L. R. Akhmetzyanova, E. K. Alekseev, A. A. Babueva, S. V. Smyshlyaev, On methods of shortening ElGamal-type signatures,
Mat. Vopr. Kriptogr., 2021, Volume 12, Issue 2, 75–91

https://www.mathnet.ru/eng/mvk367

МАТЕМАТИЧЕСКИЕ ВОПРОСЫ КРИПТОГРАФИИ
2021 Т. 12 № 2 С. 75–91

УДК 519.719.2 DOI https://doi.org/10.4213/mvk357

On methods of shortening ElGamal-type signatures

L. R. Akhmetzyanova, E. K. Alekseev, A. A. Babueva, S. V. Smyshlyaev

CryptoPro LLC, Moscow

Получено 24.XI.2020

Abstract. Development of signature schemes with short signatures is a quite relevant non-trivial challenge. The most perspective existing schemes are multivariate schemes and schemes based on Weil pairing. But the cryptographic tools used in these schemes are still not supported by most cryptographic software, this complicates their use in practice. We propose three methods of shortening standard ElGamal-type signatures and analyze how these methods affect the security. Applying all three methods to the GOST signature scheme with elliptic curve subgroup of order $q \in (2^{255}, 2^{256})$ may reduce the signature size from 512 to 320 bits providing sufficient security and acceptable (for non-interactive protocols) signing and verifying time.

Keywords: short signature scheme, ElGamal-type signature scheme, GOST, provable security

О способах сокращения длины подписей типа Эль-Гамаля

Л. Р. Ахметзянова, Е. К. Алексеев, А. А. Бабуева, С. В. Смышляев

ООО «КРИПТО-ПРО», Москва

Аннотация. Разработка схем подписи, порождающих подписи малого размера, — актуальная и нетривиальная криптографическая задача. Наиболее перспективными сейчас являются многомерные схемы и схемы на основе спариваний Вейля. Но криптографические механизмы, используемые в этих схемах, пока не поддерживаются большей частью криптографического программного обеспечения, что затрудняет их использование на практике. В настоящей работе предложены три способа укорачивания подписи для стандартных схем типа Эль-Гамаля и проанализировано их влияние на стойкость. Применение всех трех способов к схеме подписи ГОСТ с порядком подгруппы группы точек эллиптической кривой $q \in (2^{255}, 2^{256})$ позволяет уменьшить размер подписи с 512 до 320 бит, сохраняя достаточный уровень стойкости и приемлемое (для неинтерактивных протоколов) время генерации и проверки подписи.

Citation: Matematicheskie Voprosy Kriptografii, 2021, v. 12, № 2, pp. 75–91 (Russian)

Ключевые слова: короткие подписи, схемы подписи типа Эль-Гамаля, ГОСТ, доказуемая стойкость

1. Introduction

A signature scheme is one of the most widely used cryptographic protocol in practice. It is a self-supporting protocol replacing a handwritten signature and is used as a primitive in a huge amount of multiple protocols (e.g. TLS Handshake [1] and IKEv2 [2]). Therefore, the operational characteristics of signature scheme such as sizes of keys and signature, time complexity of signing and verifying, are crucial for applications. Although all parameters are important, in the current paper we focus only on the size of signature values.

One of the applications requiring short signatures is the systems where a human is asked to manually key in the signature. For example, product registration systems often ask users to key in a signature provided on a CD label. Also, the size of signature influences the requirements on a channel capacity which may be essential for low-bandwidth communication environments (e.g. Internet of Things and QR codes).

1.1. Related works

Due to the relevance of the considered issues many short signature schemes have been proposed since the late 1980's. One of these schemes is known as a BLS signature scheme [3] based on Weil pairing and providing 160 bits signatures with security sufficient for practice.

Other type of short signature schemes is multivariate schemes based on Hidden Field Equations (HFE) such as Quartz [5], Gui [8], SFLASH [4], UOV [10], Rainbow [11]. Although several of these schemes have been broken due to newly developed attacks (see [6, 7]), a number of multivariate schemes such as UOV, Rainbow withstood cryptanalysis (for suitable parameter sets) for more than 20 years. Also in 2016 the work [9] proposed technique reducing the signature size of almost every multivariate signature scheme by 10 to 15 % without increasing the key sizes or slowing down the scheme significantly. The authors claim that by applying their technique to the Gui signature scheme they obtain signatures of size only 110 bits, «which are the shortest signatures of all existing digital signature schemes». However, the scheme has relatively large public key (about 100 KByte) and slow verification time for the smallest signatures.

In light of the above, the BLS signature scheme is treated as a most favourable solution. Unfortunately, currently Weil pairing is still a non-typical cryptographic tool requiring generation and consequent deep analysis of so called «pairing-friendly» curves. Hence, not any cryptographic software supports this type of curves (unlike typical well-investigated elliptic curves used in standard signature schemes such as ECDSA and GOST). Therefore, the problem of providing shorter signatures using typical cryptographic primitives is relevant.

1.2. Our contribution

In the current paper we consider the standard ElGamal-type signature schemes [12] (particularly GOST [16–19]) with two-component signatures $\overline{r}\|\overline{s}$, where $\overline{r}$ and $\overline{s}$ are $\lceil \log_2 q \rceil$-bit strings (here q is the prime order of the used elliptic curve subgroup), and propose three methods of shortening this type of signature:

– The first method is to replace an internal function f (a mapping from a random elliptic curve point to an integer $r \in \mathbb{Z}_q$) by the hash function with truncated output that implies the reduction of the $\overline{r}$ component size.

