Trends in Mathematics
Information Center for Mathematical Sciences
Volume 8, Number 1, June, 2005, Pages 167–175

CRYPTANALYSIS OF ELGAMAL TYPE DIGITAL SIGNATURE SCHEMES USING INTEGER DECOMPOSITION

IKKWON YIE

Abstract. For an ElGamal-type signature scheme using a generator g of order q, it has been well-known that the message nonce should be chosen randomly in the interval $[1, q-1]$ for each message to be signed. Two different approaches to attack ElGamal-type signature scheme have been studied in this regard. One is to focus on reducing the number of known bits for the nonces [5], and the other is to focus on reducing the number of signatures [2]. In this paper, we follow the second approach. In [2], H. Kuwakado and H. Tanaka proposed a polynomial time algorithm that gives the private key of the signer if two signatures with message nonces $0 < k_1, k_2 \leq O(\sqrt{q})$ are available. We employ the integer decomposition method suggested by R. Gallant, R. Lambert, and S. Vanstone (See [3]) to improve Kuwakado-Tanaka Algorithm. We improve the efficiency and completeness of their algorithm and achieve a rigorous complexity analysis.

1. Introduction

Since T. ElGamal proposed a signature scheme based on discrete logarithms at Crypto'84 Conference [1], many variants of his scheme have been developed and some of them became national standards for digital signature.

The security of ElGamal-type signature schemes is based on the hardness of DLP. Every ElGamal-type signature scheme uses a cryptographic hash function and produces randomized signatures via invoking a random nonce for each message. An extreme care is required in choosing the message nonce random because otherwise the weakness of the nonces will lead to a total break of the signature scheme.

In [2], H. Kuwakado and H. Tanaka proposed a polynomial time algorithm (which we will call Algorithm KT) that recovers the private key of the signer if two signatures with message nonces $0 < k_1, k_2 \leq O(\sqrt{q})$ are available, where q is the size of the multiplicative subgroup on which ElGamal signature scheme is built.

In this paper, we improve Algorithm KT so that it is more efficient, works even when the original algorithm fails to produce an answer (Note that in [2] the success rate of Algorithm KT gets smaller as the bound for the nonces grows). In [3], in order to accelerate the scalar multiplication in elliptic curves, Gallant *et al.* introduced a method of decomposing integers into a linear combination of certain

Key words and phrases. digital signature algorithm, nonce, extended Euclidean algorithm.

things with small coefficients. They used the extended Euclidean algorithm to find two short linearly independent vectors. We will apply their idea to find a suitable vector (γ_1, γ_2) which fits into Algorithm KT. We also generalize Algorithm KT to work in the case when two signatures with message nonces k_1, k_2 such that $|k_1|, |k_2| \leq O(\sqrt{q})$ are available. As a result, we see that Algorithm KT is a deterministic algorithm that always produces an answer as long as the assumptions are satisfied. Thus, we make Algorithm KT more complete, improve the efficiency and obtain precise running time complexity of the algorithm, which wasn't provided in [2].

Before we close this section, we would like to mention the paper of Nguyen and Shparlinski [5]. Nguyen and Shparlinski present a polynomial time algorithm that recovers the secret key of the signer when a few consecutive bits of the nonces k are known for a certain number of signatures. More precisely, their algorithm runs in polynomial time when approximately $(\log q)^{1/2}$ bits are known for a number of signatures linear in $\log q$. However, their focus is to reduce the number of known bits in the price of subexponential running time instead of polynomial time. And they made various experiments with only two or three known bits. Algorithm KT and our extension are on the other extreme, that is, we need to have very small nonces, say, bounded by $O(\sqrt{q})$, but only for two signatures. We also note that selecting signatures corresponding to small nonces k by timing or power attack is considered quite feasible. Thus this paper serves as an another precaution that when an ElGamal-type signature scheme is used, the message nonces should be chosen with urgent care.

2. ElGamal-type Signature Schemes

In this section, we briefly review ElGamal signature scheme and its variations.

A signature scheme consists of three algorithms, that is, the Key Generation Algorithm, the Signature Generation Algorithm, and the Signature Verification Algorithm. A pre-fixed hash function H is also used. The key generation algorithm generates the following system parameters.

