

(43) International Publication Date
24 November 2011 (24.11.2011)(10) International Publication Number
WO 2011/144247 A1(51) International Patent Classification:
H04L 9/32 (2006.01)(21) International Application Number:
PCT/EP2010/057002(22) International Filing Date:
20 May 2010 (20.05.2010)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant (for all designated States except US):
HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P. [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

(72) Inventor; and

(75) Inventor/Applicant (for US only): **CHEN, Liquan** [GB/GB]; Mail Stop M21, Filton Road, Stoke Gifford, Bristol BS34 8QZ (GB).(74) Agent: **REES, Simon John Lewis**; Haseltine Lake LLP, Redcliff Quay, 120 Redcliff Street, Bristol, Avon BS1 6HU (GB).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: DIGITAL SIGNATURE METHOD AND APPARATUS

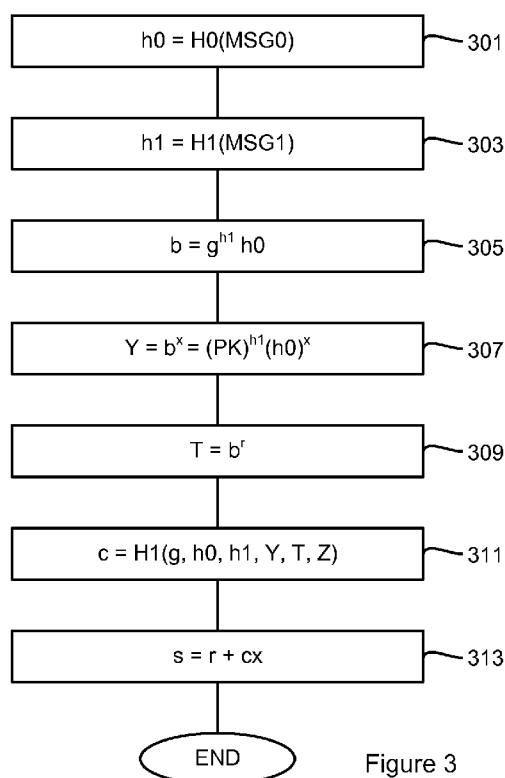


Figure 3

(57) Abstract: The digital signature method and apparatus enable a message to be signed anonymously but only once. The examples described in the various implementations offer a digital signature scheme that allows a verifier to detect that two signatures are from the same signer. The implementations offer a digital signature scheme that allows the verifier to determine the identity of the signer where two signatures are from the same signer, such that if any signer acts maliciously, i.e. submits two signatures on the same subject, the anonymity of the signer will be broken because the public key of the signer can be computed from the two signatures.

WO 2011/144247 A1



Published:

— *with international search report (Art. 21(3))*

DIGITAL SIGNATURE METHOD AND APPARATUS

Background

Electronic transactions frequently involve the use of digital signatures. One
5 feature of such digital signatures is the privacy of the signer.

The privacy of a signer in a digital signature scheme usually includes two
properties. One is signer anonymity, which means that a digital signature does
not reveal the identity of the signer. The other is signature unlinkability, which
10 means that it is not possible to tell whether two signatures have been signed by
the same signer or not.

There are several known methods of providing user privacy. These include ring
signatures, group signatures and a scheme known as Direct Anonymous
15 Attestation (DAA). Figure 1 shows how the degree of user privacy for each of
these schemes compares to a standard public key, i.e. which has no user
privacy. As can be seen DAA provides a more robust version of user privacy
compared to the others, since it provides both an anonymous and unlinkable
digital signature scheme in which the anonymity and unlinkability properties do
20 not rely on a fully trusted third party.

Sometimes, however, this signer privacy feature can be too strong, because a
malicious but still legitimate signer can abuse the system by creating multiple
signatures in the way that a verifier would think that the signatures might have
25 been created by multiple signers.

This can be a problem in applications whereby a legitimate user is asked to
make a signature on a certain message anonymously, but once only. The
following are examples of such applications:

30

- In an e-commerce system, each customer may be asked to provide a
single set of comments on particular goods or a particular buyer/seller, for

example when recommending certain goods or a certain buyer/seller in eBay or Amazon.

- In an e-voting system, each voter is allowed to make a single vote to
5 an arbitrary voting result according to his own choice.

- In an e-review system, each reviewer is required to make a single review report in relation to a certain item (for example a paper or a book).

10 It will be appreciated that user privacy can cause problems in such applications, because if a signer is malicious but still legitimate, the signer can successfully submit multiple signatures and pretend they have come from multiple users.

It follows that anonymity/unlinkability and the provision of multiple signing
15 controls are conflicting properties.

Brief description of the drawings

For a better understanding of the present invention, and to show more clearly
20 how it may be carried into effect, reference will now be made, by way of example only, to the following drawings in which:

Figure 1 illustrates how the degree of user privacy varies for different signature
25 schemes;

Figure 2a illustrates an example of an implementation of a signed message;

Figure 2b illustrates the steps performed by a signer computing entity according to an example of an implementation;

30 Figure 3 illustrates the steps performed by a signer computing entity according to an example an implementation;

Figure 4 illustrates the steps performed by a verifier computing entity according to an example of an implementation of validating a single signature;

- 5 Figure 5a illustrates the steps performed by a verifier computing entity according to an implementation of validating two signatures;

Figure 5b illustrates the steps performed by a verifier computing entity according to another implementation of validating two signatures;

10

Figure 6 illustrates the steps performed in an implementation for deriving the public key of a signer;

- 15 Figure 7 illustrates the steps performed by a signer computing entity according to an example of another implementation;

Figure 8 illustrates the steps performed by a verifier computing entity according to an example of another implementation of validating a single signature;

- 20 Figure 9 illustrates an example of an implementation of a signer computing entity apparatus; and

Figure 10 illustrates an example of an implementation of a verifier computing entity apparatus.

25

Detailed description

The implementations described below relate to a digital signature method and apparatus and in particular, but not exclusively, to a digital signature method and apparatus that enables a message to be signed anonymously but only

30 once.

The examples described in the various implementations offer a digital signature scheme that allows a verifier to detect that two messages relate to the same subject or topic, and that two signatures are from the same signer. The implementations described below also offer a digital signature scheme that

5 allows the verifier to determine the identity of the signer where two signatures are from the same signer. As such, if a signer acts maliciously, i.e. submits two signatures on the same subject, the anonymity of the signer will be broken because the public key of the signer can be computed from the two signatures. However, in circumstances where a signer has not acted maliciously, the

10 anonymity of the signer remains intact.

Before discussing the different examples of the implementations, an explanation will first be given of three types of anonymous signature schemes as follows:

15 1. Ring signatures. A ring signature convinces an external verifier that a message has been signed by one of a number of possible independent signers without allowing the verifier to identify the signer. Thus, if a single signer creates two signatures, from the point of view of a verifier these two signatures could have been signed by any one or two signers within the signer ring. As such, it is

20 not possible to determine whether a malicious signer has anonymously signed a message more than once.

2. Group signatures. If a single group member creates two signatures, a group manager could in theory determine that this has occurred, since the group

25 manager has authorisation rights to identify any signer from his signature. However, in many applications, potential problems can arise as follows. For example, a super manager can be a bottle neck because the super manager can break signer privacy. Also, such a super manager is most likely to be operating "offline", and therefore only available when an actual dispute occurs.

30 However, since multiple signing is not detectable, nobody will ever report such a "dispute" for the super manager to investigate.

3. Direct Anonymous Attestation (DAA). DAA provides a stronger scheme of user privacy compared to the ring signature scheme and group signature scheme described above, but as a consequence also has the largest problem with respect to multiple signing controls. If a single signer creates two

5 signatures by using two different basenames, from the point of view of any third party (including the DAA issuer) these two signatures could have been signed by any one or two signers within the whole set of DAA signers associated with the same DAA issuer.

10 A conventional signature scheme based on a public key operates as follows:

Let the private key of a signer be x .

Let the public key of the signer be y ,

15

From the point of view of a verifier who receives a message m signed by a public key y , the signature s equals:

$$s = \text{Sign}(x, m).$$

20

The verifier can determine whether the signature is valid with respect to y by checking:

$$1/0 = \text{Verify}(y, m, s),$$

25

where "1" indicates *accept* and "0" *reject*.

Now consider that the signer is part of a group, the group having a group public key G_y and the signer, say i , within that group having a group membership

30 private key x_i .

If a verifier receives a message signed by a signer from that group, i.e. (m, s, Gy) , where:

$$s = GSign(x_i, m), \text{ and}$$

5

$$1/0 = GVerify(Gy, m, s),$$

then the message is anonymously signed to the extent that a verifier cannot tell who signed the message, only that someone from that group signed the
10 message.

Figure 2a shows an example of an implementation for providing signing controls on a message m , such as multiple signing controls. For certain applications, for example where each signer is asked to make a single signature on a message
15 m , the signed message m is partitioned into first and second parts. A first part $MSG0$ relates to a message title (or message identifier) and the second part $MSG1$ relates to the message body.

