Blind Attacks on Engineering Samples

Vanessa Gratzer[1] and David Naccache[1+2]

[1] Université Paris II, Panthéon-Assas
Hall Goullencourt, casier 55
12 place du Panthéon, F-75231 Paris, CEDEX 05, France
vanessa.gratzer@gmail.com

[2] École Normale Supérieure
Département d'Informatique, Équipe de Cryptographie,
45 rue d'Ulm, F-75230 Paris, CEDEX 05, France
david.naccache@ens.fr

Abstract. In addition to its usual complexity assumptions, cryptography silently assumes that information can be physically protected in a single location. As we now know, real-life devices are not ideal and confidential information leaks through different physical channels.

Whilst most aspects of side channel leakage (cryptophthora) are now well understood, no attacks on totally unknown algorithms are known to date. This paper describes such an attack.

By *totally unknown* we mean that no information on the algorithm's mathematical description (including the plaintext size), the microprocessor or the chip's power consumption model is available to the attacker.

We successfully experimented the attack on a commercially available device produced by a non-European smart-card manufacturer.

1 Introduction

In addition to its usual complexity postulates, cryptography silently assumes that secrets can be physically protected in tamper-proof locations.

All cryptographic operations are physical processes where data elements must be represented by physical quantities in physical structures. These physical quantities must be stored, sensed and combined by the elementary devices (*gates*) of any technology out of which we build tamper-resistant machinery. At any given point in the evolution of a technology, the smallest logic devices must have a definite *physical extent*, require a *minimum time* to perform their function and dissipate a minimal *switching energy* when transiting from one state to another.

The rapid development of sophisticated digital communication systems have created new academic interest in physical secret information leakage (cryptophthora) [1]. According to our estimates more than two hundred papers and twenty Ph.D. mémoires were published on the topic so far.

Whilst most aspects of side channel leakage are now well understood, no attacks on totally unknown algorithms are known to date.

By *totally unknown* we mean that no information on the algorithm's mathematical description (including the plaintext size), the microprocessor or the chip's power consumption model is available to the attacker.

This paper describes such a *blind side channel attack* that we experimented successfully on a commercially available device produced by a non-European smart-card manufacturer.

The precise assumptions we make are that the attacker is given a device $\mathbf{H}_{\mathfrak{K}}(\bullet)$ keyed with an unknown key $\mathfrak{K}$ and a physically similar blank device $\mathbf{H}_{\bullet}(\bullet)$ that he can re-key at wish. The attacker only knows the plaintext size and the key size (in other words the target is not even assumed to return the decrypted plaintext to the attacker).

Given a key k and a ciphertext c, $\mathbf{H}_k(c)$ will denote the power consumption curve obtained when decrypting c under k. Although variants of our attack with variable ciphertexts exist, throughout this paper we will work with a constant c (*e.g.* the all-zero ciphertext). We will hence abridge notations by writing $d_k = \mathbf{H}_k(c)$.

Knowledge of the microprocessor's word size w (8-bit, 16-bits or 32-bits) is not mandatory but may accelerate the attack. Usual countermeasures might slow-down the attack or thwart it.

2 The Intuition

The intuition behind our idea is the following: we start by collecting the power traces of the target and average these to get a cleaner representation of $d_{\mathfrak{K}}$.

Then, assuming that the target is an 8-bit machine, we know that the device's power consumption when decrypting c under an arbitrary candidate key k should in principle coincide with $d_{\mathfrak{K}}$ on all samples until the moment where k and $\mathfrak{K}$ start being manipulated by the devices.

Consequently, the sample-wise subtraction $\Delta_k = d_{\mathfrak{K}} - d_k$ produces a differential curve which beginning is flat up to a point where Δ_k suddenly becomes noisy.

Now, being an 8-bit machine, the microprocessor cannot manipulate more than one byte at a time. Hence as soon as we guess correctly $\mathfrak{K}$'s first byte, we expect to see the flat part of Δ_k extended. We can therefore progressively guess key bytes ahead, until we get a completely flat Δ_k. At that point $\mathfrak{K} = k$.

Note that since that the microprocessor does not necessarily begin to work with the key's first byte[3], the experiment needs to be restarted for each byte position, until the key is entirely discovered.

Hence, if $\mathfrak{K}$ is n-word long, recovering $\mathfrak{K}$ will require $2^{w-1}(n^2+n) < 2^w|\mathfrak{K}|^2/w^2$ differential experiments.