- prime numbers p, q such that q divides $p-1$;
- an element g of the prime field $\mathrm{GF}(p)$ which generates a (multiplicative) cyclic group of order q;
- a random number x and $Y \equiv g^x \pmod{p}$.

And then x will be the signer's secret key, and p, q and Y will be made public.

For each message M to be signed, a random number k, which is called the (message) nonce, is to be selected in the interval $[1, q-1]$. Thus, the ElGamal signature is (M, r, s), where $r \equiv g^k \pmod{p}$ and $s \equiv (H(M) - xr)/k \pmod{q}$. The verification algorithm checks if $g^{H(M)} \equiv Y^r r^s \pmod{p}$ and accepts the signature as valid if it is so.

Most of variants of ElGamal signature uses the same system parameters and message nonce as above and alter the Signature Generation-Verification Algorithm.

In the verification process of plain ElGamal signature, essentially the equation

$$u \equiv xv + kw \pmod{q} \tag{1}$$

is verified. Note that the values $u = H(M)$, $v = r$, and $w = s$ are parts of the signature and hence are known to public once the signature is published while the private key x of the signer and the nonce k are kept secret from any adversary. Variations of ElGamal signature scheme use similar equations as this which are summarized in the Table 1 (See [4]).

scheme	signature	signing equation
ElGamal	$(M, (r, s))$	$H(M) \equiv xr + ks \pmod{q}$
DSA	$(M, (r, s))$	$H(M) \equiv x(-r) + ks \pmod{q}$
KCDSA	$(M, (r, s))$	$-(r \oplus (H(Y\|M) \mod q) \equiv xs - k \pmod{q}$
Schnorr	$(M, (r, s))$	$s \equiv xe + ks \pmod{q}$, where $e = H(M\|(g^k \mod p))$

TABLE 1. signing equations for ElGamal-type signature schemes

3. ALGORITHM KT

In this section, we briefly review the algorithm proposed by Kuwakado and Tanaka in [2]. Although their algorithm works equally well with the various variations of ElGamal signature, they display it specially for the original ElGamal signature for simplicity and we will adopt the same attitude.

One basic requirement for the sake of security is that the message nonce k should be chosen randomly from $1 \leq k \leq q - 1$. On the contrary, suppose we have signatures for two messages m_1 and m_2 with respective message nonces k_1 and k_2. Then from the equation (1) we have the following simultaneous equations:

$$u_1 \equiv xv_1 + k_1w_1 \pmod{q},$$
$$u_2 \equiv xv_2 + k_2w_2 \pmod{q}.$$

After eliminating x, we get an indeterminate congruence equation

$$u_1v_2 - u_2v_1 \equiv k_1(w_1v_2) + k_2(-v_1w_2) \pmod{q} \tag{2}$$

in k_1 and k_2.

Algorithm KT is to solve this equation and can be summarized as the following three steps:

Step 1: Find a vector $(\gamma_1, \gamma_2) \in \mathbb{Z} \times \mathbb{Z}$ that satisfies

- $\gamma_1 \equiv w(w_1v_2) \pmod{q}$ and $\gamma_2 \equiv w(-v_1w_2) \pmod{q}$ for some single number $w \in \mathbb{Z}$;
- $0 \leq \gamma_1, \gamma_2 \leq O(\sqrt{q})$;
- γ_1 and γ_2 are relatively prime.

Step 2: Let K_1, K_2 be integers such that $K_1\gamma_1+K_2\gamma_2 = 1$. Set $\gamma_3 \equiv w(u_1v_2-u_2v_1) \pmod q$.

Step3: Do the exhaustive search for k' such that $g^{k_1} \equiv g^{k'} \pmod p$ in the set

$$\mathcal{S} = \{\, (lq+\gamma_3)K_1 + \gamma_2 t \mid l, t \in \mathbb{Z}\}.$$

The reason why we search for k' in $\mathcal{S}$ is the following. Multiplying equation (2) by w, we get

$$\gamma_3 \equiv k_1\gamma_1 + k_2\gamma_2 \pmod q.$$

Thus we have

$$lq + \gamma_3 = k_1\gamma_1 + k_2\gamma_2 \tag{3}$$

for some integer l. Now by multiplying both sides by K_1 and rearranging terms, we have

$$k_1 = (lq+\gamma_3)K_1 + \gamma_2(k_1K_2 - k_2K_1).$$

Since the purpose of this algorithm is to find a single k_1, we may replace $k_1K_2-k_2K_1$ by t and go on searching for the correct t.

