DSA Signature Scheme Immune to the Fault Cryptanalysis

Maciej Nikodem

The Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
maciej.nikodem@pwr.wroc.pl

Abstract. In this paper we analyse the Digital Signature Algorithm (DSA) and its immunity to the fault cryptanalysis that takes advantage of errors inducted into the private key a. The focus of our attention is on the DSA scheme as it is a widely adopted by the research community, it is known to be vulnerable to this type of attack, but neither sound nor effective modifications to improve its immunity have been proposed. In our paper we consider a new way of implementing the DSA that enhances its immunity in the presence of faults. Our proposal ensures that inducting errors into the private key has no benefits since the attacker cannot deduce any information about the private key given erroneous signatures. The overhead of our proposal is similar to the overhead of obvious countermeasure based on signature verification. However, our modification generates fewer security issues.

1 Introduction

In recent years a variety of implementations based on tamper-proof devices (e.g. smart cards) have been proposed in order to provide better support for data protection. The main reason for this trend originate from the fact that such devices are expected to be characterized by high reliability and security. This is obtained thanks to their ability to perform complex arithmetical operations, to control incoming and outgoing communication, and to prevent unauthorized access. Cryptographic devices, on the other hand, are endangered by faults which can compromise their security.

In 1997 Bao et al.[3] and Boneh et al.[6] showed that if faults occur when the device performs cryptographic operation, then they may decrease security and leak the key stored inside of the device. The described problem has been presented for most of modern cryptographic algorithms such as the RSA encryption and signature scheme, identification protocols, and signature schemes based on the discreet logarithm problem (e.g. ElGamal, Schnorr, and DSA). The same year, Biham and Shamir [4] demonstrated that secret key cryptosystems are vulnerable to the fault cryptanalysis as well.

Since 1997, many researchers have been investigating the problem of fault cryptanalysis, in an effort to discover methods of enhancing security of different cryptographic schemes. However, a handful solutions have been proposed only for the RSA algorithm [2,12,21] and symmetric key cryptosystems [9,10,13]. Less attention has been paid to signature schemes, of which security is based on the Discrete Logarithm Problem (DLP).

All of existing solutions can be divided into three groups:

– fault prevention,
– error detection and error correction,
– error diffusion, also refereed to as *ineffective computation* [21].

The main purpose of using fault prevention techniques is to minimise the probability of fault injection. This is achieved by hardware circuits responsible for shielding a device, and protecting it from reverse engineering, radiation, ion beams, power spikes, and signal glitches [1,16]. Unfortunately, none of these methods work properly – some have design bugs while others can be simply switched off [16]. If fault prevention fails then the device is susceptible to errors, and other types of countermeasures have to be taken.

Error detection and error correction aim to detect and report an error in order to prevent successful cryptanalysis. Existing error detection schemes are based on well-known error detection techniques that utilise reverse computations (e.g. ciphertext decoding or signature verification) or parity checks and residue codes. Such simple solutions allow to enhance immunity to the fault analysis, and may be easily implemented in symmetric encryption algorithms [14]. Some steps to protect the RSA algorithm have been taken too [8]. Unfortunately reverse computations usually introduce delays and computation overhead which may be impractical. Moreover both methods require comparing procedure, that actually verifies whether computations were error free or not, which is may be a target of fault injection, and therefore, may become a bottleneck of the whole solution [15,20,21]. This threat is not present when the error correction is used, since it is performed regardless of if errors have been inducted or not, and thus requires no comparisons. Unfortunately, only one error correction scheme for the AES algorithm has been proposed so far [9].

The error diffusion is another method of preventing fault cryptanalysis. In contrast to previous solutions it attempts to modify a cryptographic algorithm in such a way that any inducted error is spread among different cryptographic computations and the output. The goal of the error diffusion is to produce erroneous output that is useless for the attacker. This is possible, since most fault attack scenarios assume that the attacker induces particular types of errors. Moreover, it is often required that errors are inserted into selected part of the algorithm. Changing and dispersing the error increases the overhead of fault cryptanalysis causing such attacks to be ineffective. For that reason this type of countermeasure is also refereed to as ineffective computations. Although error diffusion was proposed five years ago [21], so far it has been adopted only for the RSA-CRT algorithm. In last year a number of different error diffusion techniques for the RSA-CRT were proposed (e.g. [5,8,21]) but most of them have been broken [15,22]. Moreover, no countermeasures of this type have been proposed for other cryptographic algorithms.

