(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 January 2016 (28.01.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/014048 A1

(51) International Patent Classification:
H04L 9/08 (2006.01) *H04L 9/30* (2006.01)

(21) International Application Number:
PCT/US2014/047773

(22) International Filing Date:
23 July 2014 (23.07.2014)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).

(72) Inventors: **CHEN, Liqun**; Long Down Avenue, Stoke Gifford Bristol BS34 8QZ (GB). **EL KAAFARANI, Ali**; Long Down Avenue, Stoke Gifford Bristol BS34 8QZ (GB). **PEARSON, Siani**; Long Down Avenue, Stoke Gifford Bristol BS34 8QZ (GB). **LINDSAY, Bob**; Long Down Avenue, Stoke Gifford Bristol BS34 8QZ (GB). **REID, Peter**; 5400 Legacy Drive, Plano, Texas 75024 (US). **PAPANIKOLAOU, Nikolaos**; Long Down Avenue, Stoke Gifford Bristol BS34 8QZ (GB).

(74) Agents: **PATEL, Milin N.** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (US).

(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, BN, 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, IR, IS, JP, KE, KG, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, 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, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, 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, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: ATTRIBUTE-BASED CRYPTOGRAPHY

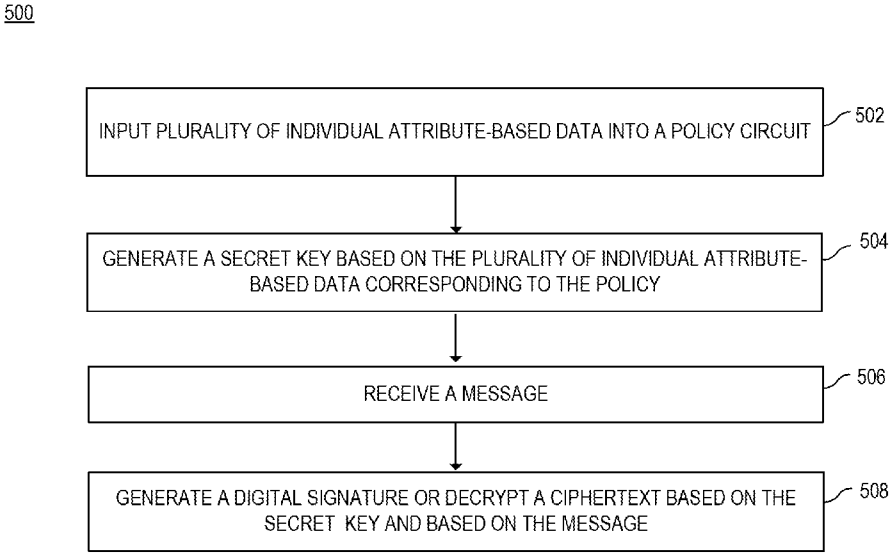

FIG. 5

(57) Abstract: A method and system for providing attribute-based cryptography which include inputting a plurality of individual attribute-based data into a circuit that implements a cryptography policy, generating a secret key via the circuit based on the plurality of individual attribute-based data corresponding to the cryptography policy, receiving a message, and generating a digital signature or decrypting a ciphertext based on the secret key and based on the message. Each of the plurality of individual attribute-based data corresponds to an individual attribute.

WO 2016/014048 A1

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

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

ATTRIBUTE-BASED CRYPTOGRAPHY

BACKGROUND

[0001] Cryptography allows for secure communication in the presence of third parties over an insecure communication channel. Two types of modern cryptography include digital signatures and encryption/decryption. A digital signature is a mathematical scheme for demonstrating the authenticity of a digital message (e.g., a document). A digital signature, once verified, gives a recipient reason to believe that the message was created by a known sender such that the sender cannot deny having sent the message and that the message was not altered in transit. Encryption is the process of encoding a message (e.g., a document) in a way that authorized parties are able to read it. Decryption is the process of decoding the encrypted message such so that the authorized parties are able to read it.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] For a detailed description of various examples, reference will now be made to the accompanying drawings in which:

[0003] Figure 1 shows a block diagram of an attribute-based cryptography system, according to an example;

[0004] Figure 2 shows a block diagram of an attribute-based cryptography system implementing an attribute-based digital signature scheme, according to an example;

[0005] Figure 3 shows a block diagram of an attribute-based cryptography system implementing an attribute-based decryption scheme, according to an example;

[0006] Figure 4 shows an example circuit;

[0007] Figure 5 shows a flowchart of an attribute-based cryptography method, according to an example;

[0008] Figure 6 shows a flowchart of an attribute-based signature method, according to an example;

[0009] Figure 7 shows a flowchart of an attribute-based decryption method, according to an example; and

[0010] Figure 8 is a block diagram of a computing device to provide attribute-based cryptography, according to an example.

DETAILED DESCRIPTION

[0011] Some digital signature techniques include a signing policy which is expressed as a public key and a certificate. In some such techniques, a verifier can lean who created the digital signature. Thus, the identity of the signer is not a secret. In an identity-based signature technique, the signing policy is expressed as a signer's identifier. Again, a verifier can learn who created the digital signature. Anonymous signatures use a group key and the signing policy is expressed as a group structure. Any group member can produce a valid signature on behalf of the group. A verifier cannot learn which group member has signed a signature, but does know the group structure.

[0012] Another technique for creating and using digital signatures is referred to as an "attribute-based signature" technique. The signature is computed based on one or more attributes of the signer. Examples of attributes may include the position of the user within a company (e.g., job role or title), the location of the signer, etc. In attribute-based signatures, those users that have a valid set of attributes can successfully generate a valid signature for a given message.

[0013] An attribute-based signature scheme may be used to provide services such as entity authentication, data origin authentication, non-repudiation, and data-integrity. In attribute-based signature schemes, a signer who has enough attributes to satisfy the policy can produce a valid signature A verifier knows that the signer has satisfied the policy. The set of attributes that has been used to sign along with the identity of the signers remain hidden. Unfortunately, the predicate itself (the policy) is known to the verifier in some attribute-based signature schemes.

[0014] In accordance with the disclosed implementations, various examples are provided herein that relate to attribute-based signature systems and methods to provide signer privacy, data origin authentication, non-repudiation, data-integrity, to ensure entity authentication, or a combination thereof. In the disclosed attribute-based signature techniques, the signer's identity, the signer's attributes, and/or the signing policy may remain hidden from the verifier.

[0015] Similarly, sharing encrypted data on the cloud allows for sharing of the data at a coarse level. In other words, sharing encrypted data requires giving a private key to any user that the data is to be shared with for decryption.

[0016] In accordance with the disclosed implementations, various examples are provided herein that relate to attribute-based decryption systems and methods which allows for fine-grained sharing of encrypted data. In other words, attribute-based decryption schemes allow a user to encrypt data with respect to a certain policy, and users who have the requisite credentials (attributes) to satisfy the policy are granted access to the data.

[0017] In an example, a master secret key is generated along with public keys which may be utilized for later verification of a digital signature or for use in decrypting a document. The attribute-based cryptography system may utilize a circuit which comprises a plurality of logic gates coupled together. These coupled logic gates implement a signing policy or decryption policy. The mathematical representation of the circuit receives as inputs a plurality of individual attribute-based keys or individual attribute-based hashes. All references herein to the "circuit" mean the hardware logic that implements the mathematical representation of the circuit. Each individual attribute-based key or hash corresponds to a distinct characteristic (attribute) that is representative of a user seeking to sign or decrypt a message. A secret key is generated by the circuit based on the individual attribute-based keys or hashes corresponding to the signing policy and based on the master secret key previously generated.