[3] *e.g.* the algorithm might start mixing the ciphertext's fifth byte with the key's third byte.

Substituting a few practical values ($w = 8, 16, 32$ and $\mathfrak{K} = 64, 96, 128, 160, 256$) into this formula we get:

	$\lvert\mathfrak{K}\rvert = 64$	$\lvert\mathfrak{K}\rvert = 96$	$\lvert\mathfrak{K}\rvert = 128$	$\lvert\mathfrak{K}\rvert = 160$	$\lvert\mathfrak{K}\rvert = 256$
$w = 8$	13	14	15	16	17
$w = 16$	19	20	21	22	23
$w = 32$	34	35	35	36	37

Table 1. Attack Complexity. Entries Express 2^x Experiments.

However, as we will soon see, the attack requires a very limited number of interactions with the target as the big bulk of experimentations is done *with the engineering sample*. Hence, the above workfactors can be divided by any factor representing the number of blank engineering samples on which the attacker will run the attack in parallel.

3 Notations and Statistical Tools

Statistics provide procedures for evaluating likelihood, called *significance tests*. In essence, given two collections of samples, a significance test evaluates the probability that both samples could rise by chance from the same parent group. If the test's answer turns out to be that the observed results could arise by chance from the same parent source with very low probability we are justified in concluding that, as this is very unlikely, the two parent groups are most certainly different. Thus, we judge the parent groups on the basis of our samples, and indicate the degree of reliability that our results can be applied as generalizations. If, on the other hand, our calculations indicate that the observed results could be frequently expected to arise from the same parent group, we could have easily encountered one of those occasions, so our conclusion would be that a significant difference between the two samples was not proven (despite the observed difference between the samples). Further testing might, of course, reveal a genuine difference, so it would be wrong to claim that our test *proved* that a significant difference did not exist; rather, we may say that a significant difference was *not demonstrated on the strength of the evidence presented*, which of course, leaves the door open to reconsider the situation if further evidence comes to hand at a later date.

We run the target and the engineering sample ℓ times and collect ℓ physical signals for each device. We denote by i the acquisition's serial number. The device's emanation can be an array $\{d_k[i,0], d_k[i,1], \ldots, d_k[i,\tau-1]\}$ (*e.g.* power consumption) or a table representing the simultaneous evolution of ℓ quantities (*e.g.* electromagnetic emanation samples or microprobes) during τ clock cycles. For the sake of simplicity, we will not treat the table case in this paper and focus on power curves[4]. The blind attack will hence operate on $d_k[i,t]$ and use (existing) significance tests as a basic building block:

[4] Generalizing the attack to spatial signals such as electromagnetic radiations is straightforward.

Definition 1. *When called with two sufficiently large samples X and Y, a significance test $S(X, Y)$ returns a probability α that an observed difference in some feature of X and Y could rise by chance assuming that X and Y were drawn from the same parent group. The minimal sample size required to run the test is denoted size(S).*

While we arbitrarily restrict our choice in this work to the three most popular hypothesis tests: the *distance of means*, *goodness of fit* and *sum of ranks*, a variety of other hypothesis tests can be used for implementing our attack. The reader may find the description of these procedures in most undergraduate textbooks (*e.g.* [2, 3]) or replace them by any custom procedure compatible with definition 1.

test S	**notation**	**description**
distance of means	`DoM-test`	[3], (pp. 240–242) 7.9
goodness of fit	`GoF-test`	[3], (pp. 294–295) 9.6
sum of ranks	`SoR-test`	[3], (pp. 306–308) 10.3

Table 2. Hypothesis Tests.

4 The Attack

We start by selecting a significance test S (*e.g.* amongst `DoM-test`, `GoF-test` and `SoR-test`) and run the target $\ell = \text{size}(S)$ times, collecting at each run an emanation curve $d_{\mathfrak{K}}[i, 0 \leq t \leq \tau - 1]$. We will not need the target anymore during the attack.

As the attack starts we reset a global clock cycle counter $c \leftarrow 0$. c will track the attack's progression on the power curve and represent the rightmost curve coincidence point between the target and the engineering sample.