– The main idea behind the second method is to make certain bits of the $\overline{r}$ signature component to be constant and then to cut them out. This method leads to increasing signing time.

– The last method is to directly truncate signature ($\overline{r}$ and/or $\overline{s}$ component). This methods leads to increasing verification time.

All these methods are independent and may be applied together in any combination.

The idea to use hash function for the ElGamal-type signature shortening was briefly mentioned in [26]. Unlike our first method, the method presented in [26] modifies the original signature scheme significantly due to hashing the message and the $\overline{r}$-component together. Moreover, as we understand, the author proposes to use the same hash function as for message processing in the original signature scheme. Therefore, it is not clear by what exact means signature shortening is made since the hash function output is usually as long as the original components. Even if truncation of the hash output is implicitly supposed to be done, the author presents only asymptotic security results (in terms of polynomial adversaries and negligible success probabilities). However, such a result cannot be used for

choosing hash output length securely for practical application and concrete security bounds should be presented. The paper [9] proposes similar technique as in the last method but for multivariate signature schemes specifically. Unlike our method, signature verification in [9] does not imply signature recovering.

We analyse how applying the methods affects the security by obtaining concrete SUF-CMA-security bounds for modified schemes in the random oracle model. The first method changes the internal structure of the base scheme (function f) and in fact provides a new instance of the ElGamal-type signature scheme. For presentation purposes we obtain security bounds for the modified GOST signature scheme (named GOST-H) only, although we believe that the proof may be easily generalized. Using standard techniques we show that the hardness of elliptic curve discrete logarithm problem (ECDLP) and standard security properties of the hash function imply SUF-CMA-security of the GOST-H signature scheme. The second and the third methods are considered in general: for them we reduce the security of the base unmodified scheme to the security of the corresponding modified signature scheme. Application of all three methods to the GOST signature scheme with EC subgroup order q, $2^{255} < q < 2^{256}$, may reduce the signature size from 512 to 320 bits. The modified scheme provides sufficient security and acceptable time for non-interactive protocols signing (≈ 6 seconds) and verifying (≈ 3 seconds).

1.3. Paper organization

The remainder of the paper is organized as follows. In Section 2 basic definitions and notation are introduced. Section 3 introduces ElGamal-type signature schemes and describes the main object of the research — three methods of shortening signatures of such type. In Section 4 we formally define basic security notions for signature schemes and accompanying primitives. Section 5 is devoted to the security analysis of the proposed methods. We draw our conclusions in Section 6. Detailed proofs of our theorems are relegated to the appendices due to space limitations.

2. Basic notation and definitions

By $\{0,1\}^s$ we denote the set of s-component bit strings and by $\{0,1\}^*$ we denote the set of all bit strings of finite length including the empty string. For bit strings a and b we denote by $a\|b$ their concatenation. Let $|a|$ be the bit length of the string a.

For a bit string u and a positive integer $l \leqslant |u|$ let $\mathsf{msb}_l(u)$ ($\mathsf{lsb}_l(u)$) be the string consisting of the l rightmost (leftmost) bits of u. For integer $r \geqslant 0$ let $\mathsf{str}(r)$ (or just $\overline{r}$) be the $(\lfloor \log_2(r) \rfloor + 1)$-bit representation of $r > 0$ with the least significant bit on the left and zero bit if $r = 0$. For a bit string u let $\mathsf{int}(u)$ be the integer r such that $\mathsf{str}(r) = u$.

If p is a prime number then the set $\mathbb{Z}_p$ is a finite field with characteristic p. We assume the canonical representation of the elements in $\mathbb{Z}_p$ as a natural number in the interval $[0, p-1]$. Each non-zero element x in $\mathbb{Z}_p$ has an inverse $1/x$. We define $\mathbb{Z}_p^*$ as the set $\mathbb{Z}_p$ without zero element.

We denote the group of points of elliptic curve over the field $\mathbb{Z}_p$ as $\mathbb{G}$, the order of the prime subgroup of $\mathbb{G}$ as q and elliptic curve point of order q as P. We denote the group generated by P as $\langle P \rangle$ and neutral element in $\mathbb{G}$ as O.

For any set A and B let $Func(A, B)$ be the set of all mappings from A to B. If the value s is chosen from a set S uniformly at random, then we denote $s \xleftarrow{\mathcal{U}} S$.

If the variable x gets the value val then we denote $x \leftarrow val$. Similarly, if the variable x gets the value of the variable y then we denote $x \leftarrow y$. If the variable x gets the result of a probabilistic algorithm A we denote $A \xrightarrow{\$} x$ ($x \xleftarrow{\$} A$). If we need to emphasize that A is deterministic than we denote it by $A \to x$ ($x \leftarrow A$). The event when A returned value val as a result is denoted by $A \to val$.

We define security properties using the notion of «experiment» played between a challenger and an adversary. The adversary and challenger are modelled using consistent interactive probabilistic algorithms. The challenger simulates the functioning of the analysed cryptographic scheme for the adversary and may provide him access to one or more oracles. The parameters of an adversary $\mathcal{A}$ are its computational resources (for a fixed model of computation and a method of encoding) and oracles query complexity. The query complexity usually includes the number of queries. Denote by $\mathsf{Adv}^{\mathrm{m}}_S(\mathcal{A})$ the measure of the success of the adversary $\mathcal{A}$ in realizing a certain threat, defined by the security notion M for the cryptographic scheme S. The formal definition of this measure will be given in each specific case.

