SM9 identity-based cryptographic algorithms

Part 2: Digital signature algorithm

Contents

1 Scope 1

2 Normative references 1

3 Terms and definitions 1
3.1 message 1
3.2 signed message 1
3.3 signature key 1
3.4 signature master key 1
3.5 identity 2
3.6 key generation center (KGC) 2

4 Symbols 2

5 Algorithm parameters and auxiliary functions 3
5.1 Overview 3
5.2 System parameters 3
5.3 Generation of the signature master key and the user's signature private key 4
5.4 Auxiliary functions 4
5.4.1 Overview 4
5.4.2 Cryptographic hash functions 4
5.4.3 Random number generators 5

6 Digital signature generation algorithm and its process 5
6.1 Digital signature generation algorithm 5
6.2 Digital signature generation process 6

7 Digital signature verification algorithm and its process 7
7.1 Digital signature verification algorithm 7
7.2 Digital signature verification process 7

SM9 identity-based cryptographic algorithms

Part 2: Digital signature algorithm

1 Scope

This part of GM/T 0044–2016 specifies an identity-based digital signature algorithm built upon pairings on elliptic curves, including the digital signature generation and verification algorithms together with their corresponding processes.

This part of GM/T 0044–2016 is applicable for a receiver to use the signer's identity to verify data integrity and the sender's identity; and for a third party to verify authenticity of a signature and of the signed message.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

GM/T 0004–2016, SM3 cryptographic hash algorithm

GM/T 0044.1–2016, SM9 identity-based cryptographic algorithms — Part 1: General

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1 message

bit string of finite length

3.2 signed message

group of data elements that consists of a message and its digital signature

3.3 signature key

private key of the signer; the secret data element used by the signer in the digital signature generating process

3.4 signature master key

topmost key in the key hierarchy of an identity–based cryptographic system, composed of the signature master private key and the signature master public key. The signature master public key is publicly available, while the signature master private key is kept secret by the KGC. The KGC generates the user's signature private key by using the signature master private key and the user's identity. In an identity–based cryptographic system, the signature master private key is usually generated by the KGC using random number generators, while the signature master public key is generated with the signature master private key and the system parameters

3.5 identity

information that can be used to confirm the identity of an entity, composed of non-repudiable information about the entity, such as its distinguished name, email address, identity card number, and telephone number

3.6 key generation center (KGC)

trusted authority responsible for the selection of system parameters, generation of the signature master keys, and generation of users' signature private keys (in this part)

4 Symbols

The following symbols apply to this part.

A, B: two users using the identity-based cryptographic system

cf: cofactor of the order of an elliptic curve relative to N

cid: curve identifier that indicates the type of elliptic curve, denoted by one byte, where 0x10 represents an ordinary curve (a non-supersingular curve) over F_p (the prime number $p > 2^{191}$), 0x11 represents a supersingular curve over F_p , and 0x12 represents the ordinary curve and its twisted curve over F_p

ds_A: signature private key of the user A

e: a bilinear pairing from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$

eid: bilinear pairing identifier to distinguish the type of the bilinear pairing e, denoted by one byte, where 0x01 represents the Tate pairing, 0x02 represents the Weil pairing, 0x03 represents the Ate pairing, and 0x04 represents the R-Ate pairing

$\mathbb{G}_T$: a multiplicative cyclic group of prime order N

$\mathbb{G}_1$: an additive cyclic group of prime order N

$\mathbb{G}_2$: an additive cyclic group of prime order N

g^u: g to the power of u, where g is an element in the multiplicative group $\mathbb{G}_T$ and u is a positive integer, that is $g^u = \underbrace{g \cdot g \cdot \cdots \cdot g}_{u\ g's}$

$H_v()$: a cryptographic hash function

$H_1(), H_2()$: cryptographic functions derived from the cryptographic hash function

hid: identifier of the signature private key generating function, denoted by one byte, selected and made public by the KGC

(h, S): the sent signature

(h', S'): the received signature

ID_A: the identity of the user A that uniquely determines the public key of A

M: the message to be signed

M': the message to be verified

mod n: the operation of modulo n, for example, $23 \bmod 7 = 2$

N: the order of the cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, which is a prime number greater than 2^{191}

P_{pub-s}: the signature master public key

P_1: a generator of $\mathbb{G}_1$

P_2: a generator of $\mathbb{G}_2$

ks: the signature master private key

$\langle P \rangle$: the cyclic group generated by the element P

$[u]P$: the u multiple of the element P in the additive groups $\mathbb{G}_1$ or $\mathbb{G}_2$

