Computer Science Journal of Moldova, vol.**14**, no.3(42), 2006

Cryptoschemes Based on New Signature Formation Mechanism

A.A.Moldovyan, D.N.Moldovyan, L.V.Gortinskaya

Abstract

Several variants of new digital signature schemes (DSS) based on the discrete logarithm and factorization problems have been proposed. Considered DSS are characterized in that a novel mechanism of the signature generation is used, in which two parameters of the (k, S) or (R, S) signature are defined after solving a system of two congruences. In the case of composite modulus additional restrictions conditions have been introduced for selection of the public key.

1 Introduction

Public key cryptosystems based on hard mathematical problems are well approved for information authorization, i.e. for generating digital signatures. For such application the most important of such problems are the following: 1) factorization of a large integer number and 2) finding discrete logarithm modulo large prime number. The first problem underlies the RSA cryptosystem and the second problem is the base of ElGamal's DSS, US standard FIPS 186, and Russian GOST P.34.10-94. The large modulo exponentiation procedure is used in these schemes. The RSA cryptosystem uses composite modulus n that is a product of two randomly chosen prime numbers r and q: $n = rq$. In the cryptosystems of the second type the prime modulus p is considered.

In RSA the public key is represented by pair of numbers (e, n), and the private key is a number d, which is calculated using the following formula: $ed \bmod n = 1$, where $\varphi(n) = (r-1)(q-1)$ is Euler phi function of n. Security of this system is based on difficulty of calculating d while φ(n) is an unknown value. The φ(n) value can be easily calculated after factorization of the modulus n, therefore divisors of n have to be kept in secret (or to be annihilated after the e and d keys have been generated). The signature corresponding to a plaintext M is a value S, which satisfies the following verification equation: $S^e \bmod n = H(M)$, where $H(M) = H$ is the hash function value corresponding to M. The signature generation equation is the following one: $S = H^d \bmod n$. In RSA the signature length is approximately equal to the n modulus length (denoted as $|n|$). At present in different practical applications the used n modulus has the length 1024 bits or more (for example, 1024-2048 bits). In ElGamal's DSS the public key y is calculated exactly as in Diffie-Hellman's public-key distribution system: $y = \alpha^x \bmod p$, where x is private key (an integer number of sufficiently large size) and α is a primitive element modulo p. The signature is represented by a pair of numbers (R, S) that are calculated depending on the both private key and message to be signed. The signature generation procedure is described as follows:

1) generate a random number k that is coprime with $p - 1$;

2) calculate $R = \alpha^k \bmod p$;

3) calculate the S value satisfying the following equation:

$$H = (xr + kS) \bmod (p - 1)\,,$$

where H is the hash function value calculated from the plaintext.

The (R, S) signature is considered as valid one if the following signature verification equation is satisfied: $\alpha^H = y^R R^S \bmod p$. To ensure required security the modulus p size has to be equal to $|p| \geq 1024$ bits. Thus, the (R, S) signature size is 1024 bits or more.

In this paper several new DSS are considered, which are based on difficulty of the discrete logarithm and factorization problems. Additional restriction requirements to selection of the composite modulus are formulated.

2 New Signature Generation Mechanism

The ElGamal-like DSS allow one to calculate the (R, S) and H values, which satisfy the verification equation, without using the private key. In practice this theoretic attack is eliminated due to using secure hash functions, since the attack produces random values H. Indeed, the mentioned attack is based on the preliminary generating the R parameter with special structure that permits calculation of the "correct" S and H values that play a part of adjusting parameters. Representing the R parameter as $R = \alpha^u y^t \bmod p$ and using two fitting parameters attacker can avoid necessity to solve discrete logarithm problem, i. e. he can easily find "satisfactory" values S and H, but the last is a random one. While using strong hash functions, the attacker can indicate no plaintext corresponding to the H hash value calculated while attacking. Because of this fact the described attack has theoretical character and no practical significance in majority of applications. However, the possibility of mentioned attack is an unwanted property of DSS.