[0018] By way of example, a signing policy or decryption policy may be a particular message can only be signed or decrypted by a junior level vice president that is located in the firm's London or New York offices. That policy can be implemented as a Boolean expression:

(junior level vice president) AND (London OR New York)

A circuit implements the Boolean expression that defines the policy. The policy implemented by the circuit, such as a circuit 110 of FIG. 1, a circuit 210 of FIG. 2, or a circuit 310 of FIG. 3, thus may specify that certain combinations of attributes must be true. In the above example, the job title (junior level vice president) and a certain location (New York or London) must both be true per the policy. To

continue the example, the signing policy may also require that that the signer may be either (a) a junior vice president and based in the London or New York offices or (b) a senior vice president at any office. Any types and number of logic gates can be used to implement the circuit 110, 210, or 310, such as AND gates, OR gates, NOT gates, NAND gates, NOR gates, XOR gates, XNOR gates, or any other logic gate. The fan-outs, or number of logic gates in the circuit 110, 210, or 310 are not limited to one as in the less expressive types of policies. In some implementations, the circuit is implemented as a mathematical representation of the Boolean expression and thus implemented in instructions executable by a hardware processor. In other implementations, the circuit may be an actual, physical circuit including various types of logic gates (AND gates, OR gates, etc.). All references herein to "circuit" refer to either or both types of implementations.

[0019] A digital signature is then created or a ciphertext is decrypted based on the secret key and based on the message. In other words, the digital signature is created based on the individual attribute-based keys validly passing through the mathematical representation of the policy signing based circuit. The digital signature may then be verified based on the public key. Likewise, the ciphertext may be decrypted based on the individual attribute-based hashes validly passing through the mathematical representation of the policy decryption based circuit.

[0020] Figure 1 is a block diagram of an attribute-based cryptography system, in accordance with an example. In Figure 1, a computing system 102 includes a key generation engine 104, a cryptography engine 106, and a cryptography policy circuit 110. The computing system 102 may be, for example, a notebook computer, a tablet computing device, a portable reading device, a wireless email device, a mobile phone, a workstation, a server, a desktop computer, or any other computing device. The attribute-based cryptography system may allow for a valid digital signature based on a user's attributes corresponding to a signing policy or allow for the decryption of an encrypted message based on the user's attributes corresponding to a decryption policy.

[0021] The key generation engine 104 generates a secret key that is utilized by the cryptography engine 106 to generate a digital signature or decrypt an encrypted message. The key generation engine 104 implements cryptography

policy circuit 110. In some examples, cryptography policy circuit 110 may include the key generation engine 104. In other examples, the key generation engine 104 may be implemented using cryptography policy circuit 110. Two such cryptography policy circuits include signing policy circuit 210 from Figure 2 and decryption policy circuit 310 from Figure 3. Cryptography policy circuit 110 corresponds to a cryptography policy such as a signing policy or a decryption policy. In other words, cryptography policy circuit 110 includes Boolean logic that corresponds to a policy of which users may sign or decrypt an individual message.

[0022] Cryptography policy circuit 110 receives individual attribute-based data as inputs. Such attribute-based data may include attribute-based keys corresponding to individual attributes of a user or attribute-based hashes also corresponding to individual attributes of a user. Cryptography policy circuit 110 outputs a secret key based on the individual attribute-based data corresponding to the cryptography policy.

[0023] Cryptography engine 106 receives a message that is to be signed or a ciphertext to be decrypted and generates a digital signature or decrypts the ciphertext based on the secret key and based on the message.

[0024] Figure 2 is an attribute-based cryptography system 200 implementing an attribute-based digital signature scheme, according to an example. In Figure 2, the system 200 includes a computing system 201 which also includes the key generation engine 104, the cryptography engine 106, and a signing policy circuit 210, as well as a setup engine 202. The cryptography engine 106 may include signature engine 206. The computing system 201 may be, for example, a notebook computer, a tablet computing device, a portable reading device, a wireless email device, a mobile phone, a workstation, a server, a desktop computer, or any other computing device. The signing policy circuit 210 may be one type of cryptography policy circuit 110.

[0025] System 200 also includes a verification engine 204 coupled to computing system 201 through a communications network 212. The communication network 212 can include wired communications, wireless communications, or combinations thereof. Further, the communication network 212 can include multiple sub-communication networks such as data networks, wireless networks, telephony

networks, etc. Such networks can include, for example, a public data network such as the Internet, local area networks (LANs), wide area networks (WANs), metropolitan area networks (MANs), cable networks, fiber optic networks, combinations thereof, or the like. In various examples, wireless networks may include cellular networks, satellite communications, wireless LANs, etc. Further, the communication network 212 can be in the form of a direct network link between devices. Various communications structures and infrastructure can be utilized to implement the communication network 212.

[0026] By way of example, the computing systems 201 and verification engine 204 may communicate with each other and other components with access to the communication network 212 via a communication protocol or multiple protocols. A protocol can be a set of rules that defines how nodes of the communication network 212 interact with other nodes. Further, communications between network nodes can be implemented by exchanging discrete packets of data or sending messages. Packets can include header information associated with a protocol (e.g., information on the location of the network node(s) to contact) as well as payload information.

[0027] Setup engine 202 may generate a master secret key, a public key, and other parameters which may be utilized by the key generation engine 104. The setup engine 202 receives as inputs a security parameter (λ), an upper boundary depth of all circuits (ℓ) and an upper boundary input size for all attributes (n). A trusted third party may produce these values. The security parameter (λ) is a variable that measures the resource requirements needed, such as the length, in bits, of the secret signing key to be computed. The upper boundary depth of all circuits (ℓ) is the upper boundary depth, based on the number of logic gates, for signing policy circuit 210. The upper boundary input size for all attributes (n) is the upper boundary number of attribute-based keys that may be input into signing policy circuit 210.

[0028] Setup engine 202 generates groups $G = (\mathbb{G}_1, \ldots, \mathbb{G}_{k+1})$ of prime order p, with canonical generators, $g_1, \ldots, g_{k+1}$ where $g = g_1$. Setup engine 202 also generates random values $\alpha, \beta \in \mathbb{Z}_p$, and $h_1, \ldots, h_n \in \mathbb{Z}_p$. In an example, setup engine 202 generates the master secret key as:

$$(g_{k-1})^{\alpha}, \beta, h_1, \ldots, h_n.$$

[0029] Setup engine 202 may also generate a public key. The public key may be a one-time signature (OTS) public key. An OTS public key is a public key that is utilized to verify a single message. In an OTS scheme, a secret signing key is also used to sign a single message. The underlying OTS scheme can be instantiated by a suitable one-time strongly unforgeable signature scheme. Such schemes may include those that are based on the computational Diffie-Helman CDH assumption in the standard model, or for instance, Shnorr signatures that are based on the Discrete Logarithm problem Dlog in the random oracle model using the Fiat-Shamir heuristic.

[0030] In an example, the public key generated by setup engine 202 includes the group sequence $G = (\mathbb{G}_1, \ldots, \mathbb{G}_{k+1})$ in addition to:

$$K = g_k^{\alpha}, d_0 = g^{\beta}.$$

[0031] Setup engine 202 also generates additional parameters such as by solving for $d_i = h_i^{\beta}$ for each $i \in [1, n]$. The values $d_i$may be utilized by the key generation engine 104.

[0032] The key generation engine 104 generates a secret signing key to be used to produce a digital signature. The secret signing key is a key that is not known outside of computing system 102, 201. Thus, verification engine 204 does not know the secret signing key, nor the circuit used to generate the secret signing key.

[0033] The inputs to the signing policy circuit 210 include attribute-based keys that are computed based on the set of attributes of the potential signer. Each attribute-based key corresponds to an individual attribute, or characteristic, of a user to be used to generate that user's secret signing key. For example, an attribute-based key may correspond to the position level of a user, the location the user is based, the age of the user, or any other individual characteristic of the user. A lookup table may be utilized to generate the attribute-based keys. For

example, a lookup table may match a key to a specific attribute. In another example, the attribute-based key may be based on an existing identity based scheme.

