An Efficient DSA Approach for Batch Verification

Khaldun Ibraheem Arif
Computer Science Dept.
College of Education
Univ. of Thi-Qar

Abstract

Batch verification is the ability to verify multiple digital signatures together. This method can reduce computational cost required to verify these signatures individually. In this paper, we present a fast algorithm based on DSA-type algorithm. Our algorithm gives the same level of security as compared with Bellare et al. algorithm and at the same time it is more efficient since it does not require computing of modular inverse neither at the signing side nor at the verifying side. The proposed algorithm need to compute modular inverse at generating public key and this is done once.

Keywords: Digital Signature Algorithm; Batch Verification; Modular Inverse.

1. Introduction

In 1991, the Digital Signature Standard(DSS) proposed by the US government as a federal standard to allow federal government agencies to use the Digital Signature Algorithm(DSA) to sign electronic documents[1]. The DSA algorithm is of type ElGamal signature scheme and the difficulty of breaking the scheme is based on solving the discrete logarithm problem[6]. Verifying process of ElGamal –type signature scheme is slower than signing process since it uses more modular exponentiations. Since modular exponentiation is a computationally intensive operation, we need an efficient algorithm to speed up verification process. To satisfy this purpose, Naccache et al. proposed the notion of batch verifying multiple signatures and constructed an interactive DSA-type batch verifying algorithm[2]. Lim and Lee[3] pointed out that this scheme is insecure since any attacker can easily forge signatures with satisfying the batch verification criterion. For this reason, Bellare et al.[4,5] described a high confidential technique which does not allow false values to be mixed into the batch. This technique is called Small Exponent Test.

In this paper, an efficient and secure algorithm has been presented. The proposed algorithm uses Small Exponent Test as in Bellare et al.'s scheme to keep security level but, in comparison, our scheme is more efficient since it does not need to compute modular inverse neither at the signing side nor verifying side. Only we need to compute modular inverse to the secret key in order to generate public key and this process is done once.

The rest of this paper is organized as follows. We review Bellare et al.'s batch verifying algorithm in section 2. Then section 3 shows the proposed algorithm. Section 4 gives complexity analysis of both Bellare et al.'s scheme and ours. In section 5, we discuss conclusion.

2. Bellare et al. Batch Verifying Algorithm

In [4,5], Bellare et al. proposed a secure batch verifying algorithm for multiple signatures. This algorithm is proven to be secure through using a technique called Small Exponents Test in order to prohibit introducing forged signatures into batch. Before giving the algorithm, it is necessary to list the DSA-type algorithm parameters firstly.

p is a large prime.

q is a prime divisor of p-1.

g is an element of order q in GF(P); i.e, $g=h^{(p-1)/q} \pmod p$, where h is any integer with $1 < h < p$ such that $g=h^{(p-1)/q} \pmod p > 1$.

x is the secret key of the signer in GF(q).

y is the public key of the signer such that $y = g^x \bmod p$.

Assume that there are n signatures $(r_1,s_1),(r_2,s_2),\ldots,(r_n,s_n)$ of n different messages $m_1,m_2,\ldots,m_n$ respectively. The signing and verifying processes are shown below.

2.1.Signature generation

The signer can generate each pair of the signature (r_i,s_i), where $i=1,2,\ldots,n$ as follows:

Step1. Selects a random integer k_i

Step2. Computes $r_i=g^{k_i} \bmod p$

Step3. Computes $s_i=k_i^{-1}(m_i+xr_i) \bmod q$

2.2.Batch Verification

The verifier can verify multiple signatures as follows:

Step1. Chooses ℓ -bit random number b_i, where i=1,2,…,n

Step2. Verifies the following Batch Verification equation:

$$\prod_{i=1}^{n} r_i^{b_i} \equiv^{?} g^{\sum_{i=1}^{n} m_i s_i^{-1} b_i \bmod q} \, y^{\sum_{i=1}^{n} r_i s_i^{-1} b_i \bmod q} \pmod p$$

If the equation holds, the signatures are accepted. Otherwise, they are rejected.