To eliminate such attacks we propose to use a new mechanism of the (R, S) signature generation in the DSS based on difficulty of solving DLP. The idea is to calculate simultaneously both parameters R and S each of which play a part of a variable defining the exponent value and a part of a multiplier in the signature verification equation. In new schemes the R and S parameters form two different functions $F_1(R, S)$ and $F_2(R, S)$. The signature parameters are changed in such a way that the $F_1(R, S)$ function changes, while the $F_2(R, S)$ function keeps its value constant. Due to variability of the F_1 function and invariability of the F_2 function we have potential possibility to calculate the values R and S satisfying the verification equation. Assuming that we consider only such values of the R and S signature parameters, which save the value $F_2 = Z = \text{const}$, it is potentially possible to simplify the verification equation and calculate some signature value $(R\,(Z)\,, S\,(Z))$ depending on Z. While constructing some concrete verification equation the (F_1, F_2) pair of functions can be used, for example, in one of the following forms:

$$(R/S \bmod p, RS \bmod p),$$

$$(RS \bmod p, R/S \bmod p),$$
$$(RS^H \bmod p, RS \bmod p),$$
$$(RS \bmod p, RS^2 \bmod p),$$
$$(RS^Z \bmod p, RS \bmod p),$$
$$(RS^Z \bmod p, R/S \bmod p),$$

where $Z = RS \bmod p$, the R and S values are supposed to have the following structure: $R = \alpha^k \bmod p$ and $S = \alpha^g \bmod p$. The values k and g are to be calculated simultaneously as solution of the system of two congruences, one of the lasts defining invariability of the F_2 function and having the form: $k + g \equiv \text{const} \bmod (p-1)$. Taking into account that signature verification expression should contain multipliers depending on the hash value and public key we have the following examples of DSS based on the described signature formation mechanism:

$$R/S = y^{(RS \bmod p)^H} \alpha^{(RS \bmod p)} \bmod p,$$
$$R = Sy^{H(RS \bmod p)} \alpha^{(RS \bmod p) \bmod \delta} \bmod p,$$
$$(R/S)^{(RS \bmod p)} = y^{(RS \bmod p) \bmod \delta} \alpha^H \bmod p,$$

where δ is an arbitrary prime number such that $|\delta| \approx 0.25\,|p|$ (the $\bmod\ \delta$ operation defines a compression function depending on F_2).

Considering these concrete variants of the verification equation it is possible to note that an attacker can re-designate the signature parameters R and S and reduce the verification equation to some equivalent expression in which parameters R and S can be calculated consecutively without using the private key. For example, let us consider the DSS defined by the verification equation

$$R = Sy^{H(RS \bmod p)} \alpha^{(RS \bmod p) \bmod \delta} \bmod p.$$

Denoting $Z = RS \bmod p$ we have $Z/S = Sy^{HZ} \alpha^{Z \bmod \delta} \bmod p$. Now the signature can be forged as follows:

i) select arbitrary value Z,

ii) calculate $S^2 = Z\left(y^{HZ}\alpha^{Z \bmod \delta}\right)^{-1} \bmod p$,

iii) if the value $W = Z\left(y^{HZ}\alpha^{Z \bmod \delta}\right)^{-1} \bmod p$ is not a quadratic residue, then go to step 1, otherwise calculate $S = W^{1/2} \bmod p$ and $R = Z/S \bmod p$.

