On the Security of RSA Capable Smart Cards

Roman Novak
Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
roman.novak@ijs.si

Abstract

[1] *The security of the RSA capable smart card that is widely used for secure Internet banking, Web access and remote access to corporate networks, has been reviewed. The card has been tested against several non-invasive attacks. Aspects tested include the system interfaces, the side-channel information leakage and fault resistance against the RSA glitch attack. Several points of vulnerability have been identified. The major weakness is the power consumption information leakage. Simple power analysis (SPA) revealed macro characteristics of the RSA decryption that enable an adversary to extract the user's private key by the adaptive chosen-ciphertext type of attack. The findings give an insight into the security of using smart cards and can be used to eventually improve future implementations of smart card based RSA decryption.*

1 Introduction

Public-key cryptography and related standards and techniques underlie the security of transactions over public networks, allowing consumers to access information, purchase goods and services by computer in the comfort of their homes or workplaces. The RSA cryptosystem is the most widely used public-key system [7]. Its security is based on the intractability of the integer factorisation problem, given that the user's private key is kept secret.

The user's PC environment is not the best place to store cryptographic material and perform cryptographic operations. An adversary has numerous opportunities to steal secrets and impersonate the user. Although the key is additionally protected, it must be exposed to the system while it is used. Various Application Programming Interfaces (APIs) have been developed in order to move cryptographic information and perform cryptographic functions on safer devices.

Smart cards are often used as a ciphering device. They can provide a high level of protection against the seizure of key material. However, they have their own weaknesses that can be exploited by an adversary. Four major attack categories on smart cards have been distinguished [6]. Software attacks exploit vulnerable points in the protocols, cryptographic algorithms, or in their implementation. Eavesdropping techniques monitor the analog characteristics of all supply and interface connections and any other electromagnetic radiation produced by the card. Fault generation techniques use abnormal environmental conditions to generate malfunctions in the processor. Finally, microprobing techniques are used to access the chip surface directly. The microprobing techniques are invasive attacks while the other three are considered to be non-invasive.

In this paper, the resistance of an RSA capable smart card against several non-invasive attacks has been investigated. Weaknesses have been discovered. One of them allows an adversary to extract the private key that is stored on the card.

The card under investigation is used for secure Internet banking, Web access and remote access to corporate networks world-wide. The card provider is among the leaders in the integration of strong authentication and electronic certification technology. The card embeds a cryptoprocessor dedicated to security. On the card, the DES, Triple-DES and RSA algorithms are implemented.

The rest of the paper is structured as follows. In Section 2 the card supported security functions are identified. Section 3 analyses the weaknesses of the interfaces on frequently used Microsoft platforms, while Section 4 analyses card's weaknesses. Power consumption information leakage of the RSA card and fault resistance against the RSA glitch attack have been investigated. Other weaknesses of the RSA implementation are given. We conclude the paper by summarising our findings in Section 5.

2 Card Supported Security Functions

Today, Public Key Infrastructure (PKI) backed up with the Secure Socket Layer protocol (SSL) [10] enables the majority of secure transactions over the Internet. A typical SSL session starts with a handshake protocol in which both peer's identities can be

[1] ERK 2001, Volume B, pp. 135-138, 2001.

authenticated using asymmetric cryptography (RSA, DSS, etc.). In addition, secret session keys are calculated which are needed for data encryption after an initial handshake using symmetric cryptography (DES, RC4, etc.), and for keyed message integrity check using secure hash functions (SHA, MD5, etc.).

The card under investigation can hold the user's certificate and private key. During the handshake protocol an explicit verification of a certificate is requested. The card supports RSA signing which means that the encryption of a 36-byte structure of two hashes (one SHA and one MD5) is requested by SSL handshake protocol and performed on the card [10]. Although the user's private key is removed from the user's PC there are still session keys present, which causes certain security threats while the SSL session is active.

3 API Weaknesses

RSA Laboratories has developed a family of standards called Public-Key Cryptography Standards (PKCS). PKCS#11 specifies an Application Programming Interface (API) to devices which hold cryptographic information and perform cryptographic functions. The interface is supported by Netscape browsers. On the other hand, Microsoft browsers use their own CryptoAPI. The implementation of both interfaces on frequently used Microsoft platforms enables an adversary to gather some security-related data about the card.

