Volume 4, No. 3, March 2013 (Special Issue)

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Enhanced RSA Digital Signature Algorithm

Ashish Vijay
M.Tech. Student
SBCET, Jaipur, Rajasthan, India
ashishvijay2007@gmail.com

Priyanka Trikha
CSE Department
SBCET, Jaipur, Rajasthan, India
trikhapriyanka@gmail.com

Ashik Hussain
M.Tech Scholar
Govt. Engineering College
ashik.hussain07@gmail.com

Abstract: Generally, digital signature algorithms are based on a single hard problem like prime factorization problem or discrete logarithm problem or elliptic curve problem. If one finds solution of this single hard problem then these digital signature algorithms will no longer be secured and due to large computational power, this may be possible in future. The RSA digital signature algorithm (RSADSA) is an asymmetric cryptographic technique, whose security is related to the difficulty of factorization. But if one would solve the factorization problem then he would get the private key too. RSADSA is not only vulnerable to the prime factorization attacks but also to the small private exponent d and small public exponent e attacks. So to improve security, this paper presents a new variant of digital signature algorithm which is based on two hard problems, prime factorization and xth root problem. The proposed algorithm is a modification of the RSA digital signature algorithm.

Keywords: xth root; RSA; Factorization; Digital Signature; Cryptanalysis

I. INTRODUCTION

A digital signature algorithm is mathematical scheme which provides authenticity of a digital message and assure the recipient that the message was created by an authorized sender and was not modified in transit. RSA Digital Signature Algorithm (RSADSA) [26] proposed by Rivest, Shamir and Adleman, is a popular and well known digital signature algorithm. The RSA signature algorithm is unforgivable in the random oracle model assuming the integer factorization problem is unsolvable. This scheme is based to the RSA public key cryptosystem [26]. However if, in near future, one can solve the integer factorization problem, then he can also forge the signature created by this scheme. In this paper, a new variant of digital signature algorithm is proposed, called Enhanced RSA Digital Signature Algorithm (ERSADSA), which is based on the combined difficulties of integer factorization problem and xth root problem. The proposed algorithm is actually a modification of RSADSA algorithm.

In past, many efforts have been done for developing multiple hard problems based digital signature algorithms. In modern cryptography [7], the security of digital signature algorithms is based on the difficulty of solving some hard number theoretical problems.

These algorithms stay secure as long as the problem, on which the algorithm is based, stays unsolvable. The most used hard problems for designing a signature algorithm are prime factorization (FAC) [26] and Discrete Logarithm (DL) [8] problems. For improving the security, the algorithms may be designed based on multiple hard problems. Undoubtedly, the security of such algorithms is longer than algorithms based on a single problem. This is due to the need of solving both the problems simultaneously. Many digital signature algorithm have been designed based on both FAC and DL [9, 11, 12, 14, 17, 19, 25, 27, 29, 30] but to design such algorithms is not an easy task since many of them have been shown to be insecure [10, 18, 19, 20, 21, 28, 29, 30].

Rest of the paper is organized as follows. Section 2 describes brief overview of RSA digital signature algorithm. Section 3 contains the proposed algorithm. In Section 4, analysis is carried out for the proposed algorithm. Finally, in Section 5, paper is concluded.

II. BRIEF OVERVIEW OF RSA DIGITAL SIGNATURE ALGORITHM

RSADSA is an asymmetric digital signature algorithm as it uses a pair of keys, one of which is used to sign the data in such a way that it can only be verified with the other key. RSADSA is based on one way trap-door function. In case of RSADSA, the idea is that it is relatively easy to multiply prime numbers but much more difficult to factor. Multiplication can be computed in polynomial time where as factoring time can grow exponentially proportional to the size of the numbers. The algorithm is as follows:

2.1. Key Generation:

Followings are the key generation steps:

- Generate two large random primes, p and q.
- Compute $n = p \times q$ and $\phi = (p-1)\times(q-1)$
- Choose an integer e, satisfying $1 < e < \phi$, such that gcd $(e, \phi) = 1$
- Compute the secret exponent d, satisfying $1 < d < \phi$, such that $e \times d \bmod \phi = 1$.
- The public key is e and the private key is d. By using these keys, signature generation and signature verification are performed.

