Future Cryptography: Standards Are Not Enough

Tomáš Rosa
tomas.rosa@decros.cz

Decros, Ltd., member of ICZ group
V Olšinách 75
100 97 Praha 10, Czech Republic

Dept. of Computer Science and Engineering
Czech Technical University, Faculty of Electrical Engineering
Karlovo námestí 13
121 35 Praha 2, Czech Republic

Abstract

The development and implementation of various standards represent the main stream in the area of contemporary cryptography. Standards such as AES, SHA-1, DSA, ECDSA, RSA or standards such as PKCS, etc stand for a good example. These standards are kept up-to-date and public.

Does it mean, that anyone with a basic knowledge of the computer architecture and discrete mathematics can simply build up a secure cryptographic device following these standards? Also, does it tell us that all cryptographic devices using the same cryptographic standard have the same level of security? Unfortunately it doesn't.

Here we focus on the influence of implementation-dependent properties of cryptographic devices on their security. We will use the general notion of output and input side-channels to discuss the main principles behind various attacks based on physical properties of attacked device. Then we will show and explain general defending techniques, which should be automatically used in all cryptographic devices, which are designed and built nowadays.

Keywords: CSP, fault analysis, oracle based analysis, power analysis, side channel, time analysis, timing attack.

1. Introduction

It is becoming well known fact that the proper design of a particular cryptosystem in the ”paper form” is one thing and the proper implementation of this system into given physical device is the second one. The main security risk here is to evaluate only the mathematical properties of the designed cryptosystem, while underestimating the physical properties the system will have after its implementation in the ”real world”.

Almost all physical properties (including the electromagnetic emanation) of the cryptographic device that could be carefully measured or precisely altered can be used for some kind of attack. These attacks however are not visible in the pure mathematical description of the given cryptosystem. This is why the contemporary cryptography tends to be highly interdisciplinary science requiring balanced skills in mathematics, physics and electrical engineering.

In the following article we will show the general concept of attacks based on the physical properties of attacked devices. We will try to keep the focus on the general properties of these attacks as much as it will be possible. However sometimes we will need to introduce some practical example. In such case we will use the asymmetrical cryptosystem RSA, because it seems to be the most illustrative cryptographic mechanism for these purposes. Many of the attacks discussed later were successfully tested on it.

2. The Concept of Side-channels

With the hope that one day we would use such sophisticated apparatus as the information theory to analyse the information exchanged between the particular cryptographic device and its neighbourhood we can use the notion of side-channel. This term appeared for the first time in the article [7]. Here we will present more general definition of this term.

Definition 1. *The unplanned way which allows a physical cryptographic device to exchange some sensitive information with its physical neighbourhood we will refer to as the side-channel.*

Note that we can observe side-channels transferring information from the device (output type) as well as the other ones transferring the information to the device (input type). Both types of side-channels are dangerous – the first type allows the attacker to obtain some information about the inner variables, while the second one can be used to force the device to change its behaviour in the way to create some kind of backdoor.

There are plenty of ways which can be the particular side-channel realized in. Today attackers use mainly the timing characteristics or some kind of direct electrical coupling with the attacked device. The first principle is suitable only for output side-channels, while the second one can be used for both types. Well known principle is also the electromagnetic emanation (see [1], [12] and [13]), but the side-channels created in this way are (up to now) exploited in a bit different ways, so we won't discuss them here.

3. Listening to the Side-channel

Today the most dangerous attacks based on the output side-channels are those based on measuring the amount of time required to perform given cryptographic transformation or on sampling the power consumption during the computation with a secret data. The first kind of attacks is often referred to as the Timing Attack (see [5], [3]). This label is mainly due to the historical reasons. We will prefer the most accurate notion of *Time Analysis* (TA) attacks here.

The second kind of attacks is referred to as the *Power Analysis* (PA) attack (see [10]), while the most important subclass seems to be the attack based on the *Simple Power Analysis* (SPA) and *Differential Power Analysis* (DPA).

Here we will focus on the common properties of DPA attacks and later versions of TA attacks. In particular there is only one, but very important common characteristics of these attacks. It is the *Oracle-based Analysis*, which they intensively use.

Note that with the respect to side-channels we mean by the term "analysis" the process of transferring information through the particular type of channel. The type of particular analysis (time, power, etc.) closely specifies the type of used side-channel. By the term "attack" we mean the usage of particular analysis against a given cryptographic device.