[0034] The key generation engine 104 receives any number of attribute-based keys as inputs to the circuit 210 in accordance with the signing policy implemented by the circuit 210. If the attribute-based keys correspond to the signing policy, a valid secret signing key is produced. In other words, if the attribute-based keys validly pass through the signing policy circuit 210, then a valid secret signing key is produced by the key generation engine 104. Otherwise, an invalid secret signing key is produced—a condition for which will be detected during the verification phase. Attribute-based keys corresponding to the attributes of a user who wishes to sign the message would be input into the circuit 210. If these attributes validly pass through the circuit 210, then the attribute-based keys correspond to the signing policy and a valid secret signing key is generated.

[0035] Key generation engine 104 may receive as input the master secret key and a representation (f) of circuit 210 corresponding to a signing policy for a specific message. Key generation engine 104 outputs a secret signing key (SK), such that $f(\mathrm{SK}) = 1$. In some implementations, to represent the signing policy circuit 210, a mathematical representation may be utilized. In the mathematical representation, A may refer to $A(w)$, which represents one input wire (w) for the circuit. Additionally, B may refer to $B(w)$, a second input wire for the circuit. Circuit 210 may include $n + q$ wires including n input wires and q gate wires. The wire $n + q$ may be designated as an output wire.

[0036] The key generation engine 104 may generate random values $r_1, \ldots, r_{n+q} \in \mathbb{Z}_p$, where random value r_w corresponds with wire $w \in [1, n + q]$. The key generation engine 104 generates a header component:

$$K_H = (g_{k-1})^{\alpha - r_{n+q}}.$$

The key generation engine 104 may generate key components for every wire w. The structure of the key components depends on whether w is an input wire or a logic gate, such as an OR gate or an AND gate.

[0037] The key generation engine 104 generates key components for an input wire with $w \in [1, n]$ corresponding to the w-th input. The key generation engine 104 generates a random value $z_w \in \mathbb{Z}_p$. The key generation engine 104 then may generate the key components for w to be:

$$K_{w,1} = g^{r_w} \cdot h_w^{z_w},\ K_{w,2} = g^{-z_w}.$$

[0038] The key generation engine 104 generates key components of an OR gate with wire $w \in Gate$ and GateType(w) = OR. The OR gate has two inputs (A) and (B). Additionally, j = depth(w) is the depth of wire w. Furthermore, $r_{A(w)}$ is the r_w value associated with A while $r_{B(w)}$ is the r_w value associated with B. The key generation engine 104 may generate random values $a_w, b_w \in \mathbb{Z}_p$. The key generation engine 104 then may generate the key components w to be:

$$K_{w,1} = g^{\alpha_w}, K_{w,2} = g^{b_w}, K_{w,3} = g_j^{r_w - a_w \cdot r_{A(w)}}, K_{w,4} = g_j^{r_w - b_w \cdot r_{B(w)}}.$$

[0039] The key generation engine 104 generates key components of an AND gate with wire $w \in Gate$ and GateType(w) = AND. The AND gate has two inputs (A) and (B). Additionally, j = depth(w) is the depth of wire w. Furthermore, $r_{A(w)}$ is the r_w value associated with A while $r_{B(w)}$ is the r_w value associated with B. The key generation engine 104 may generate random values $a_w, b_w \in \mathbb{Z}_p$. The key generation engine 104 then may generate the key components w to be:

$$K_{w,1} = g^{\alpha_w}, K_{w,2} = g^{b_w}, K_{w,3} = g_j^{r_w - a_w \cdot r_{A(w)} - b_w \cdot r_{B(w)}}.$$

[0040] The key generation engine 104 may then generate the secret signing key (SK) which includes $K_H, d_1, \ldots, d_n$ together with all of the key components for the inputs and logic gate wires.

[0041] Cryptography engine 106 may comprise signature engine 206. Signature engine 206 receives a message to be signed and generates a digital signature based on the secret signing key and the message to be signed. The message to be signed may be received from an outside source or from within computer system 201. As with the key generation engine 104, SK is the secret signing key and $f(\mathrm{SK}) = 1$. The message to be signed may be represented as $m \in \mathbb{Z}_p$. A digital signature on m under SK indicates that the signature engine has computed $g_k^{\alpha \cdot \beta}$ without knowing either α or β and the proof is bound with m.

[0042] Signature engine 206 may generate a random value $t \in \mathbb{Z}_p$ and generates $c_0 = d_0^t$ where c_0 is g^s. Here, $s = \beta \cdot t$ which is unknown to signature engine 206. By using the pair (d_0, c_0), in an example, is an OTS public key and the value t as the corresponding secret key, signature engine 106 may create an OTS on m, which may be denoted by σ_{OTS}.

[0043] Signature engine 206 then may compute a knowledge proof denoted by $\sigma_s = g_k^{\alpha \cdot s}$as follows. First, signature engine 206 may generate a header as:

$$E' = e(K_H, c_0) = e\left(g_{k-1}^{\alpha - r_{n+q}}, g^s\right) = g_k^{\alpha \cdot s} \cdot g_k^{-r_{n+q} \cdot s}.$$

Signature engine 206 then may compute $g_k^{r_{n+q} \cdot s}$. This may be accomplished by evaluating the signing policy circuit 210 from the bottom up. For example, if wire w is at depth j and $f_w(\mathrm{SK}) = 1$, signature engine 206 computes $E_w = (g_{j+1})^{r_w \cdot s}$. However, if $f_w(\mathrm{SK}) = 0$, signature engine 206 does nothing for that wire. Signature engine 206 iteratively starts with computing E_1 and proceeds in order through the signing policy circuit 210 until signature engine 206 computes E_{n+q}. By computing these values in order, the computation on depth $j - 1$ wire, long as it evaluates to 1, is defined before computing for a depth j wire.

[0044] Signature engine 206 may compute for E_w where $f_w(\mathrm{SK}) = 1$ of an input wire as follows. Utilizing the convention $w \in [1, n]$, the input wire corresponds to the w-th input. If that wire w satisfies $f_w(\mathrm{SK}) = 1$, then signature engine 106 computes $c_w = d_w^t$ and then may compute:

$$E_w = e\left(K_{w,1}, c_0\right) \cdot e\left(K_{w,2}, c_w\right) = e\left(g^{r_w} \cdot h_w^{z_w}, g^s\right) \cdot e\left(g^{-z_w} \cdot h_w^s\right) = g_2^{r_w \cdot s}.$$

[0045] Signature engine 206 may compute E_w where $f_w(\mathrm{SK}) = 1$ of an OR gate with wire $w \in Gate$ and GateType(w) = OR as follows. With $j = \mathrm{depth}(w)$ being the depth of wire w and $f_w(\mathrm{SK}) = 1$, then if $f_{A(w)} = 1$ and $f_{B(w)} = 0$, signature engine 206 computes:

$$E_w = e\left(E_{A(w)}, K_{w,1}\right) \cdot e\left(K_{w,3}, c_0\right) = e\left(g_j^{r_{A(w)} \cdot s}, g^{a_w}\right) \cdot e\left(g_j^{r_w - a_w \cdot r_{A(w)}}, c_0\right) = (g_{j+1})^{r_w \cdot s}.$$

However, if $f_{A(w)} = 0$ but $f_{B(w)} = 1$, signature engine 206 computes:

$$E_w = e\left(E_{B(w)}, K_{w,2}\right) \cdot e\left(K_{w,4}, c_0\right) = e\left(g_j^{r_{B(w)} \cdot s}, g^{b_w}\right) \cdot e\left(g_j^{r_w - b_w \cdot r_{B(w)}}, c_0\right) = (g_{j+1})^{r_w \cdot s}.$$