2 Related Work

A feasibility for implementation of the fault cryptanalysis in case of DLP-based signature schemes was first announced in 1997 [3]. Attack described in this paper utilises bit-flip errors inducted into random bit of the private key a. Due to such error the resulting signature is affected with error that may be effectively guessed by the attacker at the cost of $2n$ modular exponentiations. This result was put forth in 2000 by Dottax who presented how one can implement fault cryptanalysis to ECDSA. In 2004, Giraud and Knudsen [11] extended previous results and analysed an attack on the DSA scheme that take advantage of byte errors instead of bit errors. Paper [11] presents detailed analysis of attack complexity, and the number of faulty signatures required to restrict possible private key values to the requested amount. On the other hand, it does not describe any countermeasures that can be applied in order to improve security.

The security of the DSA scheme was also analysed by Rosa [19] who presented a lattice-based fault attack. This attack can be carried out irrespectively of the fact whether a tamper-proof device performs result checking before output or not. It requires for the attacker to substitute g used during signing procedure with $g' = g\beta \bmod p$ for some $1 < \beta < p$ with $\operatorname{ord}(\beta) = d$ in Z_p^* and $\gcd(d, q) = 1$. This can be difficult to achieve with fault induction, when the DSA is implemented in hardware but, as presented in [19], it can be performed if the scheme is implemented in software. An obvious countermeasure for software implementation, as presented in [19], is to make manipulation on the parameters of the DSA scheme impossible.

The lattice-based fault attack was also presented by Naccache et al. [17]. Their attack is based on faults inducted into random integer k in order to force a number of the least significant bytes (LSBs) of k to flip to 0. Afterwards the attacker applies lattice attack on the ElGamal-like signature which can recover private key, given sufficiently many signatures such that a few bits of corresponding k are zeroed. As presented in [17], when one LSB of each k is zeroed, then 27 signatures are sufficient to disclose the private key. In their paper Naccache et al. presented theory and methodology of the attack as well as possible countermeasures (e.g. checksums, randomisation or refreshments).

Although DLP-based signature algorithms are known to be prone to fault cryptanalysis, neither sound nor effective countermeasures have been proposed for these so far. Precisely, there exist only one obvious solution that utilises signature verification procedure in order to check whether generated signature is correct or not. If verification fails then either signing or verification was affected with faults and the signature has to be rejected. This countermeasure seems to work but in fact it utilises comparison procedure which is susceptible to fault attacks and thus may become a bottleneck of the whole proposal [15,20].

In this paper we give a practical method of how to increase immunity of the DSA scheme in the presence of faults that affect private key a. We assume that the pseudo random generator that chooses k operates correctly while the attacker is able to induce random bit-flip or byte change errors into a. Our proposal is based on error diffusion which ensures that any error e, inducted into a, is spread within the signature s. This results in erroneous signature $\overline{s}$ that is affected with error E, which depends both on e and unknown random integer k. Relation between E, e and k ensures that E can neither be computed nor effectively guessed by the attacker. Therefore, in order to perform the attack, one has to verify all of 2^{160} possible values of k one by one which cause the whole attack to be ineffective. Immunity to the fault cryptanalysis is attained at the expense of the increased computational overhead which is similar to the overhead of signature verification. Relatively small overhead enables to implement the proposed scheme in small cryptographic devices like smart cards.

3 DSA Signature Scheme

The DSA signature scheme was proposed in 1991 by U.S. National Institute of Standards and Technology (NIST), and became first digital signature standard (DSS) ever recognised by any government. The DSA is a variant of the ElGamal signature scheme which requires a hash function $h : \{0,1\}^* \rightarrow Z_q$ for prime integer q. Its security is based on the discreet logarithm problem.