The checking condition

$$g^{k_1} \equiv g^{k'} \pmod p$$

in Step 3 may vary with signature schemes. As was mentioned earlier in this section, we considered only the case of plain ElGamal scheme. Once a single k' in Step 3 is found, we have

$$u_1 \equiv xv_1 + k'w_1 \pmod q.$$

Hence, the private key

$$x \equiv (u_1 - k'w_1)v_1^{-1} \pmod q$$

of the signer is compromised.

Kuwakado and Tanaka claimed in [2] that the complexity of their algorithm is of $O((\log p)^3)$ bit operations if $0 < k_1, k_2 \le O(\sqrt{q})$. The dominant part of Algorithm KT, in the complexity view point is the exhaustive search of Step 3. The complexity of Step 3 is determined by the number of trials for l, t. But this number of trials is intimately related to the size and the shape of the vector (γ_1, γ_2) found in Step 1. Thus, in order to obtain the claimed complexity analysis, it is crucial to find the vector (γ_1, γ_2) with explicit (upper and lower) bounds. However, Kuwakado and Tanaka considered only the upper bound and gave only the average size of this vector examined by a computer simulation instead of giving an explicit bound. As a result, the complexity required in Step 3 is not estimated rigorously.

4. An Improvement of Algorithm KT

In [2], H. Kuwakado and H. Tanaka used continued fraction to find the vector (γ_1, γ_2) in Step 1, provided a heuristic bound for the size of this vector but failed to give concrete bounds. Gallant *et al.* proposed in [3] a method of finding a short vector by using extended Euclidean algorithm. Their short vector is completely controlled in its size, namely, it is bounded by $\sqrt{q}$ and is very easy to add modification as necessary. In this section, by modifying their method, we improve the Step 1 of Algorithm KT and obtain an explicit bound for the size of the vector.

First, we briefly explain Gallant *et al.*'s method. Suppose a prime number q and a positive integer $\lambda < q$ are given. In the procedure of extended Euclidean algorithm to find the greatest common divisor of q and λ, we construct a sequence of equations,

$$s_i q + t_i \lambda = r_i, \quad i = 0, 1, 2, \cdots,$$

where $s_0 = 1, t_0 = 0, r_0 = q$ and $s_1 = 0, t_1 = 1, r_1 = \lambda$. Then s_i, t_i, r_i satisfy the following:

$$\begin{aligned} r_i > r_{i+1} &\geq 0, \quad i \geq 0\,; \\ |s_i| &< |s_{i+1}|, \quad i \geq 1\,; \\ |t_i| &< |t_{i+1}|, \quad i \geq 0\,; \\ r_{i-1}|t_i| + r_i|t_{i-1}| &= q, \quad i \geq 1. \end{aligned}$$

Since r_i is a decreasing sequence which eventually assumes the value 1, there exists a unique integer m such that

$$r_{m+1} \leq \sqrt{q} < r_m.$$

Moreover, from the relation $r_m|t_{m+1}| + r_{m+1}|t_m| = q$, we have

$$|t_{m+1}| < \sqrt{q}.$$

Therefore, we always have r_{m+1}, t_{m+1} which satisfy

$$s_{m+1}q + t_{m+1}\lambda = r_{m+1},$$

and $-\sqrt{q} < r_{m+1},\ t_{m+1} < \sqrt{q}$. As a summary, we have the following:

Lemma 1. (Gallant *et al.* [3]) For a given prime number q and a positive integer $\lambda < q$, there exist integers r, t such that

$$0 < r, |t| < \sqrt{q} \quad \text{and} \quad r - t\lambda \equiv 0 \pmod{q}.$$

By applying Lemma 1, we get the following improved version of Step 1 of Algorithm KT.

