NUMBER 5 — JULY 14, 1997

RSA Laboratories' Bulletin

News and advice on data security and cryptography

Comments on Some New Attacks on Cryptographic Devices

Burton S. Kaliski Jr.
Matthew J.B. Robshaw
RSA Laboratories

Introduction

During 1996, a new attack on cryptographic devices was proposed by researchers at Bellcore. This attack depends on introducing errors into key-dependent cryptographic operations through physical intrusion. Soon after, the initial Bellcore work which focused on public-key techniques was extended and applied to secret-key encryption techniques. It also motivated a series of discussions on the capabilities of secure hardware as a means of keeping the details of certain cryptographic algorithms confidential, and a variety of different threat models have now been considered as a result of their work.

The reliance of many security systems on the use of secure hardware or secure processing makes a full evaluation of the potential of fault analysis very important. For developers and users alike an increased awareness of the threat posed by new and novel methods of cryptanalysis allows the development of more secure cryptographic implementations.

In this note we will summarize these recent results and in particular we will assess their practical significance when applied to RSA, DES, and other secret-key cryptosystems.

Burt Kaliski is chief scientist and Matthew Robshaw is principal research scientist at RSA Laboratories. They can be contacted at burt@rsa.com or matt@rsa.com.

Fault Analysis of RSA

Dan Boneh, Richard DeMillo and Richard Lipton at Bellcore [5] first described their analytic technique in a press announcement during October 1996.

In attacking public-key based techniques, the basic attack is against signature generation on a cryptographic device such as a smart card, with an RSA private key. In a typical RSA signature generation, the signer computes a signature s on a message m (typically a hash value) by first computing the values s_p and s_q defined as

$$s_p = m^{d \bmod p-1} \bmod p$$
$$s_q = m^{d \bmod q-1} \bmod q$$

where p and q are the (private) prime factors of the signer's RSA modulus n, and d is the signer's RSA private exponent, and then solving the congruences

$$s \equiv s_p \bmod p$$
$$s \equiv s_q \bmod q$$

by the Chinese Remainder Theorem. The resulting s satisfies the RSA verification equation

$$s^e \equiv m \bmod n$$

where e is the RSA public exponent; this congruence also holds modulo the primes p and q.

Now suppose that the device performing the RSA operation makes a mistake in computing s_p, and computes some s_p' instead. Since s_q is still correct, the resulting incorrect signature s' will satisfy the congruence $(s')^e \equiv m \bmod q$, but it will most likely not satisfy the congruence $(s')^e \equiv m \bmod p$. As a result,

RSA Laboratories
A Division of RSA Data Security

the value $(s')^e - m$, instead of being a multiple of n, will be a multiple only of q. This enables an opponent to compute the factors of n by the greatest common divisor algorithm: $q = \gcd((s')^e - m, n)$. Given the prime q, the opponent easily obtains the signer's private key.

To carry out the attack the opponent simply introduces an error during the device's computation of s_p (or alternatively of s_q), perhaps by voltage or clock speed variations on the device, then analyzes the incorrect signature that results to recover the private key. This can be done in some environments without the device owner's knowledge; for instance, the attack might be implemented in the device reader.

The attack just described is against signature generation with an RSA private key. Similar attacks can be mounted against other schemes, including the Schnorr [12] and Fiat-Shamir [6] schemes. As noted in the Bellcore announcement, the difficulty of the attack does not depend on the size of the key, unlike traditional cryptanalytic techniques.

An attack against decryption with a private key, while similar in theory to one on signature generation, seems to be prevented in practice by the fact that the output of the decryption operation — typically a symmetric key — is generally not available to an opponent. (It is often retained in the device.) Both the input and the output of the operation are required by the attack.

Differential Fault Cryptanalysis

The work at Bellcore was concerned with exploiting errors in public-key based computations. A variant of this work was soon discovered by Eli Biham and Adi Shamir who showed that erroneous computations can also be useful in the cryptanalysis of secret-key cryptosystems (`http://www.cs.technion.ac.il/~biham/`). Biham and Shamir initially concentrated their attention on DES and they demonstrated (and confirmed with a software simulation) that by introducing errors into the DES encryption process and by comparing incorrect answers with the known correct output, information about the secret key might be derived.