In addition to information available in ISO/IEC 7816 series of standards, the knowledge of a card's operating system (COS) propriety commands, file system structure and other limitations of the card is a prerequisite for a number of attacks. This information is usually known only to application developers and held as a business secret. In the man-in-the-middle type of attack, calls to DLL functions may be intercepted, analysed and finally routed to real functions. An adversary can intercept the Personal Identification Number (PIN), obtain valuable information about the card's file system, identify the card's operating system (COS) propriety commands and intercept other secret keys that are needed for some COS commands. The introduction of a signed code can prevent the interception of calls to DLL functions. For instance, CryptoAPI functions use signed modules called cryptographic service providers (CSPs) to perform encryption and decryption, and to provide key storage and security. Furthermore, a card's PIN should not be entered through the user's PC. Other biometric-based personal identification technology should be employed.

In our case, an adversary has additional information publicly available. The card under investigation is based on the Cryptoflex card from Schlumberger for which an extensive reference manual exists [2]. We managed to restore the complete file system structure with full attribute descriptions, intercept PIN code and other keys needed for file system browsing.

4 Weaknesses of the cards

Smart cards have their own weaknesses that enable an adversary to steal secrets from them. The implementation of a cryptographic algorithm often leaks additional side-channel information that can be used to break into the system. Non-invasive attacks have been proposed based on timing information, on a device's power consumption, and on electromagnetic radiation [4]. We have analysed power consumption information leakage of the RSA card while it decrypts a given ciphertext, using low cost sampling equipment. We give a short overview of our adaptive chosen-ciphertext type of attack in which an adversary can reconstruct the private key [9].

In addition to sub-channel information leakage, fault resistance against the RSA glitch attack has also been investigated. Some other weaknesses of the RSA implementation have been identified.

4.1 Power Analysis

Simple Power Analysis (SPA) [5] of a power trace was carried out. A power trace refers to a set of power consumption measurements taken across a smart card operation. Simple Power Analysis (SPA) just interprets a circuit's power consumption while more advanced techniques, like Differential Power Analysis (DPA) and Inferential Power Analysis (IPA), allow observation of the effects correlated to data values being manipulated [5, 3, 8]. Power analysis attacks have been known for a while and effective countermeasures exist [6].

Large macro-features of the RSA decryption operation may be identified in power traces. Selective use of the cryptoprocessor may cause such variations in power consumption. Usually RSA decryption implementation makes use of the Chinese Remainder Theorem (CRT), where two modular exponentiations with smaller moduli are performed instead of one. The exponentiations can be easily identified due to the periodic pattern resulting from a repeated square and multiply algorithm [7]. We could not differentiate multiplying from squaring, which is a common type of side-channel information leakage. However, the comparison between several power traces reveals a slight difference in the computation that follows both exponentiations. In Figure 1, two similar patterns are highlighted on the upper trace while the first highlighted pattern is missing from the lower trace.

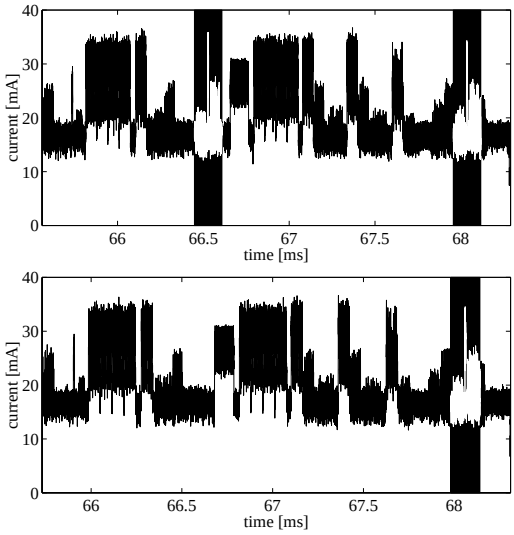

Figure 1: Two types of power trace tails

Suppose p and q are distinct primes, and let modulus $n = pq$. Let e be an encryption exponent and d a decryption exponent, respectively. Pair (n, e) is publicly known while d is kept private. The RSA encryption computes $c = x^e \bmod n$ for some $x \in \mathcal{N}$, while the decryption computes $x = c^d \bmod n$. Algorithm 1 makes use of modular representation to speed up the RSA decryption and signature generation [7].