It is easy to demonstrate that the calculated signature satisfies the initial verification equation. The last attack shows that we should make one step back in our DSS design, namely we should indicate the value R in the verification equation as $\alpha^k \bmod p$ and define the signature as (k, S). In this case the verification equation can be simplified, since there is no necessity to indicate the α parameter two times (as α^k and α^H) and to use the compression function as second type function depending on variables k and S and keeping its value constant. Let us consider the following verification equation that takes into account the mentioned remarks:

$$\alpha^{k+H} \equiv S^2 y^{(\alpha^k S \mod p)} \mod p,$$

where $\gcd(3, p-1) = 1$. Using private key it is possible to generate the (k, S) signature as follows:

i) choose random number $U < p - 1$ and calculate $Z = \alpha^U \bmod p$;

ii) solve the following system of two congruences:

$$U \equiv k + g \bmod (p-1),$$

$$k \equiv gH + x + Z \bmod (p-1),$$

where k and g are unknown parameters:

$$k = \frac{2U - H + xZ}{3} \bmod p - 1,$$

$$g = \frac{U + H - xZ}{3} \bmod p - 1$$

iii) calculate $S = \alpha^g \bmod p$.

To avoid additional restriction imposed on the p modulus one can use a generator of the γ-order group $\{\varepsilon, \varepsilon^2 \bmod p, ..., \varepsilon^{\gamma-1} \bmod p, \varepsilon^\gamma$

mod$p\}$, where γ is a prime divisor of p and $|\gamma| =$ 160-256 bit, as the α parameter in the verification equation. In this case the mentioned above system of congruences gets the form:

$$U \equiv k + g \bmod \gamma,$$

$$k \equiv gH + x + Z \bmod \gamma.$$

Therefore the k value size is $|k| =$ 160-256 bit and we get reduction of the signature length. Similarly the other variants of DSS can be constructed. For example, the following verification equations define several DSS:

$$\varepsilon^{k+H} \equiv Sy^{(\varepsilon^k S \bmod p)} \bmod p,$$

$$\varepsilon^{k} \equiv S^H y^{(\varepsilon^k S \bmod p)} \bmod p,$$

$$\varepsilon^{Hk} S^{-1} \equiv y^{(\varepsilon^k S \bmod p)} \bmod p,$$

$$\varepsilon^{-k} S \equiv y^{H(\varepsilon^k S \bmod p)} \bmod p.$$

Concrete system of congruences, which should be solved to calculate the k and g parameters, can be easily derived from the respective verification equation.

If the prime modulus p in the described DSS design approach is replaced by RSA modulus n, factorization of which is kept secret, then security of DSS will be defined by intractability of both the DLP modulo n and the factorization modulus n. Indeed, to calculate γ it is necessary to find the $\varphi(n)$ value. The public key of such schemes is (n, y, α) or (n, y, ε). Unfortunately the DLP modulo n can be reduced to problem of finding discrete logarithms modulo r and modulo q. Therefore this replacement will be actual for practice, if some new algorithms of finding discrete logarithms that avoid the factorization problem are developed. Some DSS, based only on the factorization problem, represent current interest since they provide possibility to reduce the difficulty of signature verification procedure. In the next section we consider some of such DSS variants.

3 Schemes based on difficulty of the modulus factorization

While using the RSA modulus the verification equation can be simplified due to avoiding the y or/and ε parameters. The public key in this case is (n, ε) or n, respectively. The DSS security in this case depends on difficulty to factorize the RSA modulus. Several possible variants are present in Table 1.

Table 1. Some variants of new DSS schemes

Verification equation	System of congruences	Formulas for calculating the k and g values	Public key
$\alpha^k \equiv S^{\frac{H\alpha^k}{S} \bmod n} \bmod n$	$k - g \equiv U \bmod \gamma$ $gZ \equiv U \bmod \gamma$ $\left(Z = \left(H\alpha^U \bmod n\right) \bmod \gamma\right)$	$k = \frac{ZU}{1-Z} \bmod \gamma$ $g = \frac{U}{1-Z} \bmod \gamma$	n
$\alpha^{k-H} S^{\alpha^k S \bmod n} \equiv 1 \bmod n$	$k + g \equiv U \bmod \gamma$ $k + gZ \equiv H \bmod \gamma$ $\left(Z = \alpha^U \bmod n\right)$	$k = \frac{ZU-H}{Z-1} \bmod \gamma$ $g = \frac{H-U}{Z-1} \bmod \gamma$	(n, α)
$R = S^H \alpha^{(RS \bmod n)} \bmod n$	$k + g \equiv U \bmod \gamma$ $k - Hg \equiv Z \bmod \gamma$ $\left(Z = \alpha^U \bmod n\right)$	$k = \frac{HU+Z}{H+1} \bmod \gamma$ $g = \frac{U-Z}{H+1} \bmod \gamma$	(n, α)
$R = S\alpha^{H(RS \bmod n)} \bmod n$	$k + g \equiv U \bmod \gamma$ $k - g \equiv HZ \bmod \gamma$ $\left(Z = \alpha^U \bmod n\right)$	$k = \frac{U+HZ}{2} \bmod \gamma$ $g = \frac{U-HZ}{2} \bmod \gamma$	(n, α)