2.2 Signature Generation:

Followings are the signature generation steps:

- Creates a message digest H (m) as an integer of the information to be sent between 0 and $n - 1$.

- Compute the signature by using the private key d as $s = H(m)^d \bmod n$
- s is the signature of the message m. Send s with the message m to recipient.

2.3 Signature Verification:

Signature verification steps are as follows:

- By using sender public key e, compute integer $v = s^e \bmod n$. v be the message digests calculated by sender.
- Independently computes the message digest of the message that has been signed.
- If both message digests are identical, the signature is valid.

Security of RSADSA algorithm is based on difficulty of solving the prime factorization problem. There are many efforts have been done in past to solve the prime factorization problem [13, 22, 23, 24]. In 2002, Weger [6] described a new attack for solving prime factorization problem as if there is small difference between the prime factors of modulus then a polynomial time cryptanalysis for factoring modulus is possible. In 2003, Boneh and Brumley [2] demonstrated a more practical attack capable of recovering RSA factorizations over a network connection. This attack takes advantage of information leaked by the Chinese remainder theorem optimization used by many RSA implementations. RSADSA is not only vulnerable to the prime factorization attacks but also to the private key d. Paul Kocher[16] described that if an Adversary Eve knows Alice's hardware in sufficient detail and is able to measure the decryption times for several known cipher texts, she can deduce the decryption key d quickly. Next, there are many threats if the RSA private exponent is chosen small. The first significant attack on small private exponent RSA was Wieners continued fraction attack [31]. Given only the public key (e, n), the attack factors the modulus using information obtained from one of the convergent in the continued fraction expansion of e/n. It was shown by Coppersmith [13], that an RSA modulus with balanced primes could be factored given only 1/2 of the most significant bits of one of the primes. It was later shown by Boneh, Durfee and Frankel [3] that 1/2 of the least significant bits of one of the primes was also sufficient. A theoretical hardware device named TWIRL designed by Shamir and Tromer in 2003 [15], questioned the security of 1024 bit keys. Now days due to the availability of high end resources of computation the chances of the various types of attacks have increased. It is quite possible that an organization with sufficiently deep pocket can build a large scale version of his circuits and effectively cracks an RSA 1024 bit message in a relatively short period of time.

In search of the digital signature algorithms that are vulnerable to above mentioned attacks, subsequent section contains a new variant of digital signature algorithm.

III. THE PROPOSED SIGNATURE ALGORITHM

This section proposes a new variant of digital signature algorithm based on the two NP-Complete problems named prime factorization and xth root. The algorithm is as follows:

3.1. Key Generation:

Followings are the key generation steps:

- Choose two large prime numbers p and q and calculate $n = p \times q$.
- Calculate $\phi(n) = (p-1)\times(q-1)$ and Choose e such that gcd $(e, \phi(n)) = 1$.
- Calculate d such that $d \times e \bmod \phi(n) = 1$.
- Choose random numbers b and x. Here x should not relative prime to $\phi(n)$
- Calculate c such that $b^x \times c (\bmod) n = 1$.
- public key is (n, e, c, x) and private key is (d, b).

3.2. Signature Generation:

Followings are the signature generation steps:

- Calculate $s_1 = H(m)^d \bmod n.$
- if x|s$_1$ (i.e. x is a divisor of s$_1$) then generate s$_1$ again.
- Calculate- $s_2 = (H(m) \times b^{s_1}) \bmod n.$

H (.) is a one way hash function. (s$_1$, s$_2$) is the signature of message m. Sender sends signature with the message m to receiver.

3.3. Signature Verification:

Receiver first calculates H (m) using the received message m and check the following two conditions for signature verification:

Verify, if $H(m) = s_1^e \bmod n.$ (1)

and

$$H(m)^x \equiv s_2^x \times c^{s_1} \bmod n. \quad (2)$$

then the signature is valid else reject the signature.