3. Three methods of shortening

A signature scheme consists of three algorithms $\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}$ such that: algorithm KGen generates secret signing key sk and public signature verifi-

cation key pk; algorithm Sig takes as input a signing key sk and message m and generates a signature sgn for message m; deterministic algorithm Vf takes as input verification key pk, message m and candidate signature sgn and outputs 1 (accept) or 0 (reject). It is required that for every $(\mathsf{pk}, \mathsf{sk})$ outputted by KGen and every message m it holds that

$$\mathsf{Vf}(\mathsf{pk}, m, \mathsf{Sig}(\mathsf{sk}, m)) = 1.$$

The GenElGamal framework is introduced in [12].

We follow the notation of [16] and change the definition in [12] in the following way: we represent the signature as the concatenation of vectors $\overline{t}$ and $\overline{s}$ instead of pair of elements in $\mathbb{Z}_q$, we denote r by k, h by e, t by r, x by d, X by Q. Moreover, we denote the GenElGamal signature scheme by GenEG.

The signature generated by the GenEG scheme is represented as $\overline{r}\|\overline{s}$, where $r, s \in \mathbb{Z}_q$, thus, its size is at most $2\lceil \log q \rceil$. The following sections introduce three methods of shortening signature size, we call the schemes obtained by applying these methods $\mathsf{GenEG\text{-}H}$, $\mathsf{GenEG_S}$ and $\mathsf{GenEG^V}$ respectively.

3.1. GenEG-H scheme

The r component in all GenEG schemes is computed as the result of applying function f to point R. The idea behind the first method of shortening signature is to modify function f by splitting it into two functions. At first we apply the compression function H_2 to the x-coordinate of point R and then we represent the result as an element in $\mathbb{Z}_q^*$. Note that the bit representation of $\overline{r}$ will be shorter due to compression.

The modified function f is defined in the following way:

$$f(R) = \phi(H_2(R.x)),$$

where H_2 maps $\mathbb{Z}_p$ to $\{0,1\}^b$, $b < \lceil \log q \rceil$, and ϕ maps $\{0,1\}^b$ to $\mathbb{Z}_q^*$; here and below we denote by $A.x$ an encoding of the x-coordinate of elliptic curve point A as an integer. The function H_2 may be instantiated by the function $H(1\|\overline{x}) \mod 2^b$, where H is the hash function that maps $\{0,1\}^*$ to $\mathbb{Z}_{2^{256}}$. The function ϕ is defined as follows: it maps $x \in \{0,1\}^b \setminus \{0\}$ to $\mathsf{int}(x)$ and maps zero to 2^b. This definition of the ϕ function is always correct due to the fact that $b < \lceil \log q \rceil$.

In order to separate domains of the hash function used for different cases, we redefine the hash function used for hashing messages as $H_1(x) = H(0\|\overline{x}) \mod q$.

We illustrate our method by applying it to the GOST scheme which is a special case of the GenEG scheme. Note that function f in the GOST scheme (as well as in the ECDSA scheme) is defined as

$$f(R) = R.x \mod q.$$

Here and in what follows we denote by $A.x$ an encoding of the x-coordinate of elliptic curve point A as an integer.

We define the GOST-H scheme as follows.

$\mathsf{KGen}(\ )$	$\mathsf{Sig}(d, m)$	$\mathsf{Vf}(Q, m, \overline{r}\|\overline{s})$
$d \xleftarrow{\mathcal{U}} \mathbb{Z}_q^*$	$e \leftarrow H_1(m)$	**if** $s = 0$: **return** 0
$Q \leftarrow dP$	**if** $e = 0 : e \leftarrow 1$	$e \leftarrow H_1(m)$
return (d, Q)	$k \xleftarrow{\mathcal{U}} \mathbb{Z}_q$	**if** $e = 0 : e \leftarrow 1$
	if $k = 0$: **return** $\perp$	$R \leftarrow e^{-1}sP - e^{-1}rQ$
	$R \leftarrow kP$	**if** $\phi(H_2(R.x)) \neq r$: **return** 0
	$r \leftarrow \phi(H_2(R.x))$	**return** 1
	$s \leftarrow ke + dr$	
	if $s = 0$: **return** $\perp$	
	return $\overline{r}\|\overline{s}$	

This method allows us to shorten the size of the signature from $(2\lceil \log q \rceil)$ bits to at most $(\lceil \log q \rceil + b + 1)$ bits. For instance, for $b = \lceil \log q \rceil / 2$ we can cut out one quarter of the size.

Note that we do not check the condition $r = 0$ in the GOST-H scheme, because the function ϕ is defined in a such way that it does not map any argument to zero.

3.2. $\mathsf{GenEG_S}$ scheme

The idea behind the second method is to «mine» during the signature generation procedure. We generate signature until it meets certain additional conditions: the first l bits of $\overline{r}$ should match the constant vector. Thus, we can exclude these bits from the signature and reduce the signature size by l bits.

The $\mathsf{GenEG_S}$.KGen algorithm is similar to the GenEG.KGen algorithm. The $\mathsf{GenEG_S}$.Sig and $\mathsf{GenEG_S}$.Vf procedures are defined as follows.

