OPIC
Office de la propriété intellectuelle du Canada

CIPO
Canadian Intellectual Property Office

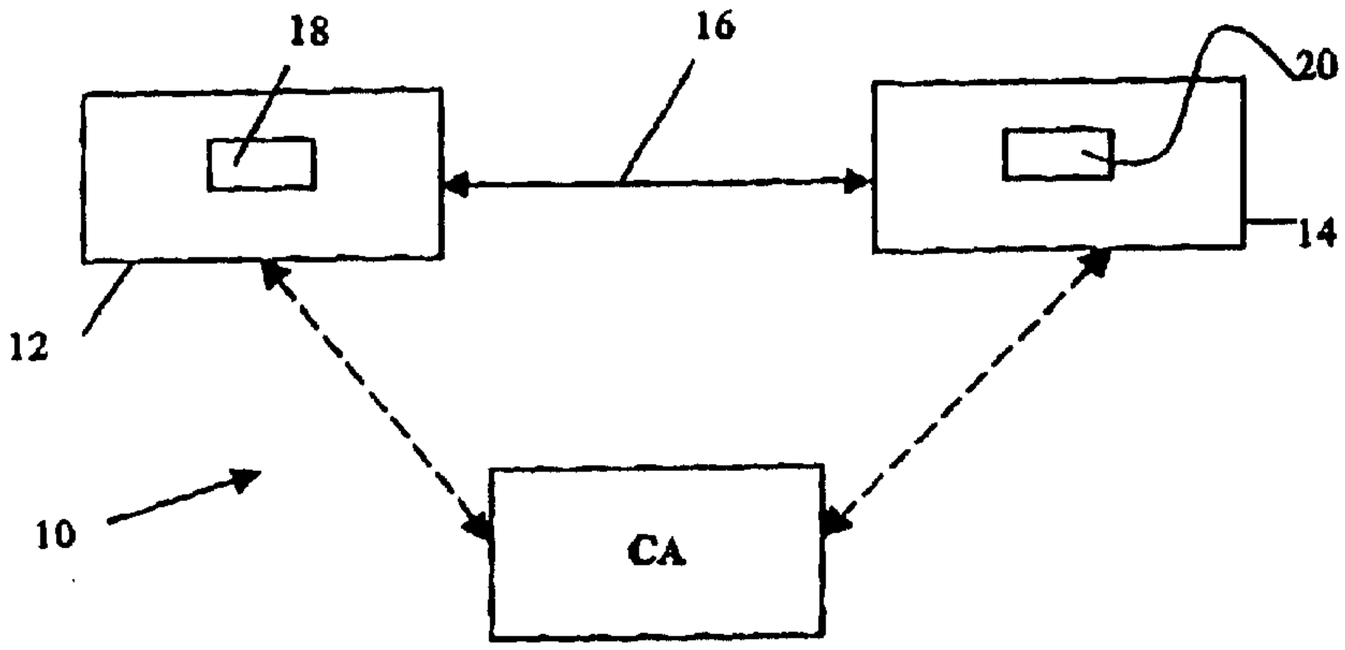
(12) (19) (CA) **Demande-Application**

(21) (A1) **2,305,896**
(86) 1998/10/14
(87) 1999/04/22

(72) JOHNSON, DONALD B., US
(71) CERTICOM CORP., CA

(51) Int.Cl.[6] H04L 9/30
(30) 1997/10/14 (08/949,781) US

(54) **PROCEDE DE VALIDATION DE CLES**
(54) **KEY VALIDATION SCHEME**



(57) L'invention concerne un procédé permettant d'assurer une sécurité améliorée dans un système de communication utilisé pour transférer des informations entre au moins deux correspondants. La communication entre les correspondants consiste en général à générer des paires de clés selon les propriétés arithmétiques d'un algorithme choisi, à communiquer une des clés, qui est une clé publique, à l'autre partie au moyen d'un certificat, de la génération et de la transmission d'une signature en utilisant une clé privée parmi les paires de clés par un des correspondants et à transmettre la signature à l'autre correspondant, la signature étant vérifiée par le destinataire. Le procédé consiste en outre à vérifier la conformité de la clé publique avec les propriétés arithmétiques requises par l'algorithme sélectionné.

