RSA Attacks

By Abdulaziz Alrasheed and Fatima

1 Introduction

Invented by Ron Rivest, Adi Shamir, and Len Adleman [1], the RSA cryptosystem was first revealed in the August 1977 issue of Scientific American. The RSA is most commonly used for providing privacy and ensuring authenticity of digital data. RSA is used by many commercial systems. It is used to secure web traffic, to ensure privacy and authenticity of Email, to secure remote login sessions, and it is at the heart of electronic credit-card payment systems.

Since its initial release, the RSA has been analyzed for vulnerabilities. Twenty years of research have led to a number of intriguing attacks, none of them is devastating. They mostly show the danger of wrong use of RSA. Our objective is to explore some of these attacks.

RSA encryption in its simple form is explained as follows. Let $N = pq$ be the product of two large primes of the same size ($n/2$ bits each). As [1] explains, a typical size for $N$ is $n=1024$ bits, i.e. 309 decimal digits. Let $e, d$ be two integers satisfying $ed = 1 \mod \varphi(N)$ where $\varphi(N) = (p-1)(q-1)$. $N$ is called the RSA modulus, $e$ is called the encryption exponent, and $d$ is called the decryption exponent. The pair $(N, e)$ is the public key. The pair $(N, d)$ is called the secret key and only the recipient of an encrypted message knows it.

A message $M$ is encrypted by computing $C = M^e \mod N$. To decrypt the ciphertext $C$, the authentic receiver computes $C^d \mod N$.

$$C^d = M^{ed} = M \mod N$$

The last equality is based on Euler’s theorem.

1.1 Factoring Large Integers

This is known as the first attack on RSA public key $(N, e)$. After getting the factorization of $N$, an attacker can easily construct $\varphi(N)$, from which the decryption exponent $d = e^{-1} \mod \varphi(N)$ can be found. Factoring the modulus is referred to as brute-force attack. Although factorizing the modulus has been improving, the current state of the art of this attack is unable to post a threat to the security of RSA when RSA is used properly. The current fastest factoring algorithm is the General Number Field Sieve with running time of $\left(\left(c + o(1)\right) n^{1/3} \log^{2/3} n\right)$ [1]

2 Elementary attacks

Let’s begin by describing some old elementary attacks. These attacks depend primarily on the misuse of RSA. We will only talk about two examples of many elementary attacks.
2.1 Common modulus

The assumption that generating the same modulus $N = pq$ for all users of a system, and user $i$ is provided with a unique pair $e_i$, $d_i$ from which user $i$ forms a public key $(N, e_i)$ and a secret key $(N, d_i)$ may seem to work providing that a trusted central authority provides the unique pairs. But as per [1] the resulting system is insecure since Bob who is unable to decipher Alice’s cipher due to not having Alice private key $d_A$ he however, can factor $N$ using his own exponents. This observation, due to Simmons, shows that an RSA modulus should only be used by one entity.

2.2 Blinding

Blinding enables Eve to obtain a valid signature on a message of his choice by asking Bob to sign a random "blinded message" [1]. In that case, Bob does not know what message he is actually signing and most signature schemes apply a "one-way hash" to the message prior to signing, thus the attack is not a serious concern.

Let $(N, d)$ be Bob's private key and $(N, e)$ be his public key. Assume that an adversary Eve wants Bob's signature on a message $M \in \mathbb{Z}_N^*$. Being a smart move, Bob should refuse to sign $M$. Otherwise Eve can compute $S = S' / \gamma \mod N$ and obtains Bob's signature $S$ on the original $M$.

Thus, $S^e = (S')^e / \gamma^e = (M')^{ed} / \gamma^e \equiv M' / \gamma^e = M \mod N$

3 Low private Exponent

Since modular exponentiation takes time linear in $\log_2 d$, a small $d$ can improve performance by at least a factor of 10, one of the misuses of RSA is to use a small value of $d$ to reduce decryption time. Unfortunately, a clever attack due to M. Wiener [2] shows that a small $d$ can result in a total break of the RSA cryptosystem.

**Theorem** (M. Wiener) Let $N = pq$ with $q < p < 2q$. Let $d < 1/3 \sqrt{N}$. Given $(N, e)$ with $ed = 1 \mod \varphi(N)$, an attacker can efficiently recover $d$.