This attack, called *Differential Fault Analysis*, is very similar to the technique of differential cryptanalysis [3]. Errors introduced during encryption or decryption effectively provide the cryptanalyst with pairs of encryptions that have some difference between them. Provided there are not too many errors, and these errors occur relatively near the end of the encryption/decryption process, it can be straightforward to compare the erroneous output with the correct one and to identify the type and location of the error. With this information it might then be possible to mount a conventional differential-style attack using the data that is already available and information about the secret encryption key might be extracted. It should be anticipated that this work will be extended in its scope and applicability as further research takes place [4].

In this attack on DES (which is also a style of attack that can be readily extended to other block ciphers including triple-DES and to both encryption and decryption operations) the important issue is not exactly where the errors take place but quite how many and at approximately which point in the encryption process. This is in contrast to the Bellcore attack on RSA signatures where only one erroneous computation is required together with the known input to that computation to recover the key. Moreover, the single erroneous computation in the Bellcore attack can consist of any number of errors, of any type, but as a limitation the errors can only take place in what would normally be one of two independent computations.

We note here that while the applicability of differential fault analysis has been concentrated on DES, all block ciphers will be potentially vulnerable to some extent to this style of analysis. And while it seems that currently these potential vulnerabilities are primarily of theoretical interest, their importance and relevance to the process of designing secure hardware should not be overlooked.

Overcoming These Basic Attacks

A general way to overcome such attacks is not to produce an output if an intrusion is detected, a method that is implemented in many cryptographic devices today [9].

Another safeguard is to verify results before outputting them. In this way, an incorrect result is not available to an opponent, so the attack is not possible. For RSA signatures, this involves very little overhead, since RSA signature verification is very fast, assuming the RSA public exponent e is small as is

typical in practice. Here, the device simply checks after computing the signature s that

$$s^e \equiv m \bmod n$$

and outputs the signature s if and only if the comparison is successful.

Signature verification is a general countermeasure for any signature scheme, and while it is little overhead for RSA there may be significant overhead for some other schemes. An area for further research is whether it is possible to verify correctness without a full signature verification.

With regard to secret-key cryptosystems the result of an encryption or decryption can be verified by immediately reversing the operation, i.e. to decrypt the proposed output of an encryption operation and vice versa, to check whether the starting input is recovered. In the case of a symmetric cipher like DES however, this will result in halving the rate of encryption or decryption since each result is checked with an operation that takes an equal length of time. As a consequence, this remedy might best be viewed as a precaution only suitable for those that are extremely concerned about this style of attack.

Apart from checking calculations, something which might have considerable impact on performance, another possible precaution is the use of randomization. By doing this it can be ensured that with a potentially high probability, the input used for the computation is not the one required or expected by the cryptanalyst.

As an example, consider using randomization to hinder the fault analysis attack against RSA signatures. Suppose that the message m is changed to some m' before computing the RSA operation, in such a way that the change can be reversed based on information in m'. As a simple example, assume that the message m is initially shorter than the RSA modulus n, and that randomization involves concatenation with a random value. (A related approach was proposed recently by Mihir Bellare and Phil Rogaway [2] so the example is quite relevant.)

If there is no error in the signature computation then m' is correctly recovered during RSA signature verification and the randomization of m can be reversed to allow the signature to be correctly verified. If, however, an error is induced in the computation then the signature block produced will not reveal the correct input m'. Moreover, because of randomization, it will not be easy to deduce m' from m. As long as the random value and hence m' are unknown to the opponent, the attack will be prevented (at least for RSA) since the opponent will not be able to compute the value $(s')^e - m'$ required to complete the attack.

Note that in some implementations, the random value that is combined with m may be output with the signature. In such circumstances the opponent would be able to compute m' and thereby complete the attack. Moreover, in signature schemes that unlike RSA do not give the opportunity to recover the message from the signature, it seems that outputting the random value is unavoidable. It is not immediately clear how one would protect other schemes against attack in this manner, though this is another area for further research.