$\mathsf{Sig}(d, m)$

1 : $cnt \leftarrow 0$
2 : **if** $cnt > W$: **return** $\perp$
3 : $cnt \leftarrow cnt + 1$
4 : $\overline{r}\|\overline{s} \leftarrow \mathsf{GenEG.Sig}(d, m)$
5 : **if** $\overline{r}\|\overline{s} = \perp$: **goto** 2
6 : **if** $\mathsf{lsb}_l(\overline{r}) \neq const$: **goto** 2
7 : $\overline{r}^* = \mathsf{msb}_{|\overline{r}|-l}(\overline{r})$
8 : **return** $\overline{r}^*\|\overline{s}$

$\mathsf{Vf}(Q, m, \overline{r}^*\|\overline{s})$

1 : $\overline{r} \leftarrow \overline{r}^*\|const$
2 : $res \leftarrow \mathsf{GenEG.Vf}(Q, m, \overline{r}\|\overline{s})$
3 : **return** res

The scheme defined above has two new parameters: l — number of fixed bits in $\overline{r}$ and W — number of attempts to generate valid signature. These parameters are not independent from each other and they are strictly related to the generation time and probability of outputting the valid signature by the $\mathsf{GenEG_S.Sig}$ procedure. Thus, they should be chosen in accordance with the generating mechanism computing power. We will discuss the appropriate values for these parameters in Section 5. The constant vector is an additional scheme parameter, it may be set to l zero bits for simplicity.

Note that the number of loop iterations needed to generate the valid signature depends on the probability of finding $\overline{r}$ satisfying the condition $\mathsf{lsb}_l(\overline{r}) = const$. In case of applying the method to the GenEG-H scheme, we can estimate this probability as 2^{-l} since the distribution of hash function output is close to uniform. We claim that the situation will not change in case of applying the method to schemes with function $f(R)$ equal to $R.x \mod q$ since function lsb_l is proven to be good entropy extractor for x-coordinate of point R (see [15] for more details).

3.3. $\mathsf{GenEG^V}$ scheme

The idea behind the third method is to truncate the signature (either $\overline{r}$ or $\overline{s}$ component) by t bits and search them during verification procedure.

The $\mathsf{GenEG^V.KGen}$ algorithm is similar to the GenEG.KGen algorithm. The $\mathsf{GenEG^V.Sig}$ and $\mathsf{GenEG^V.Vf}$ procedures are defined as follows.

$\mathsf{Sig}(d, m)$

1 : $\overline{r}\|\overline{s} \leftarrow \mathsf{GenEG.Sig}(d, m)$
2 : **if** $\overline{r}\|\overline{s} = \perp$: **return** $\perp$
3 : $\overline{s}^* \leftarrow \mathsf{msb}_{|\overline{s}|-t}(\overline{s})$
4 : **return** $\overline{r}\|\overline{s}^*$

$\mathsf{Vf}(Q, m, \overline{r}\|\overline{s}^*)$

1 : $i \leftarrow 0$
2 : **if** $i \geqslant 2^t$: **return** 0
3 : $\overline{s} \leftarrow \overline{s}^*\|\mathsf{str}_t(i)$
4 : $i \leftarrow i + 1$
5 : $res \leftarrow \mathsf{GenEG.Vf}(Q, m, \overline{r}\|\overline{s})$
6 : **if** $res = 0$: **goto** 2
7 : **return** 1

The construction defined above assumes truncating the $\overline{s}$ component of the signature, however we can define this scheme similarly up to truncating the $\overline{r}$ component. The decision which part of the signature should be truncated could depend on the possible optimization of verification process.

This method allows us to reduce the signature size by t bits. The value of parameter t is strictly related to the signature verification time and should be chosen in accordance with the verifier's computing power.

Note that the proposed method is general and may be applied not only to the GenEG signature scheme but also to any scheme with signature represented as a concatenation of two bit vectors. In particular, it may be applied to the $\mathsf{GenEG_S}$ scheme by replacing the GenEG.Sig and GenEG.Vf calls with the corresponding $\mathsf{GenEG_S}$ procedure calls.

4. Security notions

In this section we formally define basic security notions used for signature schemes and the assumptions on primitives. The security of signature schemes is traditionally analyzed in the SUF-CMA notion (Strong UnForgeability under Chosen Message Attack).

Definition 1. For a signature scheme SS

$$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{SS}}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\text{SUF-CMA}}_{\mathsf{SS}}(\mathcal{A}) \to 1\right],$$

where the experiment $\mathbf{Exp}^{\text{SUF-CMA}}_{\mathsf{SS}}(\mathcal{A})$ is defined in the following way:

$\mathbf{Exp}^{\text{SUF-CMA}}_{\mathsf{SS}}(\mathcal{A})$

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{SS.KGen}(\,)$
2 : $\mathcal{L} \leftarrow \varnothing$
3 : $(m, sgn) \xleftarrow{\$} \mathcal{A}^{Sign}(\mathsf{pk})$
4 : **if** $(m, sgn) \in \mathcal{L}$: **return** 0
5 : $res \leftarrow \mathsf{SS.Vf}(\mathsf{pk}, m, sgn)$
6 : **return** res

Oracle $Sign(m)$

1 : $sgn \leftarrow \mathsf{SS.Sig}(\mathsf{sk}, m)$
2 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{(m, sgn)\}$
3 : **return** sgn

We analyse the security of the GOST-H scheme assuming the function H_2 to be a random oracle. The random oracle model was introduced in [24] and is an idealized model that assumes the existence of a public random function H such that all parties can obtain $H(x)$ (for any desired input value x) only by interacting with an oracle computing H; parties cannot compute H (for any input) on their own. Using a random oracle is a common way to ease the cryptographic analysis by making it modular. However, one should always keep in mind that a random oracle cannot be instantiated by any real hash function and, therefore, one should use it very carefully, trying to interpret the obtained security results. We discuss the meaning of random oracle model for our proof in the next section.

