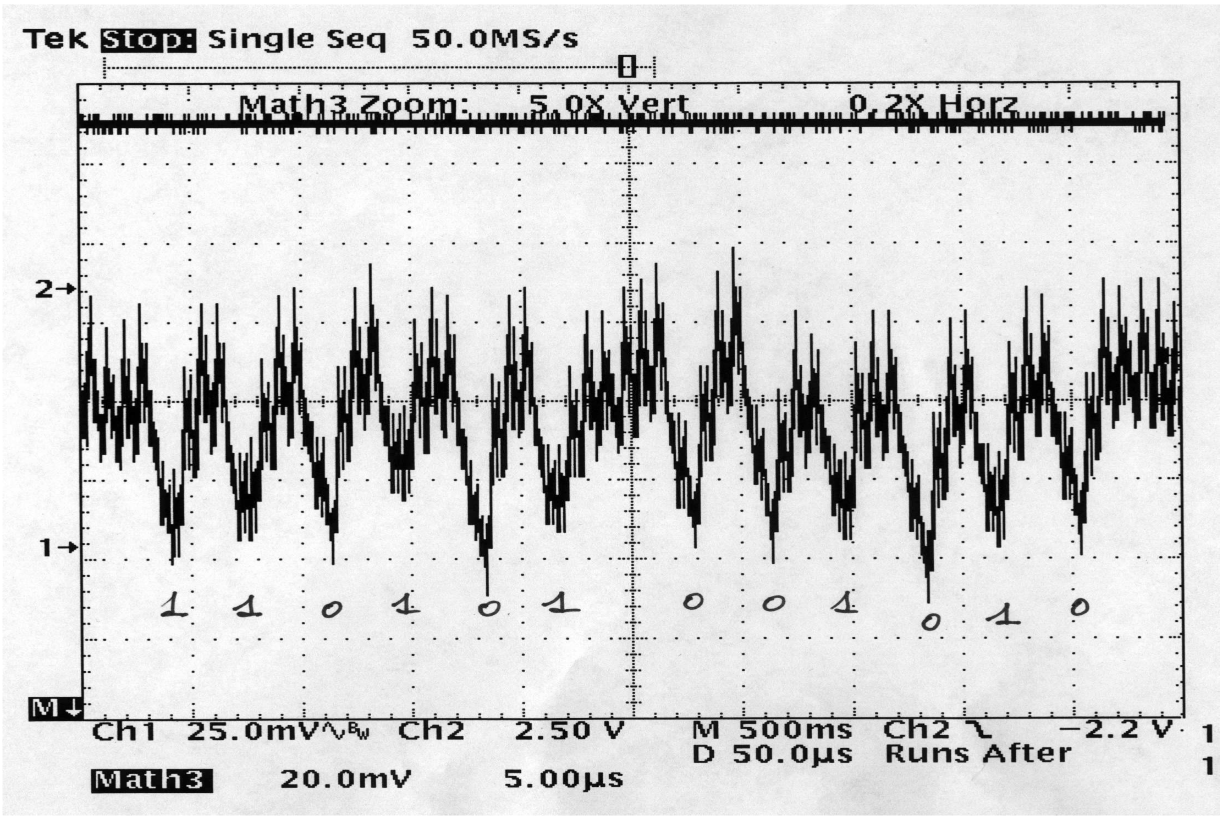
All About Side Channel Attacks
(main document to study)
Applied Crypto COMPGA12

Simple Power Analysis (SPA)

Looks at power consumption (but can also be any other side channel).
In principle, with standard silicon technology, more or less **<u>every</u>** unprotected (functional) implementation of a cryptographic algorithm, will be insecure and broken by SPA. The attacker can just see the key (or related data) if he has a good oscilloscope and "zooms in". Example for [triple] DES:



SPA Defences

The most basic defences are:

- Adding noise to the signal
- Desynchronization, dummy instructions

However noise can be eliminated with by averaging (and DPA see later).
Desynchronisation is much harder to remove.