(57) A method of providing improved security in a communication system used to transfer information between at least a pair of correspondents. The communication between the correspondents generally comprises steps of generating key pairs in accordance with the arithmetic properties of a chosen algorithm, communicating one of the keys, being a public key, to the other party by way of a certificate, generation and transmission of a signature using a private key of the key pairs by one of the correspondents and transmitting the signature to the other correspondent and verification of the signature by the recipient. The invention provides for the additional step of verifying the public key conform to the arithmetic properties dictated by the requirements of the selected algorithm.

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
H04L 9/30

A1

(11) International Publication Number: WO 99/20020

(43) International Publication Date: 22 April 1999 (22.04.99)

(21) International Application Number: PCT/CA98/00959

(22) International Filing Date: 14 October 1998 (14.10.98)

(30) Priority Data:
08/949,781 14 October 1997 (14.10.97) US

(71) Applicant *(for all designated States except US)*: CERTICOM CORP. [CA/CA]; Suite 103, 200 Matheson Boulevard, West, Mississauga, Ontario L5R 3L7 (CA).

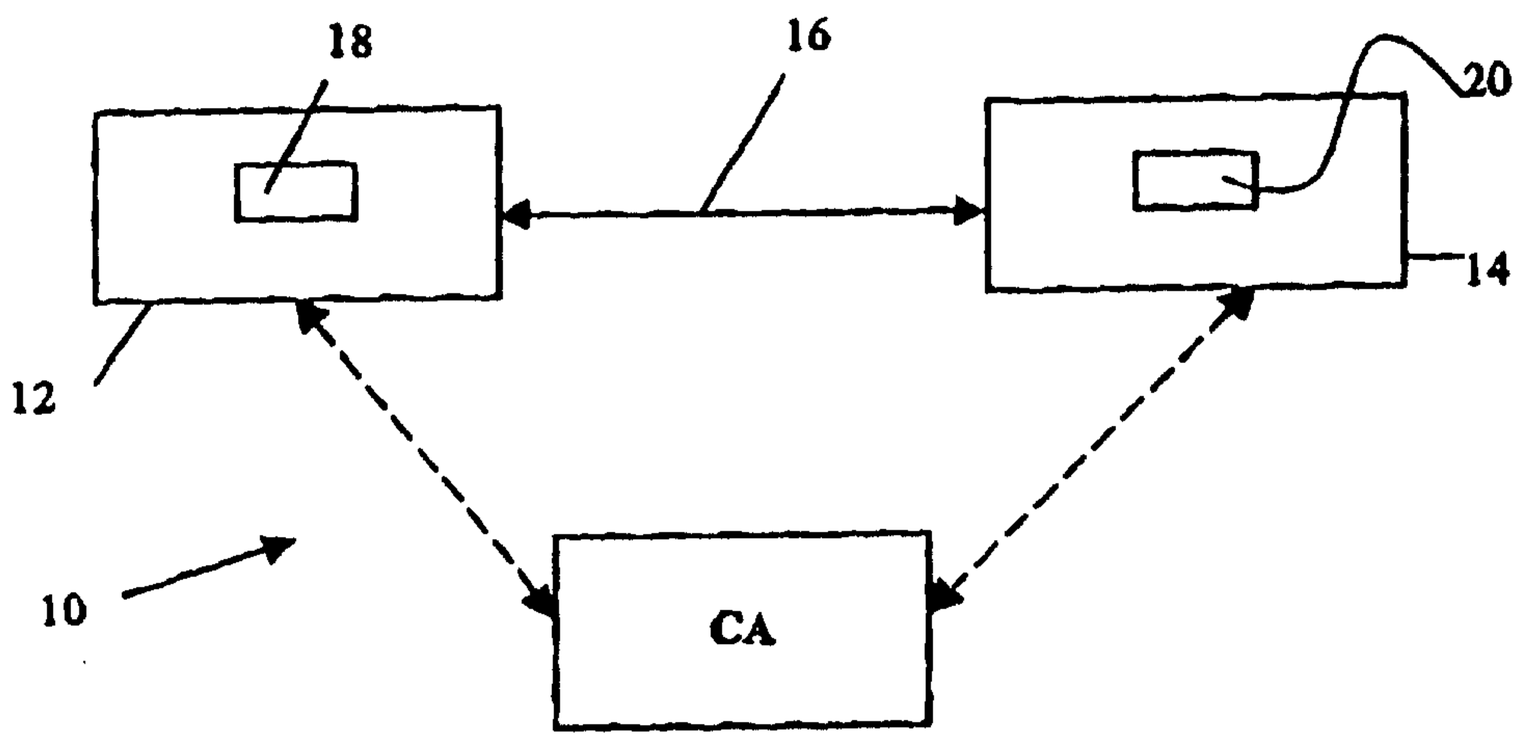
(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: JOHNSON, Donald, B. [US/US]; 7684 Knightshayes Drive, Manassas, VA 20111 (US).