3.4. Proof of correctness

This section contains the correctness proof of the proposed digital signature algorithm. First condition (equation No. 1) is a verification of RSA algorithm and proof of second condition (equation No. 2) is as follows.

R.H.S

$$= (s_2^x \times c^{s_1}) \bmod n$$
$$= (H(m) \times b^{s_1})^x \times c^{s_1} \bmod n$$
$$= H(m)^x \times b^{xs_1} \times c^{s_1} \bmod n$$
$$= H(m)^x \bmod n$$
$$= L.H.S.$$

IV. ANALYSIS OF THE PROPOSED ALGORITHM

This section contains analysis of the proposed algorithm in terms of security and performance.

4.1. Security Analysis

There are some possible areas where an adversary (Adv) can try to attack on this newly developed signature algorithm. Following are the possible attacks (not exhaustive):

Key-Only Attack: Adv wishes to obtain private key (b, d) using all information that is available from the system. For, Adv needs to solve the prime factorization problem to find p and q from modulus n which will provide Adv the value of d. next for finding b, Adv has to solve $b = c^{-1/x}$ which is NP-Complete for large b because Adv has to find xth root of c^{-1}. Let us call this problem xth root problem. xth root problem differs from the factorization problem of RSA. As in RSA, if prime factors of n are known then with the help of public key e, dth root of message (where d is a private key) could be determined. But in the xth root problem, x is not relative prime to $\phi(n)$, so its multiplicative inverse with respect to $\bmod \phi(n)$ does not exist. Therefore an Adv has to solve xth root problem and a FAC problem simultaneously for finding the complete private key. This fact makes the proposed algorithm secure enough for this type of attacks.

Known Partial Key Attack: Let us assume that Adv can solve the FAC. In this case, to forge the signature, Adv has to solve xth root problem which will provide him the values of b. Now if Adv can solve xth root but not FAC problem then to forge the signature, Adv require value of d and for that he has to solve factorization problem. Hence, the Adv cannot sign its own message using sender's signature even if he knows the part of the secret key by breaking the FAC or xth root problem. He has to solve both the problem simultaneously.

Blinding: In this attack, in case of RSADSA suppose Adv wants sender's signature on his message m. For this Adv try the following: he picks a random $r \in Z_n^*$ and calculates $m' = r^e \times m \bmod n$. He then asks sender to sign the message m'. Sender may provide his signature s' on the message m'. But we know that $s' = (m')^d \bmod n$ .Adv now computes $s = s'/r \bmod n$ and obtains sender's signature s on the original m. This technique, called blinding, enables Adv to obtain a valid signature on a message of his choice by asking Sender to sign a random blinded message. Sender has no information as to what message he is actually signing. So, RSA is vulnerable to this attack. Unlike RSA, ERSADSA is not vulnerable to Blinding and it is explained as follows: By using a random number r, Adv gets Sender's signature (s_1', s_2') for his message m, where $s_1' = (r^e \times m)^d \bmod n = r \times m^d \bmod n$ and $s_2' = r^e \times m \times b^{s_1'} \bmod n$. Adv now gets the signature (s1,s2) for message m by computing $s_1 = s_1'/r$ and $s_2 = s_2'/r$. But s_2' is related to s_1', so signature s2 will not be equivalent to the original signature of message m. For finding s2, Adv has to calculate value of b and for that he has to solve $(s_1')^{th}$ root problem as follows:

$$s_2' = r^e \times m \times b^{s_1'}$$

$$b^{s_1'} = \frac{s_2'}{m \times r^e}$$

$$b = \left(\frac{s_2'}{m \times r^e}\right)^{\frac{1}{s_1'}}$$

Hence ERSADSA is secure against this attack.