3.1 Oracle-based Analysis

Identifying usable side-channel is just the first step on the way toward the extraction of secret data held inside the device. The essential part on this way is to build an appropriate mathematical tool, which allows the attacker to extract the information leaking from the opened side-channel. There are very few cases in which the information is encoded in some directly usable format. In all the other cases it is scrambled in some device-proprietary way. Despite this, it doesn't seem to be such a problem to decode these streams, mainly because of the availability of strong statistical techniques of signals' theory.

Here we will present statistical technique, which we will refer to as *Oracle-based Analysis* (OBA). Discussing this technique seems to be important, because (despite usually not stated explicitly in the description of particular attack) it stays behind almost all major types of Power or Time Analysis. So we can derive the common characteristics of these attacks, while studying only the properties of OBA. This situation we will fully exploit when we state the *OBA-Fundamental Hypothesis*.

In the following text we will describe one pass of OBA. In the real scenario there is a number of such passes, each of them revealing part of the secret data. Different oracle as well as different condition and sometimes also different characteristic of random variables (see proposition 3) and metric are used for each particular pass of OBA.

Proposition 1. *Let us have a device with the input set I. Let S be the particular side-channel, giving for each input message the n-dimensional real information as S: I ® R^n.*

Definition 2. *The oracle will be represented by the transformation O: I ® B, where B = {0, 1}.*

The first step in the Oracle-based Analysis is to obtain a subset I_m of randomly chosen inputs (the attacker doesn't need to have a chance to alter the choice, but he needs to know their values) for which we have measured the information leaked from the particular side-channel.

Proposition 2. *Let I_m be a subset I_m Í I such that for each x Î I_m we know the appropriate value of S(x).*

Next step is to design the particular oracle. We do it according to the following proposition.

Proposition 3. *The value of oracle O splits the set I_m into the two disjunctive subsets I_1, I_2, such that for each x Î I_m we have: x Î I_1 iff O(x) = 1 and x Î I_2 iff O(x) = 0. Next we define the transformations S_1, S_2, such that S_1: I_1 ® R^n, S_2: I_2 ® R^n, $S_1(x) = S(x)$, $S_2(x) = S(x)$. By the notation S_1 or S_2 we mean the random variables taking randomly the values from the domain R^n.*

According to the selected oracle we also choose the condition "cond" depending on the part of secret data, which we want to obtain from the attacked device in this pass. The relationship between cond and O is the following:

- *(cond = false) Þ d(f(S_1), f(S_2)) £ e*
- *(cond = true) Þ d(f(S_1), f(S_2)) >> e, for some e Î R, e ³ 0.*

Here f denotes the selected characteristic of n-dimensional random variable (f: R^n ® R^n), and d denotes appropriate metric on the field R^n (d: R^n ® R).

The whole ensemble described works as a correlation based extractor of the information encoded in the signal coming from the side-channel. The oracle O and condition cond is in the particular pass of OBA selected in the way, such that if this condition is true, than the splitting process is correlated with the values measured on the output of the side-channel. If this condition is not true, then there is no (or very weak – this is the purpose of the ε) correlation observed.

The condition itself is chosen to be dependent on a part of secret data in such a way, that we get some non-trivial information about this part when we know if the selected condition is true or false. In particular the condition can be represented directly by some bits of the secret key.

There are plenty of important steps in the practical realization of OBA, mainly the process of construction of particular oracle or the statistics used to prove or disprove the validity of selected condition. Last but not least is the metric used to evaluate the distance between the values $\phi(S_1)$ and $\phi(S_2)$. But all these problems are behind the scope of this document. Also it is not necessary to present them here. It suffices to note that it seems to be true that even the simplest statistics such as the mean values in the place of ϕ are good enough to exploit the most of the information leaked through the particular side-channel. Good example is the article [3], where efficient OBA for Time Analysis was presented.

3.2 OBA-Fundamental Hypothesis

Now we are going to state the important theorem, which allows us to think about the common countermeasures against all OBA-based attacks. In fact this theorem for the first time appeared in the article [4], where it was stated according to some particular type of Power Analysis. Using the notion of OBA we are able to extend this theorem to other types of attacks (especially to the great part of attacks based on Time Analysis) and so to make it even more important.