(74) Agent: ORANGE CHARI PILLAY; Toronto Dominion Bank Tower, Toronto–Dominion Centre, Suite 3600, P.O. Box 190, Toronto, Ontario M5K 1H6 (CA).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: KEY VALIDATION SCHEME



(57) Abstract

A method of providing improved security in a communication system used to transfer information between at least a pair of correspondents. The communication between the correspondents generally comprises steps of generating key pairs in accordance with the arithmetic properties of a chosen algorithm, communicating one of the keys, being a public key, to the other party by way of a certificate, generation and transmission of a signature using a private key of the key pairs by one of the correspondents and transmitting the signature to the other correspondent and verification of the signature by the recipient. The invention provides for the additional step of verifying the public key conform to the arithmetic properties dictated by the requirements of the selected algorithm.

WO 99/20020 PCT/CA98/00959

KEY VALIDATION SCHEME

The present invention relates to secures communication systems and in particular to schemes for validating parameters and keys in such systenis.

BACKGROUND OF THE INVENTION

Secure data communications systems are used to transfer information between a pair of correspondents. At least part of the information that is exchanged is enciphered by a predetermined mathematical operation by the sender. The recipient may then perform a complimentary mathematical operation to decipher the information. For public key or symmetric key systems, there are certain parameters that must be known beforehand between the correspondents. For example, various schemes and protocols have been devised to validate the senders public key, the identity of the sender and such like. The security or validity of these systems is dependent on whether the signature is a valid signature and this is only the case if system parameters if any are valid, the public key is valid and the signature verifies. Furthermore, an asymmetric system is secure only if system parameters if any are valid, the enciphering public key is valid, the symmetric key is formatted as specified and the symmetric key recovery checks for format validity.

On the other hand a key agreement protocol is secure only if the system parameters, if any, are valid, the key agreement public keys are valid, and the shared secret and symmetric key is derived as specified in a standard. In all of these it is assumed that the public key or symmetric key, i.e. the shared secret, is derived and valid as specified in the protocol scheme. Problems, however, will arise if these parameters are either bogus or defective in some way.

The following scenarios may illustrate the implications of a defect in one or more parameters of a public key cryptographic system. For example digital signatures are used to indicate the authenticity of a sender. Thus if a Recipient A receives a certified public key from a Sender B, then A verifies the certificate, next B sends A a signed message for which A is able to verify the signature and thus assume that further communication is acceptable. In this scenario, however, if B has deliberately corrupted the public key then the Recipient A has no way of distinguishing this invalid public key. Similarly, a Participant C generates a key pair and then subsequently receives a public key certificate, the Participant C then sends the certificate and a subsequent signed message to B under the

assumption that the public key contained in the certificate is valid. The participant B can then determine key information for C. Both the above scenarios describe possible problems arising from utilizing unauthenticated parameters in signature verification.

In key transport protocols a Correspondent A may inadvertently send its symmetric key to the wrong party. For example, if Corespondent A receives a certified public key from a Sender B, the certificate is verified by A who then sends a public key enciphered symmetric key and a symmetric key enciphered message to B, thus A is comprised. Conversely, if one of the correspondents C generates a key pair and gets a public key certificate which is subsequently sent to A who public key enciphers a symmetric key and message and sends it back to C, thus, in this case, C is compromised.

In key agreement protocols, one of the correspondents, A for example, receives a certified public key from B and sends B A's certified public key. Each of A and B verify the other's certificate and agree upon a symmetric key. In this scenario A is compromised twice.