$\lceil x \rceil$: ceiling function that maps to the smallest integer not less than x, for example, $\lceil 7 \rceil = 7$, $\lceil 8.3 \rceil = 9$

$\lfloor x \rfloor$: floor function that maps to the largest integer not greater than x, for example, $\lfloor 7 \rfloor = 7$, $\lfloor 8.3 \rfloor = 8$

$x||y$: the concatenation of x and y, where x and y are bit strings or byte strings

$[x, y]$: the set of integers which are not less than x and not greater than y

β: the twisted curve parameter

5 Algorithm parameters and auxiliary functions

5.1 Overview

This part specifies an identity-based digital signature algorithm realized by pairings from elliptic curves. The signer possesses an identity and a corresponding private key for signing. The signature private key is generated by the KGC using the signature master private key and the signer's identity. The signer uses its signature private key to sign the message and to generate a digital signature. The verifier verifies the authenticity of the signature by using the signer's identity.

Before generating and verifying the signature, the message to be signed M and the message to be verified M' shall be processed by a cryptographic hash function.

5.2 System parameters

The system parameters include: the curve identifier cid, the parameters of the elliptic curve base field F_q, the parameters of the elliptic curve equation a and b, the twisted curve parameter β (if the least significant 4 bits of cid is 2), the prime factor N of the order of the curve and the cofactor cf relative to N, the embedding degree k of the curve $E(F_q)$ relative to N, a generator P_1 of the cyclic subgroup $\mathbb{G}_1$ of $E(F_{q^{d_1}})$ of order N (where d_1 divides k), a generator P_2 of the cyclic subgroup of $E(F_{q^{d_2}})$ of order N (where d_2 divides k), the bilinear pairing identifier eid of e, and optionally the homomorphism Ψ from $\mathbb{G}_2$ to $\mathbb{G}_1$.

The range of the bilinear pairing e is the multiplicative cyclic group $\mathbb{G}_T$ of order N.

For detailed descriptions of the system parameters as well as their verification, please refer to Clause 7 in GM/T 0044.1–2016.

5.3 Generation of the signature master key and the user's signature private key

The KGC generates a random number $ks \in [1, N-1]$ as the signature master private key, computes the element $P_{pub-s} = [ks]P_2$ in $\mathbb{G}_2$ as the signature master public key, and then the signature master key pair is (ks, P_{pub-s}). The KGC keeps ks secret and makes P_{pub-s} public.

The KGC selects a one-byte signature private key generating function identifier hid and makes it public.

Let ID_A denote the user A's identity. To generate the signature private key ds_A of A, the KGC first computes $t_1 = H_1(ID_A||hid, N) + ks$ over the finite field F_N. If $t_1 = 0$, it regenerates the signature master private key, computes the signature master public key and makes it public, and updates the existing signature private keys of users. Otherwise, it computes $t_2 = ks \cdot t_1^{-1}$, and then computes $s_A = [t_2]P_1$.

5.4 Auxiliary functions

5.4.1 Overview

Two types of auxiliary functions are used in the identity-based digital signature algorithm specified in this part: cryptographic hash functions and random number generators.

5.4.2 Cryptographic hash functions

5.4.2.1 Cryptographic hash function $H_v()$

The output of the cryptographic hash function $H_v()$ is a v-bit hash value. This part adopts the cryptographic hash functions approved by the State Cryptography Administration such as the SM3 cryptographic hash algorithm.

5.4.2.2 Cryptographic function $H_1()$

The input of the cryptographic function $H_1(Z, n)$ is a bit string Z and an integer n, and its output is an integer $h_1 \in [1, n-1]$. $H_1(Z, n)$ invokes the cryptographic hash function $H_v()$ internally. $H_v()$ is specified in 5.4.2.1.

Cryptographic function $H_1(Z, n)$:

Input: a bit string Z and an integer n.

Output: an integer $h_1 \in [1, n-1]$.