For example, in an e-commerce application, the first message part $MSG0$ could
20 relate to a subject such as "review on Handbook of Applied Cryptography", while the second message part $MSG1$ could relate to a response randomly created by a particular signer. In another e-commerce application, the first message part $MSG0$ could relate to a subject such as "comment on GOODS 1234", while the second message part $MSG1$ could relate to a comment
25 generated by a particular customer. In an electronic voting system the first message part $MSG0$ could relate to a subject such as "Gloucestershire Council Vote 2010", while the second message part $MSG1$ could be the vote listed by a particular signer. It will be appreciated that these are merely examples, and that the implementations described herein can be used with any application whereby
30 it is desirable to be able to determine whether a message has been signed by the same party two or more times, and/or to identify the signer in a situation

where a particular message has been signed by the same signer two or more times.

The first message part *MSG0* can be used as a form of index to a particular message subject, while the second message part *MSG1* can be used as a distinguishing symbol of a signature.

Referring to Figure 2b, step 201 shows how a message *m* to be signed is partitioned into a first message part *MSG0* and a second message part *MSG1*. The first message part *MSG0* is mapped using a first hash function *H0* to form a first digest *h0*, step 203, and the second message part *MSG1* is mapped using a second hash function *H1* to form a second digest *h1*. The first digest *h0* and second digest *h1* are combined in step 207 to form a temporary base value *b* (the temporary base value *b* being a form of ephemeral base value or session base value). In step 209 a temporary public key *PK* is formed. The message *m* is signed using the temporary base value *b* and the temporary public key *Y*, step 211.

Further details of this implementation will be described below, together with details of how a verifier computing entity is able to determine that a message has been signed twice by the same signer, and how the identify of a signer may be identified.

It is noted that the implementations can be used with any form of digital signature technique, and with any of the anonymity schemes described above.

According to an embodiment, a digital signature method uses a digital signature generated using a temporary or ephemeral base value and a corresponding public key in conjunction with a DAA signature or a group signature as described above.

A signature of this type can be a Schnorr-type digital signature, although it will be appreciated to those skilled in the art that other such digital signature schemes can be similarly employed. A suitable digital signature scheme is described in, for example, "Efficient signature generation by smart cards" C. Schnorr., in the Journal of Cryptology, 4(3):161-174, 1991, the contents of which are incorporated herein in their entirety by reference.

According to this signature scheme, there are two large primes p and q selected such that $q|p-1$, (whereby $q|p-1$ means that q is a factor of $p-1$, i.e. for an integer k , the equation $kq = p-1$ holds), which are published with an element g of $(\mathbb{Z}/p\mathbb{Z})^*$ of order q . A signer with private key x and public key $y = g^x$ signs a message m as follows:

- The signer chooses a random number $r \in \mathbb{Z}_q^*$;
- The signer computes the challenge c and signature s :

$$c = \text{Hash}(g^r, m)$$

$$s = r + x c$$
- The signer outputs (c, s) as its signature on message m .

Verification that the signature is from the sender and is on message m is effected by a verifier (who knows the public key of the signer y , i.e. g^x) by seeking to generate a value c' that should match the value of c in the signature.

The verifier does this as follows:

- The verifier first computes:

$$g^r = g^s \cdot y^{-c}$$
- The verifier then computes:

$$c' = \text{Hash}(g^r, m)$$
 where m is the received message
- The verifier then compares c' and c which, if equal, proves that the message m was signed by the party having the public key $g^x \bmod p$.

Figure 3 illustrates an example of the steps that may be performed in a signer computing entity according to one implementation. Let $G = \langle g \rangle$ be a group of large primes of order q , wherein each signer has a public key, $PK = g^x$ in G and the corresponding private key SK is the integer x such that $1 < x < q$. Each

5 signer also has a group member credential “cre” to achieve signer anonymity, since with the values of x and cre , the signer can create a DAA signature or group signature, which can be verified without the value PK .

In step 301 the first message part is mapped using a first hash function $H0$ to

10 compute a first digest $h0$, where:

$$h0 = H0(MSG0)$$

and where $H0: \{0, 1\}^* \rightarrow G$.

15

In step 303 the second message part is mapped using a second hash function $H1$ to compute a second digest $h1$, where:

$$h1 = H1(MSG1)$$

20

and where $H1: \{0, 1\}^* \rightarrow \{0, 1\}^{|q|}$, (where the value $|q|$ is the bit length of q).

In step 305, an element b of G is computed (b being a temporary or ephemeral base value) as:

25

$$b = g^{h1} h0$$

In step 307, a temporary public key Y is computed, where:

30

$$Y = b^x = (PK)^{h1} (h0)^x$$

and where x is the private key SK of the signer.

In step 309, a commitment value T is computed, where:

$$T = b^r$$

5

and where r is a random number.

In step 311, a challenge value c is computed, where:

$$c = H1(g, h0, h1, Y, T, Z)$$

10

and where Z denotes some other part of the signature used in either the DAA signature or the group signature.

15 Then, in step 313 the signature s is computed as:

$$s = r + cx$$

In the implementation of the signing method described above, the final
20 signature includes the values (Y, c, s, Z) on the message m which is comprised of a first message part $MSG0$ and a second message part $MSG1$. The values (c, s) are also part of the DAA signature or group signature.

Using the signature that has been generated according to the implementation
25 shown in Figure 3, the following steps may be performed in an implementation of the verification method by a verifier computing entity.

Referring to Figure 4, the method steps describe an implementation of a
procedure for verifying whether or not the signature of a signer is valid. In step
30 401 the first part of the received message is mapped using a first hash function $H0$ to compute a first digest $h0$, where:

$$h0 = H0(MSG0)$$

In step 403, the second part of the received message is mapped using a second hash function $H0$ to compute a second digest $h1$, where:

5

$$h1 = H1(MSG1)$$

In step 405, an element b of G is computed, (b being a temporary or ephemeral base value) where:

10

$$b = g^{h1} h0$$

In step 407, a commitment value T is computed, where:

15

$$T = b^s Y^{-c}$$

Then in step 409, it is determined whether the following equation holds true.

$$c = H1(g, h0, h1, Y, T, Z)$$

20

If the equation holds, then the signature s is deemed to be valid, step 411.

Otherwise, if the equation does not hold, the signature s is deemed to be invalid, step 413.

25 Thus, in order to determine whether two signatures were signed by the same signer or not, and/or to find the identity of the signer if they were (as will be described below), the verifier can first perform the validation procedure described above with reference to Figure 4, in order to verify the validation of each signature. Otherwise, if either of the two signatures being compared were
30 invalid, then finding the linkage or the identity of the signer would not necessarily be accurate, since any invalid signature could have been forged.

There are four possible cases regarding the message to be signed:

1. If a signer signs first and second messages $m1$, $m2$, whereby the first message part $MSG0$ and the second message part $MSG1$ are the same in both messages $m1$, $m2$, the values b and Y will appear in both the signatures. This means that any receiving party, i.e. verifier computing entity, can detect that the two signatures are in fact the same signature.

For example, referring to Figure 5a, this shows an implementation of a method that is first able to determine that two received messages relate to the same subject or topic, i.e. that the first message part $MSG0$ of the first received message $m1$ is the same as the first message part $MSG0$ of the second received message $m2$. The verifier computing entity receives a first message $m1$ in step 501, and a second message $m2$ in step 503. The verifier computing entity may (or may not) have already processed these messages as shown in the implementation of Figure 4 (i.e. to first determine that each signature is valid). In step 505 the verifier computing entity uses the temporary base value b of the first received message $m1$ and the temporary base value b of the second received message $m2$ to determine whether the first message part $MSG0$ of the first received message $m1$ is the same as the first message part $MSG0$ of the second received message $m2$. As such, the verifier computing entity is able to determine using the temporary base values b whether or not two messages relate to the same subject or topic. In some applications this procedure alone can be used to provide useful information, such as how many votes have been received on a certain ballot, or how many reviews have been received against a certain book.

Figure 5b shows an implementation of a method that is able to determine that two received messages have been signed by the same signer computing entity. The verifier computing entity receives a first message $m1$ in step 511, and a second message $m2$ in step 513. The verifier computing entity may (or may not) have already processed these messages as shown in the implementation of

Figure 4 (i.e. to first determine that each signature is valid). The verifier computing entity determines whether the first and second received messages $m1$, $m2$ have been signed by the same computer signing entity, based on the temporary base value b and the temporary public key Y of the first received message $m1$ and the temporary base value b and the temporary public key Y of the second received message $m2$. For example, in the example of the implementation shown in Figure 5b, the verifier computing entity compares the values b and Y of message $m1$ with the values b and Y of message $m2$, step 515. In step 517 the verifier determines whether the values b and Y of message $m1$ match the values b and Y of message $m2$. If the signatures of messages $m1$ and $m2$ have the same values of b and Y , i.e. the temporary base values and temporary public keys match, this provides an indication that messages $m1$ and $m2$ have been signed by the same signer, step 519, such that the verifier computing entity can then take appropriate action. The action taken by a verifier computing entity can vary depending on a particular application. For example, the verifier computing entity may decide to ignore any messages that have been received from a signer that has signed twice, or the verifier computing entity may decide to only take one signature into account and ignore the remaining one (or remaining ones if multiple messages have been signed by the same signer). From the above it will be appreciated that the temporary base values b of the first and second messages $m1$, $m2$ enable the verifier computing entity to determine that the first and second messages $m1$, $m2$ relate to the same subject or topic, while the temporary public keys Y of the first and second messages $m1$, $m2$ enable the verifier computing entity to determine that the messages $m1$, $m2$ have been signed by the same signer computing entity.