Definition 2 (ECDLP problem)**.**

$$\mathsf{Adv}_{\mathbb{G}}^{\mathrm{ecdlp}}(\mathcal{A}) = \Pr\left[Q \xleftarrow{\mathcal{U}} \langle P\rangle\,;\ d \xleftarrow{\$} \mathcal{A}(Q,P) : dP = Q\right].$$

Similar to [14] for the family H_1 of hash functions we define signum-relative collision resistance property (SCR property, see Definition 3) and signum-relative division resistance property (SDR property, see Definition 4). Throughout the paper we consider implicitly keyed hash functions $H_1\colon \{0,1\}^* \mapsto \mathbb{Z}_q$ with initialization vector assumed to be an implicit key. The experiments of the up-coming security definitions should be understood as implicitly first picking a random initialization vector $IV \in \mathcal{IV}$ and giving it to the adversary.

Definition 3 (SCR property)**.** For the family of hash functions H_1

$$\mathsf{Adv}_{\mathsf{H}_1}^{\mathrm{scr}}(\mathcal{A}) = \Pr\left[(m_1,m_2) \xleftarrow{\$} \mathcal{A}\colon H_1(m_1) = \pm H_1(m_2) \wedge m_1 \neq m_2\right].$$

Definition 4 (SDR property)**.** For the family of hash functions H_1

$$\mathsf{Adv}_{\mathsf{H}_1}^{\mathrm{sdr}}(\mathcal{A}) = \Pr\Big[\beta_1,\beta_2 \xleftarrow{\mathcal{U}} \{0,1\}^b, (m_1,\Gamma) \xleftarrow{\$} \mathcal{A}_1(\beta_1), m_2 \xleftarrow{\$} \mathcal{A}_2(\Gamma,\beta_2)\colon \frac{H_1(m_1)}{\phi(\beta_1)} = \pm\frac{H_1(m_2)}{\phi(\beta_2)}\Big].$$

The SDR property is implied by the standard assumptions: zero resistance and signum-relative preimage resistance properties of H_1 (see full version of the paper [27] for formal proof and definitions of these properties).

We estimate the advantages defined above based on the best known methods of solving the corresponding security tasks. For the ECDLP problem it is the Pollard's ρ-algorithm (see [25]), for the SCR notion it is the attack based on birthday paradox and for the SDR notion (implied by the preimage resistance) it is the exhaustive search. So for the group $\mathbb{G}$ and for the family of hash functions H_1 we assume that for any adversary $\mathcal{A}$ with the time complexity at most T

$$\mathsf{Adv}^{\mathrm{ecdlp}}_{\mathbb{G}}(\mathcal{A}) \approx \frac{T^2}{q}; \quad \mathsf{Adv}^{\mathrm{scr}}_{\mathsf{H}_1}(\mathcal{A}) \approx \frac{T^2}{q}; \quad \mathsf{Adv}^{\mathrm{sdr}}_{\mathsf{H}_1}(\mathcal{A}) \approx \frac{T}{q}.$$

5. Security bounds

In this section we provide the security bounds for the schemes defined in Section 3.

For the GOST-H scheme we provide the reduction from the ECDLP problem. We claim that the proof for the other GenEG-H schemes may be obtained using the same technique.

Theorem 1. *Let $\mathcal{A}$ be an adversary with time complexity at most T in the* SUF-CMA *model for the* GOST-H *scheme, making at most Q_S queries to the Sign oracle and at most Q_O queries to the H_2 oracle. Then there exist an adversary $\mathcal{D}$ that solves the* ECDLP *problem for the used elliptic curve group $\mathbb{G}$, an adversary $\mathcal{C}$ that breaks the signum-relative collision resistant property of H_1 and an adversary $\mathcal{M}$ that breaks the signum-relative division resistant property of H_1, such that*:

$$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GOST\text{-}H}}(\mathcal{A}) \leqslant \sqrt{(Q_O+2)\left(\mathsf{Adv}^{\mathrm{sdr}}_{\mathsf{H}_1}(\mathcal{M})\cdot(Q_O+2)+\mathsf{Adv}^{\mathrm{ecdlp}}_{\mathbb{G}}(\mathcal{D})\right)} + \frac{Q_O+3}{2^b} + \mathsf{Adv}^{\mathrm{scr}}_{\mathsf{H}_1}(\mathcal{C}) + \frac{(2Q_O+Q_S+1)Q_S}{q-1}.$$

The time complexity of $\mathcal{C}$ is at most $T + c((Q_S+3)T^V_{\mathsf{GOST\text{-}H}} + Q_O)$, the time complexities of $\mathcal{D}$ and $\mathcal{M}$ are at most $2T + 2c((Q_S+4)T^V_{\mathsf{GOST\text{-}H}} + 2Q_O + 4)$, where $T^V_{\mathsf{GOST\text{-}H}}$ is computational resources needed to verify one signature by the GOST-H.Vf *procedure, c is a constant that depends only on a model of computation and a method of encoding.*

Here Q_S and Q_0 should be interpreted as a maximum number of signatures known to the adversary and as a maximum number of the hash function calls made by the adversary with the computational resources T

respectively. The Q_S value is set in accordance with application requirements and the Q_O value depends on computational model and resources T. Usually we set Q_O to $\left\lceil \frac{T}{T_H} \right\rceil$, where T_H is the resources needed to compute one hash value for one-block message in the chosen computational model. It is correct as soon as we assume sequential computational model, however we note that any parallel model of computations is equivalent to the corresponding sequential computational model with more resources [23].

Proof sketch. The idea behind the proof is similar to the idea used in [13]. The proof consists of two steps. During the first one we show that the notions of existential unforgeability under chosen message attack and under key-only attack are nearly equivalent, assuming H_1 is signum-relative collision resistant. Next, using the forking lemma we show that the hardness of the ECDLP in the group $\mathbb{G}$ and the signum-relative division resistance of H_1 imply unforgeability under key-only attack. The complete proof may be found in [27].