Algorithm 1: RSA decryption algorithm using Garner's algorithm for CRT
INPUT: ciphertext c, primes p and q, $p > q$, precomputed values $d_{p-1} = d \bmod (p-1)$, $d_{q-1} = d \bmod (q-1)$, $u = q^{-1} \bmod p$.
OUTPUT: plaintext x.

1. $x_p = c^{d_{p-1}} \bmod p$.
2. $x_q = c^{d_{q-1}} \bmod q$.
3. $t = x_p - x_q$.
4. If $t < 0$ then $t = t + p$.
5. $x = (((tu) \bmod p)\, q) + x_q$.

The last three steps do the conversion from a modular representation back to a standard radix representation and are based on the Chinese Remainder Theorem. Implementation of the above algorithm can produce the optional pattern in power trace as a result of the conditional addition in step 4. The information leakage function *diff* can be defined (1). It returns 1 if the addition in step 4 is needed and 0 otherwise. Its argument is output plaintext x. The function can be evaluated only by analysing the power trace of the RSA decryption.

$$\mathit{diff}\,(x) = \begin{cases} 1 & x \bmod p - x \bmod q < 0 \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

The information leakage function has the properties that can be used in adaptive chosen-ciphertext attack by an adversary. *diff* changes value from 0 to 1 only at multiples of prime p, and the value of *diff* remains 1 for l consecutive values of argument x, where $0 < l < q$ and $p > q$.

The reconstruction of secret key d is possible by finding prime p. An adversary may start with plaintexts x_1 and x_2, such that $\mathit{diff}(x_1) = 0$ and $\mathit{diff}(x_2) = 1$. Then, using a binary search-like algorithm and SPA information, he finds the value x, where $\mathit{diff}(x-1) = 0$ and $\mathit{diff}(x) = 1$. The value x is a multiple of prime p that can be extracted by finding the greatest common divisor of x and modulus n. The adversary can control the output x of the RSA decryption by feeding the card with $x^e \bmod n$. The attack may be classified as SPA-based adaptive chosen-ciphertext attack.

Primes p and q are in practice about the same bitlength, and sufficiently large to avoid the elliptic curve factoring algorithm [7]. Suppose modulus n has a bitlength of t bits while primes p and q have a bitlength of $t/2$ bits. In that case it can be shown that x_1 may be $2^{t/2}$ [9]. The binary search-like algorithm requires only $t/2$ power traces, while the result is a prime p instead of a mutiple of prime p. Using Algorithm 2 we managed to restore one of two secret primes and compute secret key d.

Algorithm 2: Reconstruction of p when its bitlength is half bitlength of n
INPUT: modulus n with the bitlength t, exponent e.
OUTPUT: prime p such that p divides n.

1. $x = 2^{t/2}$, $m = x/2$, $l = 0$.
2. While $m \neq l$ do:
   (a) $c = m^e \bmod n$.
   (b) Compute $\mathit{diff}(m)$ by analysing power trace while card decrypts c.
   (c) If $\mathit{diff}(m) = 1$ then $x = m$; otherwise $l = m$.
   (d) $m = (l + x)/2$.
3. Return(x).

Proper implementation of the CRT algorithm should hide SPA characteristics that make factorisation of public modulus feasible. This can be achieved by balancing conditional operations with dummy operations, or even better, by changing the algorithm to use a constant execution path. Other known protective measures should be reconsidered.

4.2 Fault Resistance

A number of attacks depend on introducing errors into key-dependent cryptographic operations. In a glitch attack, which is known to be the most useful in

practical attacks, a deliberately generated malfunction causes replacement of a single critical machine instruction with an almost arbitrary one, or corruption of data values as they are transferred between registers and memory. For instance, a glitch attack against RSA implementation based on the CRT could recover a private key using only one message and the corresponding faulty signature [1].