Chosen-Message Attack: In this attack, Adv required to get signed some messages of his choice by the authorized signatory. With the help of chosen messages and corresponding signatures, Adv generates another message and can forge sender's signature on it. The RSADSA algorithm is forgeable for this attack. For attack on RSADSA, suppose, Adv asks signer to sign two legitimate messages m1 and m2 for him. Let us assume s1 and s2 are signatures of m1 and m2 respectively. Adv later creates a new message m = m1 × m2 with signature $s = s_1 \times s_2$. Adv can then claim that signer has signed m. ERSADSA is also vulnerable to this type of attack. Suppose (s1, s2) and (s3, s4) are digital signatures of messages m1 and m2 respectively. Then Adv can generate signature for his message m3 = m1 × m2. For this Adv generates signature (t1, t2) of message m3 as follows

$$t_1 = s_1 \times s_3$$

$$t_2 = s_2 \times s_4 \times \left(\frac{s_2}{m_1}\right)^{s_3} \times \frac{m_1}{s_2} \times \frac{m_2}{s_4}$$

$$\begin{aligned} t_1 &= s_1 \times s_3 \\ &= m_1^d \bmod n \times m_2^d \bmod n \\ &= (m_1 \times m_2)^d \bmod n \\ &= (m_3)^d \bmod n \end{aligned}$$

$$\begin{aligned} t_2 &= s_2 \times s_4 \times \left(\frac{s_2}{m_1}\right)^{s_3} \times \frac{m_1}{s_2} \times \frac{m_2}{s_4} \\ &= m_1 \times b^{s_1} \times m_2 \times b^{s_3} \times \left(\frac{m_1 \times b^{s_1}}{m_1}\right)^{s_3} \times \frac{m_1}{m_1 \times b^{s_1}} \times \frac{m_2}{m_2 \times b^{s_3}} \bmod n \\ &= (m_1 \times m_2) \times b^{s_1 \times s_3} \bmod n \\ &= m_3 \times b^{t_1} \bmod n. \end{aligned}$$

The signature (t1, t2) for m3 so obtained is actually a valid signature of Sender, because:

Hence, ERSADSA is not secured against chosen message attack.

4.2. Performance Analysis

a) Changing the Modulus Length

Changing the modulus affects the other parameters of the algorithms as shown in Table 4.1. It is clear here that increasing the modulus length (bits) increases the bit length of their factors and so the difficulty of factoring them into their prime factors arises. Moreover, the length of the secret key (d) increases at the same rate as n-bit increases. As a result, increasing the n-bit length provides more security. On other hand by increasing the n-bit length increases the values of key generation time, signature generation time and signature verification time. Hence increasing the n- bit length increases the security but decreases the speed of key generation, signature generation and signature verification process as illustrated by Figure 4.1

TABLE 4.1

Effect of changing the modulus lenght on the key generation, signature generation and signature verification time with constant public key size (512 bit) and message digest size (64 bit)

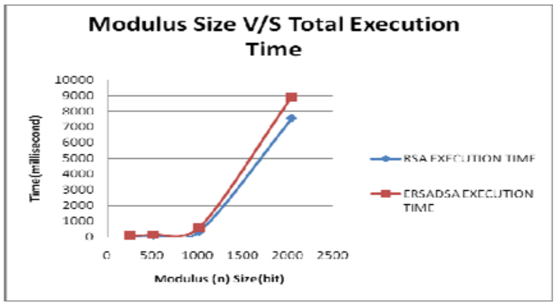

Fig.4.1. Diagram showing modulus size v/s RSADSA & ERSADSA Algorithm's execution time, with constant public key size 512 bit and message digest 64 bit.

b) Changing the Public Key Size

On the basis of simulation results of Table 4.2, following Figure 4.2 shows the effect of public key size on key generation, signature generation and signature verification time of both the algorithms. Here key generation, signature generation and signature verification time of both algorithms depends on public key size and as the public key size increases key generation, signature generation and signature verification time also increases.