It may be seen from the above scenarios that although public key systems are secure the security of the system relies to a large extent on one or both of the correspondents relying on the fact that a claimed given key is in fact the given key for the particular algorithm being used. Typically the recipients receive a string of bits and then make the assumption that this string of bits really represents a key as claimed. This is particularly a problem for a symmetric key system where typically any bit string of the right size may be interpreted as a key. If a bit in the key is flipped, it may still be interpreted as a key, and may still produce a valid crypto operation except that it is the wrong key.

In an asymmetric private key system the owner of the private key knows everything about the private key and hence can validate the private key for correctness. However, should a third party send the owner system a public key, a question arises as to whether the received key conforms to the arithmetic requirements for a public key or the operations using the claimed public key is a secure crypto operation. Unless the owner system performs a check it is unlikely to know for certain and then only by the owner.

From the above it may be seen that key establishment may be insecure. In a paper written by Lim and Lee presented at Crypto '97, this problem was demonstrated in context of the Diffie-Hellman scheme using a bogus public key that did not have the correct order

and thus one party was able to find information about the other party's private key. In the RSA or Rabin scheme, which gets its security from the difficulty of factoring large numbers, the public and private keys are functions of a pair of large prime numbers. The keys are generated from the product of two random large prime numbers. Suppose, however, that n is a prime instead of the products of two primes then phi(n) = n-1 so anyone can determine d from the bogus "public key" (n,e). These are just examples of the problems a user of a public key can get into if they cannot validate the arithmetic properties of a claimed public key for conformance with the requirements of the algorithm.

SUMMARY OF THE INVENTION

This invention seeks to provide an improved validation in a secure communication system. Furthermore the invention seeks to allow such a validation to be performed by anyone at anytime using only public information.

In accordance with this invention there is provided a method of validating digital signatures in a public key communication system, said method comprising the steps of :

verifying the arithmetic property the public key conforms to the system algorithm; and

verifying said digital signature.

A further step provides for the verification of the system parameters.

A still further step provides for including within a certificate information indicative of the claimed public key having been validated for arithmetic conformance with the algorithm and, where appropriate, the amount of validation performed.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the present invention will now be described by way of example only with reference the accompanying drawings in which:

Figure 1 is a schematic representation of a communication system.

DETAILED DESCRIPTION OF A PREFFERED EMBODIMENT

Referring to Figure 1 a data communication system 10 includes a pair of correspondents designated as a sender 12 and a recipient 14 who are connected by

communication channel 16. Each of the correspondents 12, 14 includes an encryption unit 18, 20 respectively that may process digital information and prepare it for transmission through the channel 16. In addition the system 10 may include a certification authority 22.

Embodiments of the invention shall be described with reference to the following aspects of public key algorithms. Key agreement has six routines which are defined as system parameter generation, system parameter validation, key pair generation, public key validation, shared secret derivation and symmetric key derivation. In the key validation step, anyone at anytime can validate a public key using only public information. These routines validate the range and order of the public key. If a public key validates, it means that an associated private key can logically exist, although it does not prove it actually does exist.

For an elliptic curve Digital Signature Algorithm(ECDSA) there are also six routines, defined as system parameter generation, system parameter validation, key pair generation, public key validation, signature generation and signature verification. On the other hand a first type of DSA has four routines, namely system parameter generation, key pair generation, signature generation and signature verification. In a more recent DSA has five routines, namely, system parameter generation, (implicit) system parameter validation, key pair generation, signature generation and signature verification. In order to provide key validation the DSA parameters p, q, and g are assumed to have already been validated. The public key $y = g^x \bmod p$, where x is the private key. The range of y is validated to ensure $1 < y < p$ and the order of y is validated to ensure $y^q \bmod p = 1$. These tests ensure that a claimed DSA public key meets the arithmetic requirements of such a key. They can be performed by anyone at anytime using only public information.

In RSA or Rabin signatures there are generally three routines, namely key pair generation, signature generation and signature verification. Validating an RSA public key (n, e) involves three steps. Firstly validate e, secondly validate n and thirdly validate e and n are consistent with each other. In order to validate the public exponent e, use of made of the fact that the exponent $2 <= e <= 2^{(k-160)}$ where k is the length of the modulus n. The requirement that this range be as it is specified is specifically to allow this check. If $e > 2$ then e should be odd. Furthermore, if for a closed network, it is known that the public exponent e must all meet other criteria, e.g., it must be = 3 or 65537 or be a random number larger than 65537, these checks can also be done to further confirm the validity of

the key. These checks may be incorporated as part of the specification of an RSA public key partial validation routine. Even though the above test for *e* appears trivial, this test ensures that *e* was selected before *d* as intended by the RSA/Rabin algorithm since, it may be shown that *de* = 1 mod (*lcm*(p-1,q-1)) and there are at least 160 high order zeroes in *e* when compared with modulus *n*, and this is infeasible to achieve by selecting d first.