**Proof** The proof is based on approximations using continued fractions. Since $ed = 1 \mod \varphi(N)$, there exists a $k$ where $ed - k \varphi(N) = 1$. Therefore,

$$\left| \frac{e}{\varphi(N)} - \frac{k}{d} \right| = \frac{1}{d\varphi(N)}$$

Since $\varphi(N) = N-p-q+1$ and $p+q-1 < 3\sqrt{N}$ an attacker can use $N$ to approximate $\varphi(N)$.

In order to avoid this attack, and since $N$ is 1024 bits, $d$ must be at least 256 bits long. This is unfortunate for smart cards or low powered devices.
4 Low public exponent

In order to reduce encryption or signature-verification time, a small public exponent \( e \) is customary used. The smallest possible value according to [source] is 3, but to defeat certain attacks the value \( e = 2^{16} + 1 \) is recommended. When the value \( 2^{16} + 1 \) is used only 17 multiplications are required for signature verification as opposed to roughly 1000 when a random \( e < \varphi(N) \) is used. Unlike the attack of low private exponent, attacks that apply when a small \( e \) is used are far from a total break.

4.1 Coppersmith theorem

The most powerful attacks on low public exponent RSA are based on a Copper-smith theorem.

**Theorem** Let \( N \) be an integer and \( f \in \mathbb{Z}[x] \) be a monic polynomial of degree \( d \). Set \( X = N^{1/d} \) for some \( \epsilon \geq 0 \). Then, given \((N, f)\) an attacker can efficiently find all integers \( |x_0| < X \) satisfying \( f(x_0) = 0 \mod N \). The running time is dominated by the time it takes to run the LLL algorithm on a lattice of dimension \( O(w) \) with \( w = \min(1/\epsilon, \log_2 N) \) [1].

The theorem provides an algorithm for efficiently finding all roots of \( f \) modulo \( N \) that are less than \( X = N^{1/d} \). The algorithm's running time decreases as \( X \) gets smaller. The strength of this theorem is its ability to find small roots of polynomials modulo a composite \( N \).

Application of Coppersmith's Theorem [3]:

- Attack stereotyped messages in RSA (sending messages whose difference is less than \( N^{1/e} \) can compromise RSA)
- Security proof of RSA-OAEP (constructive security proof).
- Affine Padding
- Polynomially related RSA messages (sending the same message to multiple recipients)
- Factoring \( N = pq \) if the high bits of \( p \) are known.
- An algorithm that can get the private key for RSA in deterministic polynomial time can be used to factor \( N \) in deterministic polynomial time.
- Finding integers with a large smooth factor in a proscribed interval.
- Finding roots of modular multivariate polynomials.

4.2 Hastad's broadcast attack

The first application of Coppersmith's theorem and the improvement to an old attack is Hastad's Broadcast Attack. Suppose Bob wishes to send an encrypted message \( M \) to a number of parties \( P_1, P_2, \ldots, P_k \). Each party has its own RSA key \((N_i, e_i)\). We assume \( M \) is less than all the \( N_i \)'s. Idealistically, to send \( M \), Bob encrypts it using each of the public keys and sends out of the \( i \)th ciphertext to \( p_i \). An attacker Eve can eavesdrop on the connection out of Bob's sight and collect the \( k \) transmitted ciphertexts.
For simplicity, suppose all public exponents $e_i$ are equal to 3. A simple argument shows that Eve can recover $M$ if $k \geq 3$. Indeed, Bob obtains $C_1, C_2, C_3$, where

$$C_1 = M^3 \mod N_1, \quad C_2 = M^3 \mod N_2, \quad C_3 = M^3 \mod N_3.$$ 

Assume that $\gcd(N_i, N_j) = 1$ for all $i \neq j$ since otherwise Eve can factor some of the $N_i$'s. Hence, applying the Chinese Remainder Theorem (CRT) to $C_1, C_2, C_3$ gives a $C' \in \mathbb{Z}_{N_1N_2N_3}$ satisfying $C' = M^3 \mod N_1N_2N_3$. Since $M$ is less than all the $N_i$'s, we have $M^3 < N_1N_2N_3$. Then $C' = M^3$ holds over the integers. Thus, Eve may recover $M$ by computing the real cube root of $C'$. More generally, if all public exponents are equal to $e$, Eve can recover $M$ as soon as $k > e$. The attack is feasible only when a small $e$ is used.