In the implementation above, if the two signatures have the same value of b and Y the verifier computing entity is able to determine that these two signatures are signed by the same signer, but cannot determine the identity of the signer. However, in certain applications where one wants to avoid this from happening, the signer computing entity and verifier computing entity can agree to introduce some randomness into the second message part $MSG1$ during the original

signing procedure. For example, let $MSG1 = MSG1 \parallel Z$ (i.e. let $MSG1$ be the concatenation of the original $MSG1$ and other part of the DAA or group signature). As a result the values " b " and " Y " do not have the same value, in any two signatures. Such a modification provides the same result as in the next

5 case, in which the identity of the signer can be discovered from any two signatures with the same $MSG0$ and different $MSG1$ s signed by the same signer. In other words, this means that the identity of a malicious signer computing entity can be determined (for example as described below in the implementation where it is assumed that the second message parts $MSG1$ of

10 the first and second messages are purposely made different by the signer computing entity).

2. If a signer computing entity signs first and second messages $m1$, $m2$, such that the first and second messages $m1$, $m2$ each have the same first message

15 part $MSG0$ but have different second message parts $MSG1$, (for example the first message $m1$ having a second message part $MSG1_\alpha$ and the second message $m2$ having a second message part $MSG1_\beta$), there will be one digest value $h0$, two digest values $h1$, two temporary base values b and two temporary public key values Y . These are shown in steps 601 to 613 of Figure 6, whereby:

20
$$h0 = H0(MSG0),$$

$$h1_\alpha = H1(MSG1_\alpha),$$

$$h1_\beta = H1(MSG1_\beta),$$

25

$$b_\alpha = g^{h1_\alpha} h0,$$

$$b_\beta = g^{h1_\beta} h0,$$

30

$$Y_\alpha = b_\alpha^x,$$

$$Y_\beta = b_\beta^x.$$

In this implementation the value of the public key PK of the signer can be computed in step 615 as follows:

$$5 \quad PK = (Y_\alpha / Y_\beta)^{1/(h1\alpha - h1\beta)}$$

From the above, it can be seen that the implementation enables the identity of the signer to be determined.

- 10 It is noted that, if two signatures with the same *MSG0* are signed by two different signer computing entities, since these two signers have different *x* values, say *x1* and *x2*, then $h0^{x1}$ will not be equal to $h0^{x2}$, and hence $(Y_\alpha / Y_\beta)^{1/(h1\alpha - h1\beta)}$ will not be equal to either *PK1* or *PK2*, but a random value with a very high probability. As such, it is not possible to determine the public key of a
- 15 signer when two signatures with the same *MSG0* are signed by two different signers, hence keeping the anonymity of a signer in a legitimate scenario.

3. If a signer computing entity creates two signatures with different first message parts *MSG0* but the same second message part *MSG1*, these two
- 20 signatures are unlinkable, since $H0(MSG0_\alpha)^x$ and $H0(MSG0_\beta)^x$ do not show the fact that the same *x* was used. As such, the anonymity and unlinkability of the signer is maintained. This is a desired result, because the signer will not have been behaving maliciously, but merely providing comments on two different subjects (i.e. two different first message parts *MSG0s*).

25

- For example, in an e-commerce application, a signer may wish to give a comment such as "Good Book" on two separate books, one book being "Handbook of Applied Cryptography" and the other book being "Digital Signature Techniques". In such a scenario, although the second message part
- 30 *MSG1₁* of the first message *m1* is the same as the second message part *MSG1₂* of the second message *m2* (i.e. both relating to "Good Book"), the first message part *MSG0₁* of the first message *m1* will differ from the first message

MSG0₂ of the second message *m2* (i.e. *MSG0₁* relating to "review on Handbook of Applied Cryptography" and *MSG0₂* relating to "review on Digital Signature Techniques"). It will be appreciated that such a scenario is a legitimate signature procedure by the signer, and as such the unlinkability of the two
5 signatures is a desirable result.

4. If a signer computing entity creates two signatures with different first message parts *MSG0* and different second message parts *MSG1*, these two signatures are unlinkable. As in the third scenario described above, this is a
10 desirable result since this is a legitimate signing procedure by the signer.

The implementations described above have the advantage of enabling a malicious signing behaviour by a signer to be detected, and where necessary the identity of the malicious signer to be determined.
15

The digital signature scheme described in the implementations above can form part of an anonymous digital signature scheme, such as a group signature scheme or a direct anonymous attestation (DAA) scheme.

20 For example, in one implementation, when used in the DAA scheme, the first and second message parts *MSG0*, *MSG1* may be used as a replacement to the basenames used in DAA (i.e. in place of the basenames of DAA). Such an implementation involves the computing of a challenge *c* as described in further detail later in the application.

25 According to another implementation, when used in the DAA scheme, the first message part and the second message part *MSG0*, *MSG1* may be used alongside, or in parallel to the basenames, i.e. rather than being a replacement of the basename. Such an implementation may be used when the basename
30 might still need to play its own role of the user-controlled linkability. Such an implementation involves the computing of a challenge *c* as described in further detail later in the application.

The following is an example of an implementation where the first and second message parts *MSG0*, *MSG1* are used as replacements to the basenames in DAA.

5

Figure 7 shows the steps performed by a signing computing entity during a signing procedure in such an implementation, where BSN is a basename of a DAA scheme.

10 In step 701 a digest value *h* is computed, where:

$$h = H0(BSN)$$

In step 703 a value *X* is computed, where:

15

$$X = h^x$$

where *x* is the private key *SK* of the signer.

20 In step 705 a value *R* is computed, where:

$$R = h^r$$

where *r* is a random number.

25

In step 707 the challenge *c* is computed as:

$$c = H1(h, X, R, Z)$$

30 where *Z* denotes some other part of the signature used in the DAA signature.

Then, in step 709 the signature *s* is computed as:

$$s = r + cx$$

Figure 8 shows the steps performed during verification by a verifier computing
 5 entity in relation to a message signed by a signer computing entity as described
 in Figure 7.

In step 801 a digest value h is computed, where:

$$10 \quad h = H0(BSN)$$

In step 803 a value R is computed, where:

$$15 \quad R = h^s X^{-c}$$

In step 805 it is determined whether the following equation holds true:

$$c = H1(h, X, R, Z)$$

20 If the equation holds, then the signature is deemed to be valid, step 807.
 Otherwise, if the equation does not hold, the signature is deemed invalid, step
 809.

It is noted that in an implementation where the first and second message parts
 25 $MSG0$, $MSG1$ are used in place of the basenames in DAA, the values Y , T , b ,
 $h0$, $h1$ are used in place of the value X , R , h shown above. In an
 implementation where the first and second message parts $MSG0$, $MSG1$ are
 used in parallel with the basenames, all of the values are kept (i.e. all of h , $h0$,
 $h1$, X , Y , R , T), in which case the challenge is written as:

$$30 \quad c = H1(h, h0, h1, b, X, Y, R, T, Z)$$

where Z denotes some other part of the signature used in the DAA signature.

Other parts of the implementations described above for determining whether or not two messages have been signed by the same signer, or determining the identity of the signer, are performed the same regardless of whether the first and second message parts $MSG0$, $MSG1$ are used in place of, or in parallel with the basenames in a DAA scheme.

It is therefore noted that in any group signature scheme or direct anonymous attestation schemes, an implementation can have either or both of the basename and $MSG0/MSG1$. This implementation enables a flexible threshold signature verification to be built. For example, an arbitrary verifier can find out how many signers have signed a certain event $MSG0$ without revealing who they are. Based on the total number of second message parts $MSG1$ s or the number of positive or negative second message parts $MSG1$ s, the verifier can decide whether the result passes a threshold or not.

The implementations described above differ from a basename of a verifier in the DAA scheme in that the implementations provide a message specific signature, instead of a signature specific to a verifier's input. The implementations described above therefore break the anonymity of a malicious signer, which is not otherwise possible in DAA.

Figure 9 shows an implementation of digital signature apparatus, and in particular a signer computing entity apparatus 901. The signer computing entity apparatus 901 is adapted to provide a digital signature by which the signer computing entity is able to perform a one-time anonymous signature on a message m . The signer computing entity apparatus 901 comprises a partitioning unit 903 adapted to partition the message m into a first message part $MSG0$ and a second message part $MSG1$. A mapping unit 905 is adapted to map the first message part $MSG0$ with a first hash function $H0$ to form a first digest $h0$, and map the second message part $MSG1$ with a second hash

function $H1$ to form a second digest $h1$. A combining unit 907 is adapted to combine the first digest $h0$ and the second digest $h1$ to form a temporary base value b . A forming unit 909 is adapted to form a temporary public key Y . The public key Y may be formed from the temporary base value b and the secret
5 key SK . A signing unit 911 is adapted to sign the message m using the temporary base value b and the temporary public key Y (which is based on the secret key SK).