In order to validate the modulus *n*, the sizes of *n* may be determined. It is known that *n* is supposed to contain exactly (1,024 plus 128s) bits, where s = 0,1,2,3... etc. This can be easily validated and can be part of a partial key validation. Determining whether the modulus *n* is odd given that n is supposed to be the product of two primes and that all primes after 2 are odd may perform a further validation of the modulus *n*. Therefore the product of odd numbers is odd so *n* should be odd. Furthermore, for Rabin when *e* = 2 we know *p* should be equal to 3 mod *n* and *q* should be 7 mod 8. This means *n* = *pq* should be = 21 mod 8 = 5 mod 8. This can be validated by ensuring that if *e* = 2, then *n* = 5 mod 8. Furthermore, we know *n* should not be a perfect power thus this ensures there be two distinctive prime factors and this can be validated by a simple check as documented in the Handbook of Applied Cryptography by Menezes, van Oorschot, and Vanstone.

It is also known that *n* should be a composite number thus if *n* is prime the transformation is easily invertible and hence is completely insecure. The fact that *n* should be composite can be validated by running the Miller-Rabin probable prime test expecting it to actually prove that *n* is composite. An additional test for validating the modulus *n* is based on knowing that *n* is supposed to be the product of two large primes and is supposed to be hard to factor. Therefore attempt to factor it in some simple way, expecting it to fail. For example calculate GCD (*n*, *i*) where *i* runs through all the small odd primes up to a certain limit, say the first 50K odd primes.

From the previous two tests above, it may be seen from the former that at least one factor must be of a size of half the bits of the modulus or less. From the latter it may be seen that each factor must be larger than the largest prime tested. Furthermore there are now only a limited number of potential factors (p, q, r,...) depending on the size of the largest prime test.

The multiple tests above in combination have a synergistic effect. The goal of which is to greatly reduce the freedom of action of an adversary. Even if an attack is not

totally impossible, partial key validation can make an attack much more difficult. hopefully infeasible or at least uneconomical.

Furthermore in validating the modulus *n*, p and q are not supposed to be too close in value therefore assume they are and try to factor *n*. Use the square root of *n* as a starting guess for *p* and *q*. Then let *p* decrease while *q* increases and determine if *n* can be factored up to a predetermined limit. Furthermore we know for a set of RSA moduli, no prime should repeat therefore given a set of RSA moduli *n1*, *n2* the GCD (*ni, nj*) can be calculated to ensure the results all equal one.

Offline tests as described above have their limitations. These tests may be extended since the owner of the parameters knows particular information, for example the factorization of *n*. Thus the owner may be used as an online oracle. By determining if the answers to these questions asked of the oracle are incorrect anyone may declare public key invalid.

It is shown in the Handbook of Applied Cryptography Vanstone et. al. That the owner can take square roots mod *n*, but others cannot. The validater can determine if a random number mod *n* has a Jacobi symbol 1 or –1, then half are 1 and the other half are -1. If 1, then number is either a square or not a square, again half each. Validater can square a number mod n. A square always has Jacobi symbol = 1.

The validater selects either a known square u or a random element r with Jacobi symbol = 1. Asks owner "If this is a square?" for these two types of elements. The owner responds either Yes or No. If u was selected, the owner must say Yes, else key modulus is invalid. If r was selected the owner should say Yes about 1/2 the time and No about 1/2 the time, else key modulus is invalid.