e Size	RSADSA Execution Time(ms)				ERSADSA Execution Time(ms)			
	Key Generation Time(ms) (a)	Signature Generation Time(ms) (b)	Signature verification time(ms) (c)	Total Execution time(ms) (a +b +c)	Key Generation Time(ms) (A)	Signature Generation Time(ms) (B)	Signature verification time(ms) (C)	Total Execution time(ms) (A+B+C)
64	1828	344	16	2188	1921	688	1047	3656
128	2218	344	15	2577	2312	687	1047	4046
256	5688	344	47	6079	5781	688	1031	7500
512	5703	344	94	6141	5796	688	1031	7515

TABLE 4.2

Effect of changing the public key size on key generation, signature generation, signature verification time with constant modulus lenght size (2048 bit) and message digest size (64 bit)

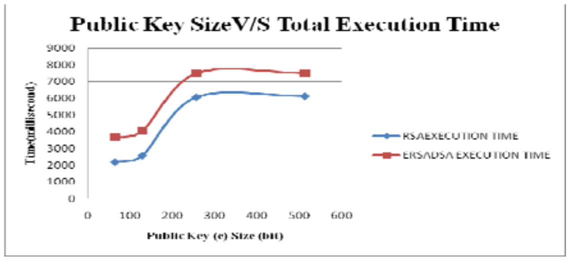

Fig. 4.2: Diagram showing public key size v/s RSADSA & ERSADSA Algorithm's execution time, with constant modulus length 2048 bit and message digest 64 bit.

Using the criterion presented in [4], the complexity of each method is estimated as a function of number of bit operations required. The basic exponential operation here is $a^b \bmod n$ and time complexity of this operation is $O(\log b \times M(n))$, where $M(n)$ the complexity of multiplying two n bit integers. In the proposed algorithm signature generation requires 2 modular exponentiation and signature verification requires 4 modular exponentiation which leads to the complexity of the algorithm to be $O(2\times\log^3 n)$ and $O(4\times\log^3 n)$ for signature generation and verification respectively as here $b = O(n)$ and time complexity of multiplying two n bit integers is $O(\log^2 n)$. Therefore the overall complexity for signature generation and verification is $O(\log^3 n)$. if the complexity of proposed DSA is compared with other DSA algorithms of same category (i.e. DSA algorithms that are based on multiple hard problems) then we see that the Dimitrios Poulakis signature algorithm [25] requires 6modular exponentiation in signature generation and 2modular exponentiation in signature verification. Ismail E. S signature algorithm [14] requires 5modular exponentiation in signature generation and 5modular exponentiation in signature verification. ShiminWei signature algorithm [30] requires 5 modular exponentiation in signature generation and modular exponentiation in signature verification. So it is clear that the complexity of the proposed algorithm is equivalent to most of the digital signature algorithms which are based on prime factorization and discrete logarithm and that is $O(\log^3 n)$.

As a future research perspective, the proposed algorithm can be improved for following reason: There are many organizations where signature generations arise frequently than signature verification at a single station like banks etc. For example, a Bank can generate many digital signatures in a day (in cash voucher, receipts), while the bank customer that receives this signature, has usually a much smaller load. So, the performance of signature generation process of this algorithm can be improved by using M-Prime RSA [1, 5], in place of RSA, with xth root problem.

V. Conclusion

In this paper, a new variant of digital signature algorithm is

n Size	d Size	RSADSA Execution Time(ms)				ERSADSA Execution Time(ms)			
		Key Generation Time(ms) (a)	Signature Generation Time(ms) (b)	Signature verification time(ms)(c)	Total Execution time(ms) (a+b+c)	Key Generation Time(ms) (A)	Signature Generation Time(ms) (B)	Signature verification time(ms)(C)	Total Execution time(ms) (A+B+C)
256	2048	47	15	15	77	47	16	16	79
512	2048	94	16	15	125	110	16	47	173
1024	2048	297	47	16	360	328	94	187	609
2048	2048	7125	344	94	7563	7218	688	1031	8937

proposed which is based on the two hard problems called prime factorization and xth root. It is shown that the new algorithm is secure enough against various attacks and one have to solve both the problems for cryptanalysis of the proposed algorithm. However, the proposed algorithm is not secure against Chosen-message attack like RSADSA. The performance of the proposed algorithm is comparatively equivalent to the most of the digital signature algorithms which are based on multiple hard problems.

