Exam

Hardware Security

January 27, 2014, 12:30 – 14:00

You can score a maximum of **100** points. Each question indicates how many points it is worth. You are NOT allowed to use books/slides/notes etc., and also NOT to use a calculator, (smart) phone or any other device. You may answer in Dutch or in English. Please write clearly and don't forget to put your name and student number on each page.

1. (**10 points**) Briefly explain what Java Card transactions can or should be used for. As a part of your answer, explain:
   - against which *attack vector* transactions can protect
   - whether this concerns *transient* and/or *persistent* memory
   - what *type of security requirement* transactions can protect.
2. (**5 points**) Give two reasons for using transient rather than persistent memory for data.
3. (**5 points**) The order of branches in code can make a difference when it comes to the susceptibility to certain physical attacks. (e.g. writing `if (b) then s1 else s2` versus `if (!b) then s2 else s1`)
   Explain why, and for what kind of physical attacks there can be a difference (e.g. by giving an example).
4. (**10 points**) List three countermeasures against power analysis attacks. Briefly explain why they work against the attacker.
5. (**20 points**) Given below is a password verification algorithm which compares the entered password P' with the correct password P.

Algorithm 1 Password verification algorithm.

Require: $P' = (P'[0], \ldots, P'[7])$, $P = (P[0], \ldots, P[7])$
Ensure: 'TRUE' *or* 'FALSE'
1: **for** i from 0 to 7 **do**
2: if($P'[j] \neq P[j]$) then return 'FALSE'
3: **end for**
4: **return** 'TRUE'

Suppose that in the password verification routine of the above algorithm, the comparison is done in a random order. More specifically, let S be the set of permutations on the set $\{0, 1, \ldots, 7\}$. At each execution a permutation s is randomly chosen in S and the comparison (Step 2 of the algorithm) is replaced with

2: if($P'[s(j)] \neq P[s(j)]$) then return 'FALSE'.

Answer the following questions:

(a) Is this implementation secure against timing attacks? If no, briefly explain your attack.

(b) Can you extend your attack if a random delay is added before returning the status?

6. (**20 points**) AES encryption can be randomized in the following way:
   Encrypt a 128-bit plaintext m under the key K as $C = (c_1, c_2)$ with $c_1 = AES_K(r)$ and $c_2 = m \oplus r$ for a 128-bit random r. Plaintext m can then be recovered from ciphertext $C = (c_1, c_2)$ using key K, as $m = c_2 \oplus AES_K^{-1}(c_1)$.

   (a) Is this implementation secure against DPA attacks?

   (b) Can you mount a DPA attack against this randomized version of AES? If yes, briefly explain how.

7. (**30 points**) PRESENT is a lightweight block cipher which encrypts 64-bit plaintexts and uses an 80-bit key to do so. The algorithm follows the SPN (Substitution Permutation Network) structure (see Figure) where the round function consists of 16 4×4 S-boxes (i.e. 4-bit input, 4-bit output) and a bit permutation for the whole state (64-bits). PRESENT requires 31 iterations of the round function described above to produce the ciphertext. Given the information about the block cipher PRESENT, step-by-step explain how to mount a DPA attack to recover the first 4 bits of the 1st round key

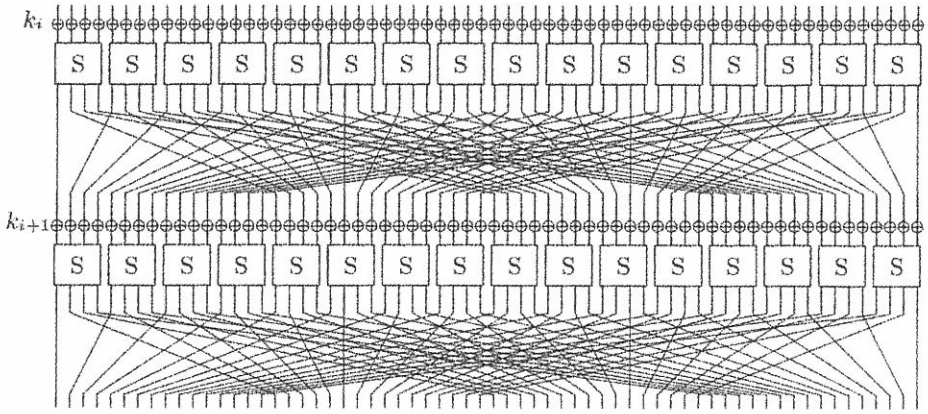

Figure 1: Round function of PRESENT (figure from the original paper)

(a) with difference of means as the distinguisher for the MSB (most significant bit) of the target intermediate value.

(b) with (Pearson) correlation as the distinguisher and Hamming distance as the power model.

Outline the attack, providing information on the number of candidates, variables to be considered, functions to be used, etc...