[0046] Signature engine 206 may compute E_w where $f_w(\mathrm{SK}) = 1$ of an AND gate with wire $w \in Gate$ and GateType(w) = AND as follows. With $j = \mathrm{depth}(w)$ being the depth of wire w and $f_w(\mathrm{SK}) = 1$, then $f_{A(w)}(\mathrm{SK}) = 1$ and $f_{B(w)}(\mathrm{SK}) = 1$, and signature engine 206 computes:

$$E_w = e\left(E_{A(w)}, K_{w,1}\right) \cdot e\left(E_{B(w)}, K_{w,2}\right) \cdot e\left(K_{w,3}, c_0\right) =$$

$$e\left(g_j^{r_{A(w)} \cdot s}, g^{a_w}\right) \cdot e\left(g_j^{r_{B(w)} \cdot s}, g^{b_w}\right) \cdot e\left(g_j^{r_w - a_w \cdot r_{A(w)} - b_w \cdot r_{B(w)}}, c_0\right) = (g_{j+1})^{r_w \cdot s}.$$

[0047] Thus, if $f(\mathrm{SK}) = f_{n+q}(\mathrm{SK}) = 1$, signature engine 206 generates $E_{n+q} = g_k^{r_{n+q} \cdot s}$. Signature engine 206 then may compute $\sigma_s = E' \cdot E_{n+q} = E_k^{\alpha \cdot s}$. The entire digital signature may be expressed as:

$$\sigma = \left(c_0, \sigma_{(OTS)}, \sigma_s\right).$$

[0048] Verification engine 204 is connected to computing system 201 through the communication network 212. Although shown separately, in some examples, verification engine 204 is instead a part of computing system 201. Verification engine 204 verifies the validity of the digital signature created by signature engine 206 based on the public key created by setup engine 202.

[0049] In order to verify the digital signature $\sigma = \left(c_0, \sigma_{(OTS)}, \sigma_s\right)$, verification engine 204 may perform a suitable verification. One such verification includes verification engine 204 verifying $\sigma_{(OTS)}$ under the OTS public key (d_0, c_0). This may be accomplished through following the verification of the underlying OTS scheme. By this verification, verification engine 204 may determine that $c_0 = d_0^t$ and that the value t is known by the signature engine. The signing policy is not involved in any of the verification equations utilized by verification engine 204 to verify the validity of the signature. Thus, the verification engine 204 does not need to know, and in fact may not know, the signing policy.

[0050] Another verification that verification engine 204 may perform is to verify σ_s by determining whether $e(\sigma_s, g) = e(K, c_0)$ holds true. This allows verification engine 204 to determine that $\sigma_s = K^s$ for some value of s that is involved in the secret signing key.

[0051] While two different verifications are presented, either one may be performed or both verifications may be performed. If verifications performed indicate that the signature σ is valid, verification engine 204 accepts the digital

signature. However, if the verifications performed indicate that the signature σ is not valid, verification engine 204 rejects the digital signature.

[0052] Because the verification engine 204 does not have access to the mathematical representation of the circuit 210 or to any of the inputs into the circuit 210, the systems described herein may have privacy of the policy and privacy of the attributes. That is to say, that the verification engine 204 does not know the signing policy for the message nor does verification engine 204 know what attributes the user whose digital signature is created has.

[0053] Figure 3 shows a block diagram of an attribute-based cryptography system 300 implementing an attribute-based decryption scheme, according to an example. In Figure 3, the system 300 includes a computing system 301 which also includes the key generation engine 104, the cryptography engine 106, and a decryption policy circuit 310, as well as a setup engine 302 and encryption engine 304. The cryptography engine 106 may include decryption engine 306. The computing system 301 may be, for example, a notebook computer, a tablet computing device, a portable reading device, a wireless email device, a mobile phone, a workstation, a server, a desktop computer, or any other computing device. Decryption policy circuit 310 may be one type of cryptography policy circuit 110.

[0054] Setup engine 302 may generate a master secret key and a set of public parameters $(\mathcal{P})$ which may be utilized by encryption engine 304, the key generation engine 104, and decryption engine 306. The setup engine 302 receives as inputs a security parameter (λ), an upper boundary depth of all circuits (ℓ) and a number of Boolean inputs (n). A trusted third party may produce these values. The security parameter (λ) is a variable that measures the resource requirements needed, such as the length, in bits, of the secret signing key to be computed. The upper boundary depth of all circuits (ℓ) is the upper boundary depth, based on the number of logic gates, for circuit 310. The Boolean inputs (n) is the input size for all ciphertexts that will be input into circuit 310.

[0055] Setup engine 302 generates groups $G = (\mathbb{G}_1, \ldots, \mathbb{G}_{k+1})$ of prime order p, with canonical generators, $g_1, \ldots, g_{k+1}$ where $g = g_1$. Setup engine 302 also generates random values $\alpha \in \mathbb{Z}_p$ and $h_1, \ldots, h_n \in \mathbb{G}_1$. The public parameters $(\mathcal{P})$ consist of the group sequence description $= (\mathbb{G}_1, \ldots, \mathbb{G}_{k+1})$ in addition to:

$$\mathcal{P} = g_k^{\alpha}, h_1, \ldots, h_n$$

[0056] Setup engine 302 may map an individual attribute with a group element h_i. To accomplish this, setup engine 302 may utilize a hash to point technique to hash the attribute $a_i \in U$ into the point $h_i \in \mathbb{G}$ (i.e. $h_i = \mathcal{H}(a_i)$ where $\mathcal{H}: \{0,1\}^* \rightarrow \mathbb{G}$ is a suitable hash to point type of hash function. Setup engine 302, in an example, may hash the attributes into group elements (i.e. $\mathcal{H}: \{0,1\}^* \rightarrow \mathbb{G}$), so that $h_i = \mathcal{H}(a_i)$.

[0057] In an alternative example, setup engine 302 may hash the decryption policy (Ψ) along with the attributes, so that users that have secret keys that correspond with the decryption policy may decrypt. In this example, the hash function is $\mathcal{H}_2: \{0,1\}^* \times \{0,1\}^* \rightarrow \mathbb{G}$ where $h_i = \mathcal{H}_2(a_i \parallel \Psi)$. In this example, if a user uses $\{h_i\}_i$ as a basis to encrypt a particular message, any user who wishes to decrypt that data should have a secret key that is specifically corresponds to the decryption policy (e.g., being the attribute manager is not enough, by itself, to decrypt, the user is required to own a credential associated with $manager \parallel \Psi$).

[0058] Utilizing this hashing technique, setup engine 302 may generate a set of public group elements $\{h_i\}, i = 1, \ldots, n$ as follows:

$$h_i = \mathcal{H}(att_i).$$

Thus, the public parameters may be reduced by the setup engine 302 to:

$$\mathcal{P} = g_k^{\alpha}.$$

Hence the setup engine 302 may generate the master secret key as:

$$g_{k-1}^{\alpha}.$$

[0059] Encryption engine 304 may receive the public parameters $\mathcal{P}$ and a message M to create ciphertext for the message, and thus, encrypt the message. Encryption engine 304 first generates $s \in \mathbb{Z}_p$ based on the public parameters $\mathcal{P}$, the input x that describes which attributes are being used, and a message bit $M \in G_k$. Encryption engine 304 then may generate $C_M = M \cdot (g_k^{\alpha})$. With Sbeing

the set of i for which $x_i = 1$, encryption engine 304 generates the following ciphertext:

$$\text{CT}= (C_M, g^s, \forall i \in S \qquad C_i = h_i^S).$$

[0060] The key generation engine 104 generates a secret signing key to be used by the decryption engine 306 for decryption of the encrypted message. The inputs to the decryption policy circuit 310 include the attribute-based hashes generated by setup engine 302. Each attribute-based hash corresponds to an individual attribute, or characteristic, of a user to be used to generate that user's secret key. For example, an attribute-based hash may correspond to the position level of a user, the location the user is based, the age of the user, or any other individual characteristic of the user.