Figure 10 shows an implementation of digital signature apparatus, and in particular a verifier computing entity apparatus 1001. The verifier computing
10 entity apparatus 1001 is adapted to determine whether first and second messages have the same first message part $MSG0$, and comprises a receiving unit 1003 adapted to receive a first message $m1$ and a second message $m2$ which have been signed using a signer computing entity apparatus, for example
15 an apparatus as described in Figure 9 above. The verifier computing entity apparatus 1001 comprises a processing unit 1005 adapted to determine whether the first message part $MSG0$ of the first received message $m1$ is the same as the first message part $MSG0$ of the second received message $m2$, based on the temporary base value b of the first received message $m1$ and the
20 temporary base b of the second received message $m2$.

The verifier computing entity apparatus 1001 may be further configured such that the processing unit 1005 is adapted to determine whether the first and second received messages $m1$, $m2$ have been signed by the same computer
25 signing entity, based on the temporary public key Y of the first received message $m1$ and the temporary public key Y of the second received message $m2$.

The verifier computing entity apparatus 1001 may be further configured such
30 that the processing unit 1005 is adapted to determine the public key PK of the signer computing entity if a second message part $MSG1_\alpha$ of the first received

message $m1$ differs from a second message part $MSG1_{\beta}$ of the second received message $m2$.

The processing unit 1005 may be adapted to determine the public key PK of the
5 signer computer entity by computing:

$$PK = (Y_{\alpha} / Y_{\beta})^{1/(h1_{\alpha} - h1_{\beta})}, \text{ where:}$$

$$h1_{\alpha} = H1(MSG1_{\alpha}),$$

10

$$h1_{\beta} = H1(MSG1_{\beta}),$$

$$b_{\alpha} = g^{h1_{\alpha}} h0,$$

15

$$b_{\beta} = g^{h1_{\beta}} h0,$$

$$Y_{\alpha} = b_{\alpha}^x, \text{ and}$$

$$Y_{\beta} = b_{\beta}^x.$$

20

The implementations described above have the advantage of being easily incorporated into any group signature scheme or DAA signature scheme.

The implementations described above also have the advantage of enabling a
25 flexible threshold signature scheme to be provided for free, for example without a trusted threshold share distributor.

The implementations described above are also efficient in terms of computational complexity.

30

It will be appreciated that the security of the implementations in the examples described above are based on the security of the Schnorr signature scheme,

which holds a known provable security. It is noted, however, that other signature schemes, i.e. other than the Schnorr signature scheme may be used.

The implementations described above have the advantage of holding the nature
5 of user-controlled privacy, and offer a good balance between strong privacy and robust security.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design many
10 alternative embodiments without departing from the scope of the appended claims. The word "comprising" does not exclude the presence of elements or steps other than those listed in a claim, "a" or "an" does not exclude a plurality, and a single processor or other unit may fulfil the functions of several units recited in the claims. Any reference signs in the claims shall not be construed
15 so as to limit their scope.

CLAIMS

1. A digital signature method by which a signer computing entity is able to perform a one-time anonymous signature on a message m , the method comprising the signer computing entity:
 - partitioning the message m into a first message part $MSG0$ and a second message part $MSG1$;
 - mapping the first message part $MSG0$ with a first hash function $H0$ to form a first digest $h0$;
 - mapping the second message part $MSG1$ with a second hash function $H1$ to form a second digest $h1$;
 - combining the first digest and second digest to form a temporary base value b ;
 - forming a temporary public key Y ; and
 - signing the message m using the temporary base value b and the temporary public key Y .
2. A method as claimed in claim 1 wherein:
 - the step of forming the temporary base value b comprises computing

$$b = g^{h1} h0 \text{ in } G,$$
 whereby $G = \langle g \rangle$, a group of a large prime order q , wherein each signer computing entity has a public key PK , wherein $PK = g^x$ in G , wherein x is an integer corresponding to a private key SK of the signer computing entity, and wherein x is greater than one and less than q ; and
 - the step of forming the temporary public key comprises computing

$$Y = b^x = (PK)^{h1} (h0)^x \text{ in } G.$$
3. A method as claimed in claim 2, wherein:
 - the first digest $h0 = H0(MSG0)$,
 - the first hash function $H0$ relates to the function $\{0, 1\}^* \rightarrow G$.
 - the second digest $h1 = H1(MSG1)$;
 - the second hash function $H1$ relates to the function $\{0, 1\}^* \rightarrow \{0, 1\}^{|q|}$;

and wherein the method further comprises the steps of:

computing a commitment value T , wherein $T = b^r$ in G , and wherein r is a random number; computing a challenge value c , wherein $c = H1(g, h_0, h_1, Y, T, Z)$, and wherein Z denotes some other part of the signature used in either
5 a direct anonymous attestation (DAA) signature or a group signature; and
wherein the signature s of the message m is computed as
$$s = r + cx \bmod q.$$

4. A digital signature method in a verifier computing entity, the method
10 comprising the steps of:
receiving a first message m_1 which has been signed according to the method defined in claim 1,
receiving a second message m_2 which has been signed according to the method defined in claim 1; and
15 determining whether the first message part MSG_0 of the first received message m_1 is the same as the first message part MSG_0 of the second received message m_2 , based on the temporary base value b of the first received message m_1 and the temporary base value b of the second received message m_2 .

20

5. A method as claimed in claim 4, wherein the verifier computing entity further performs the steps of:
determining whether the first and second received messages m_1, m_2 have been signed by the same computer signing entity, based on the temporary
25 public key Y of the first received message m_1 and the temporary public key Y of the second received message m_2 .

6. A method as claimed in claim 5 wherein, if a second message part $MSG_{1\alpha}$ of the first received message m_1 differs from a second message part $MSG_{1\beta}$ of the second received message m_2 , the verifier computing entity further performs
30 the step of determining the public key PK of the signer computing entity.

25

7. A method as claimed in claim 6, wherein the step of determining the public key PK of the signer computer entity comprises computing:

$$PK = (Y_{\alpha} / Y_{\beta})^{1/(h1_{\alpha} - h1_{\beta})}, \text{ where:}$$

5

$$h1_{\alpha} = H1(MSG1_{\alpha}),$$

$$h1_{\beta} = H1(MSG1_{\beta}),$$

10

$$b_{\alpha} = g^{h1_{\alpha}} h0,$$

$$b_{\beta} = g^{h1_{\beta}} h0,$$

$$Y_{\alpha} = b_{\alpha}^x, \text{ and}$$

15

$$Y_{\beta} = b_{\beta}^x.$$

8. A method as claimed in claim 1, wherein the digital signature scheme forms part of an anonymous digital signature scheme, a group signature scheme or a direct anonymous attestation (DAA) scheme.

9. A method as claimed in claim 8, wherein the step of signing a message comprises the steps of computing a challenge value c, where

25

$$c = H1(g, h0, h1, Y, T, Z), \text{ and where:}$$

Z denotes some other part of the signature used in an anonymous signature, a direct anonymous attestation (DAA) scheme or a group signature;

30

$$h0 = H0(MSG0);$$

26

$$h1 = H1(MSG1);$$

$$b = g^{h1} h0 \text{ in } G;$$

5 $T = b^s Y^{-c}$ in G , where T is a commitment value; and

$$Y = b^x = (PK)^{h1} (h0)^x.$$

10. A method as claimed in claim 8, wherein the first message part MSG0 and
 10 second message part MSG1 are used in place of the base names of a direct
 anonymous attestation (DAA) signature scheme.

11. A method as claimed in claim 10, further comprising the steps of
 computing a challenge c , where:

15

$$c = H1(g, h0, h1, Y, T, Z), \text{ and where:}$$

Z denotes some other part of the signature used in a direct
 anonymous attestation (DAA) scheme other than the basename;

20

$$h0 = H0(MSG0);$$

$$h1 = H1(MSG1);$$

25

$$b = g^{h1} h0;$$

$$T = b^s Y^{-c}, \text{ where } T \text{ is a commitment value; and}$$

$$Y = b^x = (PK)^{h1} (h0)^x.$$

30

12. A method as claimed in claim 8, wherein the first message part (MSG0) and second message part (MSG1) are used in parallel with base names of a direct anonymous attestation (DAA) signature scheme.

- 5 13. A method as claimed in claim 12, further comprising the step of computing a challenge c , where:

$$c = H1(h, h0, h1, b, X, Y, R, T, Z), \text{ and where;}$$

10 $h = H0(BSN);$
 $h0 = H0(MSG0);$

$$h1 = H1(MSG1);$$

15 $b = g^{h1} h0;$

$$X = h^x;$$

$$Y = b^x = (PK)^{h1} (h0)^x;$$

20 $R = h^s X^{-c};$

$$T = b^s Y^{-c}, \text{ where } T \text{ is a commitment value; and}$$

25 Z denotes some other part of the signature used in a direct anonymous attestation (DAA) signature.

14. A signer computing entity apparatus adapted to provide a digital signature by which the signer computing entity is able to perform a one-time anonymous signature on a message m , the apparatus comprising:

- 30 a partitioning unit adapted to partition the message m into a first message part MSG0 and a second message part MSG1;

a mapping unit adapted to map the first message part MSG0 with a first hash function H0 to form a first digest h0, and map the second message part MSG1 with a second hash function H1 to form a second digest h1;

5 a combining unit adapting to combine the first digest h0 and the second digest h1 to form a temporary base value b;

a forming unit adapted to form a temporary public key Y; and

a signing unit adapted to sign the message m using the temporary base value b and the temporary public key Y.