Key generation in the DSA scheme is done as follows:

- select a prime number q such that q is 160 bit long ($2^{159} < q < 2^{160}$),
- choose t so that $0 \leq t \leq 8$, and select a prime number p where $2^{511+64t} < p < 2^{512+64t}$ with the property that $q|(p-1)$,
- select a generator g of the unique cyclic group of order q in Z_p^*,
- select a random integer a such that $1 \leq a \leq q-1$,
- compute $y = g^a \bmod p$,
  the public key is $\langle p, q, g, y \rangle$, and the private key is a.

After key generation the device stores the private key a and system parameters p, q, g for future use. Signing for a given message m goes as follows:

- select a random integer $0 < k < q$,
- compute $r = \left(g^k \bmod p\right) \bmod q$,
- compute $s = k^{-1} \left(h(m) + ar\right) \bmod q$,
  the signature for m is a pair $\langle r, s \rangle$.

Given the message m, signature $\langle r, s \rangle$, and the public key $\langle p, q, g, y \rangle$ one can verify whether the signature is actually correct. This is done as follows:

- compute $w = s^{-1} \bmod q$,
- compute $v_1 = g^{h(m)w} \bmod p$ and $v_2 = y^{rw} \bmod p$,
- verify if $v_1 \cdot v_2 \bmod p \bmod q \overset{?}{=} r$.

If last equation holds, then the signature is accepted, otherwise the signature is rejected.

4 Fault Cryptanalysis of the DSA Signature Scheme

As presented in [3,11,17,19] hardware implementations of the DSA scheme can be compromised with fault cryptanalysis. This can be done in a few ways: by affecting random integer k [17], public parameter g [19] or by inducting errors into the private key register during signing procedure [3,11]. In the remaining part of the paper we focus on the fault cryptanalysis that attempts to affect a private key a. Its purpose is to generate a faulty signature which is then used to deduce the key. Let us briefly present an attack scenario described in [3].

Assume that the attacker has a possibility to induct random bit-flip errors into the private key register. Since errors are inducted randomly and the private key register is of the size $n = \log_2 a$ so the probability that i-th bit will be affected with error equals $1/n$. Moreover, inducting exactly one bit-flip error into the i-th bit of the register cause a change of the private key which is now equal to $\overline{a} = a \pm 2^i$ where sign $\pm$ depends on the original value of this bit.

The erroneous signing procedure executes as follows:

- select a random integer $0 < k < q$,
- compute $r = \left(g^k \bmod p\right) \bmod q$,
- compute $\overline{s} = k^{-1}\left(h(m) + \overline{a}r\right) \bmod q$,
  erroneous signature for m is a pair $\langle r, \overline{s}\rangle$.

Because the error has changed the private key into $\overline{a} = a \pm 2^i$, therefore, an element $\overline{s}$ of the erroneous signature $\langle r, \overline{s}\rangle$ is equal

$$\begin{aligned}\overline{s} &= k^{-1}\left(h(m) + \overline{a}r\right) \bmod q = k^{-1}\left(h(m) + \left(a \pm 2^i\right) r\right) \bmod q \\ &= k^{-1}\left(h(m) + ar\right) \pm 2^i r k^{-1} \bmod q \\ &= s \pm 2^i r k^{-1} \bmod q. \end{aligned} \tag{1}$$

Due to (1) and because the fact attacker knows $r = g^k \bmod p \bmod q$ the fault analysis can be performed. This is done in a similar way as for the signature verification:

- compute $w = \overline{s}^{-1} \bmod q$,
- compute $v_1 = g^{h(m)w} \bmod p$ and $v_2 = y^{rw}\left(g^{rw}\right)^{\pm 2^i} \bmod p$,
- look for the fault value $\pm 2^i$ for which following equation holds

$$r = v_1 v_2 \bmod p \bmod q = g^{h(m)w} y^{rw}\left(g^{rw}\right)^{\pm 2^i} \bmod p \bmod q. \tag{2}$$

According to (2), in order to perform cryptanalysis the attacker needs to find a value of the inducted fault $\pm 2^i$. Since we assume errors are inducted at random hence the attacker does not know which bit of the private key was flipped. Therefore, to perform cryptanalysis the attacker needs to check all possible fault values and find the one for which (2) holds. The time required to perform this attack is dominated by $2n$ exponentiations that have to be computed.

