# Timing attack against a DES implementation

Alessandro Menduni

May 4, 2015

## 1 Introduction

A timing attack is a side channel attack, which consists in exploiting information given by the time taken by the hardware to execute a cryptographic alghoritm. In particular, it is taken into account the dependancy between the input provided to the system and the time required to complete the operation, thus, by the use of precise measurements, an attacker can procede backwards from the output to the input. In this report it will be considered the attack against a poor implementation of the P permutation of the DES library.

## 2 The attack target: the P permutation

As stated in the introduction, the main target of the attack will be the implementation of the P permutation that is performed on the output of the SBoxes; this stage of the algorithm is implemented in such a way that it makes it feasible to exploit time measurements to deduce the input key of the 16th round of the DES; in fact, as we can see in the snippet below, the time needed by the hardware to perform the permutation depends on the number of bits set to 1, since it procedes bit by bit and, everytime it encounters a bit to 0 in the input string, it skips all the loops and it copies directly such bit to the result string.

```
// Permutation table. Input bit #16 is output bit #1 and
// input bit #25 is output bit #32.
p_table = {16, 7, 20, 21,
          29, 12, 28, 17,
           1, 15, 23, 26,
           5, 18, 31, 10,
           2,  8, 24, 14,
          32, 27, 3,  9,
          19, 13, 30, 6,
          22, 11, 4, 25};

p_permutation(val) {
  res = 0;                  // Initialize the result to all zeros
  for i in 1 to 32 {        // For all input bits #i (32 of them)
    if get_bit(i, val) == 1 // If input bit #i is set
      for j in 1 to 32      // For all 32 output bits #j (32 of them)
   if p_table[j] == i  // If output bits #j is input bit #i
         k = j;             // Remember output bit index
       endif
      endfor                // output bit #k is now input bit #i
      set_bit(k, res);      // Set bit #k of result
    endif
  endfor
  return res;               // Return result
}
```

## 3 Attack's working principles

The reasoning behind the attack is the following:

- Since the attacker owns the cipertexts, she can extract L16 and combine it with her guess for the 16th round key;
- The result of such a combination can then be passed to the SBoxes, and their output would present a given quantity of 1s, which is hopefully reflected in the timing measurments for the ciphertexts - meaning, the higher the count of 1s, the higher the corresponding timing;
- In order to make the relationship between the timing measurments and the number of 1s in the output of the SBoxes apparent, the attack tries to find the key for which the correlation between all the timing values and all the hamming weights is the highest possible. In other words, it is computed some kind of ranking for every key, in a way that associates an higher rank to the key for which the link between the timing values and the quantity of 1s is stronger.

A computationally feasible procedure would carry on the attack on one SBox at a time, as follows:

- It is generated every possible combination of the first 6 bits of the key
- For every key generated as such, it is computed the correlation between all the measurements and the corresponding hamming weights
- It is kept as good the key whom correlation is the highest
- Repeats for the next 6 bits of the key until all the 48 bits of 16th round key have been "guessed"

At this point the remaining bits of the 64bits key can be brute-forced by inverting the key schedule algorithm and trying all the keys over a couple plaintext-ciphertext, which is needed in order for the attack to automatically recognise whether it's been successful or not.

## 4 The ranking system

Two systems have been tried out for such a purpose. The first one consists in computing the difference between the average of the timing values for the chipertexts whom hamming weight is 0 and the average of the timings for the ones whom hamming weight is 1; such a difference should be high for the correct key, since it would mean that the measurements vary depending on the number of 1s present in the output of the SBoxes according to what it is stated by the timing model. Such a method has given out medium results by yielding the correct key with more than 12000 of pairs ciphertext - timing value. The second method relies on the Pearson product-moment correlation coefficient to measure the desired correlation; this strategy takes into account all the chipertexts in the data set (not only the ones whom hamming weight is either 1 or 0) and it is therefore much more stable. Such PCC coefficient is computed by considering as X variable the timing measurements, whereas the Y variable contains the hamming weights of the outputs. Such a method yields the correct key with 1704 pairs chipertext - timing value. The second approach has been thus introduced in the final attack.

## 5 Results and environment

The machine used to run the experiments is a laptop ASUS K550JK mounting an i7-4710HQ with 4GB of RAM DDR3L-1600. The data file used to test the solution is the one provided and the recovred key is 52e94a5e04dafba2 with a stable minimum of 1704 experiments needed.