10 15. A verifier computing entity apparatus adapted to determine whether first and second messages have the same first message part MSG0, the apparatus comprising:

15 a receiving unit adapted to receive a first message m1 and a second message m2 which have been signed using an apparatus as defined in claim 14;

20 a processing unit adapted to determine whether the first message part MSG0 of the first received message m1 is the same as the first message part MSG0 of the second received message m2, based on the temporary base value b of the first received message m1 and the temporary base b of the second received message m2.

16. An apparatus as claimed in claim 15, wherein the processing unit is further adapted to determine whether the first and second received messages m1, m2 have been signed by the same computer signing entity, based on the temporary public key Y of the first received message m1 and the temporary public key Y of the second received message m2.

17. An apparatus as claimed in claim 16, wherein the processing unit is further adapted to determine the public key PK of the signer computing entity if a second message part MSG1_α of the first received message m1 differs from a second message part MSG1_β of the second received message m2.

18. An apparatus as claimed in claim 17, wherein the processing unit is adapted to determine the public key PK of the signer computer entity by computing:

5
$$PK = (Y_{\alpha} / Y_{\beta})^{1/(h1_{\alpha} - h1_{\beta})}, \text{ where:}$$

$$h1_{\alpha} = H1(MSG1_{\alpha}),$$

$$h1_{\beta} = H1(MSG1_{\beta}),$$

10

$$b_{\alpha} = g^{h1_{\alpha}} h0,$$

$$b_{\beta} = g^{h1_{\beta}} h0,$$

15

$$Y_{\alpha} = b_{\alpha}^x, \text{ and}$$

$$Y_{\beta} = b_{\beta}^x.$$

1/11

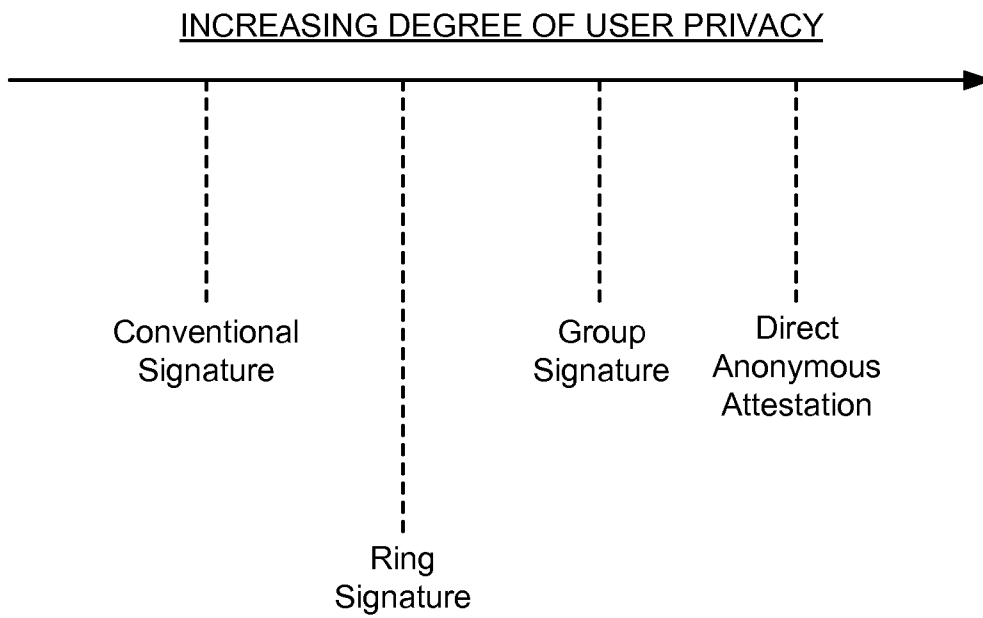


Figure 1

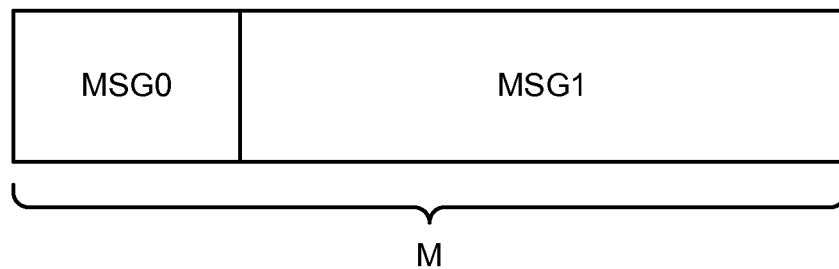


Figure 2a

2/11

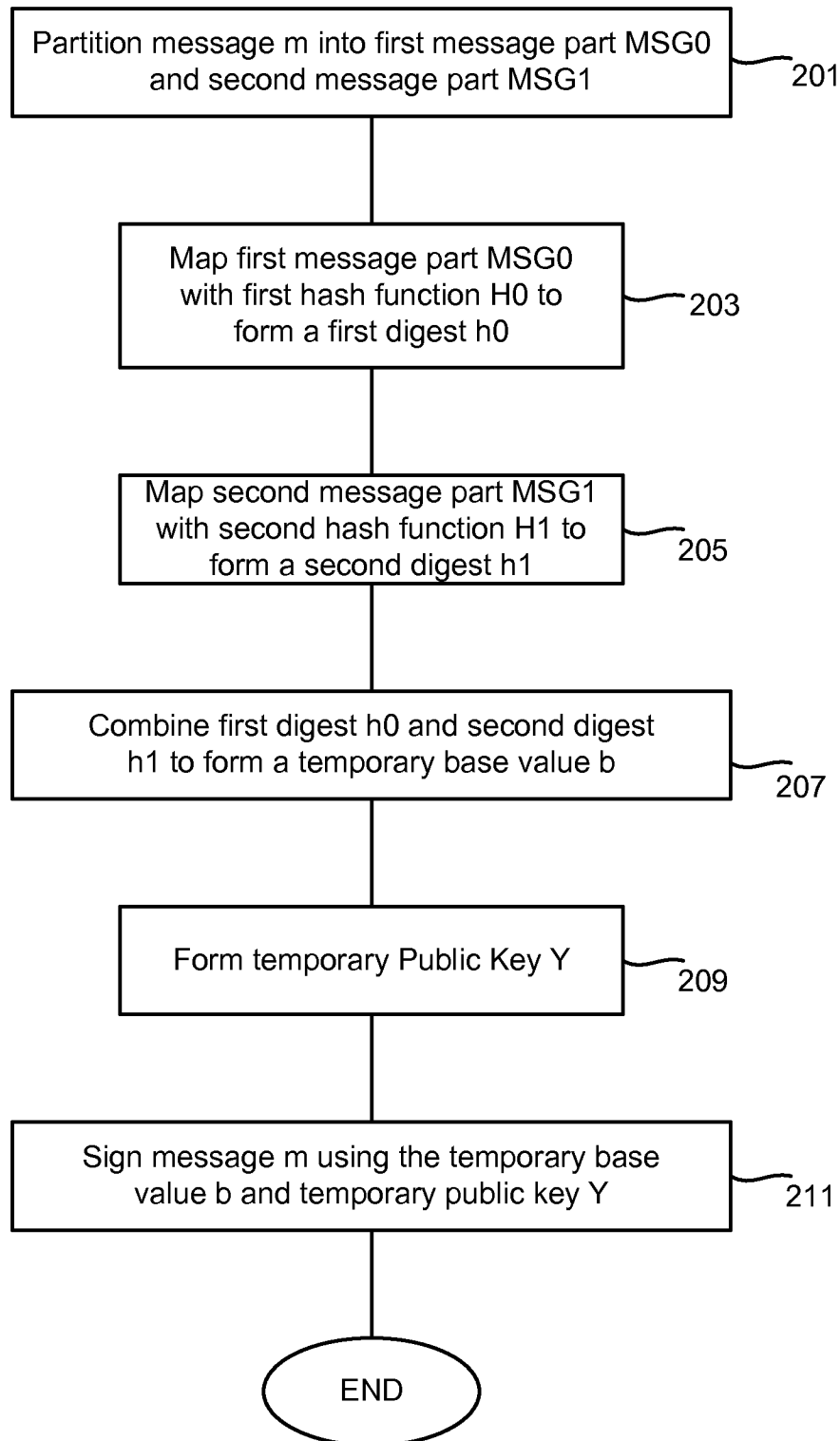


Figure 2b

3/11

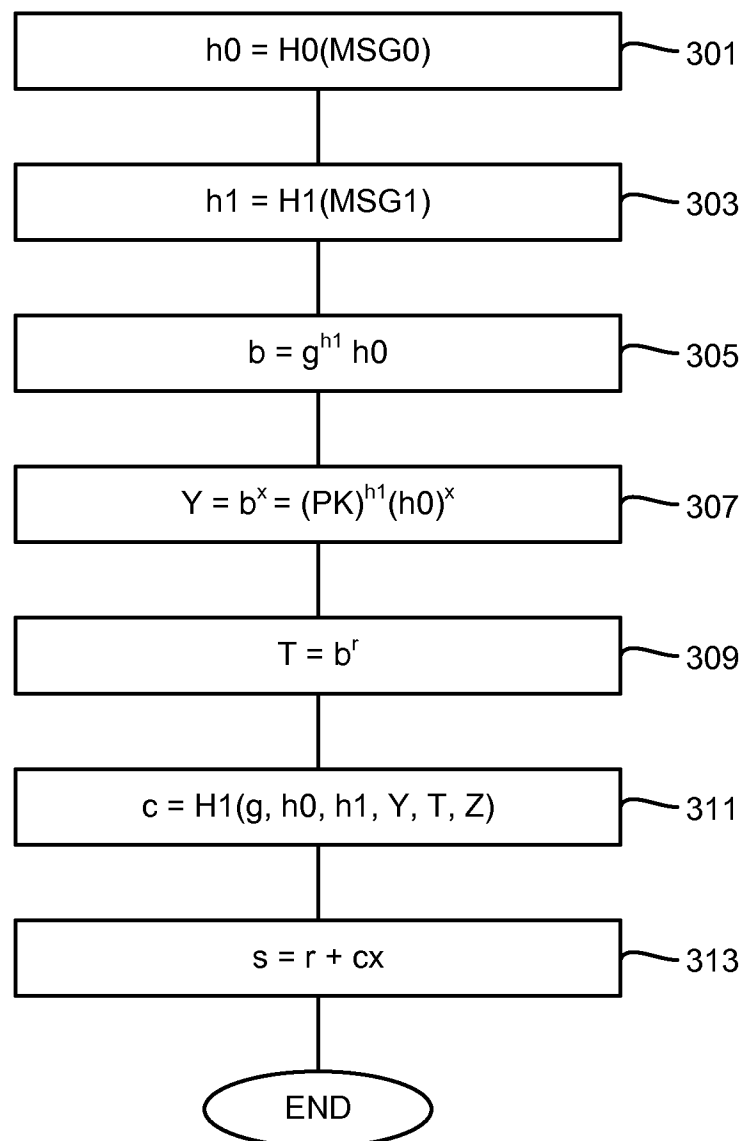


Figure 3

4/11

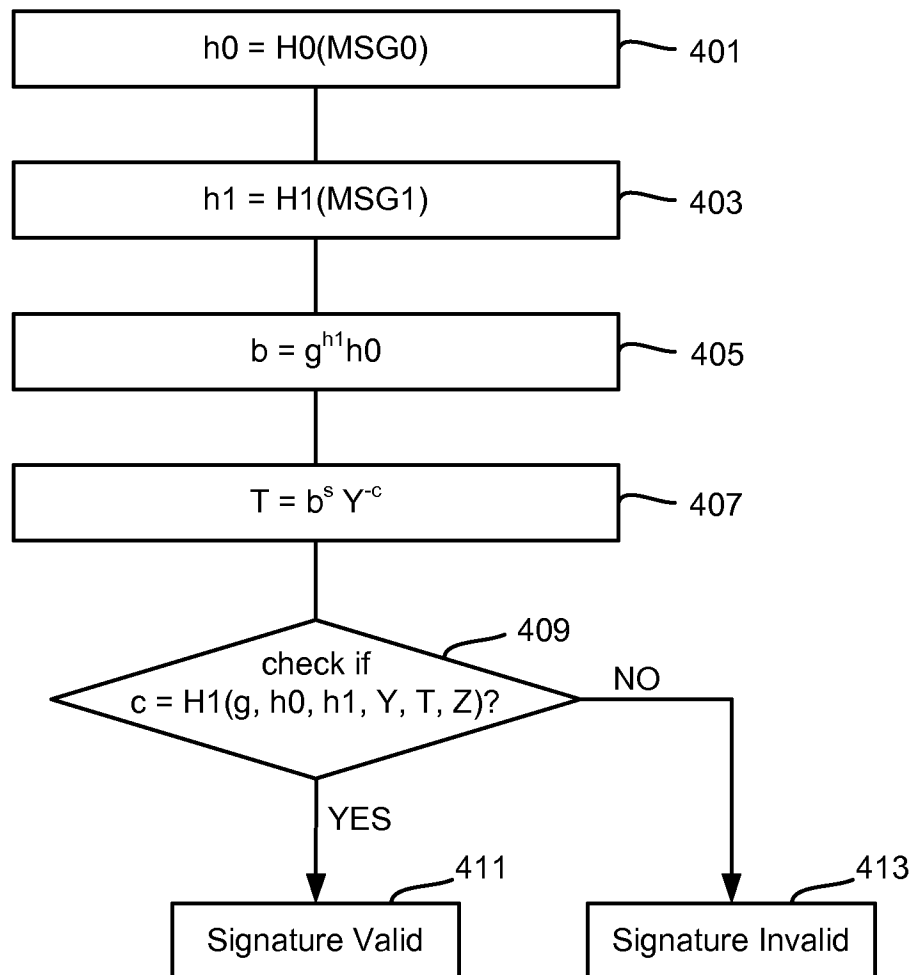


Figure 4

5/11

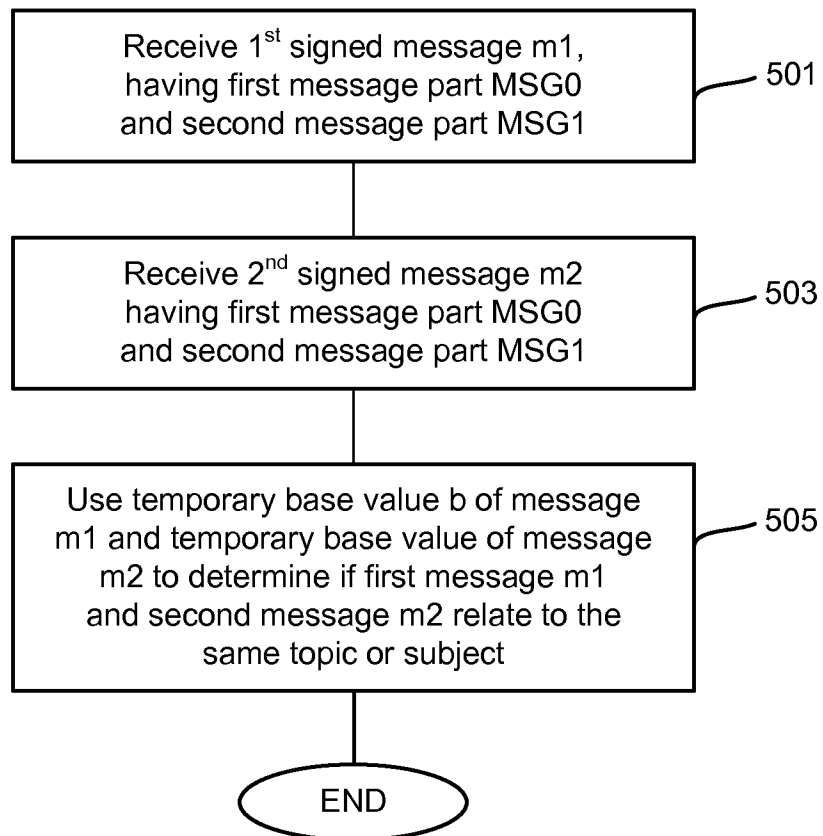


Figure 5a

6/11

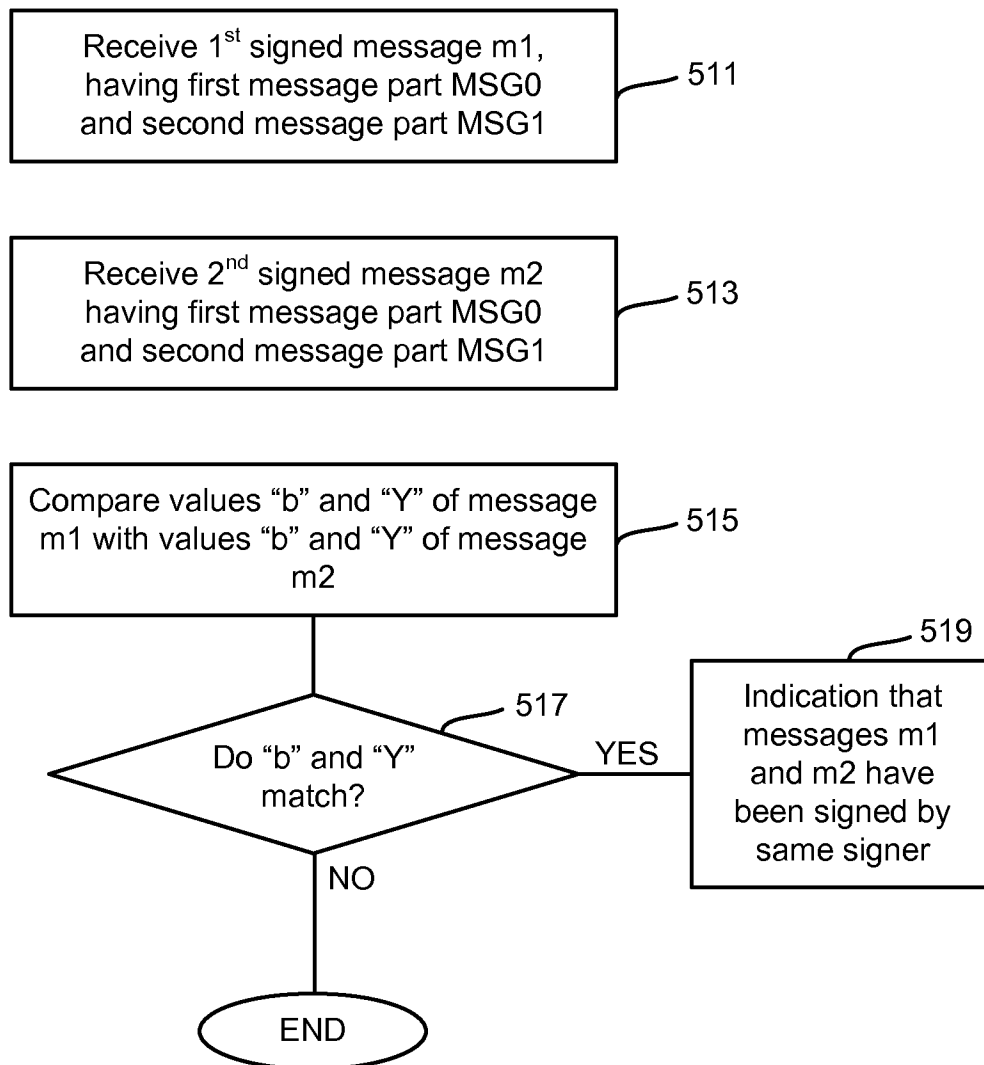


Figure 5b

7/11

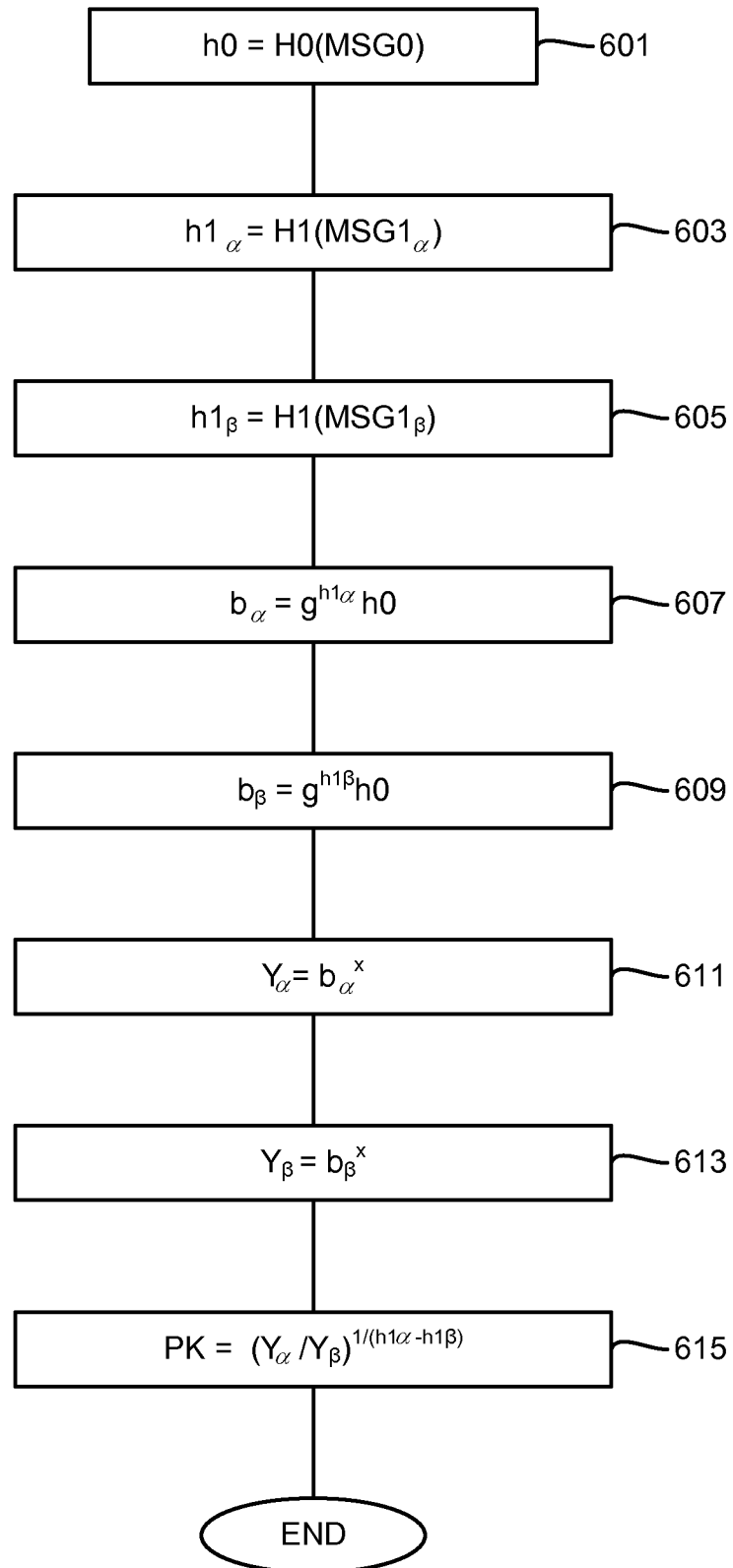


Figure 6

8/11

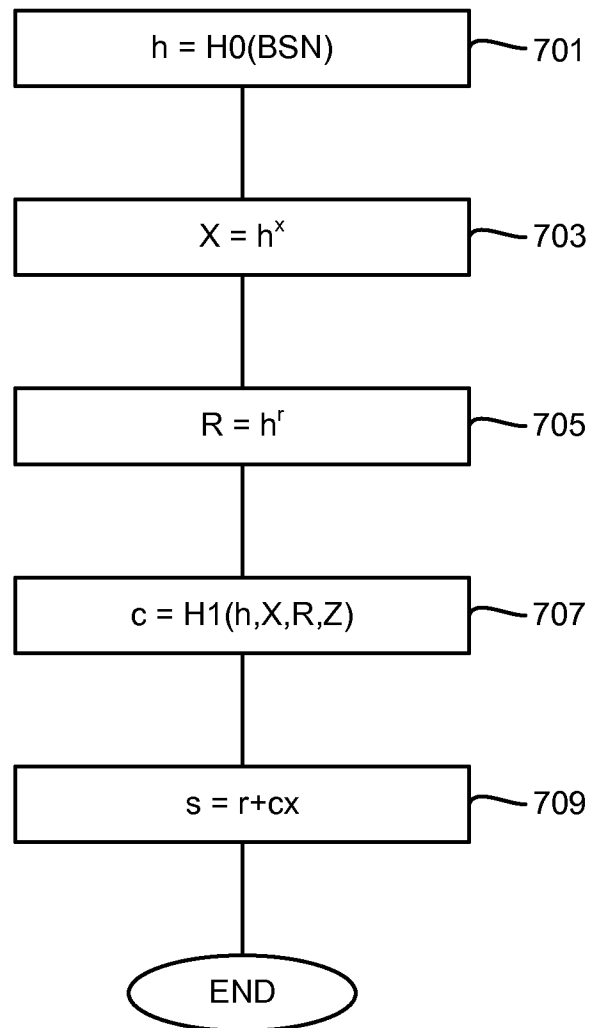


Figure 7

9/11

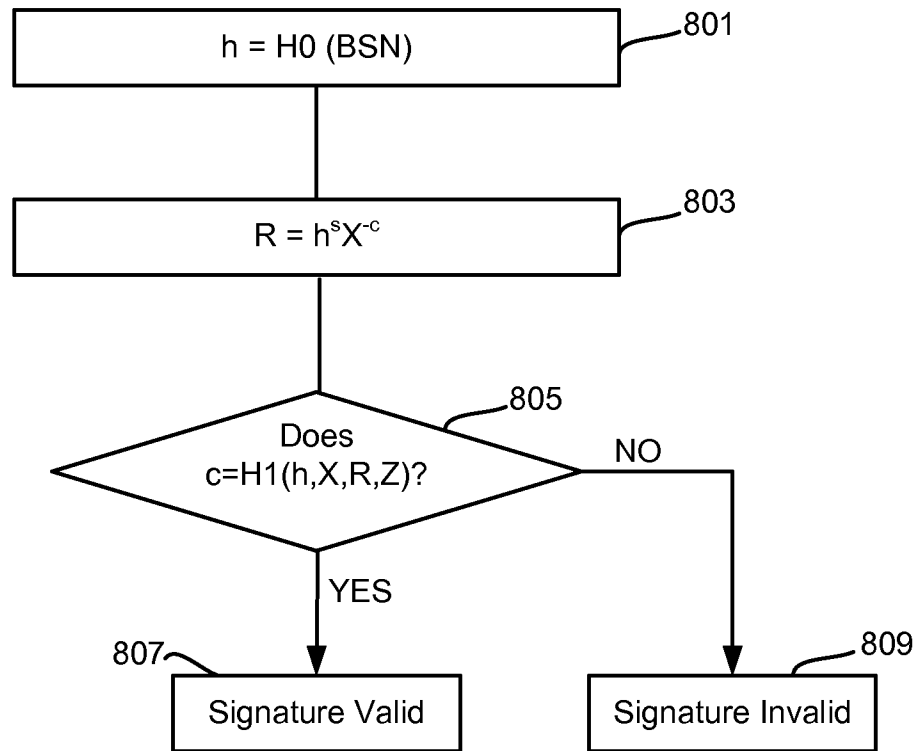


Figure 8

10/11

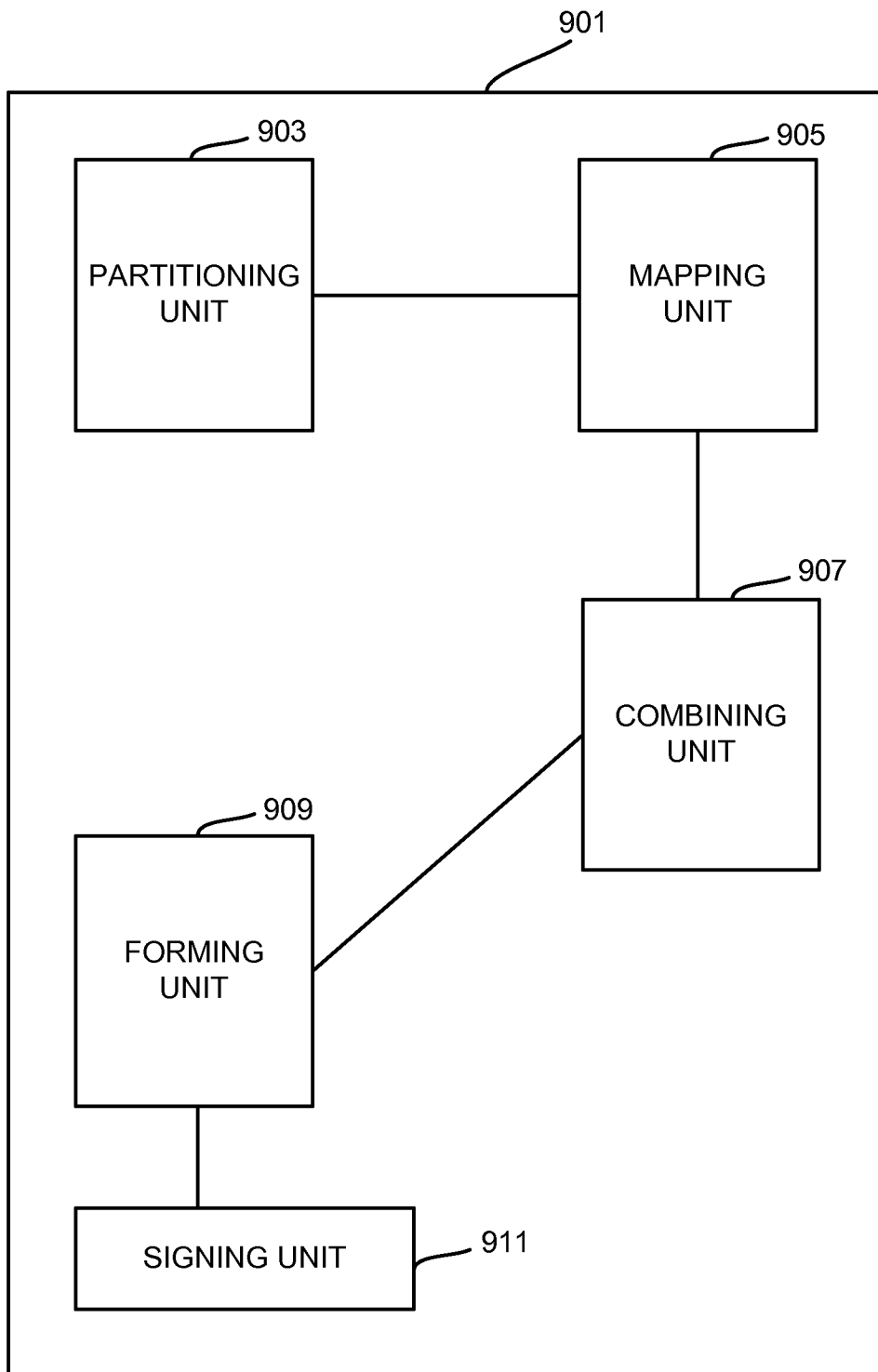


Figure 9

11/11

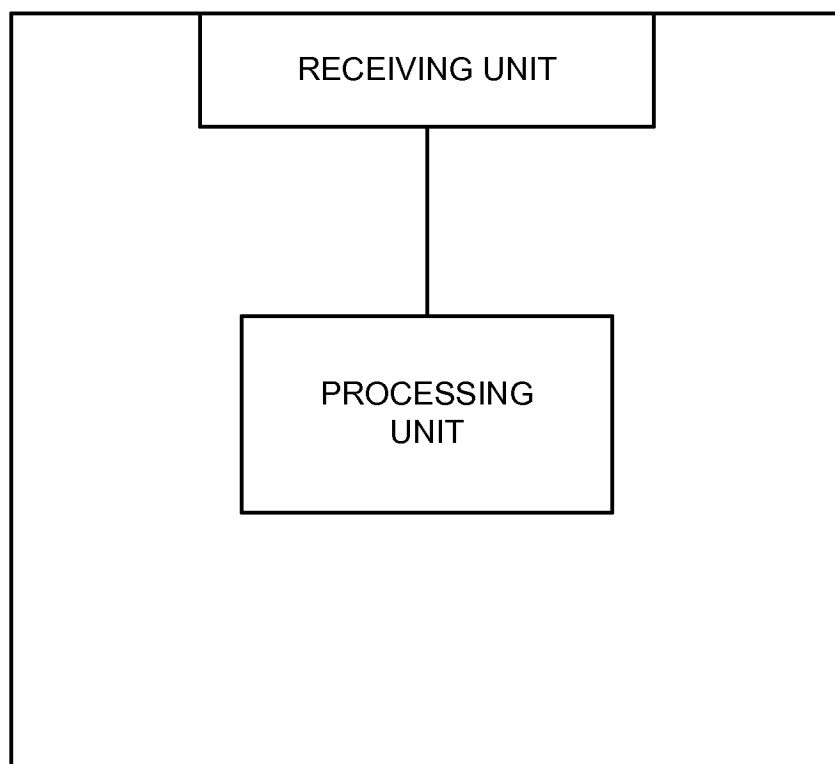


Figure 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/057002

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/32