Theorem. Let q be a prime and $0 < x, y < q$ be given integers. Then there exist integers w, γ_1, γ_2 with $0 < \gamma_1, |\gamma_2| \leq \sqrt{q}$ such that

$$wx \equiv \gamma_1 \quad \text{and} \quad wy \equiv \gamma_2 \pmod{q},$$

where γ_1, γ_2 are relatively prime.

Proof. Since q is a prime, there are integers a_1 and a_2 such that $a_1x \equiv 1 \pmod{q}$ and $a_2y \equiv 1 \pmod{q}$. By applying Lemma 1 for q and $\lambda = (-a_2a_1^{-1} \bmod q)$, we get integers γ_1, γ_2 such that

$$\gamma_1 + \gamma_2\lambda \equiv 0 \pmod{q} \quad \text{and} \quad 0 < \gamma_1, |\gamma_2| < \sqrt{q}.$$

Then $w = (a_1\gamma_1 \mod q)$ $(= (a_2\gamma_2 \mod q))$ is what we wanted, since it satisfies that

$$wx \equiv \gamma_1 \quad \text{and} \quad wy \equiv \gamma_2 \pmod{q}.$$

Now suppose that γ_1 and γ_2 are not relatively prime. Then we may use $(\gamma_1', \gamma_2') = (\gamma_1/d, \gamma_2/d)$ instead, where $d = \gcd(\gamma_1, \gamma_2)$. Note that $\gamma_1' + \gamma_2'\lambda \equiv 0 \pmod{q}$. Therefore, if we take $w' \equiv a_1\gamma_1' \equiv a_2\gamma_2' \pmod{q}$, then we still have

$$w'x \equiv \gamma_1' \quad \text{and} \quad w'y \equiv \gamma_2' \pmod{q},$$

with smaller values γ_1', γ_2'.

5. The Complexity

Now we can find the vector (γ_1, γ_2) in Step 1 of Algorithm KT with an explicit bound $\sqrt{q}$. Thus we carry out the complexity analysis of Algorithm KT with rigor. Before we start the analysis, let us fix the notation once and for all.

Let p be a prime and g be an element of the prime field $\mathrm{GF}(p)$ of multiplicative order q. Assume that q is a prime. We also assume that ElGamal signature scheme is run on the subgroup generated by g.

The starting point is that we are given two signatures with message nonces k_1, k_2. Then we can derive an indeterminate congruence equation (2) from these signatures. By applying Theorem 1, we find a vector (γ_1, γ_2) in Step 1 of Algorithm KT with an explicit bound $\sqrt{q}$.

Let K_1, K_2 and γ_3 be the numbers obtained in Step 2. Then we have

$$k_1 = (lq + \gamma_3)K_1 + \gamma_2(k_1K_2 - k_2K_1)$$

for some integer l. Hence we can search for k_1 from the set

$$\mathcal{S} = \{\, (lq + \gamma_3)K_1 + \gamma_2 t \mid l, t \in \mathbb{Z}\}.$$

Assume that the nonces k_1, k_2 are bounded by $B\sqrt{q}$. In [2], these numbers were set to be positive. However, everything works exactly the same if we replace either one or both of these numbers by its q-complement (i.e., replace X by $q - X$). Hence for the simplicity of our notation, we will just write $|k_1|, |k_2| \leq B\sqrt{q}$. In order to estimate the complexity of the algorithm, the followings should be considered:

- the computational complexity to find γ_1, γ_2,
- for any fixed pair (l, t), the computational complexity to compute $g^{k'} \mod p$, where $k' = (lq + \gamma_3)K_1 + \gamma_2 t$,
- the number of pairs l, t to try in S until we find a match $g^{k_1} \equiv g^{k'} \pmod{p}$.

In order to find γ_1 and γ_2, the extended Euclidean algorithm is used and its complexity is $O((\log_2 p)^2)$. Also, for a fixed pair (l, t), the complexity to compute $g^{k'} \mod p$ is $O((\log_2 p)^3)$. Thus it is enough to estimate the number of pairs (l, t) to try in the exhaustive search in Step 3.

First, let us estimate the number of l's to try without reference to t. From the equation (3), we have

$$|lq + \gamma_3| = |k_1\gamma_1 + k_2\gamma_2| \leq |k_1||\gamma_1| + |k_2||\gamma_2|.$$

Again by our assumption $|k_i| \leq B\sqrt{q}$ and the fact $|\gamma_i| \leq \sqrt{q}$ for $i = 1, 2$, we have

$$|lq + \gamma_3| \leq 2Bq.$$

Therefore, the number of l's which satisfy this inequality is exactly $4B$.