Note that for several steps we provide more accurate reductions than article [13] does (there are several unclarified statements in [13] which seem to be incorrect). Note that providing accurate reductions is quite important since potential mistakes may lead to practical vulnerabilities (see e.g. [20, 21]).

The interpretation of the random oracle model in our case is as follows. If the signature scheme turns out to be insecure, then, due to the proof sketch, the ECDLP problem is solved or the used hash function does not sufficiently disrupt the link between the domain and the range.

Remark 1. Note that the obtained reduction is not tight: there are no known cryptanalytic attacks breaking the signature scheme with the specified probability and computational resources. This is the common problem of reductions obtained using forking lemma. Moreover, several negative results are known. Paillier and Vergnaud [22] show that the forgeability of several discrete log based signatures cannot be equivalent to solving the discrete log problem in the standard model, assuming the so-called one-more discrete log assumption and algebraic reductions.

The only term depening on b is $\frac{Q_O + 3}{2^b}$. Applying the assumed bounds for $\mathsf{Adv}^{\mathrm{sdr}}_{\mathsf{H}_1}$, $\mathsf{Adv}^{\mathrm{scr}}_{\mathsf{H}_1}$ and $\mathsf{Adv}^{\mathrm{ecdlp}}_{\mathbb{G}}$ we have the biggest term in the bound of order $\frac{\sqrt{Q_O} \cdot T}{\sqrt{q}}$. Thus, assuming $T > \sqrt{Q_O}$ (that is reasonable due to $Q_O \leqslant T$) we

obtain the following surprising result: reducing b up to $\frac{\lceil \log_2 q \rceil}{2}$ does not significantly change the final bound. Thus, this method allows to shorten the size of the signature from $2\lceil \log_2 q \rceil$ bits to $\frac{3}{2}\lceil \log_2 q \rceil$ bits without harming security.

Theorem 2. *Let $\mathcal{A}$ be an adversary with time complexity at most T in the* SUF-CMA *model for the* $\mathsf{GenEG_S}$ *scheme, making at most Q_S queries to the Sign oracle. Then there exists an adversary $\mathcal{B}$ for the* GenEG *scheme in the* SUF-CMA *model that makes at most $Q_S \cdot W$ queries to the Sign oracle, such that*

$$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GenEG_S}}(\mathcal{A}) = \mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GenEG}}(\mathcal{B}).$$

Furthermore, the time complexity of $\mathcal{B}$ is at most $T + cQ_SW$, where c is a constant that depends only on a model of computation and a method of encoding.

The proof may be found in [27].

Consider the Q_S parameter in detail. Unlike the previous theorem, here Q_S cannot be interpreted as a number of signatures known to the adversary, since the scheme may return the failure indicator very often (depending on the parameters l and W). Therefore, if N is a required for application number of signatures, then Q_S should be set to $\frac{N}{pr}$, where pr is the probability to return valid signature for one signing call. We assume that $pr \approx 1 - (1 - 2^{-l})^W$.

Note that W parameter should be chosen in such a way that the error probability is small enough for practice. The optimal way is to choose $W = 2^l$ (in this case the error probability is smaller than e^{-1} for all l).

Theorem 3. *Let $\mathcal{A}$ be an adversary with time complexity at most T in the* SUF-CMA *model for the* $\mathsf{GenEG^V}$ *scheme, making at most Q_S queries to the Sign oracle. Then there exists an adversary $\mathcal{B}$ for the* GenEG *scheme in the* SUF-CMA *model that makes at most Q_S queries to the Sign oracle, such that:*

$$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GenEG^V}}(\mathcal{A}) = \mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GenEG}}(\mathcal{B}).$$

Furthermore, the time complexity of $\mathcal{B}$ is at most $T + c \cdot 2^t \cdot T^V_{\mathsf{GenEG}}$, where T^V_{GenEG} is computational resources needed to verify one signature by the $\mathsf{GenEG.Vf}$ *procedure, c is a constant that depends only on a model of computation and a method of encoding.*

The proof may be found in [27].

Note that parameter t affects the security bound via the time complexity of the adversary $\mathcal{B}$. If we consider this parameter to be very large then the computational resources of $\mathcal{B}$ become too large, the GenEG scheme breaks and, as a result, the security bound degenerates.

Theorems 2 and 3 are presented not in the random oracle model. However, if the security bound for the GenEG scheme is provided in the random oracle model we can change the theorems accordingly. If the adversary $\mathcal{A}$ makes at most Q_O queries to H_2 oracle, then in case of the Theorem 2 the adversary $\mathcal{B}$ makes the same number of queries to its own H_2 oracle and in case of the Theorem 3 the adversary $\mathcal{B}$ makes at most $Q_O + 2^t$ queries to its own H_2 oracle.

The GOST-H$^{\mathsf{V}}_{\mathsf{S}}$ scheme. Let us introduce the GOST-H$^{\mathsf{V}}_{\mathsf{S}}$ scheme – the result of applying all three methods to the GOST scheme which is the special case of the GenEG scheme. The GOST-H$^{\mathsf{V}}_{\mathsf{S}}$.KGen algorithm is similar to the GOST.KGen algorithm. The GOST-H$^{\mathsf{V}}_{\mathsf{S}}$.Sig and GOST-H$^{\mathsf{V}}_{\mathsf{S}}$.Vf procedures are defined as follows.