[0061] The key generation engine 104 receives any number of attribute-based hashes as inputs to circuit 310 in accordance with the decryption policy implemented by circuit 310. If the attribute-based hashes correspond with the decryption policy, a valid secret key is produced. In other words, if the attribute-based hashes validly pass through the decryption policy circuit 310, then a valid secret key is produced by the key generation engine 104. Otherwise, an invalid secret signing key is produced—a condition for which is detected during the decryption phase. Attribute-based hashes corresponding to the attributes of a user who wishes to decrypt the message would be input into the circuit 310. If these attributes validly pass through the circuit 310, then the attribute-based hashes correspond to the decryption policy and a valid secret key is generated.

[0062] The key generation engine 104 may receive as inputs the master secret key generated by setup engine 302 and a representation (f) of circuit 310 corresponding to a decryption policy for a specific message. The representation (f) of circuit 310 has a total of $n + q$ wires with n inputs, and q gates. The wire $n + q$ represents the output wire of circuit 310. The key generation engine 104 then may generate a set of keys (i.e. the header key (K_H), the input wire keys (K_{in}), the OR gate keys (K_{ORs}), and the AND gate keys (K_{ANDs})) in a similar way as discussed under Figure 2. Thus, the secret key generated by the key generation engine 104 is:

$$\text{SK}= \{K_H, K_{in}, K_{ORs}, K_{ANDs}\}$$

[0063] Decryption engine 306 may validly decrypt the message given a ciphertext CT corresponding with input $x \in \{0,1\}^n$ and a secret key SK associated with representation (f) of circuit 310 if $f(x) = 1$. In other words, decryption engine 306 may decrypt the message if the attributes validly pass through the circuit 310. Decryption engine 306 generates $g_k^{\alpha s}$ to retrieve the message M, since $M = C_M / g_k^{\alpha s}$. Utilizing the header key K_H, decryption engine 306 may generate the following:

$$A = e(K_H, g^s) \text{ which is } g_k^{\alpha s} \cdot g_k^{-r_{n+q} \cdot s}.$$

Therefore, decryption engine 306 may generate:

$$g_k^{\alpha s} = A / g_k^{-r_{n+q} \cdot s}.$$

Thus, $g_k^{\alpha s}$ may be reduced to determining $g_k^{-r_{n+q} \cdot s}$ which is possible if circuit 310 is evaluated from the bottom up so long as $f(x) = 1$.

[0064] Figure 4 shows an example of cryptography circuit 110. The circuit depicted in Figure 4 may also represent circuit 210 and circuit 310. Circuit 110 from Figure 4 includes input wires 402, 404, 406, and 410, AND gate 408, OR gate 412 and output wire 414. The collection of wires and gates implements a particular cryptography policy such as a signing policy or a decryption policy. While a specific configuration is depicted in Figure 4, circuit 110, which is mathematically represented by the key generation engine 104 and cryptography engine 106, may be any combination of logic gates, such as AND gates, OR gates, NOT gates, NAND gates, NOR gates, XOR gates, XNOR gates, or any other logic gate that maps to a signing policy for a particular message.

[0065] In the example shown in Figure 4, the cryptography policy requires attribute-based keys or attribute-based hashes input as input wires 402 and 404 to validly pass through AND gate 408. Thus, both attribute-based keys input as input wires 402 and 404 may meet the signing policy or both attribute-based hashes input as input wires 402 and 404 may meet the decryption policy. For example, both inputs 402 and 404 may conform to the policy. The output of AND gate 408 acts as an input wire 410 along with input wire 406 for OR gate 412. In this case one of input wire 410 or input wire 406 may conform to the requirements for the output 414 of the circuit 110 to be valid. In the case of a digital signature,

E_{n+q} is the output 414 of the cryptography policy circuit 110. As discussed previously under Figure 2, this is multiplied by E' in order to get σ_s which is a part of the final signature. Moreover, inputs 402, 404, and 406 are algebraic group elements, thus, they are multi-bit attribute based keys.

[0066] Figures 5, 6, and 7 are flowcharts of attribute-based cryptography methods 500, 600, and 700. Although the execution of methods 500, 600, and 700 is described below with reference to systems 102, 200, and 300, other suitable components for execution of methods 500, 600, and 700 can be utilized (e.g., computing device 800). Additionally, the components for executing the methods 500, 600, and 700 may be spread among multiple devices. Methods 500, 600, and 700 may be implemented in the form of processor executable instructions stored on a non-transitory machine-readable storage medium, such as the machine-readable storage medium 820, and/or in the form of electronic circuitry.

[0067] Method 500 begins at 502 with inputting a plurality of individual attribute-based data into circuit 110. As explained above, the circuit 110 comprises a plurality of logic gates which implement a Boolean expression that defines a signing or decryption policy for a message. Each of the plurality of individual attribute-based data may correspond to an individual attribute. Each individual attribute may correspond to a characteristic representative of a user seeking to sign a message. The individual attribute-based data may include individual attribute-based keys and individual attribute-based hashes.

[0068] The method continues at 504 with generating, by the key generation engine 104, a secret key, such as a secret signing key, based on the plurality of individual attribute-based data corresponding to the signing or decryption policy. The method continues at 506 with receiving, by the cryptography engine 106, a message to be signed or decrypted. The message to be decrypted may be ciphertext. The method continues at 508 with generating, by the cryptography engine 106, a digital signature or decrypt, by the cryptography engine 106, based on the secret key and based on the message.

[0069] Method 600 begins at 602 with generating a master secret key by the setup engine 202. The method continues at 604 with generating a public key by the setup engine 202. The public key may be an OTS public key. The method continues at 606 with inputting a plurality of individual attribute-based keys into the signing policy circuit 210. The circuit 210 comprises a plurality of logic gates. The logic gates correspond to a signing policy for a message.

[0070] The method continues at 608 with generating, by the key generation engine 104, a secret signing key based on the plurality of individual attribute-based keys corresponding to the signing policy. The method continues at 610 with receiving, by the signature engine 206, a message to be signed. The method continues at 612 with generating, by the signature engine 206, a digital signature based on the secret signing key and the message to be signed. The method continues at 614 with verifying, by verification engine 204, the digital signature.

[0071] Method 700 begins at 702 with generating a master secret key by the setup engine 302. The method continues at 704 with generating public parameters by the setup engine 302. The method continues at 706 with encrypting, by encryption engine 304 a message. The method continues at 708 with inputting a plurality of individual attribute-based hashes into the decryption policy circuit 310. The circuit 310 comprises a plurality of logic gates that correspond to a decryption policy for a message.

[0072] The method continues at 710 with generating, by the key generation engine 104, a secret key based on the plurality of individual attribute-based hashes corresponding to the decryption policy. The method continues at 712 with receiving, by the decryption engine 306 the encrypted message as ciphertext. The method continues at 714 with decrypting, by the decryption engine 306, the encrypted message.

[0073] Figure 8 is a block diagram of a computing device 800 to provide attribute-based cryptography, according to an example. The computing device 800 includes, for example, a processing resource 830, and a non-transitory machine-readable storage medium 820 including instructions 802, 804, 806, and 808 for providing attribute-based cryptography including an attribute-based digital

signature and attribute-based decryption. Computing device 800 may be, for example, a notebook computer, a tablet computing device, a portable reading device, a wireless email device, a mobile phone, a workstation, a server, a desktop computer, or any other computing device.

[0074] Processing resource 830 may include a single processor, multiple processors, a single computer, a network of computers, or any other type of processing device suitable for retrieval and execution of instructions stored in machine-readable storage medium 820. For example, the processing resource 830 may include multiple cores on a chip, include multiple cores across multiple chips, multiple cores across multiple devices (e.g., if the computing device 800 includes multiple node devices), or combinations thereof. Processing resource 830 may fetch, decode, and execute instructions 802-808 to implement methods 500, 600, and 700. As an alternative or in addition to retrieving and executing instructions, processing resource 830 may include at least one integrated circuit (IC), other control logic, other electronic circuits, or combinations thereof that include a number of electronic components for performing the functionality of instructions 802-808.