Step 1: Initialize a 32-bit counter $ct = 0x00000001$;

Step 2: Compute $hlen = 8 \times \lceil (5 \times (\log_2 n))/32 \rceil$;

Step 3: For $i = 1$ to $\lceil hlen/v \rceil$:

Step 3.1: Compute $Ha_i = H_v(0x01||Z||ct)$;

Step 3.2: ct++;

Step 4: If $hlen/v$ is an integer, set $Ha!_{\lceil hlen/v \rceil} = Ha_{\lceil hlen/v \rceil}$. Otherwise, set $Ha!_{\lceil hlen/v \rceil}$ to be the leftmost $(hlen - (v \times \lfloor hlen/v \rfloor))$ bits of $Ha_{\lceil hlen/v \rceil}$.

Step 5: Set $Ha = Ha_1 || Ha_2 || \cdots || Ha_{\lceil hlen/v \rceil - 1} || Ha!_{\lceil hlen/v \rceil}$. Convert the data type of Ha to integer as specified in Clauses 6.2.4 and 6.2.3 of GM/T 0044.1–2016.

Step 6: Compute $h_1 = (Ha \bmod (n-1)) + 1$.

5.4.2.3 Cryptographic function $H_2()$

The input of the cryptographic function $H_2(Z, n)$ is a bit string Z and an integer n, and its output is an integer $h_2 \in [1, n-1]$. $H_2(Z, n)$ invokes the cryptographic hash function $H_v()$ internally. $H_v()$ is specified in 5.4.2.1.

Cryptographic function $H_2(Z, n)$:

Input: a bit string Z, an integer n.

Output: an integer $h_2 \in [1, n-1]$.

Step 1: Initialize a 32-bit counter $ct = 0x00000001$;

Step 2: Compute $hlen = 8 \times \lceil (5 \times (\log_2 n))/32 \rceil$;

Step 3: For $i = 1$ to $\lceil hlen/v \rceil$:

Step 3.1: Compute $Ha_i = H_v(0x02 || Z || ct)$;

Step 3.2: ct++;

Step 4: If $hlen/v$ is an integer, set $Ha!_{\lceil hlen/v \rceil} = Ha_{\lceil hlen/v \rceil}$. Otherwise, set $Ha!_{\lceil hlen/v \rceil}$ to be the leftmost $(hlen - (v \times \lfloor hlen/v \rfloor))$ bits of $Ha_{\lceil hlen/v \rceil}$.

Step 5: Set $Ha = Ha_1 || Ha_2 || \cdots || Ha_{\lceil hlen/v \rceil - 1} || Ha!_{\lceil hlen/v \rceil}$. Convert the data type of Ha to integer as specified in Clauses 6.2.4 and 6.2.3 of GM/T 0044.1–2016.

Step 6: Compute $h_2 = (Ha \bmod (n-1)) + 1$.

5.4.3 Random number generators

This part adopts random number generators approved by the State Cryptography Administration.

6 Digital signature generation algorithm and its process

6.1 Digital signature generation algorithm

Let M be the message to be signed. In order to obtain the signature (h, S) of the message M, user A performs the following operations as signer:

A1: Compute the element $g = e(P_1, P_{pub-s})$ in the group $\mathbb{G}_T$;

A2: Generate a random integer $r \in [1, N-1]$;

A3: Compute the element $w = g^r$ in the group $\mathbb{G}_T$, and convert the data type of w to a bit string as specified in Clauses 6.2.6 and 6.2.5 of GM/T 0044.1–2016.

A4: Compute the integer $h = H_2(M||w, N)$;

A5: Compute the integer $l = (r - h) \bmod N$; if $l = 0$, go to Step A2;

A6: Compute the element $S = [l]ds_A$ in the group $\mathbb{G}_1$;

A7: Convert the data type of h to a byte string as specified in Clause 6.2.2 of GM/T 0044.1–2016, convert the data type of S to a byte string as specified in Clause 6.2.8 in GM/T 0044.1–2016. Output (h, S) as the signature of message M.

6.2 Digital signature generation process

The process of the digital signature generation algorithm is shown in Figure 1.