Theorem 1. *Let we have a device, which is prone to be broken by OBA-based attack. Then there exists an intermediate value, which appears during the computation of the cryptographic algorithm. Its value is dependent on the input data and on some small part of secret material (the key) in the way that knowing this part of the key we are able to decide for two randomly chosen inputs if they produce the same value of this variable or don't, with the probability significantly higher than 1/2.*

Proof (sketch). Denote the variable v_{temp} and suppose that it doesn't exist. From the feasibility of OBA-based attack we have that the attacker is able to build up the oracle, which is correlated with the signal from the side-channel if and only if the selected condition is true. This condition directly depends on some part of the key. Knowing this part of the key, we know the value of the condition and from here the behavior of the oracle. The oracle is able to give us quite precise answer on the question if two randomly selected messages belong to same subset I_j or don't. Clearly the output from the oracle seems to be good candidate for v_{temp}. But the oracle exists. From this contradiction we deduce that variable v_{temp} exists too.

4. Fault Analysis

Fault Analysis (FA) differs from the previously discussed applications of side-channels in that it uses these channels also in the reversed direction - from the attacker to the device. This direction is used to alter the behavior of the inner cryptographic process in the way that the output is corrupted with some special kind of error. Using these outputs from the particular device the attacker is able to get some information about secret parameters inside the device. It means that the information sent from the attacker to the device opens up the output side-channel in the direction from the device to the attacker.

Today FA attacks can be used on almost every kind of cryptographic mechanisms. We will focus here on asymmetrical techniques, more concretely on RSA. Unfortunately it would be still out of the scope of this document to describe all known FA attacks on RSA, so we will focus on the most dangerous ones. The criterion here is the number of required interactions in the direction from the attacker to the device. Attacks presented in the following text typically require only one interaction to be done in this direction.

We shall also note that although we expect the faulty behavior to be caused by the effort of some attacker, there can be also errors of the "natural" type. It is as important to avoid these faults as to defend the device against attackers. This is so because when the error, which causes the leakage of the secret information, happens, it doesn't matter what was its reason. We may imagine the worldwide spying agency, which passively monitors every important system (for example major certification authorities - CAs) waiting for its faulty output. When it happens the agency gains the secret information from this particular system (usually the private key of this CA) and continues spying the other stations. If designers of these systems were not aware of FA attacks, then this agency would sooner or later have private keys of major part of certification authorities in the world.

4.1 Important FA Attacks on RSA

First we will show very basic lemma, which stays behind the great part of FA attacks on RSA. Despite its simplicity, it is very important for us, because it describes the nature of all attacks presented here.

Lemma 1. *Let we have x, y, n Î Z, such that n = p*q, where p, q are both primes, x º y (mod p) and x ¹ y (mod q). Than it is easy to compute p as p = gcd((x-y), n).*

Proof. From the first congruence we have that $p \mid (x-y)$ while from the second we have that q doesn't divide (x-y). Obviously also $x \neq y$. So p = gcd((x-y), n).
Using this lemma we can easily derive various types of FA attacks, all based on the assumption that the attacker is able to obtain the pair of integers (x, y) with the properties stated in the premise of lemma 1. In fact we can have two basic types of such attacks, depending on the role of particular values from the pair (x, y). The first type we will refer to as *Signature-Signature* (*S-S*) while the second we will call *Known Message-Signature* (*KM-S*).

Before focusing on particular types of FA mentioned, we will briefly discuss the main steps in RSA signature generation based on the *Chinese Remainder Theorem* (CRT). The signing transformation will be described in the following proposition.

Proposition 4. *Let us have the RSA instance with the public key (n, e) and the private key (p, q, d_p, d_q, pInv), where n = p*q is the modulus, e is the public exponent, p, q are primes, e*d_p º 1 (mod (p-1)), e*d_q º 1 (mod (q-1)) and p*pInv º 1 (mod q).*

Signature of a message m, m Î Z_n, is then computed in the following steps:

1. $s_p = m^{dp} \bmod p$
2. $s_q = m^{dq} \bmod q$
3. $h = pInv*(s_q - s_p) \bmod q$
4. $s = s_p + p*h$

It can be easily verified that the signature s satisfies: s º s_p (mod p) and s º s_q (mod q). This property we will write with the respect to the particular RSA instance as s = CRT(s_p, s_q). It is important to note that by the value m we don't mean here the raw message signed by the user, but its preprocessed and properly formatted image (for example according to PKCS#1).

4.1.1 Signature-Signature Type of FA

The S-S type of attacks in fact directly exploits the lemma 1. It is assumed that the attacker is able to obtain the pair of two signatures (s_1, s_2) computed with the same private key, which fulfills the condition stated in lemma 1.