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2001/011351 A1 (SAKO KAZUE [JP]) 2 August 2001 (2001-08-02) paragraphs [0074] - [0102] -----	1,2,4-8, 10,12, 14-17
A	CAMENISCH J ET AL: "EFFICIENT GROUP SIGNATURE SCHEMES FOR LARGE GROUPS", ADVANCES IN CRYPTOLOGY - CRYPTO '97. SANTA BARBARA, AUG. 17 - 21, 1997; [PROCEEDINGS OF THE ANNUAL INTERNATIONAL CRYPTOLOGY CONFERENCE (CRYPTO)], BERLIN, SPRINGER, DE, vol. CONF. 17, 17 August 1997 (1997-08-17) , pages 410-424, XP000767547, ISBN: 978-3-540-63384-6 pages 416-418 page 421 ----- -/--	1-18



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

28 February 2011

Date of mailing of the international search report

04/03/2011

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Horbach, Christian

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/057002

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	ERNIE BRICKELL ET AL: "Direct Anonymous Attestation", PROCEEDINGS OF THE 11TH ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY : WASHINGTON, DC, USA, OCTOBER 25 - 29, 2004; [ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY], NEW YORK, NY : ACM PRESS, 25 October 2004 (2004-10-25), pages 132-145, XP007917346, DOI: DOI:10.1145/1030083.1030103 ISBN: 978-1-58113-961-7 pages 5-15 -----	1-18