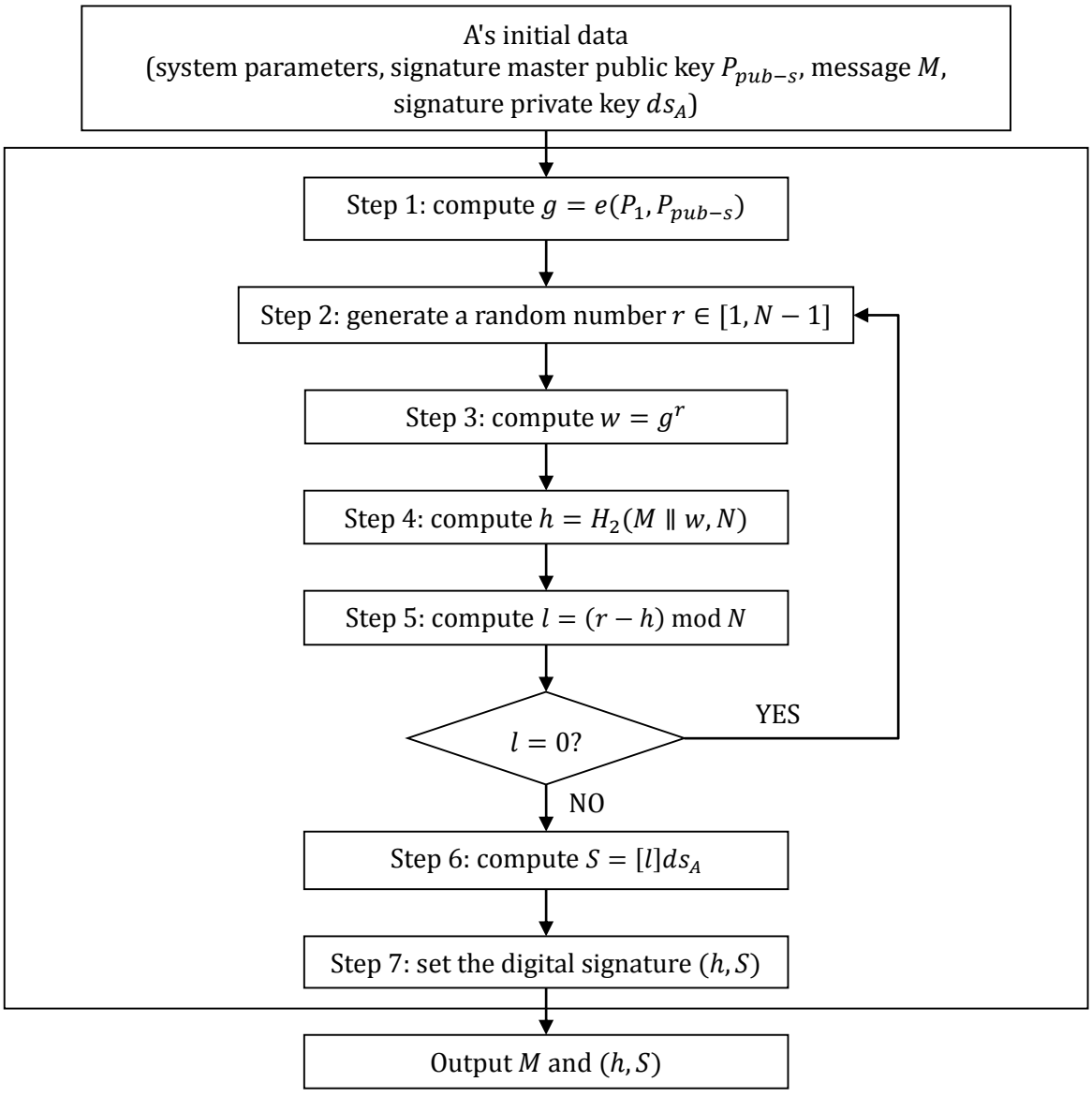

Figure 1: Digital signature generation process

7 Digital signature verification algorithm and its process

7.1 Digital signature verification algorithm

To verify the received message M' and its digital signature (h', S'), user B performs the following operations as receiver:

B1: Convert the data type of h' to an integer as specified in Clause 6.2.3 of GM/T 0044.1–2016. Check whether $h' \in [1, N-1]$ holds true. If it does not, the verification fails;

B2: Convert the data type of S' to a point on the elliptic curve as specified in Clause 6.2.9 of GM/T 0044.1–2016, then check whether $S' \in \mathbb{G}_1$ holds true as specified in Section 4.5 of GM/T 0044.1–2016. If it does not hold, the verification fails;

B3: Compute the element $g = e(P_1, P_{pub-s})$ in the group $\mathbb{G}_T$;

B4: Compute the element $t = g^{h'}$ in the group $\mathbb{G}_T$;

B5: Compute the integer $h_1 = H_1(ID_A || hid, N)$;

B6: Compute the element $P = [h_1]P_2 + P_{pub-s}$ in the group $\mathbb{G}_2$;

B7: Compute the element $u = e(S', P)$ in the group $\mathbb{G}_T$;

B8: Compute the element $w' = u \cdot t$ in the group $\mathbb{G}_T$. Convert the data type of w' to a bit string as specified in Clauses 6.2.6 and 6.2.5 of GM/T 0044.1–2016;

B9: Compute an integer $h_2 = H_2(M' || w', N)$ and check whether $h_2 = h'$. If so, the signature is valid. Otherwise, the validation fails.