Each iteration of the above procedure allows the attacker to recover one bit of the private key. Attacker can then repeat this procedure to get remaining bits, but since errors are inducted at random it could happen that successive errors affect bits already known. This is a difficulty that increases attack overhead but as presented in [6] the attacker can repeat above procedure as long as demanded amount of the private key bits is known. Afterwards he can perform exhausting search in order to find remaining bits.

5 DSA Scheme Immune to the Fault Cryptanalysis

As mentioned in the previous section, the fault analysis countermeasure proposed in this paper is based on the error diffusion. The purpose of this solution is to check whether the public key a, used during signature generation, was affected with error or not. This has to be done without any comparisons and conditional operations since these may be a bottleneck of the whole proposal, similarly as in case of error detection schemes. Therefore, we propose to use public key y in order to verify the correctness of a. The outcome of this verification is called an error diffusion term T, and it is equal zero only if all computations were error free. Later on this term is used in signature generation in such a way that for $T = 0$ the signature s is correct while for $T \neq 0$ erroneous value of $\overline{s}$ depends both on error inducted e and random integer k. The relation between $\overline{s}$, e, and k ensures that the attacker will find it difficult to guess the error e given erroneous signature $\langle r, \overline{s} \rangle$ for message m.

Implementation of the above countermeasure requires that computation of the signature s is split into two steps

$$v = k + ar \bmod q, \tag{3}$$

$$s = k' \left(h(m) + v\right) - 1 \bmod q. \tag{4}$$

These two steps are separated by one additional and one modified computation:

– additional computation of error diffusion term T

$$T = \left(y^{-r} g^{v} \bmod p \bmod q\right) - r \bmod q, \tag{5}$$

– modified computation of multiplicative inverse

$$k' = (k \oplus T)^{-1} \bmod q. \tag{6}$$

With these modifications the whole signing procedure goes according to the scheme 1.

According to the 4-*th* step of the proposed signature scheme the error diffusion term $T = 0$ if and only if

$$y^{-r} g^{v} \bmod p \bmod q = r. \tag{7}$$

If the attacker inducts an error e into the private key a, then the erroneous value of $\overline{v}$ equals

$$\overline{v} = k + \overline{a}r \bmod q = k + (a + e)r \bmod q = k + ar + er \bmod q. \tag{8}$$

Scheme 1. Proposed modification of the DSA scheme

```
Require: message $m$, private key, and public key of the DSA scheme
Ensure: the DSA signature $\langle r, s \rangle$
1: select a random integer $k$ with $0 < k < q$,
2: compute $r = g^k \bmod p \bmod q$,
3: compute $v = k + ar \bmod q$,
4: compute $T = \left(y^{-r} g^v \bmod p \bmod q\right) - r \bmod q$,
5: compute $k' = (k \oplus T)^{-1} \bmod q$,
6: compute $s = k' \left(h(m) + v\right) - 1 \bmod q$
```

Due to faulty value of $\overline{v}$ left side of (7) equals

$$\begin{aligned} y^{-r} g^{\overline{v}} \bmod p \bmod q &= g^{-ar} g^{v+er} \bmod p \bmod q = g^{-ar} g^{k+ar+er} \bmod p \bmod q \\ &= g^{k+er} \bmod p \bmod q. \end{aligned} \tag{9}$$

This equals r if and only if

$$k + er \bmod q = k, \tag{10}$$

which is equivalent to

$$er \bmod q = 0. \tag{11}$$

However, $r < q$ since it was computed modulo q, and e is not a multiplicity of q – otherwise (i.e. $q|e$) $\overline{a} = a + e \bmod q = a$ and no error affects the private key. Therefore, $er \bmod q \neq 0$ for any error e, and hence, $T = 0$ only if no errors were inducted, and $T \neq 0$ otherwise. It is worth to mention that non-zero value of T equals

$$T = g^{k+er} \bmod p \bmod q - r \bmod q = g^{er} \bmod p \bmod q. \tag{12}$$

According to (12) and assuming that the attacker inducts particular type of errors (e.g. bit-flip errors) he is able to guess possible values of $T = g^{er} \bmod p \bmod q$, given erroneous signature $\langle r, \overline{s} \rangle$. This information, however, do not simplify the attack considerably.