Practical example of S-S attack was for the first time presented in [2]. Here authors assume that the attacker knows the public modulus n and can obtain one good and one faulty signature for the same message m under the same private key. This can happen for example when these signatures have been computed according to the proposition 4. Here it could be relatively easy for the attacker to affect the signing process (running on the attacked device) in the way that the sub-signature s_p is computed correctly, while the sub-signature s_q is corrupted. Resulting signature is then s_{faulty} = CRT(s_p, s_q). Without affecting the device the attacker also computes the proper signature s_{good}, which has both its sub-signatures correct. Now using the lemma 1 he can easily compute one factor (here p, because s_q is invalid) of n as p = gcd((s_{good} - s_{faulty}), n).

4.1.2 Known Message-Signature Type of FA

The K-MS attack can be derived from the S-S when the attacker knows the whole public key (n, e) and the formatted message m. Then it can be shown that only one faulty signature suffices to obtain the factors of n.

To see this, assume that we have the faulty signature s_{faulty}, such that $s_{faulty} \equiv m^{dp} \pmod p$, but $s_{faulty} \neq m^{dq} \pmod q$. Then the value $y = s_{faulty}^{e} \bmod n$ has the property, such that $y \equiv m \pmod p$, but $y \neq m \pmod q$. Clearly the pair (y, m) can be used according to the lemma 1 to factorize the public modulus n.

The particular instance of the K-MS attack was for the first time presented in the article [8]. Author here again exploits the properties of signing process described in the proposition 4. In fact the process is assumed to be altered in the same way as in the case of S-S attack described in [2], except that only one (faulty) signature is generated. This extension makes the attack more dangerous, because only one faulty signature may broke the whole system.

4.2 Importance of Checking the Integrity of Private Keys

Up to now all presented instances of S-S or K-MS attacks have been concerned on affecting the main computing process. With the respect to the architecture of today's cryptographic devices it is important to note that there is also another way, how to build the particular instance of these attacks. In this way we exploit the properties of the *key containers* – the place where the particular private keys are stored. There are still plenty of cryptographic applications, which store these keys in some file in the filesystem. Usually they are protected by some symmetric encryption to avoid their disclosure. But there is not only the requirement for

the good encryption scheme – also the integrity of these records must be achieved. Otherwise the attacker can mount either S-S or K-MS attacks by corrupting selected values in the data representing the private key.

For example suppose that the private key is stored as a quintuple (p, q, d_p, d_q, pInv), which is properly encrypted, but which integrity is not checked. Then it is an easy exercise to show that corrupting some of these values (for instance q or pInv) leads to the possibility of mounting either S-S or K-MS attack. In the case of K-MS attacks the attacker simply alters (using only the ciphertext form!) selected values and then waits for the (faulty) signature. Only one such signature then suffices to recover the whole private key!

It seems that RSA instances implemented with the help of CRT are more vulnerable to the presented attacks than the other ones. The reason is that it is often easier for the cryptosystems implemented according to the proposition 4 to find the way, how to affect the signing process. But it doesn't mean that these attacks cannot be mounted on cryptosystems computing the signing transformation directly as $s = m^d \bmod n$.

Suppose that the attacker forces the device to use the value d' instead of d, such that $d'*e \equiv 1 \pmod{(p-1)}$, but $d'*e \not\equiv 1 \pmod{(q-1)}$, where $n = p*q$ is the public modulus of RSA, p, q are primes of the appropriate length, e is the public exponent and d is the private exponent. It could be done for example when the device uses the public key to derive some parts of the private rather then store the whole private key (the main reason to do this can be saving the space in the key container). Now, obtaining one signature computed as $s_{faulty} = m^{d'} \bmod n$ for known m allows the attacker to use the KM-S attack to factorize n. To see this note that $(s_{faulty})^e \equiv m^{d'*e \bmod (p-1)} \equiv m \pmod p$, but $(s_{faulty})^e \equiv m^{d'*e \bmod (q-1)} \pmod q$, where $d'*e \bmod (q-1) \neq 1$. Therefore with high probability we have $(s_{faulty})^e \neq m \pmod q$.

5. Countermeasures

All attacks discussed in this document use the physical properties of attacked device. One could think, that countermeasures against these attacks would be based merely on hardware improvements. However the practical situation shows that at least one half of these countermeasures can (or even must) be done in the phase of mathematical description of the cryptosystem. Roughly speaking we can say that as much the attack requires the sophisticated mathematical techniques, as great part of the whole countermeasures must be done in the mathematical description of the system.