3. The Proposed Algorithm

A fast and secure algorithm is reviewed in this section. This algorithm includes the same level of security as that of Bellare et al.'s algorithm. On the other hand, the proposed algorithm is more efficient since it does not need to compute modular inverse neither at the signing nor at the verifying processes. Our scheme uses the same parameters mentioned previously except for computing public key which is computed as follows:

$$y = g^{x^{-1}}$$

3.1.Signature generation

The signer can generate each signature pair (r_i,s_i), where i=1,2,…,n by the following steps:

Step1. Chooses integer k_i randomly

Step2. Computes $r_i=g^{k_i} \bmod p$

Step3. Computes $s_i=x(m_ik_i-r_i) \bmod q$

If (s_i=0) goto step 1.

3.2.Batch Verification

After receiving multiple signatures $(r_1,s_1),(r_2,s_2),\ldots,(r_n,s_n)$, the verifier can verify the signature pairs according to the following steps:

Step1. Picks $b_1,b_2,\ldots,b_n \in \{0,1\}^{\ell}$ randomly

Step2. Checks the following Batch Verification equation:

$$\prod_{i=1}^{n} r_i^{m_i b_i \bmod q} \equiv^{?} y^{\sum_{i=1}^{n} s_i b_i \bmod q} \, g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

If the equation holds, the verifier accepts the signatures; otherwise, rejects.

Here, we prove that the two parties of the above Batch Verification equation are equivalent.

$$\prod_{i=1}^{n} r_i^{m_i b_i \bmod q} = y^{\sum_{i=1}^{n} s_i b_i \bmod q} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{x^{-1}(\sum_{i=1}^{n} s_i b_i \bmod q)} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{(\sum_{i=1}^{n} x^{-1} s_i b_i \bmod q)} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{(\sum_{i=1}^{n} x^{-1}(x(miki-ri) \bmod q) b_i \bmod q)} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{(\sum_{i=1}^{n} ((miki-ri)\, b_i \bmod q)} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{\sum_{i=1}^{n} (m_i k_i b_i - r_i b_i) \bmod q} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$

$$= g^{\sum_{i=1}^{n}(m_i k_i b_i) \bmod q - \sum_{i=1}^{n} r_i bi \bmod q} g^{\sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$
$$= g^{\sum_{i=1}^{n}(m_i k_i b_i) \bmod q - \sum_{i=1}^{n} r_i bi \bmod q + \sum_{i=1}^{n} r_i b_i \bmod q} \pmod p$$
$$= g^{\sum_{i=1}^{n}(m_i k_i b_i) \bmod q} \pmod p$$
$$= g^{\sum_{i=1}^{n} k_i (m_i b_i) \bmod q} \pmod p$$
$$= \prod_{i=1}^{n} r_i^{m_i b_i \bmod q} \pmod p$$

4. Complexity Analysis

Now, we compare our scheme with Bellare et al.'s scheme. In signature generation side, Bellare et al.'s scheme requires (n) modular exponentiations to compute r_i (for i=1,…,n) and (2n) modular multiplication to compute s_i (for i=1,…,n). furthermore, it requires (n) modular inverse to compute k_i^{-1} (for i=1,…,n). on the other hand, our scheme is similar to above except that it does not need to compute modular inverse. In batch verification side, Bellare et al.'s scheme [5] needs (ℓ_b+n ℓ_b/2) modular multiplications, where ℓ_b denotes bit number of small exponents test, to compute ($\prod_{i=1}^{n} r_i^{b_i}$) with small exponents. In addition, it requires (4n) modular multiplication to compute ($\sum_{i=1}^{n} m_i s_i^{-1} b_i \bmod q$) and ($\sum_{i=1}^{n} r_i s_i^{-1} b_i \bmod q$). In total, it takes ($\ell_b$+n(4+ ℓ_b/2)) modular multiplications, (2) modular exponentiations, and (n) modular inverses to complete batch verification. On the other hand, our scheme takes some more modular multiplications (ℓ_q+n ℓ_q/2) modular multiplications, where ℓ_q denotes bit number of prime q, to compute ($\prod_{i=1}^{n} r_i^{m_i b_i \bmod q}$) with small exponents. In addition it takes (3n) modular multiplications to compute ($m_i b_i \bmod q$; for i=1,…,n) and ($\sum_{i=1}^{n} s_i b_i \bmod q$) and ($\sum_{i=1}^{n} r_i b_i \bmod q$). In total, our proposed scheme takes (ℓ_q+n(3+ℓ_q/2)) modular multiplications and two modular exponentiations to complete batch verification. Moreover, it needs none of modular inverse. The complexity comparison between Bellare et al.'s scheme and ours is showed in table 1.