Error diffusion in the proposed scheme is obtained by the modified inverse computation

$$k' = (k \oplus T)^{-1} \bmod q. \tag{13}$$

Because inverse is a nonlinear transformation thus the value of k' depends on the term T and the random integer k in a nonlinear way. Therefore, if $T \neq 0$ then $\overline{k'} = k^{-1} + E$ where E is a non-linear function of T and k. Since k is unknown to the attacker, he cannot compute the error E even if he knows the value of T. Moreover, there are $q - 2$ possible values of E because k is chosen randomly every iteration of the scheme. Finally, using $\overline{k'}$ for the computation of the signature s yields its erroneous value $\overline{s}$ to be affected both with error e and E. Therefore, to perform a cryptanalysis the attacker needs to guess both errors, and since there are 2^{160} possible values of E such attack is infeasible.

On the other hand, if no errors were inducted into the private key a, then the proposed scheme outputs correct DSA signature. This is quite obvious since in such situation we obtain

$$\begin{aligned} T &= \left(y^{-r} g^{v} \bmod p \bmod q\right) - r \bmod q = \left(g^{-ar} g^{k+ar}\right) \bmod p \bmod q - r \bmod q \\ &= g^{k} \bmod p \bmod q - r \bmod q = 0, \end{aligned} \tag{14}$$

and

$$k' = (k \oplus T)^{-1} \bmod q = k^{-1} \bmod q. \tag{15}$$

This gives the signature equal

$$\begin{aligned} s &= k' \left(h(m) + v\right) - 1 \bmod q = k^{-1} \left(h(m) + k + ar\right) - 1 \bmod q \\ &= k^{-1} \left(h(m) + ar\right) \bmod q, \end{aligned} \tag{16}$$

which is a standard DSA signature.

6 Security of the Proposed Scheme

The attack scenarios presented in [3,11] assume that the attacker inducts particular types of errors. This allows him to guess inducted error effectively, and use this knowledge to restrict the number of possible private keys.

In our proposal erroneous signature $\overline{s}$ is affected with error e and E, where the value of E depends on inducted error e and random integer k, chosen by the device. Because k is unknown and cannot be computed by the attacker, thus he can perform no better than guessing. Precisely, for each possible value of e the attacker has to find t for which following equation holds

$$\left(g^{h(m)} y^{r} g^{er}\right)^{\overline{s}^{-1}} g^{t} \bmod p \bmod q = r. \tag{17}$$

Taking into account that $\overline{s} = k' \left(h(m) + ar + er\right) \bmod q$, $k' = (k \oplus T)^{-1} = k^{-1} + E$ and assuming that the attacker guessed the inducted error e correctly, we can simplify the above equation

$$g^{(k^{-1}+E)^{-1}} g^{t} \bmod p \bmod q = g^{k \oplus T + t} \bmod p \bmod q = r. \tag{18}$$

Equation (18) shows that the attacker has to guess proper value of t such that

$$k \oplus T + t \bmod q = k, \tag{19}$$

or equivalently

$$k \oplus \left(g^{er} \bmod p \bmod q\right) + t \bmod q = k. \tag{20}$$

Accordingly, the sought value of t is a function of e and k. Furthermore, because k is chosen at random from the set $(1, q)$, there are $q - 2$ possible values of t for each e. Therefore, to perform an attack and to recover partial information on the private key a, the attacker has to guess the inducted error e and find t

that satisfies (18). This requires at least $n2^{161}$ exponentiations, assuming that the attacker inducts single bit-flip errors.