This is repeated a number of times to be confident. If the Validater gave the owner all squares, owner should always respond Yes. If the Validater gave the owner all random elements with Jacobi Symbol = 1, owner should respond 1/2 of the time Yes and ½ of the time No. Owner of bogus key only knows that at least half the answers should be Yes. However, owner of the private key knows the factorization of n, they know the squares and thus just need to lie about the pseudosquares, saying some are squares, in order to fool the validater. What is needed is a way for the validater to ask the "Is this a square?" question using a known pseudosquare. Normally, determining if a number is a pseudosquare for a given modulus without knowing the factorization of the modulus is a

infeasible problem, however, the owner must respond to the above questions with an answer that says that some of the Jacobi = 1 numbers are pseudosquares. The validater can form arbitrary known pseudosquares by multiplying a known pseudosquare by a square modulo the modulus. The result will be a value that the validater knows is a pseudosquare. This third type of value t (known pseudosquare) can be asked of the owner and now lies by the owner saying that some pseudosquares are squares can be detected by the validater.

In order to validate e and n together GCD(e, p-1) = 1 and GCD(e, q-1) = 1. If e is odd, we know p should not be of form $xe + 1$ for some integer x and q should not be of form $ye + 1$ for some integer y. If both p and q are bad then n should not be of form $xye^2 + xe + ye + 1$ and $n \neq 1$ mod e.

A further method of validating e and n together. It is know that the GCD(e,phi(n)) should be 1. If it is known that phi(n) = (p-1)(q-1), then this is two equations in two unknowns and therefore the validater can factor n.

Assuming the other requirements on a key pair are met, the reason GCD(e, phi(n)) = 1 is needed is to ensure the operation using e is a one-to-one (invertible) function. Else, the operation using e is many-to-one. If the operation using e is many-to-one then d (the inverse of e) does not exist, at lest as normally conceived. The owner should give evidence that d actually exists, but the question should not be under the private key owner's control, that is, a self-signed certificate request may not be enough evidence.

The challenge can send the claimed owner some dummy messages to sign. The owner of the private key can verify that they are dummy messages, sign them, and return them to the challenger. This is an online probabilistic oracle test that d exists.

Thus anyone can do offline validation at any time. Anyone can do online validation if owner is online. Owner can do offline and online validation to assure him/herself public key is valid. CA can do online validation and tell others exactly what and how much it validated in the public key certificate.

In the ECDSA the system parameters are field size q = p or 2^m. An optional seed that generates (a, b) with (a, b) defining an elliptic curve over F_q, P a distinguished point on the curve, n, the large prime order of P, h, the cofactor such that the order of curve is hn. The field size, EC defined by (a, b) and point P are primary parameters.

It is important to verify not only the EC system parameters but also the EC public key. For example, given an elliptic curve public key Q, check that Q is on E. In key agreement, and utilizing a prime order curve, then we do not need to check the order of Q since Q certainly has the correct order if it is on the curve. Checking that Q is on the curve is important since an erroneous key may give away the private key a in computing aQ, if Q is not on the curve. Verifying the public key is on the curve may be achieved by substitution into the curve or testing.

Thus it may be seen that key validation may reduce exposure to attacks and help detect inadvertent errors and is also is a valuable service for a CA to perform. Those of ordinary skill in the art will appreciate that the above techniques and methods may be implemented on a suitable processor to carry out the steps of the invention. In addition although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware or in more specialized apparatus constructed to perform the required method steps.

WE CLAIM:

1. A method for validating digital information transmitted by one correspondent to another in a data communication system, said method comprising the steps of:

a) generating a public key in accordance with a predetermined cryptographic scheme having predetermined arithmetic properties and system parameters;

b) verifying said public key conforms to said arithmetic properties of said scheme; and

c) transmitting said verified public key to a recipient.

2. A method as defined in claim 1, including transmitting an information along with said verified public key, for indicating said public key is validated.

3. A method as defined in claim 1, said public key being an elliptic curve public key Q and said cryptographic scheme being an elliptic curve scheme.

4. A method as defined in claim 3, said steps verifying said public key including verifying said public key Q is on said elliptic curve E.

5. A method as defined in claim 1, including the step of verifying said system parameters.

6. A method of providing a secure asymmetric communication system, having a public key and symmetric key, said method comprising the steps of:

a) verifying said public key is valid;

b) verifying said symmetric key is of a predetermined format;

c) recovering said symmetric key; and

d) verifying said recovered symmetric key is of a predetermined valid format.