In the last two DSS variants with composite modulus we indicate directly the R value in the verification equation. The attack based on using arbitrary value Z and calculating the respective value S is prevented because the S value is to be calculated as $S = \left(\frac{Z}{\alpha^{HZ}}\right)^{1/2} \bmod n$ or as $S = \left(\frac{Z}{\alpha^{Z}}\right)^{\frac{1}{H+1}} \bmod n$. In both cases it is difficult to calculate S without factorizing the modulus.

4 Additional restriction requirements for choosing the RSA modulus

Let us consider the scheme from Table 1, where (n, α) is a public key, where α is a generator of the cyclic γ-order group, i. e. we have $\alpha^{\gamma} \equiv 1 \bmod n$. Some internal relation between the values α^{γ} and n provides potentially some additional possibilities to factorize modulus n. Let us assume that γ is a divisor of $r-1$ and α does not divide $q-1$. In practice to generate the α generator the following expression is used: $\alpha = \beta^{\varphi(n)/\gamma} \bmod n \neq 1$, where β is a random number for which we have $\gcd(\beta, n) = 1$. (Using the Euler's theorem one can demonstrate that $\alpha^{\gamma} = \beta^{\varphi(n)} \equiv 1 \bmod n$, where $\varphi(n) = (r-1)(q-1)$, i. e. if $\beta^{\varphi(n)/\gamma} \bmod n \neq 1$, then the $\beta^{\varphi(n)/\gamma} \bmod n$ value is a generator of the γ-order group). We have:

$$\begin{aligned}
&\alpha = \beta^{\varphi(n)/\gamma} \bmod n \equiv \left(\beta^{(q-1)}\right)^{(r-1)/\gamma} \bmod n \Rightarrow \\
&\alpha \equiv \left(\beta^{(q-1)}\right)^{(r-1)/\gamma} \bmod q \Rightarrow \alpha \equiv 1^{(r-1)/\gamma} \bmod q \Rightarrow \\
&\alpha \equiv 1 \bmod q \Rightarrow \alpha - 1 \equiv 0 \bmod q \Rightarrow q | \alpha - 1 \Rightarrow \gcd(\alpha - 1, n) = q
\end{aligned}$$

Thus, in the considered case it is possible to factorize modulus using extended Euclidean algorithm. Therefore some restrictions should be imposed on generation of the public key. A way preventing the described factorization method is to use such numbers r and q that both of them contain the same required large divisor γ and γ^2 does not divide neither $r-1$ no $q-1$. If this additional requirement is imposed, then the α parameter can be generated as follows: $\alpha = \beta^{L(n)/\gamma} \bmod n \neq 1$, where $L(n) = \operatorname{lcm}[r-1, q-1]$ is the generalized Euler's function. Thus, $\alpha = \beta^{uv} \bmod n \neq 1$, where $u = (r-1)/\gamma$ and $v = (q-1)/\gamma$. If we use, while generating the α value, a value that is simultaneously primitive element modulo r and primitive element modulo q as the β value (i.e. β is "double" primitive element), then we will have simultaneously $\alpha \not\equiv 1 \bmod q$ and $\alpha \not\equiv 1 \bmod r$. While using a "double" primitive element we deterministically generate a "strong" α value. But it is not strictly necessary to use "double" primitive elements. We can generate a "strong" α value selecting random β values. In this case we should check if $\alpha \not\equiv 1 \bmod q$ and $\alpha \not\equiv 1 \bmod r$ hold. Probability that the current value is not "strong" is sufficiently low (see the next section).