[0075] The various engines depicted in Figures 1, 2, and 3 are implemented as the processing resource 830 executing machine instructions such as those illustrated in Figure 8.

[0076] The non-transitory machine-readable storage medium 820 may be any electronic, magnetic, optical, or other physical storage device that contains or stores executable instructions. Thus, machine-readable storage medium may be, for example, Random Access Memory (RAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage drive, a Compact Disc Read Only Memory (CD-ROM), and the like. As described in detail herein, machine-readable storage medium 820 may be encoded with a series of executable instructions for providing a digital signature.

[0077] Setup instructions 802 can be used to generate a public key. The public key may by an OTS public key. Setup instructions 802 may also be used to generate a master secret key. Key generation instructions 804 may be used to input a plurality of individual attribute-based keys or attribute-based hashes into a

mathematical representation of a circuit. The circuit may comprise a plurality of coupled logic gates which correspond to a policy, such as a signing policy or a decryption policy. Each of the plurality of individual attribute-based keys or hashes may correspond to an individual attribute. The individual attribute may correspond to a characteristic representative of a user seeking to sign or decrypt a ciphertext.

[0078] Key generation instructions 804 may also be used to generate a secret key based on the plurality of individual attribute-based keys or hashes corresponding to the signing or decryption policy. The secret key may also be based on the master secret key.

[0079] The cryptography instructions 806 may receive a message to be signed or decrypted. The cryptography instructions 806 may also generate a digital signature or decrypt the message based on the secret key and the message to be signed. Verification instructions 808 may verify the digital signature based on the public key.

[0080] The above discussion is meant to be illustrative of the principles and various embodiments of the present disclosure. Numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.

CLAIMS

What is claimed is:

1. An attribute-based cryptography system, comprising:

a key generation engine to generate a secret key, the key generation engine to implement a cryptography policy circuit that defines a cryptography policy, the cryptography policy circuit to receive as an input a plurality of individual attribute-based data and to output the secret key based on the individual attribute based data corresponding to the cryptography policy; and

a cryptography engine to receive ciphertext or a message and to decrypt the ciphertext or generate a digital signature based on the secret key and based on the message;

wherein each of the plurality of individual attribute-based data corresponds to an individual attribute.

2. The attribute-based cryptography system of claim 1, further comprising:

a verification engine to accept or reject a digital signature, the verification engine not having access to the cryptography policy, the cryptography policy being a signing policy;

wherein the cryptography engine comprises a signature engine to receive the message and to generate the digital signature,

wherein the cryptography policy circuit is a signing policy circuit comprising Boolean logic that implements the signing policy, and

wherein each of the plurality of individual attribute-based data is an individual attribute-based key.

3. The attribute-based cryptography system of claim 2, wherein each individual attribute is selected from a group comprising a position level of a user, a location the user is based, and an age of the user.

4. The attribute-based cryptography system of claim 2, further comprising a setup engine to generate a public key, the public key being one-time signature (OTS) public key.

5. The attribute-based cryptography system of claim 1,

wherein the cryptography engine comprises a decryption engine to receive the ciphertext and to decrypt the ciphertext based on the individual attribute-based data corresponding to a cryptography policy, the cryptography policy being a decryption policy;

wherein the cryptography policy circuit is a decryption policy circuit comprising Boolean logic that implements the decryption policy; and

wherein each of the individual attribute-based data is a hash of the individual attribute.

6. A non-transitory machine-readable storage medium storing instructions that, if executed by a processing resource of an attribute-based cryptography system, cause the processing resource to:

generate a public key;

input a plurality of individual attribute-based data into a circuit defining a cryptography policy;

generate a secret key via the circuit based on the plurality of individual attribute-based data corresponding to the cryptography policy; and

generate a digital signature or decrypt a ciphertext based on based on the secret key and based on the public key;

wherein each of the plurality of individual attribute-based data corresponds to an individual attribute.

7. The non-transitory machine-readable storage medium of claim 6, further comprising instructions that, if executed by the processing resource, causes the processing resource to generate a master secret key based on an upper

boundary depth of the circuit, wherein the secret key is further based on the master secret key.

8. The non-transitory machine-readable storage medium of claim 6, further comprising instructions that, if executed by the processing resource, wherein the public key is a one-time signature (OTS) public key used by a verification engine to verify a single message.

9. The non-transitory machine-readable storage medium of claim 6,

wherein the plurality of individual attribute-based data is a plurality of individual attribute-based keys;

wherein the circuit is signing policy circuit comprising Boolean logic that implements a signing policy; and

wherein the individual attribute corresponds to a characteristic representative of a user seeking to sign a message.

10. The non-transitory machine-readable storage medium of claim 6,

wherein the plurality of individual attribute-based data is a plurality of individual attribute-based hashes;

wherein the circuit is a decryption policy circuit comprising Boolean logic that implements a decryption policy; and

wherein the individual attribute corresponds to a characteristic representative of a user seeking to decrypt the ciphertext.

11. An attribute-based cryptography method, comprising:

inputting a plurality of individual attribute-based data into a mathematical representation of a plurality of coupled logic gates corresponding to a cryptography policy;

generating, by at least one processor, a secret key based on the plurality individual attribute-based data corresponding to the cryptography policy;

receiving, by the at least one processor, a message; and

generating a digital signature or decrypting the ciphertext based on the secret key and based on the message;

wherein each of the plurality of individual attribute-based keys corresponds to an individual attribute.

12. The attribute-based cryptography method of claim 11, further comprising verifying the digital signature such that a signing policy is not known when the digital signature is verified;

wherein the individual attribute-based data is a plurality of individual attribute-based keys;

wherein the cryptography policy is a signing policy; and

wherein the individual attribute corresponds to a characteristic representative of a user seeking to sign the message.

13. The attribute-based cryptography method of claim 12, further comprising generating a master secret key and a public key, wherein a secret signing key is further based on the master secret key and the public key, the public key being a one-time signature (OTS) public key.

14. The attribute-based cryptography method of claim 11, further comprising encrypting the message;

wherein the individual attribute-based data is a plurality of individual attribute-based hashes;

wherein the cryptography policy is a decryption policy; and

wherein the individual attribute corresponds to a characteristic representative of a user seeking to decrypt the ciphertext.

15. The attribute-based cryptography method of claim 14, further comprising generating a master secret key and public parameters;

wherein the generating the secret key is further based on the master secret key and the public parameters.