SPA on RSA

Square and Multiply: =def=

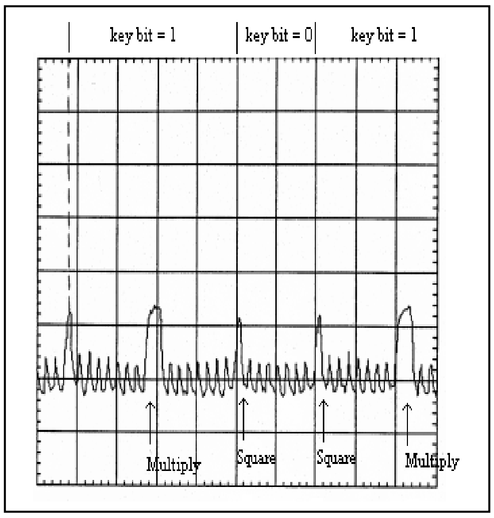
Let $d = \Sigma_{i=1}^{k} d_i$ be the binary expansion of d.

- Compute x, x^2, x^4, x^8, …, $x^{2^\wedge(k-1)}$,
- Multiply all those for which d_i=1.

In many implementations it is done as follows:

- If a bit of the private key is d_i = 1, we square+multiply.
- If the bit is 0, we just square.

Again the secret key is visible to the attacker!



Defences For RSA

In addition to general defences (same as before) there are specific methods which exploit the mathematics of RSA to the advantage of the defender.

- Computation never done twice in the same way.
- Blind the exponent =def= replace d by d'=d+A*(p-1)*(q-1) where A is random.
- Blind the modulus as well: use a small multiple of N instead of N.

Differential Power Analysis (DPA)

The name is misleading. This has NOTHING to do with differential cryptanalysis, but is a CONDITIONAL attack where we compare two conditional average curves (averaging power consumption curves removes noise and amplifies useful info.).

Here is a short description of a DPA attack on an implementation of DES / triple DES.

1.
The key observation in this attack is that if we know 6 bits of the key in DES, for example those that are XORed to the state before applying the S-box S1, we can predict in a computer simulation the output of this S-box. In particular we can compute bit number, say number 9, that goes into the next round of DES.