A careful reader may realise that the proposed modification may be simplified since some surplus operations are performed. In fact, the proposed modification can be simplified, and will still work if we substitute equations (3–5) with following

$$v = ar \bmod q, \tag{21}$$

$$s = k' \left(h(m) + v\right) \bmod q, \tag{22}$$

$$T = y^{-r} g^{v} \bmod p \bmod q, \tag{23}$$

which are created by simply removing k, 1 and r from equations (3), (4), and (5) respectively. According to these changes the simplified signing goes according to the scheme 2.

Scheme 2. Simplified modification of the DSA scheme

```
Require: message $m$, private key, and public key of the DSA scheme
Ensure: the DSA signature $\langle r, s\rangle$
1: select a random integer $k$ with $0 < k < q$,
2: compute $r = g^{k} \bmod p \bmod q$,
3: compute $v = ar \bmod q$,
4: compute $T = y^{-r} g^{v} \bmod p \bmod q$,
5: compute $k' = (k \oplus T)^{-1} \bmod q$,
6: compute $s = k' \left(h(m) + v\right) \bmod q$
```

Let us now briefly analyse the security of the simplified scheme. It is a quite simple task to verify that this simplified version has properties similar to the previous scheme (scheme 1):

– for any error e inducted into the private key a the error diffusion term $T = y^{-r} g^{ar+er} \bmod p \bmod q = g^{er} \bmod p \bmod q = 0$ only if $er \bmod q = 0$. Similarly to the scheme 1 this holds only if $e = 0$,
– for any error e, k' is affected with error E that depends on e and the random integer k. Since E may not be computed thus the attacker can do no better than guessing. However this is infeasible since there are 2^{160} possible values of E.

It seems that the simplified scheme offers the same security as first proposal at the cost of smaller computation overhead.

However, this is not true as the simplified scheme can be attacked with lattice based fault cryptanalysis. This type of attack utilises errors in order to simplify the attack down to the problem of solving hidden number problem (HNP). HNP problem states that given pairs $\left\langle u_i, t_i^{(l)} \right\rangle$, where u_i is a random integer, $t_i^{(l)}$ denotes l subsequent bits of $t_i = b + au_i \bmod q$, and a, b are constant (usually it is assumed that $t_i^{(l)}$ denotes most significant bits of t_i but this is not a requirement), one has to deduce the exact values of a and b [7]. As presented in [7], HNP problem can be solved if $l > \epsilon\sqrt{\log q}$ for any fixed $\epsilon > 0$. Moreover, an algorithm that solves HNP may be used to attack the ElGamal-like signature scheme [7,18]. In such case a partial knowledge on k enables the attacker to recover the private key with less than 30 signatures. A similar attack can be also applied to the simplified modification of the DSA scheme.

To achieve this the attacker needs to induct errors into v used in the last step of the simplified signing procedure. Due to a such error the erroneous value $\overline{s}$ equals

$$\overline{s} = k'\left(h(m) + v + e\right) \bmod q, \tag{24}$$

and may be used to guess the inducted error. This can be done by searching for e that satisfies

$$g^{s^{-1}h(m)} y^{s^{-1}r} g^{s^{-1}e} \bmod p \bmod q = r. \tag{25}$$

When proper e is found then the attacker gets partial information about v. Since $v = ar \bmod q$ and r is known, thus the attacker may collect sufficiently many data and solve the HNP problem for a.

Such an attack cannot be carried out in case of the previously proposed scheme (scheme 1) since it computes v as

$$v = k + ar \bmod q, \tag{26}$$

and k is randomly chosen every iteration of the protocol. On the other hand, this modification requires additional additions to be performed in 4-*th* and 6-*th* step of signing.

Proposed modification of the DSA scheme is also immune to multiple fault attacks that can be inducted in practice [15]. Since there is no comparison procedure in our proposal, hence possible attack scenario may focus on inducting two errors: first error e into the private key a and the second error into one of the successive computations. Purpose of the second error is to mask error e during the inverse computation and thus force the device to output the erroneous signature that is suitable for fault cryptanalysis. Although this is possible scenario, it will be very difficult to achieve. This is due to error diffusion term T that depends on e and r which are unknown to the attacker during execution of the signing procedure. Therefore, probability that inducting multiple faults enables the attacker to break the proposed scheme is negligible.