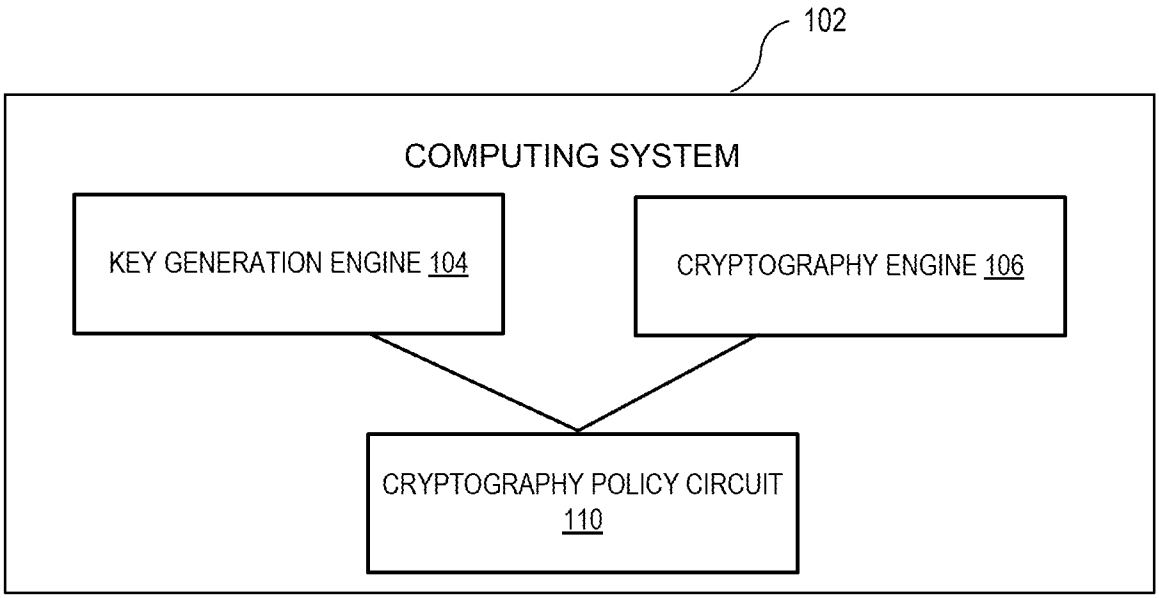
102
COMPUTING SYSTEM
KEY GENERATION ENGINE 104
CRYPTOGRAPHY ENGINE 106
CRYPTOGRAPHY POLICY CIRCUIT 110

FIG. 1

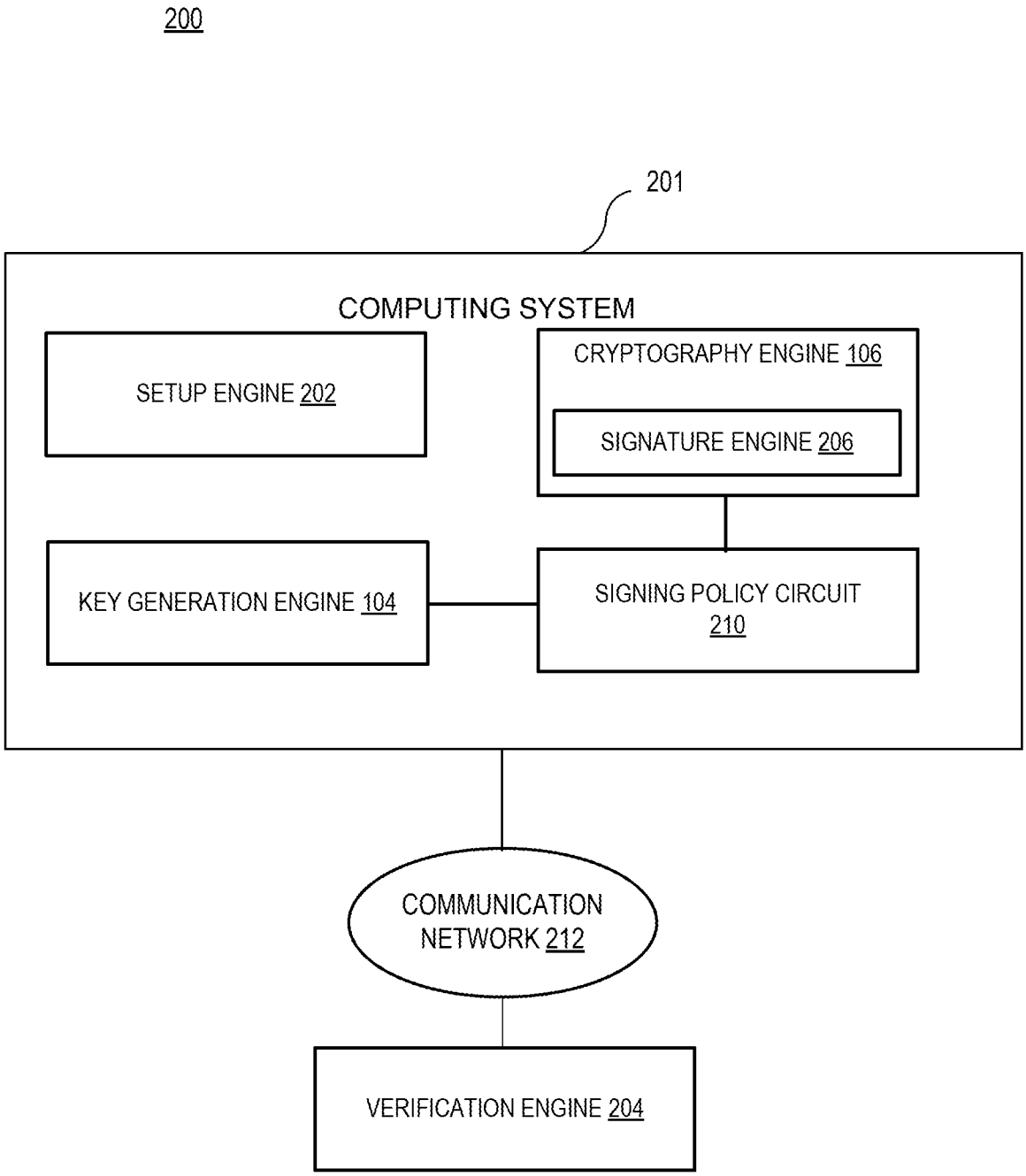
200
201
COMPUTING SYSTEM
SETUP ENGINE 202
CRYPTOGRAPHY ENGINE 106
SIGNATURE ENGINE 206
KEY GENERATION ENGINE 104
SIGNING POLICY CIRCUIT 210
COMMUNICATION NETWORK 212
VERIFICATION ENGINE 204

FIG. 2

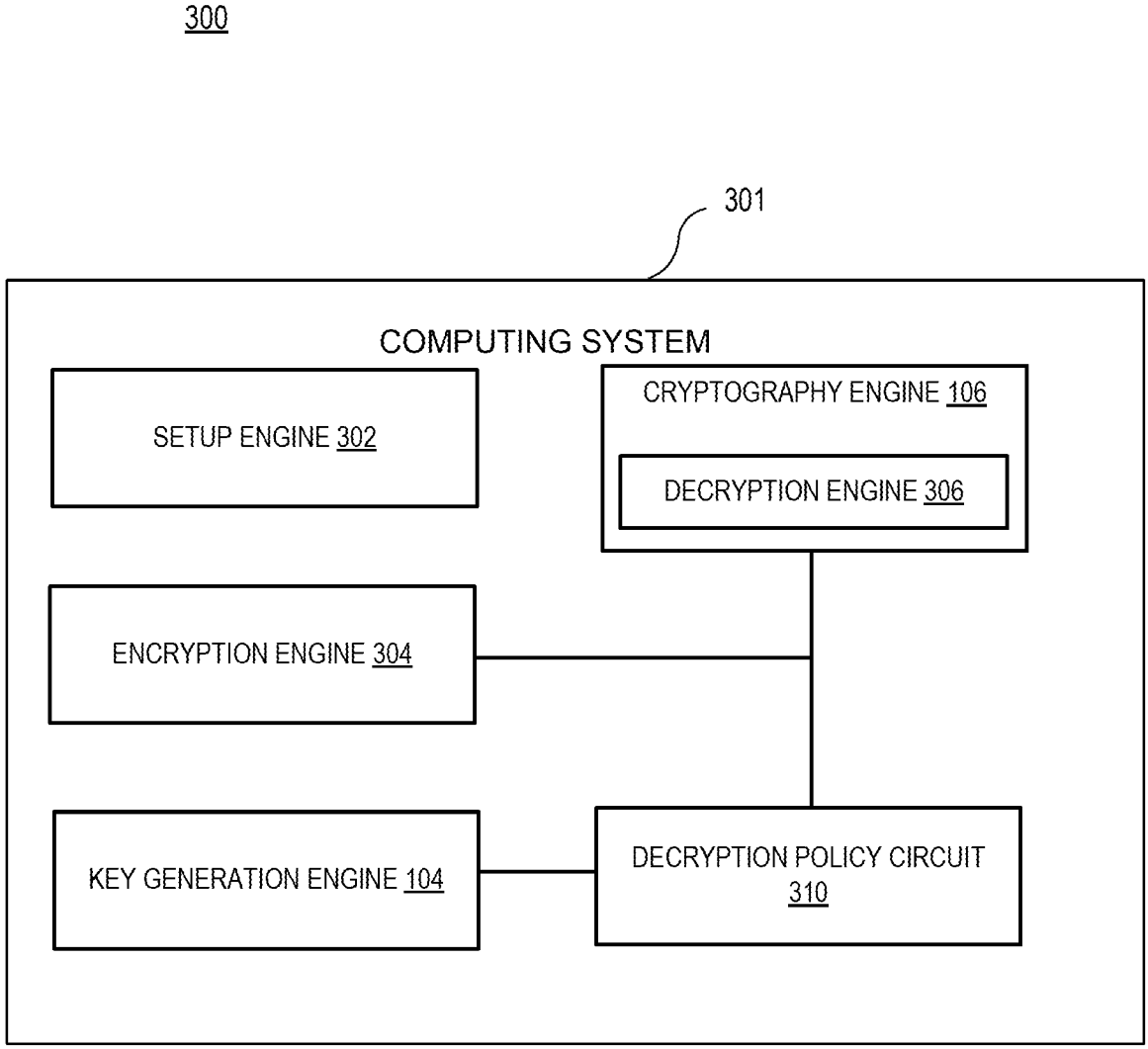
300
301
COMPUTING SYSTEM
SETUP ENGINE 302
CRYPTOGRAPHY ENGINE 106
DECRYPTION ENGINE 306
ENCRYPTION ENGINE 304
KEY GENERATION ENGINE 104
DECRYPTION POLICY CIRCUIT 310