The second way to generate "strong" public key is to use composite value γ, i. e. $\gamma = \gamma'\gamma''$, where $\gamma'|r-1$ and $\gamma''|q-1$ and γ' and γ'' do not divide $q-1$ and $r-1$, correspondingly. For generating the α parameter we have the following formula: $\alpha = \beta^{L(n)/\gamma} \bmod n \neq 1$, i. e $\alpha = \beta^{uv} \bmod n \neq 1$, where $u = (r-1)/\gamma'$ and $v = (r-1)/\gamma''$. Analogously to the first case, while using the β value that is "double" primitive element, we get $\alpha \not\equiv 1 \bmod q$ and $\alpha \not\equiv 1 \bmod r$.

Thus, we have two different ways to define difficulty of the n modulus factorization in the considered DSS. Unfortunately in the first way we have a problem to avoid possibility to calculate the secret parameter γ without factorizing the n modulus. Indeed, we have:

$$n-1 = (u\gamma+1)(v\gamma+1) - 1 = uv\gamma^2 + u\gamma + v\gamma = (uv\gamma + u + v)\gamma,$$

Usually the $n-1$ value can be easily factorized; therefore the secret γ can be recovered, if no new restriction requirements are imposed on selection of the n modulus. In the second way factorization of the $n-1$ value does not allow one to determine the γ secret. Thus, we have shown that the second way is preferable in practice.

The considered above restrictions are also actual for the DSS schemes, where the α parameter is not used directly in the verification equation. Indeed, parameters R and S are generated in accordance with the following formulas: $R = \alpha^k \bmod p$ and $S = \alpha^g \bmod p$, therefore, if one of the congruences $\alpha \equiv 1 \bmod r$ or $\alpha \equiv 1 \bmod q$ are valid, then we have: $R \equiv \alpha^k \equiv S \equiv a^g \equiv 1 \bmod r$ or $R \equiv \alpha^k \equiv S \equiv a^g \equiv 1 \bmod q$, correspondingly. Thus, if the additional restrictions for generating modulus n are not taken into account, then it becomes possible to factorize n as follows: $\gcd(S-1, n) = d$ or $\gcd(R-1, n) = d$, where $d = r$ or $d = q$.

5 Experiments

5.1 Portion of “double” primitive elements

Since some new restrictions have been imposed on the n modulus selection it becomes interesting if the portion of the required values is sufficiently large and the generation of the required parameters is not difficult. Some experiments have been performed to clarify this problem. We have investigated the probability of appearance of “double” primitive element modulo r and modulo q for the case $|r| = |q|$.

In the first experiment two arrays $\{q_1, q_2, ..., q_{100}\}$ and $\{r_1, r_2, ..., r_{100}\}$ of random prime numbers q and r were fixed and for all couples q_i and r_j, where $i, j = 1...100$, random values β were checked whether they were primitive elements both modulo q_i and modulo r_j. The number of cases in which the β value is a “double” primitive element were calculated. The portion of “double” primitive elements had been estimated as ratio of successful attempts to total number of the checked values β.

The second experiment was analogous to the first one except the numbers r_j had special structure such that $r_j - 1 = 2r'$, where r' was a prime (q was a random prime).

The third experiment was analogous to the first one except both of the numbers q_i and r_j had special structure such that $r_j - 1 = 2r'$ and $q_i - 1 = 2q'$, where r' and q' were primes. Results of the experiments are presented in Table 2, where probability of appearance of ”double primitive element” for different r and q are shown.