Suppose a glitch occurs during the first step of Algorithm 1. The algorithm makes a mistake in computing x_p, and computes some x'_p instead. Since x_q is still correct, the resulting incorrect plaintext x' will satisfy the congruence $(x')^e \equiv c \pmod{q}$, but probably not satisfy the congruence $(x')^e \equiv c \pmod{p}$. As a result, the value $(x')^e - c$, instead of being a multiple of n, will be a multiple only of q. This enables an adversary to compute the factors of n by the greatest common divisor algorithm, $q = gcd((x')^e - c, n)$, and then to reconstruct the user's private key.

A countermeasure against such attacks is not to produce an output if an intrusion is detected. Another way to overcome such attacks is to verify results before outputting them.

Power glitches of different duration and amplitude were applied to the test card during the exponentiation operation. We did not succeed in introducing any error in computation. The card entered the reset state or produced the correct ciphertext. Further testing is needed to verify the existence of on-card countermeasures.

4.3 Other Weaknesses

Several properties of the RSA decryption implementation have been observed that can simplify various attacks by an adversary. The implementation allows decryption of small values, which is usually forbidden in publicly available crypto packages. The same value may be decrypted repeatedly without restriction, which offers additional help for an attacker in studying implementation properties. The impact of noise can be averaged out by repeated measurements or, in our case, the presence of other countermeasures can be detected due to different timing characteristics of repeated power traces.

5 Conclusion

Smart cards are often viewed as a way of increasing security. We have shown that security can easily be compromised, even when state-of-the-art cryptographic solutions are used, due to improper implementation of the interfaces and cryptographic operations on the card.

The security of using a RSA capable smart card has been studied. We have reviewed a fast RSA decryption algorithm that is implemented on the card used for secure Internet banking. Several weaknesses have been discovered, the sub-channel information leakage being the largest. We have shown that the cryptoprocessor carries an additional threat to security, due to the easily detectable patterns of its use. An adversary can use the information about the algorithm's execution path in the adaptive chosen-ciphertext attack. Several other weaknesses have been discovered, but the well-known RSA glitch attack did not succeed using the available equipment. Protection of the user's PC is still required because the SSL session keys are handled by the system and the implementations of the interfaces enable an adversary to gather security-related data about the card in the man-in-the-middle type of attack.

References

[1] D. Boneh, R. DeMillo, and R. Lipton. On the Importance of Checking Cryptographic Protocols for Faults. *Lecture Notes in Computer Science*, Springer-Verlag, 1233:37–51, 1997.

[2] Cryptoflex Card Reference Manual, Version 1.55. *Schlumberger document SC003-155-9907-t*, http://www.cryptoflex.com/Support, 2000.

[3] P.N. Fahn and P.K. Pearson. IPA: A New Class of Power Attacks. *Lecture Notes in Computer Science*, Springer-Verlag, 1717:173–186, 1999.

[4] P. Kocher. Timing Attacks on Implementation of Diffie-Hellman, RSA, DSS and Other Systems. *Lecture Notes in Computer Science*, Springer-Verlag, 1109:104–113, 1996.

[5] P. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis: Leaking Secrets. *Proceedings of Crypto'99*, pp. 15–19, Santa Barbara, California, August 1999.

[6] O. Kömmerling and M.G. Kuhn. Design Principles for Tamper-Resistant Smartcard Processors. *Proceedings of the USENIX Workshop on Smartcard Technology Smartcard'99*, pp. 9–20, Chicago, Illinois, 1999.

[7] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. Handbook of Applied Cryptography. *CRC Press Series on Discrete Mathematics and Its Applications*, 1996.

[8] R. Novak. Java Card Sensitivity to Differential Power Analysis. *Proceedings of SCI'2000*, vol. VI, part II, pp. 156–160, Orlando, Florida, 2000.

[9] R. Novak. SPA-Based Adaptive Chosen-Ciphertext Attack on RSA Implementation. *submitted for publication*, 2001.

[10] A.O. Freier, P. Karlton, and P.C. Kocher. The SSL Protocol Version 3.0. *Netscape Communications Corporation, Internet Draft*, March 1996.