The scheme	Exponentiation	Multiplication	Inversion
Bellare et al.			
Signing	n	2n	n
Verifying	2	ℓ_b+n(4+ ℓ_b/2)	n
Ours			
Signing	n	2n	0
Verifying	2	ℓ_q+n(3+ ℓ_q/2)	0

Table 1: The complexity comparison between Bellare et al.'s scheme and ours.

In addition to the complexity comparison, we give practical results (computed in milliseconds) of each scheme using PC has Intel Core 2 Duo processor 1.60 GHz and 1 GB DDR2 RAM, as shown in table 2. Since the computational complexity of each modular inverse computation is almost equivalent to a modular exponentiation[6], our scheme is more efficient than Bellare et al. scheme at the signature generation side and at the batch verification side.

No. of Signatures	Bellare scheme		The proposed scheme	
	Signing	Verifying	Signing	Verifying
10000	141	171	63	78
15000	219	250	78	125
20000	297	344	94	156
25000	360	422	140	188
30000	437	516	156	234
35000	516	594	172	281
40000	594	672	203	312
45000	688	765	235	343
50000	734	828	266	391
55000	812	938	281	422

Table 2: The practical results measured in milliseconds.

5. Conclusion

In this paper, we have presented an efficient DSA-type scheme for Batch Verification. Our scheme is efficient since it does not need to compute modular inverse, which is time consuming, at the signing and verifying processes. In our scheme, public key (y) of a signer is obtained by computing modular inverse of the corresponding secret key (x) and this operation is computed one time only.

References

[1] NIST, "Digital Signature Standard", FIPS PUB 56 (169) (1991).

[2] D. Naccache, D. M'Raihi, D. Rapheali, S. Vandenay: Can DSA Be Improved: Complexity Tradeoffs With The Digital Signature Standard, Proceedings of Advances in Cryptology-EURO-CRYPT '94, LNCS 950,1995, PP. 77-85.

[3] C.H. Lim, P.J. Lee, Security of Interactive DSA Batch Verification, Electronics Letters 30 (19) (1994) 1592-1593.

[4] C.H. Lin, R.H. Hsu, L. Harn, Improved DSA Variant for Batch Verification, ELSEVIER, Applied Mathematics and Computation 169 (2005) 75-81.
[5] M. Bellare, J. A. Garay, T. Rabin, Fast Batch Verification for Modular Exponentiation and Digital Signatures, Proceedings of Advances in Cryptology- EUROCRYPT '98, LNCS 1403,1998, pp.236-250.
[6] C.A. Jan, D.L. Van, Basics Methods of Cryptography, Faculty of Information Technology and Systems, Delft University, 1998.

ملخص البحث

التحقق من الدفعه (Batch verification) هو القدره على التحقق من عدة تواقيع رقميه معا. هذه الطريقه يمكن ان تقلل الكلفه الحسابيه المطلوبه للتحقق من هذه التواقيع منفرده. في هذا البحث، نقدم خوارزميه سريعه تعتمد على الخوارزميه من النوع(DSA). الخوارزميه المقترحه تعطي نفس مستوى الامنيه بالمقارنه مع خوارزميه(Bellare et al.) وفي نفس الوقت فانها اكثر كفاءه لانها لا تحتاج لحساب المعكوس(modular inverse) لا في جانب توليد التواقيع ولا في جانب التحقق منها. هذه الخوارزميه تتطلب حساب المعكوس فقط عند توليد المفتاح المعلن(public key) وهذه العمليه تحسب لمره واحده.