Now fix l and let

$$f_l(t) = |(lq + \gamma_3)K_1 + \gamma_2 t|.$$

In order to search for the correct k' and to make this search possible, we find t_0 that gives minimal $k'_0 = f_l(t_0)$. Usually it is okay to take $t_0 = -\lceil (lq + \gamma_3)K_1/\gamma_2 \rceil$. It follows that $0 \leq |k'_0| \leq |\gamma_2|$ and hence

$$(i-1)|\gamma_2| \leq |f_l(t_0 \pm i)| \leq (i+1)|\gamma_2|$$

for any $i \geq 1$. Recall that we are assuming $|k_1| \leq B\sqrt{q}$. Thus, it is enough to consider only the case

$$|f_l(t_0 \pm i)| \leq B\sqrt{q}. \tag{4}$$

Therefore the number of t's which satisfy this inequality for a fixed l is bounded by a constant multiple of $\frac{B}{|\gamma_2|}$.

In most cases, applying Theorem 1 produces γ_2 with $|\gamma_2| \approx O(\sqrt{q})$. But, sometimes $|\gamma_2|$ can get as small as $\frac{\sqrt{q}}{1000}$ or smaller for which case the exhaustive search can take much longer than usual. If this happens, without our improvement, one might consider giving up on this pair of signatures and trying to get another pair of signatures to apply Algorithm KT.

With our improvement however, the role of γ_1 and γ_2 is symmetric. Therefore, if too small $|\gamma_2|$ gives us trouble, then we can switch the role of γ_1 and γ_2 and search for k_2 instead. In fact, when we apply Algorithm KT we make γ_2 slightly bigger than $\sqrt{q}$ as in the example below to get around this problem. We can always find $\gamma_2 > \sqrt{q}$ and even smaller γ_1 by modifying the Euclidean algorithm. Thus the overall complexity of the exhaustive search in the set $\mathcal{S}$ is $O((\log_2 p)^3)$, where the constant coefficient is a small constant multiple of $\frac{B^2(|\gamma_1|+|\gamma_2|)}{\max(|\gamma_1|,|\gamma_2|)} < B^2$. Thus the complexity of the algorithm is $O((\log_2 p)^3)$ as claimed in [2].

6. An Example

In this section, we show, by an example, how our improved Algorithm KT works. By running the Key Generation Algorithm, we first generate the following parameters: a 1024-bit prime

$p =$

```
FFFFFFFF FFFFFFFF CB47A437 683930DF 5FFE707F 0146C929 1E11DAF4 C4F04CA4
802C8962 FBDDCBEF 95C64EC5 5E6E2DE4 34C15608 988B0DE0 BB67C6F1 897E3524
7AE4F8E2 357C5C6F 846D66A6 BE2E9535 03865E6A 584B6D90 E08E529F 1BEED0CF
39340EB4 09A229F3 4AB4E9AE 84573A2C 25C103DC 252DAE89 FFFFFFFF FFFFFFFF
```

and an element

$g =$

```
F673E579 5E963571 89F8BABC 9DCB3D2F 77D69D82 A3AC49FA 1983830F 8767B079
C653E71B 921AA8FA 31538E5F 471905C6 16C51043 2FEF992B 96CAE38E C1A2CA9A
E0A31D48 577F1CD8 A4A13D55 C4659186 D1120112 D1A02156 06EC8E23 66B972EA
D563400A 964D8607 9DA15F95 BA48C849 27D02FA4 74F588FF 9E13ABDA C8178DFC
```

of GF(p) of multiplicative order q, where

$$q = \texttt{ 97B16FBB 1F6E9D93 0DAC5350 C07043F9 558A7F45}$$

is a 160-bit prime q dividing $p - 1$ and the signer's secret key

$$x = \texttt{ 5669EE54 FCC7D19B 17D23994 CFC675E7 BE00C7B6.}$$

In this example, we generate two signatures using message nonces

$$k_1 = \texttt{1DC28733 53DC818B 082FDB0D}$$

and

$$k_2 = \texttt{0B8905F7 8E33061F 90296997}$$

that are bounded by $B\sqrt{q}$, where $B = 2^{13}$. While generating signatures, we have used the equation (1) instead of choosing messages and using hash since the equation (1) is all that matters. Thus we have the following numbers as signatures:

$$u_1 = \texttt{86E4155E D4B08E2A 59701A89 9A482A3F C73DE59E}$$
$$v_1 = \texttt{4D668963 08F04FEC DD8DB68D 097C6894 64EE4A73}$$
$$w_1 = \texttt{873E9323 35A095B0 51031574 371D6771 E299B227}$$

and

$$u_2 = \texttt{743C6A82 E7BBA6FE C9E6A392 0B6FFB83 43139DC4}$$
$$v_2 = \texttt{450C542D D97DAD8D 0B493D98 FDDB6013 DA1ABECD}$$
$$w_2 = \texttt{66EC0899 26D0DA6E 1F5B14C7 C910042F 626DC2CE.}$$

Now we have two signatures generated using small nonces. So we apply Algorithm KT step by step. First, by applying Euclidean algorithm we obtain

$$w = \texttt{8348484A 7D88CFAF 47923F2B A2BC58DB 079F944E}$$

and

$$\gamma_1 = \texttt{0000116A 481ECCD8 7CB58C19}$$
$$\gamma_2 = \texttt{00050F9F 9431606F FFBA0C91.}$$

Here, we execute Euclidean algorithm just until we have $\gamma_2 > \sqrt{q}$ (In fact, $\frac{\gamma_2}{\sqrt{q}} \approx 6.57$.) since we found that the algorithm works best when this is so and it is much easier this way to handle various delicate matters in programming. The next step is to find

$$K_1 = \texttt{0000A093 261A8813 CB5DE1B4}$$
$$K_2 = \texttt{- 00000228 8B0622EE ED41F6E3.}$$

Now in the last step of exhaust search, we limit the search space $\mathcal{S}$ to

$$\{(lq + \gamma_3)K_1 + \gamma_2 t \mid |l| \leq (\gamma_1 + \gamma_2)B\sqrt{q} \text{ and } |t - t_0| \leq \frac{2B}{\gamma_2}\}$$

as suggested in the previous section, where $t_0 = -\lceil (lq + \gamma_3)K_1/\gamma_2 \rceil$. Finally we find a match $k_1 = (lq + \gamma_3)K_1 + \gamma_2 t$ when $l = 26096$ and $t = t_0 - 1505$.

7. Conclusion

In this paper, we applied the extended Euclidean algorithm as in Gallant *et al.*'s method to Algorithm KT [2] and we modified a crucial step of Algorithm KT which makes the idea of Algorithm KT more complete and rigorous. Thus, we extended their result to the case when the message nonce k with $|k| \leq O(\sqrt{q})$ and made Algorithm KT work symmetrically in the two parameters γ_1 and γ_2. As a result, we made the algorithm always produce the correct result by providing an explicit domain where the exhaustive search is made and a way of getting around bad cases.

Acknowledgment : This work was supported by KOSEF grant R05-2002-000-01360-0.

References

[1] T. ElGamal, "A Public Key Cryptosystem and a signature scheme based on discrete logarithms", Advances of Cryptology-CRYPTO'84, LNCS 196, 1985, pp. 10–18.

[2] H. Kuwakado and H. Tanaka, "On the Security of the ElGamal-Type Signature Scheme with small parameters", IEICE Trans. Fundamentals, Vol. E82-A. No. 1 Jan. 1999, pp. 93–97.

[3] R. Gallant, R. Lambert, and S. Vanstone, "Faster Point Multiplication on Elliptic Curves with Efficient Endomorphism", Advances in Cryptology-Crypto'2001, 2001, pp. 190–201.

[4] A. Menezes, P. Ooschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

[5] P. Q. Nguyen and I. E. Shparlinski, "The Insecurity of the Digital Signature Algorithm with Partially Known Nonces", J. Cryptology, Vol. 15, 2002, pp. 151–176.

Math. Dept. Inha Univ., Korea
E-mail address: `ikyie@math.inha.ac.kr`