Denoting by $n = |\mathfrak{K}|/w$ the key-size measured in machine words, we assume that we are given a procedure denoted $\mathsf{Exhaust}(k, i, c)$. Here k stands for an intermediate key candidate and $0 \leq i < n$ is a new word position to exhaust in k. $\mathsf{Exhaust}$ returns a new key value k_i and an associated latest coincidence clock cycle $c_i \geq c$.

$$(k_i, c_i) \leftarrow \mathsf{Exhaust}(k, i, c)$$

The attack progressively fills an n-bit string $s = \{s_0, \ldots, s_{n-1}\}$, where positions set to one stand for key words that has been already exhausted in k. Before the attack is launched we initialize $s \leftarrow \{0, \ldots, 0\}$ and $k \leftarrow \{0_{2^w}, \ldots, 0_{2^w}\}$.

```
while  s ≠ {1, ..., 1}
  {
    for i ← 0 to n − 1 {if s_i = 0  then  (k_i, c_i) ← Exhaust(k, i, c)}
      j  ← index for which c_j = max_{s_i=0}(c_i)
      s_j ← 1
      k  ← k_j
      c  ← c_j
  }
```

In practical implementations, one can keep at each for iteration two or three best scoring j values and backtrack if the algorithm hits a dead-end (due to measurement inaccuracies or noise).

At the end of the process k is output as a hypothesis for $\mathfrak{K}$'s value (or confirmed if the target returns a plaintext).

4.1 The Exhaust Routine

Exhaust gets as input a partially filled key k, a word index i and a latest coincidence point c. The routine exhausts the 2^w possible values at position i, selects the value which optimizes the coincidence with the target and returns this optimal coincidence point c_i to the caller.

We will denote by k_e the key k where the i-th word was set to e and by $\{d_{k_e}[u,0],\ldots,d_{k_e}[u,\tau-1]\}$ the τ-sample acquisition collected during the u-th experiment, where $\mathbf{H}$ was keyed with k_e (we perform ℓ such experiments per k_e value). We compute for $0 \leq e \leq 2^w-1$ and $0 \leq t \leq \tau-1$:

$$\alpha_e[t] \leftarrow S(\{d_{k_e}[1,t], d_{k_e}[2,t],\ldots,d_{k_e}[\ell,t]\},\{d_{\mathfrak{K}}[1,t], d_{\mathfrak{K}}[2,t],\ldots,d_{\mathfrak{K}}[\ell,t]\})$$

and match the $\alpha_e[t]$ to entire clock cycles by computing:

$$\gamma_e[n] = \sum_{t\in\text{cycle } n} \alpha_e[t]$$

Assuming that at each clock cycle $n \geq c$ the random variable $\gamma_X[n]$ follows a normal distribution (with mean μ_n and standard deviation σ_n that we can easily estimate from the 2^w measurements available), we compute for each e the probability λ_e (P-value) that $\gamma_e[n]$ would fall beyond the number of σ_n-s that separate $\gamma_e[n]$ from μ_n. The lowest λ_e determines our choice of e but the exploration of the curve will continue until no deviation bigger than $2\sigma_n$ (*i.e.* a P-value of 0.95%) is found, in which case we consider that we have hit the next de-synchronization point and report it by returning to the caller.

To distinguish complete coincidence[5] from complete de-synchronization both characterized by deviations smaller than $2\sigma_n$, we compare the value of σ_n to a threshold σ' set experimentally. Cycles for which no deviation bigger than $2\sigma_n$ was found and $\sigma_n < \sigma'$ are considered are completely coinciding whereas cycles for which no deviation bigger than $2\sigma_n$ was found and $\sigma_n \geq \sigma'$ are considered an indication of complete de-synchronization.

5 Practical Implications and Further Research

The experiments reported in this paper underline the risk of distributing engineering samples of improperly protected tamper-resistant devices. We showed

[5] *e.g.* operations that do not manipulate the key nor the ciphertext

that this is important even in contexts where the algorithm is unknown to the attacker. We think while the new scenario does not represent a significant risk in telecommunications and financial contexts it should certainly be taken into account in pay-TV applications.

From a technical standpoint, it would be interesting to devise variants of the attack that use other decision criteria (*e.g.* exploring the correlation coefficient curve between the target's power consumption and the engineering sample) or devise specific countermeasures against blind attacks.

As a final note, we observed in several of our experiments that the key was disclosed before the algorithm even started to work. The attack had actually detected key's transfer from non-volatile memory to the target's RAM.

References

1. P. Kocher, J. Jaffe, B. Jun, *Differential power analysis*, Advances in Cryptology CRYPTO'99, Springer-Verlag, LNCS 1666, pp. 388–397, 1999.
2. R. Langley, *Practical statistics*, Dover publications, New-York, 1968.
3. I. Miller, J. Freund, R. Johnson, *Probability and statistics for enginners*, Prentice Hill, 1990.