VI. References

[1] C. Alison and C.A.M. Paixao, “An efficient variant of the RSA cryptosystem 2003”.

[2] D. Boneh, G. Durfee, and Y. Frankel. “Exposing an RSA private key given a small fraction of its bits”, Full version of the work from Asia crypt, 98, 1998.

[3] D. Boneh and H. Shacham, “Fast variants of RSA”, CryptoBytes (RSA Laboratories), 5:1-9, 2002.

[4] T. Collins, D. Hopkins, S. Langford, and M. Sabin, “Public key cryptographic apparatus and method”, October 7 2008, US Patent RE40,530.

[5] B. De Weger, “Cryptanalysis of RSA with small prime difference”, Applicable Algebra in Engineering, Communication and Computing, 13(1):17-28, 2002.

[6] W. Diffie and M. Hellman, “New directions in cryptography”, Information Theory, IEEE Transactions on, 22(6):644-654, 2002.

[7] W. H. He, “Digital signature scheme based on factoring and discrete logarithms”, Electronics Letters, 37(4):220-222, 2002.

[8] ES Ismail, NMF Tahat, and RR Ahmad, “A New Digital Signature Scheme Based on Factoring and Discrete Logarithms”, Journal of Mathematics and Statistics, 4(4):222-225, 2008.

[9] P. Kocher, “Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems”, In Advances in Cryptology CRYPTO96, pages 104-113, Springer, 1996.

[10] C.S. Laih and W.C. Kuo, “New signature schemes based on factoring and discrete logarithms”, IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, 80(1):46-53, 1997.

[11] NY Lee, “Security of Shao’s signature schemes based on factoring and discrete logarithms”, In Computers and Digital Techniques, IEE Proceedings-, volume 146, pages 119-121, IET, 2002.

[12] N.Y. Lee and T. Hwang, “Modified Harn signature scheme based on factorizing and discrete logarithms”, In Computers and Digital Techniques ,IEEE Proceedings-,volume143,pages196-198.IET,2002.

[13] N.Y. Lee and T. Hwang, “The security of He and Kiesler’s signature schemes”, In Computers and Digital Techniques, IEE Proceedings-, volume 142, pages 370-372,IET, 2002.

[14] D. Poulakis, “A variant of Digital Signature Algorithm”, Designs, Codes and Cryptography, 51(1):99-104, 2009.

[15] R.L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signature sand public-key cryptosystems”, Communications of the ACM, 21(2):120-126, 1978.

[16] Z. Shao, “Signature schemes based on factoring and discrete logarithms”, In Computers and Digital Techniques, IEE Proceedings-, volume 145, pages 33-36, IET, 2002.

[17] Z. Shao, “Security of a new digital signature scheme based on factoring and discrete logarithms”, International Journal of Computer Mathematics, 82(10):1215-1219, 2005.

[18] S.F. Tzeng, C.Y. Yang and M.S. Hwang, “A new digital signature scheme based on factoring and discrete logarithms”, International Journal of Computer Mathematics, 81(1):9-14, 2004.

[19] S.Wei, “A New Digital Signature Scheme Based on Factoring and Discrete Logarithms”, Progress on Cryptography, pages 107-111, 2004.

[20] M.J. Wiener, “Cryptanalysis of short RSA secret exponents”, Information Theory, IEEE Transactions on, 36(3):553-558, 2002.

AUTHOR’S BIOGRAPHY

Ashish Vijay was born on 22 September 1985.He is the M.Tech. Student in S.B.C.E.T, Jaipur (Rajasthan). He has completed B.E. (CE) in 2007 from University of Rajasthan, Jaipur.

Priyanka Trikha has completed her BTech and MTech degree in Computer Science and Engineering from Faculty of Engineering and Technology, MITS, Lakshmangarh, Sikar, Rajasthan.Now She is assistant professor in comp. sc. Dept. of SBCET, Jaipur**.**