Note that a similar approach might be adopted in protecting symmetric ciphers, but the practical advantages can be slight with many of today's ciphers. The attack of Biham and Shamir require both correct and incorrect encryptions of the same input. (Note that we can equally consider decryptions instead of encryptions.) The use of randomization, perhaps by setting aside part of the input block for the introduction of a randomly generated value, would hinder this collection of encryption pairs. Instead successive encryptions of the same initial input m would be different (even in the absence of errors) since randomization would make the actual input to the encryption operation an unknown m'.

However the efficacy of this precaution will depend very much on the style of attack it is intended to protect against and in reality, such a mechanism is unlikely to be useful unless the block cipher has a large block size. In such cases the impact on the encryption/decryption data rate when using randomization might be proportionally less significant.

Perhaps the best advice that can be offered is that of prevention. All the attacks described in this bulletin serve to stress the fact that good engineering practices in the design of secure hardware are essential.

More Involved Attacks

While these attacks are interesting in themselves they have been the trigger for more involved re-

search into the potential of creating errors during some cryptographic computation. It should be realized that being able to force errors into a cryptographic computation provides the cryptanalyst with considerable power. No longer is the cryptanalyst passively recording different types of plaintext and ciphertext and analyzing the data off-line (as more traditional methods of cryptanalysis generally allow) but instead the introduction of faults within a calculation has a direct and obvious impact on the actual computations performed. It should therefore not be surprising that these attacks appear to allow very dramatic compromises in the security of some implementations.

Biham and Shamir have extended their work on differential fault analysis to cases where the key might be recovered even when the specifics of the encryption algorithm are unknown and even to deducing the structure and eventually the details of some unknown encryption algorithm [4]. This fascinating work provides evidence for the view held by some that it is difficult to keep the details of a secret cipher hidden from analysis even when secure tamper-proof hardware is used. Another interesting style of attack considers the effect of making a permanent change to the cryptographic hardware, perhaps by using engineering tools to break or modify parts of the circuit [4].

Work by Quisquater [11] has demonstrated how the use of errors might be exploited in reducing the computational requirements of a brute-force key search attack. Meanwhile Anderson and Kuhn [1] have focused more on an attack in which the machine instructions used during the encryption process are changed in some way.

Anderson and Kuhn's attack seems to be a yet more invasive attack than that of forcing errors into the data or the key being used and many ciphers are susceptible to attack or reverse-engineering under such circumstances. While realizing the scope of this style of analysis is important, we need to recognize that the equipment and the effort required for an attacker to mount the style of attack described by Anderson and Kuhn is far more extensive than might ordinarily be envisaged. In practical terms, it appears that these more involved attacks are more relevant to the task of reverse-engineering some secret cipher in hardware, where the cryptanalyst can expect to have the luxury of mounting highly invasive and repetitive analysis on some token or isolated device. In fact it might be argued that if cryptanalysts are able to actively change and alter the operations used at specific points in the encryption procedure, then there seems to be almost limitless scope to the cryptographic damage that might be inflicted.

Discussion

Manufacturers of cryptographic modules such as smart cards have been aware for some time of the importance of protecting against intrusions such as voltage and clock speed variations, which may result in the device performing unintended operations. Many cryptographic devices incorporate circuitry to detect such attacks [9]. The U.S. government standard FIPS 140-1 [10] similarly covers issues related to these types of attack. Thus, attack by physical intrusion is by no means unanticipated. Indeed Quisquater has observed that there is a whole field of study devoted to a related phenomenon, the effect of single isolated bit errors in electronic devices (see for example `http://flick.gsfc.nasa.gov:80/radhome/`). The question of how practical these attacks might be is, in many ways, one for physicists and the designers and manufacturers of secure hardware. While there is often a great deal of flexibility in the type and number of errors that can be accommodated in these attacks, there still remain practical limitations.

We note that this style of cryptanalysis by which errors are introduced into a cryptographic calculation need not be restricted to hardware devices. It is possible to envisage situations where errors in the execution of a program in software, such as overflowed arrays or exceeded boundary conditions, might corrupt the data that will be used in some other part of the cryptographic computation. Such a possibility should be among the threats considered by system designers and software engineers.