To stimulate Hastad's result, if $M$ is $m$ bits long, Bob could send $M_i = i2^m + M$ to party $P_i$. Since Eve obtains encryptions of different messages, he can't mount the attack. Unfortunately, Hastad showed that this linear padding is insecure. In fact, he proved that applying any fixed polynomial to the message prior to encryption does not prevent the attack [1].

Suppose that for each of the participants $P_1, \ldots, P_k$, Bob has a fixed public polynomial $f_i \in \mathbb{Z}_{N_i}[x]$. To broadcast a message $M$, Bob sends the encryption of $f_i(M)$ to party $P_i$. By eavesdropping, Eve learns $C_i = f_i(M)^e \mod N_i$ for $i = 1, \ldots, k$. Hastad showed that if enough parties are involved, Eve can recover the plaintext $M$ from all the ciphertexts. In more generality, Hastad proved that a system of univariate equations modulo relatively prime composites, such as applying any fixed polynomial $g_i(M) = 0 \mod N_i$, could be solved if sufficiently many equations are provided. This attack suggests that randomized padding should be used in RSA encryption.

**Theorem** Let $N_1, \ldots, N_k$ be pairwise relatively prime integers and set $N_{\text{min}} = \min_i (N_i)$. Let $g_i \in \mathbb{Z}_{N_i}[x]$ be $k$ polynomials of maximum degree $d$. Suppose there exists a unique $M < N_{\text{min}}$ satisfying

$$g_i(M) = 0 \mod N_i \quad \text{for all } i = 1, \ldots, k.$$ 

Under the assumption that $k > d$, one can efficiently find $M$ given $(N_i, g_i)^k_1 = 1$.

### 4.3 Franklin-Reiter Related Message Attack

Franklin and Reiter [4] found a smart attack when Bob sends Alice related encrypted messages using the same modulus. Let $(N, e)$ be Alice’s public key. Suppose $M_1, M_2$ are two distinct messages such as $M_1 = f(M_2) \mod N$. If Bob encrypt the messages and transmit the resulting ciphers $C_1$ and $C_2$ we will show how an attacker can easily recover $M_1$ and $M_2$.

**Lemma** Set $e = 3$ and let $(N,e)$ be an RSA public key. Let $M_1 \neq M_2$ satisfy $M_1 = f(M_2) \mod N$ for some linear polynomial $f = ax + b$ with $b \neq 0$. Then, given $(N, e, C_1, C_2, f)$ an attacker can recover $M_1$ and $M_2$ in time quadratic in $\log N$. 


Proof Since $C_1 = M_1^e \mod N$, we know that $M_2$ is a root of the polynomial $g_1(x) = f(x)^e - C_1$ and similarly $M_2$ is a root of $g_2(x) = f(x)^e - C_2$. The linear factor $x - M_2$ divides both polynomials. Therefore, an attacker may use the Euclidean algorithm to compute the gcd of $g_1$ and $g_2$. If the gcd turns out to be linear, $M_2$ is found.

4.4 Coppersmith’s short pad attack

Generally, The Franklin-Reiter attack is considered to be an artificial attack because why should Bob send Alice the encryption of related messages? Coppersmith strengthened the attack and proved an important result on padding. Coppersmith showed that if randomized padding suggested by Hastad is used improperly then RSA encryption is not secure [7].

A naive random padding algorithm might pad a plaintext $M$ by appending a few random bits to one of the ends. The following attack points out the danger of such simplistic padding. Suppose Bob sends a properly-padded encryption of $M$ to Alice. An attacker, Eve, intercepts the ciphertext and prevents it from reading its destination. Bob notice that Alice did not respond to his message and decides to resend $M$ to Alice. He randomly pads $M$ and transmits the resulting ciphertext. Eve now has two ciphertexts corresponding to two encryptions of the same message using two different random pads.

The following theorem shows that although he does not know the pads used, Eve is able to recover the plaintext.