A	RONALD L RIVEST ET AL: "How to Leak a Secret: Theory and Applications of Ring Signatures", 1 January 2006 (2006-01-01), THEORETICAL COMPUTER SCIENCE LECTURE NOTES IN COMPUTER SCIENCE;;LNCS, SPRINGER, BERLIN, DE, PAGE(S) 164 - 186, XP019029104, ISBN: 978-3-540-32880-3 pages 172-173 -----	1-18
A	CAMENISCH J ET AL: "A signature Scheme with Efficient Protocols", 20030101, vol. 2576, 1 January 2003 (2003-01-01), pages 268-289, XP002456613, DOI: DOI:10.1007/3-540-36413-7_20 ISBN: 978-3-540-24128-7 sections 2 and 4 -----	1-18
A	"Chapter 11: Digital Signatures ED - Menezes A J; Van Oorschot P C; Vanstone S A", [Online] 1 October 1996 (1996-10-01), HANDBOOK OF APPLIED CRYPTOGRAPHY; [CRC PRESS SERIES ON DISCRETE MATHEMATICS AND ITS APPLICATIONS], CRC PRESS, BOCA RATON, FL, US, PAGE(S) 425 - 488, XP001525011, ISBN: 978-0-8493-8523-0 Retrieved from the Internet: URL:http://www.cacr.math.uwaterloo.ca/hac/> pages 459-460 -----	1-18

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2010/057002

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2001011351 A1	02-08-2001	AU 780858 B2	21-04-2005
		AU 1639001 A	26-07-2001
		CA 2331991 A1	21-07-2001
		EP 1136927 A2	26-09-2001
		JP 2001202013 A	27-07-2001
		SG 108237 A1	28-01-2005