7.2 Digital signature verification process

The process of the digital signature verification algorithm is shown in Figure 2.

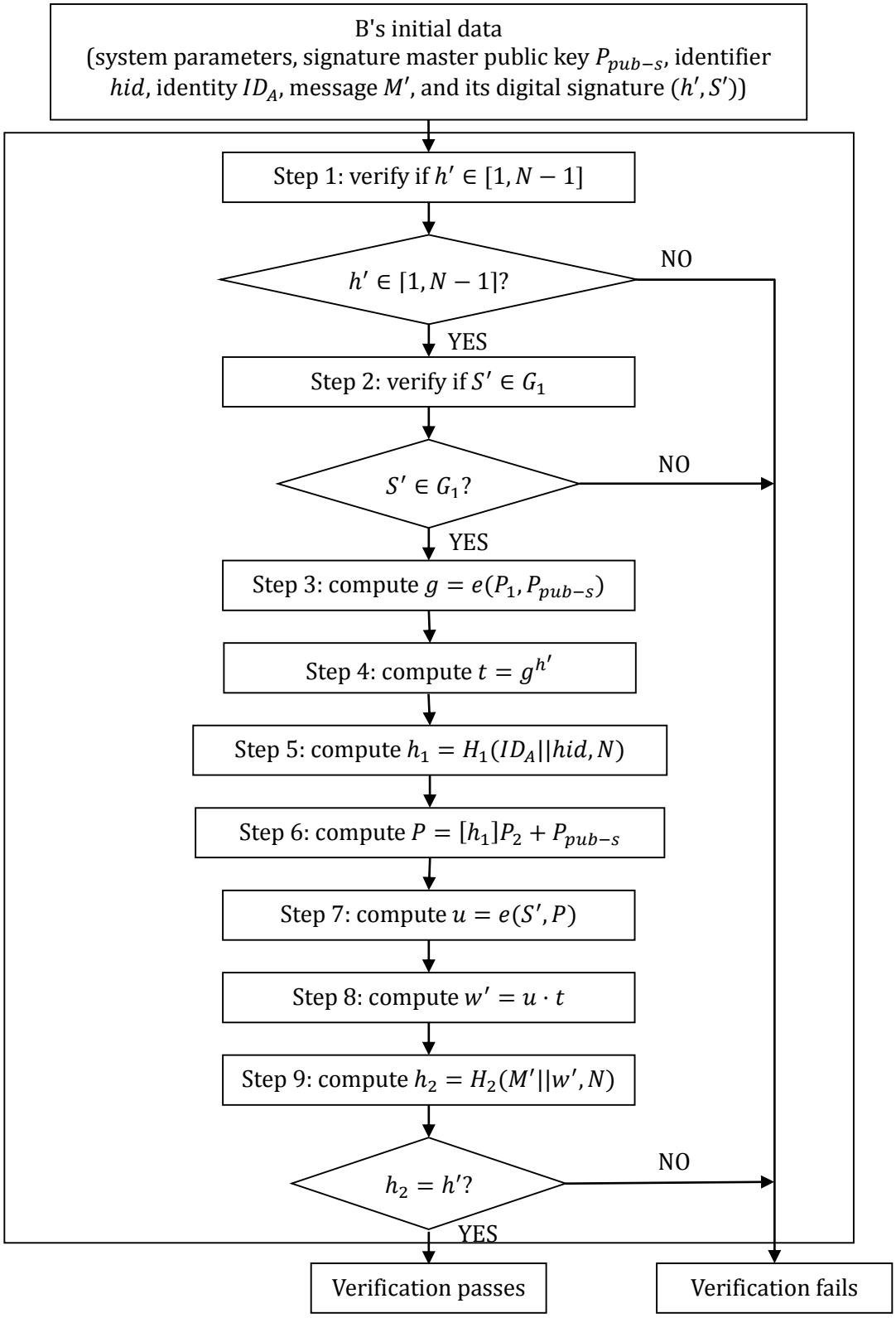

Figure 2: Digital signature verification process