Theorem Let $(N, e)$ be a public RSA key where $N$ is n-bits long. Set $m = \lfloor n/e^2 \rfloor$. Let $M \in \mathbb{Z}^*_N$ be a message of length at most $n - m$ bits. Define $M_1 = 2^m M + r_1$ and $M_2 = 2^m M + r_2$, where $r_1$ and $r_2$ are distinct integers with $0 \leq r_1, r_2 < 2^m$. If Eve is given $(N, e)$ and the encryptions $C_1, C_2$ of $M_1, M_2$ (but is not given $r_1$ or $r_2$), he can efficiently recover $M$.

4.5 Partial key exposure attack

This attack is possible when the public key is small. If an attacker exposed a fraction of the bits of $d$, s/he can, on the assumption that the modulus is small, reconstruct the rest of $d$. Boneh, Durfe, and Frankel [5] have made recent proof that $d$ can be reconstructed as long as $e < \sqrt{N}$.

Theorem Let $(N,d)$ be a private key with $N$ is n-bits long. Given the $\left\lfloor \frac{n}{4} \right\rfloor$ least significant bits of $d$, an attacker can reconstruct all of $d$ in time linear in $e \log_2 e$.

Theorem (Coppersmith) Let $N = pq$. Given the n/4 least or most significant bits of $p$, one can factor $N$ efficiently. $k$ integer exist such that: $ed - k (N - p - q + 1 ) = 1$. [1]

Since $d < \varphi(N)$, then $0 < k \leq e$. Reducing $N$ to $2^{n/4}$and setting $q = N/p$, we get:

$$(ed)p - kp(N - p + 1) + kN = p \mod 2^{n/4}$$ [1]
5 Implementation Attacks

The following attacks follow different class of attacks. Instead of attacking the underlying structure of RSA function, these attacks target the implementation of RSA.

5.1 Timing attacks

Let us consider a smartcard that stores a private RSA key. An attacker may not be able to see its content and expose the key. However, a clever attack found by Kocher [5] explains that the precise time of decryption the card takes can help an attacker find or discover the private decryption exponent d. Repeated squaring algorithm can be used for this attack which is explained as follow [1]:

- Let \( d = d_n d_{n-1} \ldots d_0 \) (binary of d)
- Set \( z = M \) and \( C = 1 \). For \( i=0 \) \ldots \( n \) do:
  - (1) if \( d_i = 1 \) set \( C = C \cdot z \mod N \)
  - (2) \( z = z \cdot z \mod N \)
- \( C \) at the end has the value \( M^d \mod N \)

5.2 Random Faults

To speed up the computation of \( M^d \mod N \), Implementations of RSA decryption and signatures frequently use the Chinese Remainder Theorem. Instead of working modulo \( N \), the signer Bob first computes the signatures modulo \( p \) and \( q \) and then combines the result using the Chinese Remainder Theorem. More accurately, Bob first computes

\[
C_p = M^{d_p} \mod p \quad \text{and} \quad C_q = M^{d_q} \mod q
\]

Where \( d_p = d \mod (p -1) \) and \( d_q = d \mod (q -1) \).

then \( C = T_1 C_p + T_2 C_q \mod N \)

where

\[
T_1 = \{ \begin{array}{ll} 1 \mod p \\ 0 \mod q \end{array} \} \quad \text{and} \quad T_2 = \{ \begin{array}{ll} 0 \mod p \\ 1 \mod q \end{array} \}
\]

The running time of the last CRT step is negligible compared to the two exponentiations. Note that \( p \) and \( q \) are half the length of \( N \). Since simple implementations of multiplication take quadratic time, multiplication modulo \( p \) is four times faster than modulo \( N \). Furthermore, \( d_p \) is
half the length of \(d\) and consequently computing \(M^{dp}\) mod \(p\) is eight times faster than computing \(M^{dp}\) mod \(N\). Overall signature time is thus reduced by a factor of four. Many RSA implementations use this method to improve performance.

6 Conclusion

Twenty years of research aimed to break the RSA produced some insightful attacks, but no serious attack has been found yet. Currently, it appears that proper RSA implementation can provide the required security in the digital world. Four main classes of RSA attacks were found: (1) elementary attacks that show the misuse of the system, (2) low private exponent to show how serious it gets when a low private is used, (3) low public exponent attacks, and (4) attacks on the RSA implementation.

Proper use of RSA and properly padding a message before encryption can defeat the explained attacks.
7 References

[1] D. Boneh, Twenty Years of Attacks on the RSA Cryptosystem