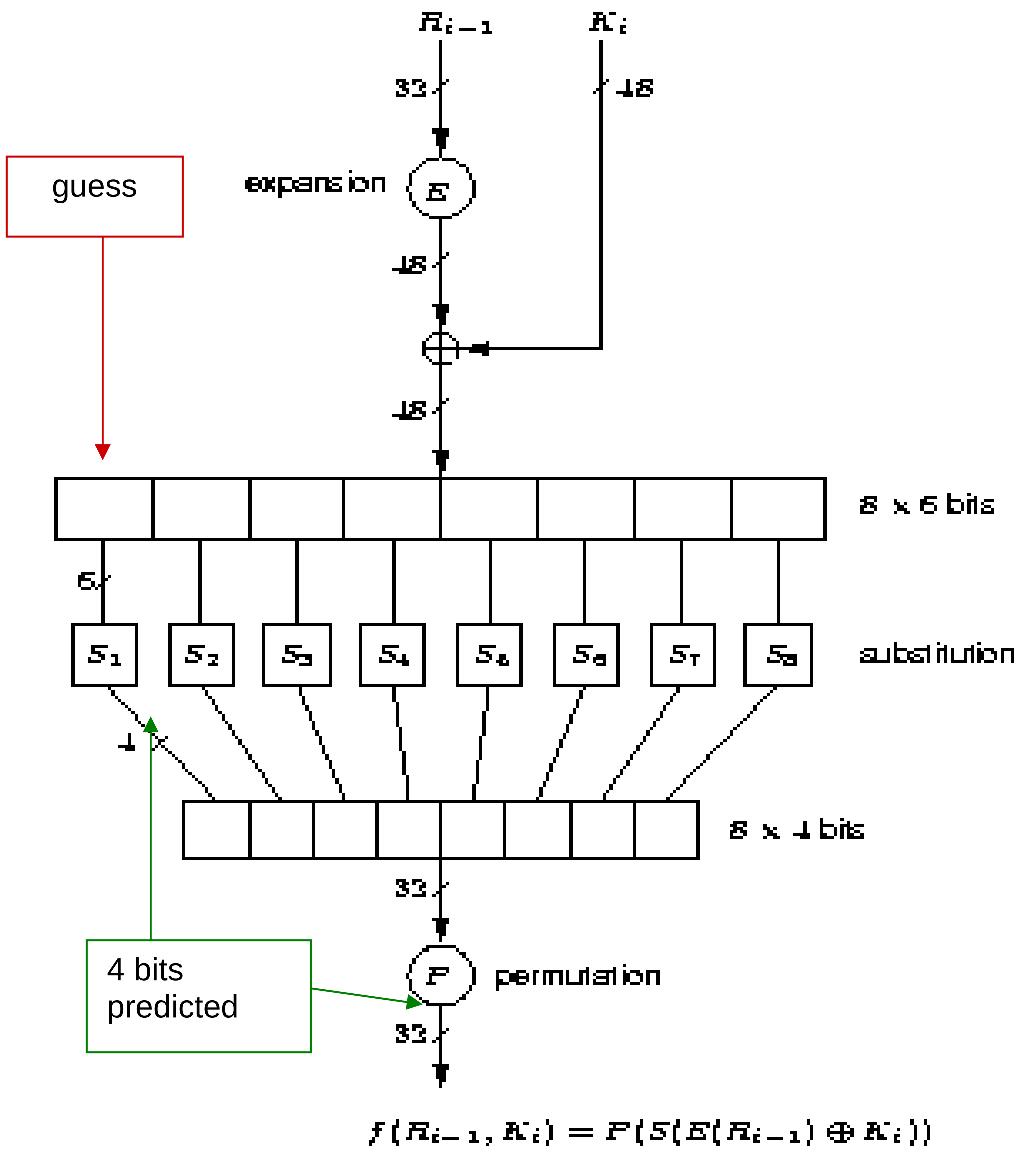


Figure 7.10: *DES inner function f.*

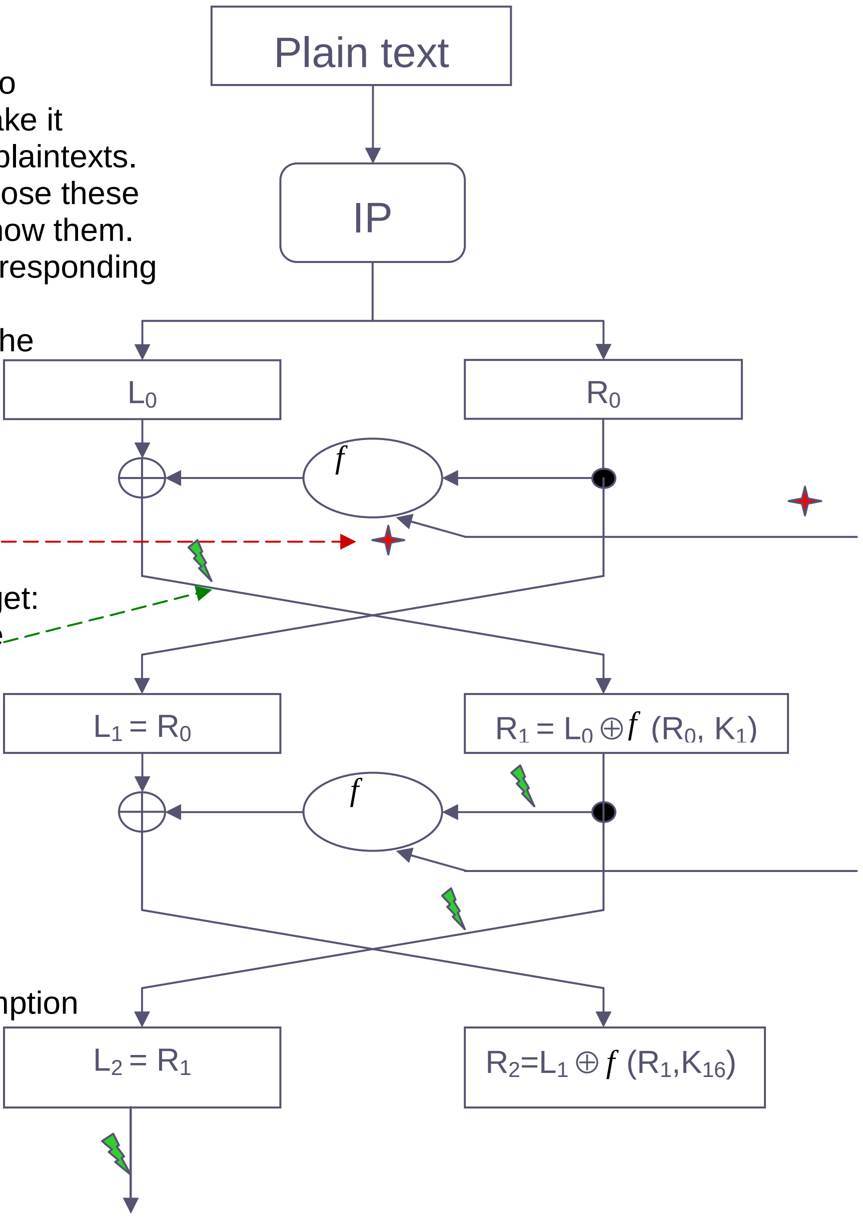
2.
Assume that we have access to
a smart card where we can make it
encrypt many different known plaintexts.
We don’t need to be able to chose these
plaintetexts, we just need to know them.
We don’t need to know the corresponding
ciphertexts at all.
For example the plaintext are the
known data of a transaction
that will be initiated
by the attacker.