Table 2. Probability of appearance of “double primitive element”

	random r and q	$r = 2r' + 1$, q is random	$r = 2r' + 1$, $q = 2q' + 1$
size16 bits	0.138	0.190	0.242
size 32 bits	0.141	0.191	0.249
size 128 bits	0.140	0.191	0.250

As you can see, the numbers r and q of special case have largest amount of ”double primitive elements”.

5.2 The probability to get $\gcd(\alpha - 1, n) \neq 1$

Fulfillment of the condition $\gcd(\alpha - 1, n) \neq 1$ means that the α value is not “strong” and permits easy factorization of the n modulus. We considered such primes r and q, that r and q had the following special forms: $2^k\gamma + 1$, $2^k\gamma t + 1$, $2^k\gamma wz + 1$, $2a^k\gamma + 1$, where γ, z w, a and t are primes, k is integer. For such structures of primes the following theoretic estimation holds: $\Pr[\gcd(\alpha - 1, n) \neq 1] = \Pr(r|\alpha - 1) + \Pr(q|\alpha - 1)$, where $\Pr[r|(\alpha - 1)] = \Pr[q|(\alpha - 1)] = \frac{1}{\gamma}$. In the experimental investigation we had generated 10,000 random values β for each r. For each case we calculated the value $\alpha = \beta^{L(rq)/\gamma^2} \bmod n$ and checked if the inequality $\gcd(\alpha - 1, r) \neq 1$ holds (successful attempt). The average probability to get $\gcd(\alpha - 1, r) \neq 1$ had been estimated as ratio of successful attempts to 10,000. The results are shown in Table 3.

Table 3. Probability to get $\gcd(\alpha - 1, r) \neq 1$

	$\gamma = 13$	$\gamma = 61$	$\gamma = 251$	$\gamma = 1021$	$\gamma = 4093$	$\gamma = 16381$
$r = 2^k\gamma t_r + 1$	0,07691	0,01635	0,00397	0,001	0,00024	0,00006
$r = 2^k\gamma w_r z_r + 1$	0,07694	0,01639	0,00396	0,00097	0,00024	0,00006
$r = 2a_r^k\gamma + 1$	0,077	0,01633	0,00397	0,00098	0,00026	0,00006
Theoretical estimation	0,07692	0,01639	0,00398	0,00098	0,00024	0,00006

6 Examples

This section presents numerical examples illustrating the performance of two variants of the DSS schemes based on the described method.

Example 1 corresponds to the verification equation

$$\alpha^H S = (\alpha^k y)^{(\alpha^k S \bmod p)} \bmod p,$$

where p is prime, α is a γ-order group generator, γ is a prime divisor of $p-1$.
$p = 1114480948460437900461137676365117526064437706171$;
$\gamma = 438842863793146886096235091634916473156863$;
$\alpha = 894976612126097602347243899051388076165101371 79$;
$H = 12345678900987654321$.
Private key x is equal to $x = 123456789421$; and public key y is $y =$
$= \alpha^x \bmod p = 7086433481212949992850706989252856068670920132 26$.

Signature generation procedure is as follows.
Choose random number $U < p-1 : U = 324567894535645$;
calculate $Z = \alpha^U \bmod p$:
$Z = 5973035889804987392524872230127491330188049562 66$.

Solve the following system of two congruences:

$$\begin{cases} H + g = Z(k+x)\mathrm{mod}\gamma \\ k + g = U\mathrm{mod}\gamma \end{cases}$$

We get

$$k = \frac{H - Zx + U}{Z+1} \bmod \gamma =$$
$$= 24322086709251169695760096421007332936051$$
$$g = \frac{UZ - H + Zx}{Z+1} \bmod \gamma =$$
$$= 41452077708389571640047499553847703475645 7$$

Now we can calculate $S = \alpha^g \bmod p$.
$S = 3424812560204624214643292756602039084168683412 55$.

Signature is a pair of integers

$$(k, S) = (24322086709251169695760096421007332936051,$$

$$3424812560204624214643292756602039084168683412 55).$$

The signature verification gives:

$$\alpha^H S = 186116586750325286758065722053257979431171344776$$

$$(\alpha^k y)^{(\alpha^k S \bmod p)} \bmod p =$$

$= 1861165867503252567580657220832579794811713448 76.$

Thus, the verification result is positive.

