Attacks on Rivest-Shamir-Adleman (RSA) cryptosystem

Klemen Krenker

University of Ljubljana, Faculty of computer and information science, Večna pot 113, 1000 Ljubljana, Slovenia
E-mail: kk8876@student.uni-lj.si

Abstract. In 1977 began an era of public-key encryption with the release of RSA, and with it began the desire to break the cryptosystem. Since its release there were quite a few attacks that have succeeded to a degree, but due to the difficult challenges that RSA presents none of them have been able to completely break it yet.

Keywords: RSA, RSA-CRT, RSA attacks

Napadi na Rivest-Shamir-Adleman (RSA) kriptosistem

Leta 1977 se je začelo obdobje šifriranja z uporabo javnih ključev z objavo RSA in s tem se je tudi začela želja po razbitju kriptosistema. Od uradne izdaje je bilo kar nekaj napadov, ki so poskusili in do neke mere uspeli razbiti sistem, vendar zaradi težkih izzivov, ki jih predstavlja RSA, nobenemu še ni povsem uspelo.

1 INTRODUCTION

In this article, we aim to introduce RSA and the important parts for its implementation. We will also showcase a number of attacks (or groups of attacks) on RSA cryptosystem.

2 RIVEST-SHAMIR-ADLEMAN: RSA

RSA published in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman is among the first public-key cryptosystems used in data transference [5]. Core idea of such cryptographic method is the use of two keys, a public encryption key *e*:

$$c = m^e \bmod n \tag{1}$$

and private decryption key *d:*

$$m = c^d \bmod n \tag{2}$$

Modulus *n* is a product of two distinct prime numbers *p* and *q*:

$$n = pq \tag{3}$$

where factorization of *n* is computationally difficult. Public key is chosen from the interval:

$$1 < e < \phi(n) \tag{4}$$

where:

$$\phi(n) = (p - 1)(q - 1) \tag{5}$$

such that:

$$gcd(e, \phi(n)) = 1 \tag{6}$$

And from *e*, *d* can be computed using the extended Euclidean algorithm:

$$ed \equiv 1 \pmod{\phi(n)} \tag{7}$$

(EV_Slovene)

2.1 Trapdoor Permutations

Trapdoor permutations is composed of three algorithms *(G, E, D)*:

- *G*: randomized algorithm that produces a key pair *(pk, sk)*,
- E: deterministic mapping for every public key, $x = E(pk, x)$, must be a bijective function,
- D: deterministic algorithm, where for every key pair the following holds true: $D(sk, E(pk, x)) = x$.

In RSA terms, for *public key pair (n, e)*, $f(x) = x^e \bmod n$ is a TDP. Given *x*, *f(x)* can be easily computed. However computing $f^{-1}(y)$ without the *private key* is difficult, but if we know *y*, compututation becomes easy.

2.2 Padding

Due to plain RSA *(described at the beginning)* being vulnerable to *chosen plaintext attacks (because of determinism)* some sort of smart padding needs to be done before encryption. Two such schemes are *PKCS#1 (v1.5)* and *OAEP*, where *OAEP* adds *randomness* to a deterministic encryption scheme, it also prevents partial decription without inverting the TDP one-way permutation *f*.

2.3 Chinese Remainder Theorem (CRT)

Theorem once used for counting soldiers in 3rd century AD nowadays offers a solution to simultaneous linear congruences with coprime moduli. For $n = pq$ (where p and q are coprime) we can then divide the message into two halves and compute each modulo seperately. One

Received
Accepted

half being $m_1 = (M^d \bmod n) \bmod p$ and $m_2 = (M^d \bmod n) \bmod q$. Due to p and q being prime integers and using the Fermat's little theorem, we can then re-write the equations, reducing the exponent d modulo $q-1$ or $p-1$, these values are often precomputed.

$$d_p = d \bmod (p-1)$$
$$d_q = d \bmod (q-1) \quad (10)$$

Therefore leaving us with $m_1 = ((M \bmod p)^{d \bmod p-1}) \bmod p$ and $m_2 = ((M \bmod q)^{d \bmod q-1}) \bmod q$.