Pure hardware improvements (some of them appear in [6]) work well against such attacks as Simple Power Analysis or some types of Fault Analysis. On the other hand these improvements (with the respect to our technological skills) are almost useless against well formed Differential Power Analysis or Fault Analysis based on the broken integrity of the private key store. This is why we will discuss the pure mathematical countermeasures, which seem to be the most effective against the attacks presented in this article.

The main defending mechanisms are the following:

- **Blinding the data being processed**. We will briefly show this technique in the case of RSA signatures. Before computing the signing transformation, the device computes the values $v_f, v_i \in Z_n$, such that $v_f^{-1} \equiv v_i^d \pmod n$, where n is the public modulus and d is the private exponent. Let the signing operation be in the form $s = m^d \bmod n$. Then the device computes the value $m_{temp} = m*v_i \bmod n$ and uses it as its ordinary input. The resulting signature $s_{temp} = m_{temp}^{\ d} \bmod n$ is then before its output unmasked as $s = s_{temp}*v_f \bmod n$.

  The idea of using the concept of blind signatures to defeat attacks based on the existence of side-channels was for the first time presented in the article [5]. There are coming up still newer and newer evidences, that this technique is really strong countermeasure. Using the OBA-Fundamental Hypothesis it is easy to show that blinding makes all OBA-based attacks impossible, because it cancels out the consequence of this theorem. Of course, we have to be careful to avoid attacking the pair (v_f, v_i) itself. Some ideas on how to do this with the less effort than to generate a brand new pair for each message are presented in [5].

  The concept of blinding can be used also for other cryptosystems than RSA. The particular example for the DES cryptosystem presented in [4] can be used as the inspiration.

- **Randomizing the cryptographic transformations.** This improvement seems to be the essential part of countermeasures against Fault Analysis of signing transformations. Observing the descriptions of these attacks in the case of RSA we can conclude that the attacker has to know the exact value of data processed through the transformation or at least he has to be able to obtain two different results produced with the same input data. Using randomized formats such as those specified in [14] or in draft of PKCS#1 ver. 2.1 (format type EMSA-PSS) obviously cancels out the possibility of these attacks.
- **Checking the integrity of keys.** We have seen that RSA signature scheme can be easily broken when the integrity of private key is not checked. There are plenty of strong and easy to implement techniques to preserve that the cryptographic transformation always works with correct keys. It only remains to use them whenever designing the structure of key containers.
- **Checking the output for faults.** Although this operation seems to be easy to implement and strong against the Fault Analysis, one must be very careful when designing the particular checking mechanism. Otherwise the countermeasure can be weak or totally ineffective against some special kinds of FA attacks. For example the test consisting of computing the main cryptographic transformation twice and then comparing the results is fatally weak against FA attacks based on the alternation of the private key. Clearly the corrupted value of the private key gives always the same result for the same input, so this test would detect nothing suspicious.

  Next, one may think that when we defend the digital signature scheme, the test can be designed to verify the validity of the signature obtained. But one must be sure that this test is done with the proper public key. Despite looking self-evident, there can be a situation when the attacker can affect the integrity of both private and public keys. In such case this test could be weak.

  The consequence is that some kind of checking procedure should be implemented, because it at least makes FA attacks more difficult. On the other hand it surely should not be the only one countermeasure employed.

6. Practical Example – CSP

The shorthand CSP stands for the *Cryptographic Service Provider*. This is the core part of the CryptoAPI subsystem on the Microsoft's operating systems platform. We use the CSP such as an example here to show that the threats, which we have discussed in this article, are very real, because even such common plain cryptographic library called CSP can be attacked in this way. Moreover CSPs can be both software and hardware modules and both of these types can be broken by attacks based on some kind of side-channel. This clearly illustrates that the area of side-channels concerns not only pure cryptographic devices. Last but not least many cryptographers participate in the process of designation of new CSPs, so the conclusions presented here can be highly practical for them. For the deeper discussion of CSP modules and CryptoAPI architecture see [9].

There are usually several CSP modules in the operating system, each of them realizes some kind of cryptographic mechanisms according to the shared interface standard. Besides the support of selected mechanisms, each CSP is also responsible for the maintenance of the persistent data structure referred to as the *key container*. This is the place where the private and public keys are stored.