Example 2 corresponds to the verification equation

$$\alpha^k \bmod n = S^{H(\alpha^k S \bmod n)} \bmod n,$$

where n is RSA modulus ($n = r * q$), α is a γ-order group generator. The γ value represents a production of two primes: $\gamma = \gamma'\gamma''$, where $\gamma' | r-1$, $\gamma' \not| q-1$ and $\gamma'' | q-1$, $\gamma'' \not| r-1$.

In the example we have generated the following values of the DSS parameters: $\gamma' = 304417319473$; $\gamma'' = 509801878007$; $\gamma = 155192521165192291530311$; $r = 186859595222449878933706 7039$;
$q = 207162539019283834092433954034 89$;
$n =$
$= 387103081863983561529983812356929823366698433937812474990 71$;
$\alpha =$
$= 184847879437491203098938318996018868688047509495509021823 15$;
$H = 1234567890987654321$.

The signature generation procedure is described as follows:
i) choose random number $U < \gamma$: $U = 445274222$
ii) calculate $Z = \alpha^U \bmod n$:

$Z = 797412200754415147882392318201658204428653141530458590298 6$

iii) solve the following system of two congruences:

$$\begin{cases} k = gHZ \bmod \gamma \\ k + g = U \bmod \gamma \end{cases}$$

Solving the system we get:

$$k = \frac{UHZ}{1+HZ} \bmod \gamma = 338182890913800769661 33;$$

$$g = \frac{U}{1+HZ} \bmod \gamma = 121374232073812659838400$$

Now the S signature element can be calculated: $S = \alpha^g \bmod n$.
$S =$
$= 7777656711905876453231146489464318441673845448603344786719.$

The digital signature is $(k, S) = (33818289091380076966133,$
$7777656711905876453231146489464318441673845448603344786719)$.

The signature verification gives:
$\alpha^k \bmod n =$
$= 305728545657482087903519125474912965654981630180212537554 20$
$S^{H(\alpha^k S \bmod n)} \bmod n =$
$= 30572854565748208790351912547491296565498163018021253755420$
Thus, the verification result is positive.

7 Conclusion.

Using a novel mechanism of the signature generation we have proposed new signature schemes based on the DLP and factorization problem. The feature of the applied signature generation mechanism consists in simultaneous calculation of the k and g parameters that define signature (k, S) or (R, S), where $R = \alpha^k \bmod p$ and $S = \alpha^g \bmod p$, in different variants of DSS. Using the composite modulus one can simplify the verification equation, but in this case some additional (relatively ones corresponding to the RSA cryptosystem) restriction requirements to the public key should be taken into account. The fulfilled experiments have shown that the additional requirements do not introduce essential restrictions for practical use of the proposed DSS based on composite modulus.

References

[1] R.L. Rivest, A. Shamir, and L.M. Adleman. *A Method for Obtaining Digital Signatures and Public Key Cryptosystems*, Communications of the ACM, 1978, vol. 21, n. 2, pp. 120–126.

[2] T.A. ElGamal. *Public key cryptosystem and a signature scheme based on discrete logarithms* // IEEE Transactions on Information Theory. 1985, Vol. IT-31, No. 4. pp. 469–472.

[3] A.J. Menezes, S.A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996. 780 p.

[4] D.N. Moldovyan. *Digital Signature Schemes Based on Difficulty of Factorizing Modulus* // Voprosy zaschity informatsii (Moscow). 2004. No. 4 (67). pp. 6–11 (in Russian).

A.A.Moldovyan, D.N.Moldovyan, L.V.Gortinskaya

Received December 6, 2005

Specialized Center of Program Systems "SPECTR",
Kantemirovskaya, 10, St.Petersburg 197342, Russia;
Phone/fax: 7-812-2453743,
E–mail: *nmold@cobra.ru*