7 Overhead of the Proposed Scheme

As presented in previous sections there is one additional (5) and three modified computations (3), (4), and (6) in the proposed modification of the DSA signature scheme (scheme 1). The overhead of the modified computations is similar to the overhead of the original computations since only two additions modulo q and one EXOR operation is added. An additional computation of the error diffusion term T requires two exponentiations, one multiplication modulo p and q, and one addition modulo q. Accordingly, the computation overhead of the proposed modification is dominated by the time required to perform two exponentiations and a multiplication. Therefore, the overhead of our proposal is smaller than the overhead of signature verification which requires three exponentiations and two multiplications modulo p and q.

Storage overhead of the proposed scheme is also higher compared to the storage required by the standard DSA scheme. It is so since our proposal utilises the public key y to verify the correctness of the private key used for signature generation. Therefore, the public key has to be stored inside of the device affecting storage requirements. However, in a real implementation this storage overhead can be neglected since cryptographic devices usually store the public key anyway.

Implementation overhead can be further reduced if we change the way error diffusion term T is computed. One of possible solutions is to limit the number of modular exponentiations. This can be achieved by using private key a instead of y, so that

$$T = \left(g^{-ar+v} \bmod p \bmod q\right) - r \bmod q. \qquad (27)$$

This reduces the computational and storage overhead significantly but also affects the security of the whole proposal. In fact, it is now susceptible to attackers that can induct permanent errors or the same random error twice: first into a during computation of v, second during computation of T. In this way conducting a multiple fault attack enables to brake the proposal. Above mentioned flaw can be eliminated if we use $a^{-1} \bmod q$ instead of a and compute T as follows

$$T = \left(g^{a^{-1}(v-k)-r} \bmod p \bmod q\right) - 1 \bmod q. \qquad (28)$$

Because multiplicative inverse is a nonlinear transformation thus the relation between error e, inducted into unknown private key a, and corresponding error affecting $a^{-1} \bmod q$ is unknown to the attacker. Therefore, probability that the multiple error attack succeeds is negligible. This is achieved at the cost of increased storage overhead which is required to store multiplicative inverse of the private key.

8 Conclusions

In this paper we have analysed the DSA scheme and its immunity to the fault cryptanalysis. We have demonstrated that introducing the error diffusion we can improve an immunity of the DSA scheme in the presence of faults affecting private key a. Our modification (scheme 1) ensures that in order to recover the private key a the attacker needs to guess error E that depends both on inducted error e and randomly selected integer k (which is unknown to the attacker). Since E cannot be computed thus the attacker needs to check all of 2^{160} possible values of E for each e, which render the whole attack ineffective. Unlike simplified scheme (scheme 2) the proposed modification is also immune to lattice-based attacks that take advantage of errors affecting v in the last step of signature generation.

Proposed modification also eliminates the comparison procedure which is an inherent part of signature verification and cause the obvious countermeasure to be susceptible to multiple fault attacks. Since there is no such procedure in our proposal thus such attacks do not apply.

The overhead of the proposed scheme is similar to the overhead of the obvious countermeasure based on the signature verification. However, computational overhead can be further reduced if error diffusion term is calculated using inverse of the private key instead of the public key.