3.
If our 6 bits are fixed, then,
for each experimentation, we get:
-one power consumption curve
-a value of bit 9 here.

4.
We will classify our curves
in two “baskets”:
in one basket all those where
the (predicted) bit 9 is 0,
and in the other basket,
where this bit is 1.
We average the power consumption
curves in each basket.

5.
Now we expect that if
our guess was correct,
the average consumption
curve in each basket are different:
For one guess we will be able
(at certain moments in time)
to eliminate the noise and see the
power consumption that characterises
the fact that bit 9 is 0
(respectively 1 in the other basket).

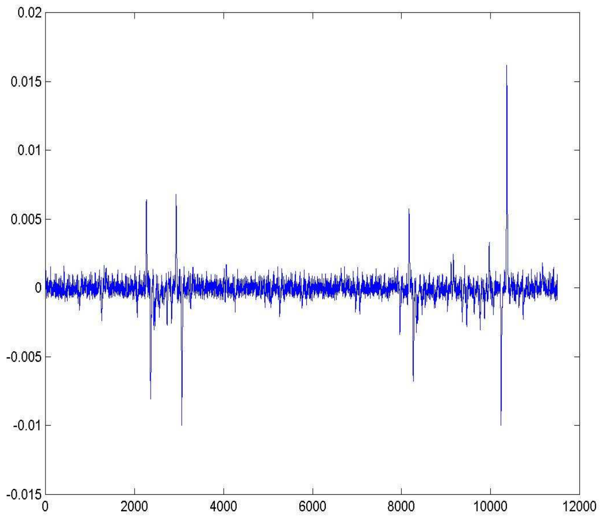
6. To see this we will compute the
difference between the two averages.
It will have characteristic “peaks”
at certain moments in time.



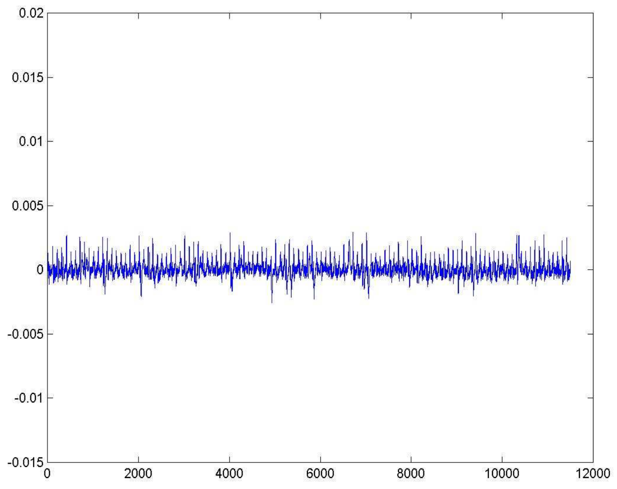
Plain text
IP
L_0
R_0
f
$L_1 = R_0$
$R_1 = L_0 \oplus f\ (R_0, K_1)$
f
$L_2 = R_1$
$R_2 = L_1 \oplus f\ (R_1, K_{16})$

- ✦ Bits guessed
- ⚡ We can predict something

Here is what we get if the assumption on 6 bits is right:



And here when the assumption on 6 bits is wrong:



In this way we can recover 6 bits of the key.

