GOST Digital Signature Scheme

GOST describes the generation and verification processes for digital signatures, based on operations with an elliptic curve points group, defined over a prime finite field. The necessity for developing this standard is caused by the need to implement digital signatures of varying resistance due to growth of computer technology. Digital signature security is based on the complexity of discrete logarithm calculation in an elliptic curve points group and also on the security of the hash function used.

The following notations are used in this:

1. V_l: set of all binary vectors of an l-bit length
2. V_all: set of all binary vectors of an arbitrary finite length
3. Z: set of all integers
4. P: prime number, p > 3
5. GF(p): finite prime field represented by a set of integers {0, 1, ..., p - 1}
6. b (mod p): minimal non-negative number, congruent to b modulo p
7. M: user's message, M belongs to V_all
8. (H1 || H2): concatenation of two binary vector
9. a, b: elliptic curve coefficients
10. m: points of the elliptic curve group order
11. q: subgroup order of group of points of the elliptic curve
12. O: zero point of the elliptic curve
13. P: elliptic curve point of order q
14. d: integer - a signature key
15. Q: elliptic curve point - a verification key
16. zeta: digital signature for the message M
17. ^ : the power operator
18. /= :non-equality
19. sqrt: square root

Digital Signature Generation Process

It is necessary to perform the following actions (steps) to obtain the digital signature for the message M belonging to V_all.

Step 1. Calculate the message hash code M: H = h(M)

Step 2. Calculate an integer alpha, the binary representation of which is the vector H, and determine:
e = alpha (mod q)
If e = 0, then assign e = 1.

Step 3. Generate a random (pseudorandom) integer k, satisfying the inequality: 0 < k < q

Step 4. Calculate the elliptic curve point C = k * P and determine: r = x_C (mod q),
Where x_C is the x-coordinate of the point C.
If r = 0, return to step 3.

Step 5. Calculate the value: s = (r * d + k * e) (mod q)
If s = 0, return to Step 3.

Step 6. Calculate the binary vectors R and S, corresponding to r and s, and determine the digital signature zeta = (R || S) as a concatenation of these two binary vectors.

The initial data of this process are the signature key d and the message M to be signed.
The output result is the digital signature zeta.

Digital Signature Verification

To verify the digital signature for the received message M, it is necessary to perform the following actions (steps).

Step 1. Calculate the integers r and s using the received signature zeta. If the inequalities 0 < r < q, 0 < s < q hold, go to the next step. Otherwise, the signature is invalid.

Step 2. Calculate the hash code of the received message M: H = h(M)

Step 3. Calculate the integer alpha, the binary representation of which is the vector H, and determine if: e = alpha (mod q)
If e = 0, then assign e = 1.

Step 4. Calculate the value: v = e^(-1) (mod q)

Step 5. Calculate the values:
z1 = s * v (mod q), z2 = -r * v (mod q)

Step 6. Calculate the elliptic curve point C = z1 * P + z2 * Q and determine: R = x_C (mod q),
where x_C is x-coordinate of the point.

Step 7. If the equality R = r holds, then the signature is accepted. Otherwise, the signature is invalid.

The input data of the process are the signed message M, the digital signature zeta, and the verification key Q.
The output result is the witness of the signature validity or invalidity.