References

1. Anderson, R.J., Kuhn, M.G.: Tamper Resistance - a Cautionary Note. In: The Second USENIX Workshop on Electronic Commerce Proceedings, pp. 18–21 (1996)
2. Aumller, C., Bier, P., Fischer, W., Hofreiter, P., Seifert, J.P.: Fault Attacks on RSA with CRT: Concrete Results and Practical Countermeasures. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 260–275. Springer, Heidelberg (2003)
3. Bao, F., Deng, R., Han, Y., Jeng, A., Narasimhalu, A.D., Ngair, T.-H.: Breaking Public Key Cryptosystems on Tamper Resistance Devices in the Presence of Transient Fault. In: Christianson, B., Lomas, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 115–124. Springer, Heidelberg (1998)
4. Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
5. Blmer, J., Otto, M., Seifert, J.-P.: A New CRT-RSA Algorithm Secure Against Bellcore Attacks. In: Proc. ACM Computer and Communications Security 2003 (ACM CCS 2003), pp. 311–320. ACM Press, New York (2003)
6. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the Importance of Checking Cryptographic Protocols for Faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
7. Boneh, D., Venkatesan, R.: Rounding in Lattices and Its Cryptographic Applications. In: SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical, Experimental Analysis of Discrete Algorithms), pp. 675–681 (1997)
8. Breveglieri, L., Koren, I., Maistri, P., Ravasio, M.: Incorporating Error Detection in an RSA Architecture. In: Breveglieri, L., Koren, I., Naccache, D., Seifert, J.-P. (eds.) FDTC 2006. LNCS, vol. 4236, pp. 71–79. Springer, Heidelberg (2006)
9. Czapski, M., Nikodem, M.: Error Correction Procedures for Advanced Encryption Standard. In: Int. Workshop on Coding and Cryptography (WCC 2007), April 16-20, 2007, pp. 89–98. INRIA (2007)
10. Dusart, P., Letourneux, G., Vivolo, O.: Differential Fault Analysis on A.E.S., ArXiv Computer Science e-prints (January 2003)
11. Giraud, C., Knudsen, E.: Fault Attacks on Signature Schemes. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 478–491. Springer, Heidelberg (2004)
12. Joye, M., Lenstra, A., Quisquater, J.J.: Chinese Remaindering Based Cryptosystems in the Presence of Faults. Journal of Cryptology 12, 241–245 (1999)
13. Karpovsky, M., Kulikowski, K.J., Taubin, A.: A Differential Fault Analysis Attack Resistant Architecture of the Advanced Encryption Standard. In: Proceedings of CARDIS 2004, pp. 177–192. Kluwer, Dordrecht (2004)
14. Karri, R., Wu, K., Mishra, P., Kim, Y.: Concurrent error detection schemes for fault-based side-channel cryptanalysis of symmetric block ciphers. IEEE Trans. on CAD of Integrated Circuits and Systems 21(12), 1509–1517 (2002)

15. Kim, C.-H., Quisquater, J.-J.: Fault Attacks for CRT Based RSA: New Attacks, New Results, and New Countermeasures. In: Sauveron, D., Markantonakis, K., Bilas, A., Quisquater, J.-J. (eds.) WISTP 2007. LNCS, vol. 4462, pp. 215–228. Springer, Heidelberg (2007)
16. Kömmerling, O., Kuhn, M.G.: Design Principles for Tamper-Resistant Smartcard Processors. In: USENIX Workshop on Smartcard Technology - Smartcard 1999, USENIX Association, pp. 9–20 (1999)
17. Naccache, D., Nguyen, P.Q., Tunstall, M., Whelan, C.: Experimenting with Faults, Lattices and the DSA. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 16–28. Springer, Heidelberg (2005)
18. Nguyen, P.Q., Shparlinski, I.E.: The Insecurity of the Digital Signature Algorithm with Partially Known Nonces. Journal of Cryptology 15(3), 151–176 (2002)
19. Rosa, T.: Lattice-based Fault Attacks on DSA - Another Possible Strategy. In: Proceedings of the conference Security and Protection of Information 2005, Brno, Czech Republic, 3-5 May 2005, pp. 91–96 (2005)
20. Yen, S.M., Joye, M.: Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis. IEEE Transactions on Computers 49(9), 967–970 (2000)
21. Yen, S.M., Kim, S., Lim, S., Moon, S.: RSA Speedup with Chinese Remainder Theorem Immune Against Hardware Fault Cryptanalysis. IEEE Transactions on Computers 52(4), 461–472 (2003)
22. Yen, S.M., Kim, D., Moon, S.: Cryptanalysis of Two Protocols for RSA with CRT Based on Fault Infection. In: Breveglieri, L., Koren, I., Naccache, D., Seifert, J.-P. (eds.) FDTC 2006. LNCS, vol. 4236, pp. 53–61. Springer, Heidelberg (2006)