We have now re-written the equation for two halves of the message, to recombine the messages, such m must be found that the following holds true $m \equiv (M^d \bmod n) \bmod p$ and $m \equiv (M^d \bmod n) \bmod q$. Due to CRT we can deduce $m \equiv (M^d \bmod n) \bmod pq$.

2.4 RSA security

RSA security depends on two computationally hard problems, first being *integer factorization (decomposition of a composite number into smaller numbers – called prime factorization if these numbers are primes)* and the second being *"RSA problem" (finding the e^{th} root of a number modulo n)*. At this time there are no *classical* algorithms for factoring large integers in polynomial time, but at the same time, the existence of such algorithm has not yet been disproven.

3 ATTACKS ON RSA

In the many years of RSA encryption quite a few possible vulnerabilities have been revealed and exploited, but RSA is yet to be broken. Attacks can be categorized by inspecting the vulnerability exploited by groups [2].

3.1 Elementary attacks

Elementary attacks exploit the incorrect use of RSA. A good example is the use of *common modulus* for multiple sub-systems within the system, where a member of one sub-system can factor n using his own exponent. If $gcd(e_1, e_2) = 1$ holds, then there exists such integers a and b that:

$$e_1 a + e_2 b = 1 \quad (10)$$

Using the *Extended Euclidean algorithm* integers a and b can be produced and message can be retrieved using the formula:

$$\begin{aligned} c_1^a * c_2^b &= (m^{e_1})^a * (m^{e_2})^b \\ &= m^{e_1 a} * m^{e_2 b} \\ &= m^{e_1 a + e_2 b} \\ &= m^1 = m \end{aligned} \quad (11)$$

A clear solution to the problem presented is to not use the same modulus to deploy sub-systems.

3.2 Implementation or side-channel attacks

Instead of exploiting the incorrect deployment of RSA, these attacks are based on the implementation of a computer system, information about *timing, power consumption, electromagnetic or sound leaks* that are revealed and then used to break the system. *Timing* attacks monitor the time it takes to complete some computation, the timing difference depends on wether or not extra reduction step is performed while using the Montgomery multiplication. Schindler has calculated the probability to be:

$$\Pr[extra\ reduction] = \frac{g \bmod q}{2R} \quad (12)$$

As g approaches either p or q, the number of extra reductions increases, but when g is equal to p or q there appears a dramatic drop in reductions [3]. For all the above attacks except the timing attacks some specialized equipment is needed, and most importantly, physical access to the system.

3.3 Low public or private exponent

To shorten the times of encryption *(public key e)* or decryption *(private key d)* small values might be used for any/both of the keys to speed up the process, especially the decryption. This poses a security threat, and in the case of low private exponent, a complete break. To ensure the safety against this specific threat keys *(private and public)* need to be of a sufficient length.

4 CONCLUSION

After so many years since the release of RSA it is remarkable that there were no serious breaks, at least not in the terms of breaking the algorithm, but poor implementations of it. Currently there still exists a way to deploy RSA encryption properly and safely and defend against the know types of attacks. However, a quantum algorithm was developed for integer factorization by Peter Shor in 1994, proving that RSA would be vulnerable to an integer factorization attack with the use of quantum computers. Needless to say, it will definitely be interesting to see what happens in the years to come in regards to RSA encryption.

REFERENCES

[1] Boneh, D. (1999). Twenty years of attacks on the RSA cryptosystem. *Notices of the AMS*, *46*(2), 203-213.

[2] Schindler, W. (2000, August). A timing attack against RSA with the chinese remainder theorem. In *International Workshop on Cryptographic Hardware and Embedded Systems* (pp. 109-124). Springer, Berlin, Heidelberg.

[3] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120-126.