7.
In a similar way, by focusing on S-box S2 and one if its output bits, we can recover another 6bits of the key,

8.
By running the attack several times for different S-boxes we can recover full 56 bits DES key, or a full 112-bit triple DES key.

Other Side Channel Attacks:

Timing Attacks

Deduce the key from a timing behaviour. The exact method depends a lot of the type of crypto algorithm and also a lot on the implementation technique used.

Electromagnetic Emanations Analysis (EMEA)

Same possibilities as with SPA and DPA but we use other type of channel.

*Binary Power Analysis (BPA – Mehdi-Laurent-Akkar)

- Special variants of DPA, but the operation attacked is a XOR on 1 bit or on 1 byte.
- Breaking an algorithm such as AES which uses M XOR K operation at the beginning of encryption with M known to the attacker and K being the key.
    - Example: One basket = average of all traces over many different executions when the predicted bit after XOR is at 1. The attacker needs to distinguish between the output bit being at 1 and the output bit being at 0. Here there is no wrong key guesses, the attacker needs to be able to see which distribution is at 0 because it has lower consumption at some key places on the averaged curve.
- Very technical and will not work on every smart card.
- A more refined version could work on bytes or few bytes at the same time. Then we would have wrong key guesses. It can for example work if there is a correlation between the HW of a certain byte (number of bits at 1 in this byte) and the power consumption of the smart card if this byte occurs during the computation. Instead of HW the author have also proposed an affine consumption model, in which each of 8 bits has a different weight (in HW model each bits has the same weight, we simply count the number of 1s).

Comparative Power Analysis (CPA)

Self-similarity attack. Applied by Shamir et al. to RSA. Presented at UCL in 2012. Works more or less only if the implementation is deterministic: running the same large block with the same data is expected to produce the same trace.

1. Make an assumption on the data such that for a larger component ALL inputs are the same (for example a large modular exponentiation).
2. Detect the similarity with a very low error rate.
3. Recover the key by mathematical analysis.

Differential Fault Analysis (DFA)

1. Provoke one faulty ciphertext computation and obtain one correct ciphertext value to compare.
2. Recover the key by a mathematical attack.

The corresponding plaintext is useful to have but is not strictly necessary.
It requires the cryptographic algorithm AND its implementation to be deterministic.
The attacker needs to be able to make the smart card execute the same operation twice or more times, with the same data and in the same way (with the same intermediate values).
The attacker works with 2 ciphertexts. One execution would be correct. Another would be faulty (incorrect ciphertext obtained).

DFA is not always possible with all products. It is not untrue to say that Fault Attacks are feasible mostly on poorly engineered products. See slides on Moodle.

The exact method to obtain the key will vary a lot depending on the crypto used.

- If we are attacking a symmetric algorithm it will be brute force, differential cryptanalysis, software algebraic attack or other.
- With RSA: a dedicated method, see below.

Fault Attack on RSA (DFA on RSA)

- A common method to speed-up RSA which is very widely used in the industry is called RSA-CRT.

With RSA-CRT method the secret key is computed using the RSA mathematics with a GCD.
Let N=pq and let d be the secret exponent (part of the private key). When a smart card decrypts a message (and also when it digitally signs a message) it computes:

$$s = h^{d} \bmod N$$

Instead of computing it modulo N many smart cards compute it modulo p and modulo q. Then there is a method to reconstruct the result from these by using the so called CRT isomorphism which is very inexpensive to implement.
Overall, this is roughly about 4-8 times faster than the normal method because it is like doing the same operation twice but on large integers which are twice smaller, which makes a big difference.

Exercise:
How do we exploit this break this smart card using a fault attack?
How can the attacker produce factors of N?

Hint:
1) assume that one “big” computation is correct; one is faulty,
2) use a GCD. The result (e.g. the signature) will be right mod p but wrong mod q.