$\mathsf{Sig}(d, m)$

1 : $cnt \leftarrow 0$
2 : **if** $cnt > W$: **return** $\perp$
3 : $cnt \leftarrow cnt + 1$
4 : $\overline{r}\|\overline{s} \leftarrow \mathsf{GOST\text{-}H.Sig}(d, m)$
5 : **if** $\overline{r}\|\overline{s} = \perp$: **goto** 2
6 : **if** $\mathsf{lsb}_l(\overline{r}) \neq const$: **goto** 2
7 : $\overline{r}^* \leftarrow \mathsf{msb}_{|\overline{r}|-l}(\overline{r})$
8 : $\overline{s}^* \leftarrow \mathsf{msb}_{|\overline{s}|-t}(\overline{s})$
9 : **return** $\overline{r}^*\|\overline{s}^*$

$\mathsf{Vf}(Q, m, \overline{r}^*\|\overline{s}^*)$

1 : $\overline{r} \leftarrow \overline{r}^*\|const$
2 : $i \leftarrow 0$
3 : **if** $i \geqslant 2^t$: **return** 0
4 : $i \leftarrow i + 1$
5 : $\overline{s} \leftarrow \overline{s}^*\|\mathsf{str}_t(i)$
6 : $res \leftarrow \mathsf{GOST\text{-}H.Vf}(Q, m, \overline{r}\|\overline{s})$
7 : **if** $res = 0$: **goto** 2
8 : **return** 1

Summarizing the results of Theorems 1, 2, 3 and the bounds presented in Section 4 we obtain the following security bound for the GOST-H$^{\mathsf{V}}_{\mathsf{S}}$ scheme:

$$\mathsf{Adv}^{\text{suf-cma}}_{\text{GOST-H}^{\mathsf{V}}_{\mathsf{S}}}(\mathcal{A}) \leqslant \sqrt{2(Q_O + 2^t + 2) \cdot \frac{(Q_O + 2^t + 2)T_1 + 2T_1^2}{q}} + \frac{Q_O + 2^t + 3}{2^b} + \frac{(2Q_O + 2^{t+1} + Q_S \cdot W + 1)Q_S \cdot W + T_1^2}{q - 1},$$

where

$$T_1 \leqslant T + 2T^V_{\mathsf{GOST\text{-}H}}(Q_S \cdot W + 2^t + 2) + 2Q_O + 4,$$

Table 1: Security bounds for the $\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}$ scheme

Fixed parameters	Variable parameter and corresponding security bound					
$l = 18,\ t = 18$	b	128	100	80	70	60
	$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}}(\mathcal{A})$	2^{-35}	2^{-35}	2^{-19}	2^{-9}	1
$b = 100,\ t = 18$	l	10	18	35	50	77
	$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}}(\mathcal{A})$	2^{-35}	2^{-35}	2^{-34}	2^{-19}	1
$b = 100,\ l = 18$	t	10	18	30	64	80
	$\mathsf{Adv}^{\text{suf-cma}}_{\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}}(\mathcal{A})$	2^{-35}	2^{-35}	2^{-35}	2^{-27}	1

$T^{V}_{\mathsf{GOST\text{-}H}}$ is computational resources needed to verify one signature by the $\mathsf{GOST\text{-}H.Vf}$ procedure.

The size of the short signature generated by the $\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}$ scheme is equal to at most $(\lceil \log q \rceil + b + 1 - l - t)$, where parameters b, l and t characterize three methods of shortening respectively. We provide the bounds for the $\mathsf{GOST\text{-}H}^{\mathsf{V}}_{\mathsf{S}}$ scheme for particular values of N, q, W, T, Q_O: we set N to 10^6 as it is reasonable number of signatures for our application, we use curve with prime subgroup order q such that $2^{255} < q < 2^{256}$, we set W to 2^l by reasons discussed above and we set T to 2^{60} assuming such computational power of potential adversary for our application. Moreover, for simplicity we estimate Q_O as T. The $\mathsf{GOST\text{-}H.Vf}$ procedure assumes two hash and two multiple point calculation, we estimate $T^{V}_{\mathsf{GOST\text{-}H}}$ as 32 assuming the computational resources measured in hash calculations.

Table 1 presents the evolution of security bound with changing one of the scheme parameter as long as other two parameters are fixed. We choose l and t equal to 18 in the first step based on the appropriate signing (≈ 6 seconds) and verifying (≈ 3 seconds) time. The computer with the following characteristics was used: Intel Core i5-8600K CPU 3.60GHz, L1 D-Cache 32 KB x 6, L1 I-Cache 32 KB x 6, L2 Cache 256 KB x 6. We set b to 100 based on the security bound obtained in the first step.

By choosing the optimal values of methods parameters ($b = 100$, $l = 18$ and $t = 18$) we reduce the signature size from 512 to 320 bits providing the sufficient security for our application.

6. Conclusion

This paper introduces three methods of shortening ElGamal-type signatures. The proposed methods do not imply increasing the key sizes and may be applied together in any combination. Applying second and/or third method leads to increasing signing and/or verifying time. The implementation of these methods do not require any special cryptographic tools.

We apply these methods to ElGamal-type signature schemes and obtain security bounds in the random oracle model. The presented theorems allow us to estimate the security of the modified scheme by the security of the used cryptographic primitives (elliptic curve group and hash function family) in case of the first method and by the security of the original scheme in case of the second and the third methods. The paper presents the security bounds for the $\mathsf{GOST\text{-}H}_{\mathsf{S}}^{\mathsf{V}}$ scheme with elliptic curve subgroup order q, $2^{255} < q < 2^{256}$ with different parameter values (see Table 1). Choosing the optimal parameter values for our application allows to reduce the signature size from 512 to 320 bits.