So which are the main threats for CSP modules? At first it is the structure of key containers, which deserves great attention. Especially in the case of pure software CSPs their architects often think only about the proper encryption of this structure while underestimating the check of its integrity. It is important to note that there are both private and public keys in the key container and that the CSP doesn't have direct access to the certificate of the public key. From here we have that the attacker can affect not only the value of encrypted private key. He can also alter the value of the public key and so cancel out the mechanism checking the correctness of the results of particular cryptographic transformation (for details see the discussion above).

In the case of hardware CSPs we must also take care of the potential possibility of Time or Power Analysis. In fact such analysis is always possible, it is only the question of noise to signal ratio on the particular side-channel. For this reason it is highly recommended that every hardware module (including the CSP) shall use the appropriate combination of countermeasures presented above. Modules (devices), which don't use these techniques, can be considered as prone to some side-channel based attacks.

Finally we shall note that the Time Analysis can be under some special circumstances possible also in the case of pure software CSPs. Although it has not been observed yet, there is the possibility that some precise tool allowing the administrator to monitor the operating system's load can in fact open up the side-channel making the Time Analysis possible. We shall keep this threat in mind when designing the environment, which will the CSP module be used in. From here we can also conclude that the possibility of attacks based on side-channels is not only the question of the architecture of particular device, but it is also the question of the environment, which will this device operate in.

7. Summary

In this document we have introduced the general definition of side-channels and the common characteristics of techniques used to build particular types of attacks using these channels. Especially the notion of Oracle-based Analysis seems to be useful tool for further studies of various kinds of Power and Time Analyses, because it shows us the major common property of these attacks.

Using the general concepts introduced for attacks based on side-channels we have derived the basic set of defending techniques, which shall be automatically implemented in the new cryptographic devices. Of course, we don't say that these techniques thwart all possible attacks based on the notion of side-channels. The reason why we have collected them here is that they defeat the most of the attacks known up to date. Also they are effective against various kinds of attacks simultaneously, so they are really good candidates for implicitly chosen mechanisms included in all device built nowadays.

For those, who are interested in building cryptographic applications based on the CryptoAPI subsystem, we have shown how the topics discussed in this article affect the security of these applications.

The general conclusion of this article should be that the security of particular cryptographic device depends not only on the cryptographic standards employed. It is also the question of the way which are these standards implemented in.

8. References

[1] Anderson, R., Kuhn, M.: Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations, in Proc. of Information Hiding ‘98, pp. 124-142, 1998.

[2] Boneh, D., DeMillo, R. A., Lipton, R. J.: On the Importance of Checking Cryptographic Protocols for Faults, in Proc. of EUROCRYPT ‘97, pp. 37-51, 1997.

[3] Dhem, J. F., Koeune, F., Leroux, P. A., Quisquarter, J. J., Willems, J. L.: A practical implementation of the timing attack, Technical Report CG-1998/1, 1998.

[4] Goubin, L., Patarin, J.: DES and differential power analysis, in Proc. of CHES ’99, pp. 158-172, 1999.

[5] Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, in Proc. of CRYPTO ‘96, pp. 104-113, 1996.

[6] Kömmerling, O., Kuhn, M.: Design Principles for Tamper-Resistant Smartcard Processors, in Proc. of USENIX Workshop on Smartcard Technology, pp. 9-20, 1999.

[7] Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Side Channel Cryptanalysis of Product Ciphers, in Proc. of ESORICS ‘98, pp. 97-110, 1998.

[8] Lenstra, A. K.: Memo on RSA signature generation in the presence of faults, manuscript, Sept. 28, 1996.

[9] Microsoft Developer’s Network Library, available partly online at http://msdn.microsoft.com/library/default.asp.

[10] Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis, in Proc. of Crypto ’99, pp. 388-397, 1999.

[11] Akkar, M. L., Bevan, R., Dischamp, P., Moyart, D.: Power Analysis, What Is Now Possible..., in Proc. of ASIACRYPT 2000, pp. 489-502.

[12] Smulders, P.: The Threat of Information Theft by Reception of Electromagnetic Radiation from RS-232 Cables, Computers & Security vol 9, pp. 53-58, 1990.

[13] van Eck, W.: Electromagnetic Radiation from Video Display Units: an Eavesdropping risk?, Computers & Security vol 4, pp. 269-286, 1985.

[14] Bellare, M., Rofaway, P.: The Exact Security of Digital Signatures – How to Sign with RSA and Rabin, in Proc. of Eurocrypt ‘96, pp. 399-416, 1996.