What is significant about the Bellcore attack and related work is that an error introduced during a cryptographic computation can produce a favorable result for the opponent. Previous attacks often needed to override the device's control logic at specific points to gain access to sensitive data, and were thus dependent on very targeted penetration, although a crude attack could potentially be effective. The new attacks do not need to be very targeted. They require much less precision on the attacker's part, and consequently much less knowledge of the

internals of the device. Indeed, as we have already mentioned, some of these techniques can be used to recover the key from an unknown cipher or even to deduce the structure of an unknown cipher in tamper-resistant hardware. This will have important implications for the viability of proposals which aim to ensure that certain cryptographic designs remain secret.

As security techniques have become more widely implemented, it has become common to move cryptographic processing from desktop and server computers to portable devices. This is often viewed as a way to increase security, since sensitive keys are no longer stored in computer memory which may be vulnerable to compromise. It is well understood that the devices must include a secure operating system that controls access to stored keys and to cryptographic operations. Furthermore, the devices must have tamper-resistant storage for the keys. These recent attacks serve as a reminder that the devices must also have tamper-resistant processing for the cryptographic operations.

Conclusion

Like the timing attacks on RSA and other cryptosystems observed last year by Paul Kocher [8] (see [7] for discussion), the attacks described in this bulletin show again that security involves more than just good algorithms. Indeed, good engineering is essential. The underlying security of RSA, DES and other algorithms has not been questioned, only the security of particular implementations against one form of physical attack.

The attack can and may already be prevented by well known hardware implementation techniques, and can also be prevented by simple modifications to the cryptographic processing. However the significance of these attacks, and their relevance to high security applications, should not be overlooked.

For further information on this and other developments in security, please contact RSA Laboratories.

References

[1] R. Anderson and M. Kuhn. Improved Differential Fault Analysis. Available from `http://www.cl.cam.ac.uk/users/rja14/`.

[2] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with RSA and Rabin. In U. Maurer (ed.), *Advances in Cryptology — Eurocrypt '96, Lecture Notes in Computer Science*, vol. 1070, pages 399-416, Springer-Verlag, 1996.

[3] E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, 1993.

[4] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. To appear in *Proceedings of Crypto '97*.

[5] D. Boneh, R. DeMillo and R. Lipton. On the importance of checking cryptographic protocols for faults. In W. Fumy (ed.), *Advances in Cryptology — Eurocrypt '97, Lecture Notes in Computer Science*, vol. 1233, pages 37-51, Springer-Verlag, 1997.

[6] U. Feige, A. Fiat and A. Shamir. Zero-knowledge proofs of identity. Journal of Cryptology, Vol. 1, pages 66-94, 1988.

[7] B. Kaliski. Timing attacks on cryptosystems. *RSA Laboratories' Bulletin*, No. 2, January 23, 1996. Available from `http://www.rsa.com/`.

[8] P. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. In N. Koblitz (ed.), *Advances in Cryptology — Crypto '96, Lecture Notes in Computer Science*, vol. 1109, pages 104-113, Springer-Verlag, 1996.

[9] David Naccache and David M'Raïhi. Cryptographic smart cards. *IEEE Micro*, pages 14-24, June 1996.

[10] National Institute of Standards and Technology (NIST). *FIPS Publication 140-1: Security Requirements for Cryptographic Modules*. October 1995. Available from `http://csrc.nscl.nist.gov/fips/`.

[11] Jean-Jacques Quisquater. Personal communication. December 1996.

[12] C. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, vol. 4, pages 161-174, 1991.

For more information on this and other recent security developments, contact RSA Laboratories at one of the addresses below.

RSA Laboratories
100 Marine Parkway, Suite 500
Redwood City, CA 94065 USA
415/595-7703
415/595-4126 (fax)
rsa-labs@rsa.com
http://www.rsa.com/rsalabs/

Copyright © 1997 RSA Laboratories, a division of RSA Data Security, Inc., a Security Dynamics Company. All rights reserved.