References

[1] Rescorla E., *The Transport Layer Security* (*TLS*) *Protocol Version* 1.3, RFC 8446, DOI 10.17487/RFC8446, 2018, https://www.rfc-editor.org/info/rfc8446.

[2] Kaufman C., Hoffman P., Nir Y., Eronen P., Kivinen T., *Internet Key Exchange Protocol Version* 2 (*IKEv*2), RFC 7296, DOI 10.17487/RFC7296, 2014, https://www.rfc-editor.org/info/rfc7296.

[3] Boneh D., Lynn B., Shacham H., "Short signatures from the Weil pairing", ASIACRYPT 2001, Lect. Note Comput. Sci., **2248**, 2001, 514–532.

[4] Patarin J., Courtois N., Goubin L., "FLASH, a fast multivariate signature algorithm", CT-RSA 2001, Lect. Note Comput. Sci., **2020**, 2001, 298–307.

[5] Patarin J., Courtois N., Goubin L., "QUARTZ, 128-bit long digital signatures", CT-RSA 2001, Lect. Note Comput. Sci., **2020**, 2001, 282–297.

[6] Dubois V., Fouque PA., Shamir A., Stern J., "Practical cryptanalysis of SFLASH", CRYPTO 2007, Lect. Note Comput. Sci., **4622**, 2007, 1–12.

[7] Courtois N.T., Daum M., Felke P., "On the security of HFE, HFEv- and Quartz", PKC 2003, Lect. Note Comput. Sci., **2567**, 2003, 337–350.

[8] Petzoldt A., Chen MS., Yang BY., Tao C., Ding J., "Design principles for HFEv-based multivariate signature schemes", ASIACRYPT 2015, Lect. Note Comput. Sci., **9452**, 2015, 311–334.

[9] Mohamed M.S.E., Petzoldt A., "The shortest signatures ever", INDOCRYPT 2016, Lect. Note Comput. Sci., **10095**, 2016, 61–77.

[10] Kipnis A., Patarin J., Goubin L., "Unbalanced Oil and Vinegar signature schemes", EUROCRYPT'99, Lect. Note Comput. Sci., **1592**, 1999, 206–222.

[11] Ding J., Schmidt D., "Rainbow, a new multivariable polynomial sgnature scheme", ACNS 2005, Lect. Note Comput. Sci., **3531**, 2005, 164–175.

[12] Fersch M., Kiltz E., Poettering B., "On the one-per-message unforgeability of (EC)DSA and its variants", TCC 2017, Lect. Note Comput. Sci., **10678**, 2017, 519–534.

[13] Fersch M., Kiltz E., Poettering B., "On the provable security of (EC) DSA signatures", Proc. 2016 ACM SIGSAC Conf. Comput. and Communic. Security, 2016, 1651–1662.

[14] Fersch M., *The Provable Security of Elgamal-type Signature Schemes*, Diss. Bochum, Ruhr-Universität Bochum, 2018.

[15] Chevalier C., Fouque PA., Pointcheval D., Zimmer S., "Optimal randomness extraction from a Diffie-Hellman element", EUROCRYPT 2009, Lect. Note Comput. Sci., **5479**, 2009, 572–589.

[16] *GOST R* 34.10-2012. *Information technology. Cryptographic data security. Signature and verification processes of electronic digital signature. National standard of the Russian Federation*, *STANDARTINFORM*, 2012 (In Russian).

[17] *GOST* 34.10-2018. *Information technology. Cryptographic data security. Signature and verification processes of electronic digital signature. Interstate standard*, *Interstate Council for Standardization*, *Metrology and Certification* (*ISC*), 2018 (in Russian).

[18] *ISO/IEC* 14888-3:2018, *IT Security techniques – Digital signatures with appendix – Part* 3: *Discrete logarithm based mechanisms – Section* 6: *Certificate-based mechanisms* – 6.9: *ECRDSA*, 2018.

[19] Dolmatov V., Degtyarev A., *GOST R* 34.10-2012: *Digital Signature Algorithm*, RFC 7091, DOI 10.17487/RFC7091, 2013, https://www.rfc-editor.org/info/rfc7091.

[20] Inoue A., Iwata T., Minematsu K., Poettering B., "Cryptanalysis of OCB2: attacks on authenticity and confidentiality", CRYPTO 2019, Lect. Note Comput. Sci., **11692**, 2019, 3–31.

[21] Koblitz N., Menezes A., *Critical perspectives on provable security*: *fifteen years of "Another Look" papers*, Cryptology ePrint Archive: Report 2019/1336, 2019.

[22] Paillier P., Vergnaud D., "Discrete-log-based signatures may not be equivalent to discrete log", ASIACRYPT 2005, Lect. Note Comput. Sci., **3788**, 2005, 1–20.

[23] Savage J.E., *Models of Computation*: *Exploring the Power of Computing*, Addison-Wesley Longman Publishing Co, Boston, 1998.

[24] Bellare M., Rogaway P., "Random oracles are practical: A paradigm for designing efficient protocols", *Proc. 1st ACM conf. Comput. and communic. security*, 1993, 62–73.

[25] Pollard, J.M., "A Monte Carlo method for factorization", *BIT*, **15** (1975), 331–334.

[26] Zheng Y., "Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption)", CRYPTO'97, Lect. Note Comput. Sci., **1294**, 1997, 165–179.

[27] Akhmetzyanova A., Alekseev E., Babueva A., Smyshlyaev S., *On methods of shortening ElGamal-type signatures* (*full version*), IACR Cryptology ePrint Archive, 2021/148.pdf, 2021 https://eprint.iacr.org/2021/148.pdf.