7. A method as defined in claim 6, including the step of verifying said system parameters are valid.

8. A method of providing a secure key agreement in a communication system having a public key, symmetric key and secret information, said method comprising the steps of:

a) verifying said public key is valid;

b) verifying said secret information is valid; and

c) verifying said symmetric key is valid.

9. A method as defined in claim 8, including the step of verifying system parameters are valid.

10. A method as defined in claim 8, including the step of including in a certificate information indicative of said key verification.

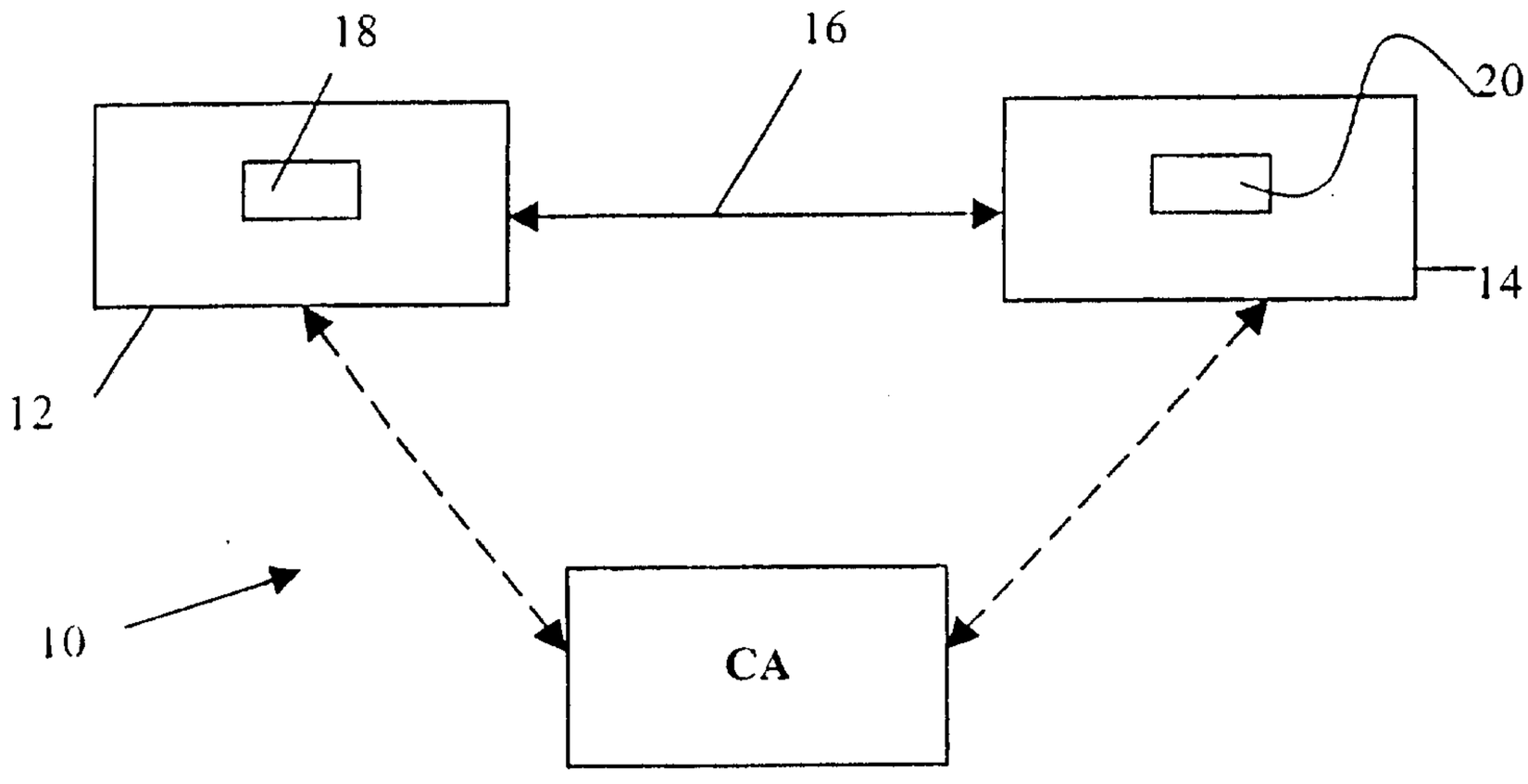


Figure 1

SUBSTITUTE SHEET (RULE 26)