General Countermeasures

- Hiding -- reduce the SNR by either increasing the noise or reducing the signal
  - Noise Generators, Balanced Logic Styles, Asynchronous Logic, Low Power Design and Shielding
- Masking/Blinding -- remove the correlation between the input data and the side-channel emissions from intermediate nodes in the functional block
- Design Partitioning -- separate regions of the chip that operate on plaintext from regions that operate on ciphertext
- Physical Security and Anti-Tamper -- denial of proximity, access, and possession

- Hiding aims at reduce the SNR of the side-channel information by either reducing the signal component or increasing the noise component of side-channel emissions. Common hiding methods include noise generators, balanced logic styles, asynchronous logic, low power design and shielding.
- Masking or blinding seeks to remove the correlation between the input data and the side-channel emissions from intermediate nodes in the functional block.
- Design partitioning separates regions of the chip that operate on plaintext from regions that operate on ciphertext in order to avoid coupling of secret signal on to other nodes and information leakage that followed.
- Physical security and anti-tamper are committed to forbid the proximity and access to the cryptographic systems by attackers.

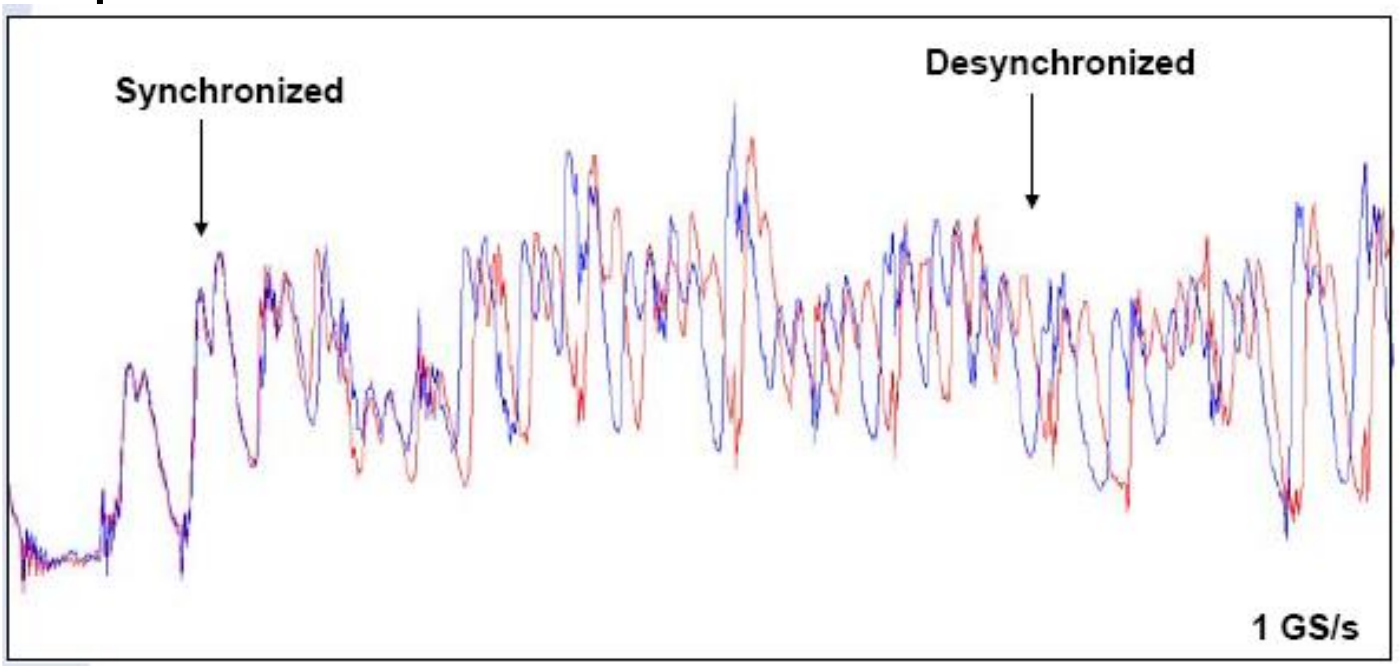
Anti-DPA countermeasures

- Applicative counter-measures : make message free randomization impossible !
  - Fix some message bytes
  - Constrain the variable bytes (ex : transaction counter)
- Decorrelate power curves from data
  - by hardware : current scramblers (additive noise)
  - by software : data whitening
- Desynchronise the N traces (curves misalignment)
  - software random delays
  - software random orders (ex : SBoxes in random order)
  - hardware wait states (dummy cycles randomly added by the CPU)
  - hardware unstable internal clock (phase shift)
- DPA is powerful, generic (to many algorithms) and robust (to model errors)...
- ... but there are counter-measures !

GEMPLUS

Anti-DPA

- Internal clock phase shift



Timing attacks

- Running time of a crypto processor can be used as an information channel
- The idea was proposed by Kocher, Crypto’96

- Running time of a crypto processor can be also used as an information channel.
- For example, imagine you put $28 in one pot and $10 in the other. If there are 10 notes in the blue pot and 7 notes in the red pot, you cannot determine what is in each pot according to whether the result is odd or even.

Timing attacks (cont'd)

- Well, normally not :

  $28 * 7 + 10 * 10 = 296$ is an even number

  and

  $10 * 7 + 28 * 10 = 350$ is also even…

- However, just by monitoring the time it takes to give the answer one can tell where each amount is!

RSA Cryptosystem

- Key generation:
  - Generate large (say, 2048-bit) primes p, q
  - Compute n=pq and $\varphi(n)=(p-1)(q-1)$
  - Choose small e, relatively prime to $\varphi(n)$
    - Typically, e=3 (may be vulnerable) or $e=2^{16}+1=65537$ (why?)
  - Compute unique d such that $ed = 1 \bmod \varphi(n)$
  - Public key = (e, n); private key = (d, n)
    - Security relies on the assumption that it is difficult to factor n into p and q
- Encryption of m: $c = m^e \bmod n$
- Decryption of c: $c^d \bmod n = (m^e)^d \bmod n = m$

How Does RSA Decryption Work?

- RSA decryption: compute $y^x \bmod n$
  - This is a modular exponentiation operation
- Naive algorithm: square and multiply

```
Let s_0 = 1.
For k = 0 upto w − 1:
  If (bit k of x) is 1 then
    Let R_k = (s_k · y) mod n.
  Else
    Let R_k = s_k.
  Let s_{k+1} = R_k^2 mod n.
EndFor.
Return (R_{w−1}).
```

Kocher's Observation

Let $s_0 = 1$.
For $k = 0$ upto $w - 1$:
  If (bit k of x) is 1 then
    Let $R_k = (s_k \cdot y) \bmod n$.
  Else
    Let $R_k = s_k$.
  Let $s_{k+1} = R_k^2 \bmod n$.
EndFor.
Return (R_{w-1}).

Whether iteration takes a long time depends on the k^{th} bit of secret exponent

This takes a while to compute

This is instantaneous

Outline of Kocher's Attack

- Idea: guess some bits of the exponent and predict how long decryption will take
- If guess is correct, we will observe correlation; if incorrect, then prediction will look random
  - This is a signal detection problem, where signal is timing variation due to guessed exponent bits
  - The more bits you already know, the stronger the signal, thus easier to detect (error-correction property)
- Start by guessing a few top bits, look at correlations for each guess, pick the most promising candidate and continue