FIG. 3

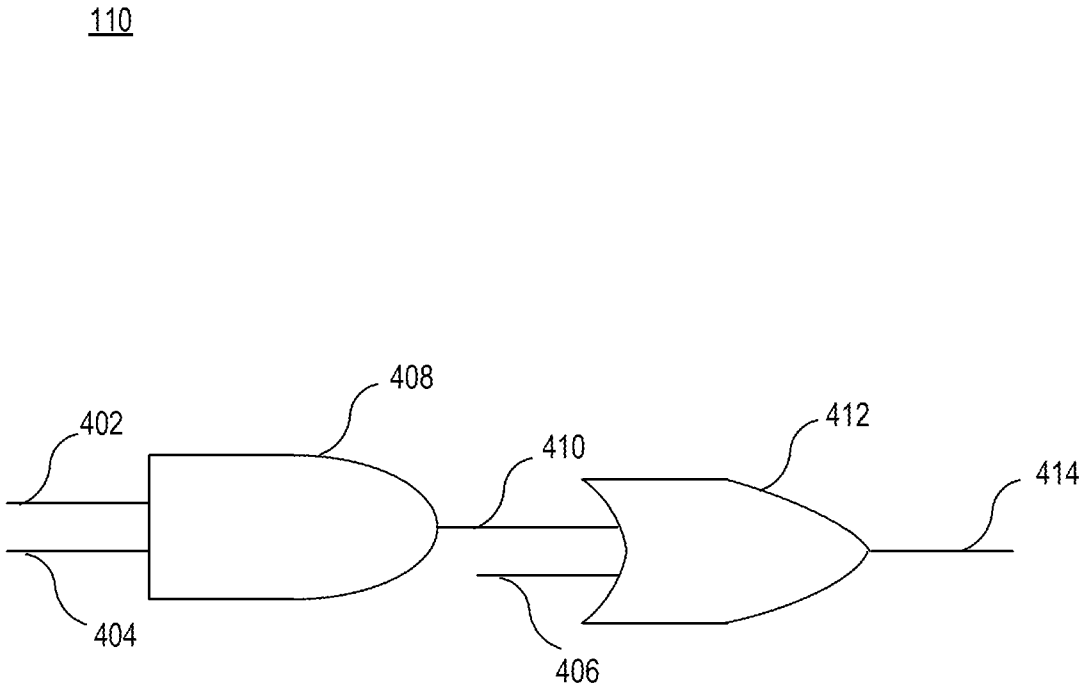

FIG. 4

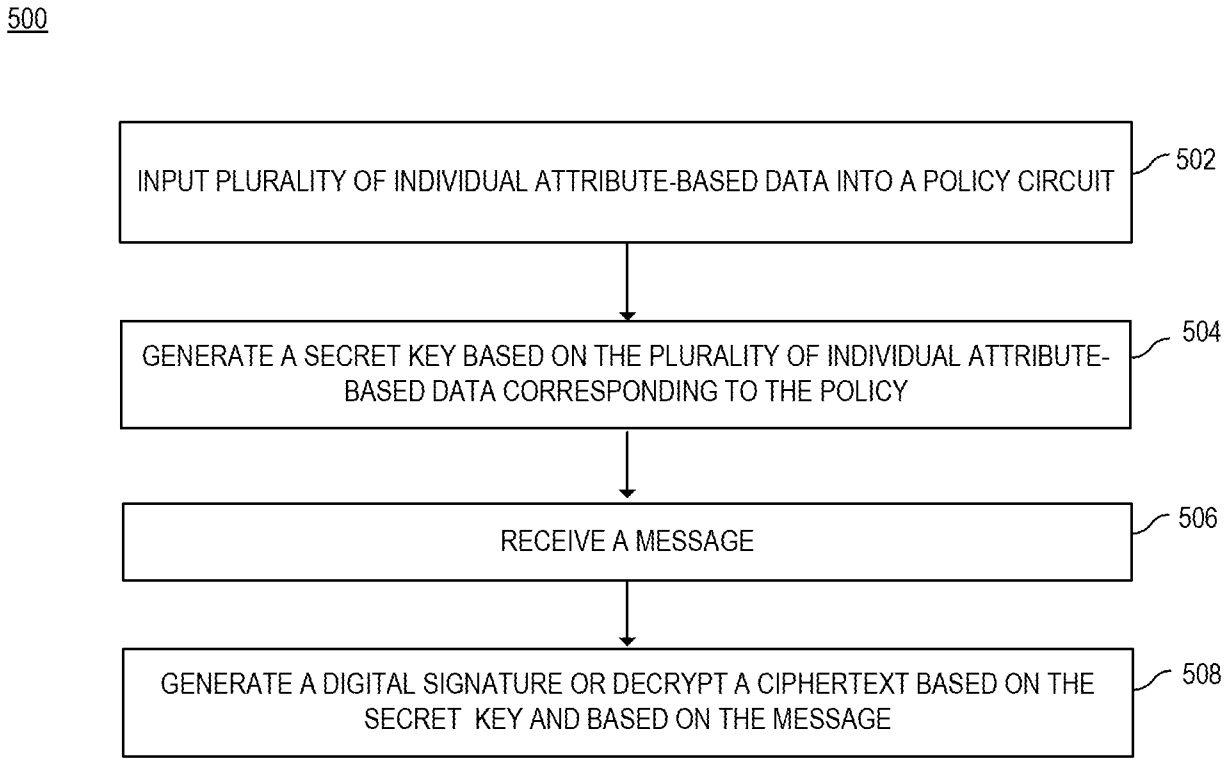
500
INPUT PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED DATA INTO A POLICY CIRCUIT
502
GENERATE A SECRET KEY BASED ON THE PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED DATA CORRESPONDING TO THE POLICY
504
RECEIVE A MESSAGE
506
GENERATE A DIGITAL SIGNATURE OR DECRYPT A CIPHERTEXT BASED ON THE SECRET KEY AND BASED ON THE MESSAGE
508

FIG. 5

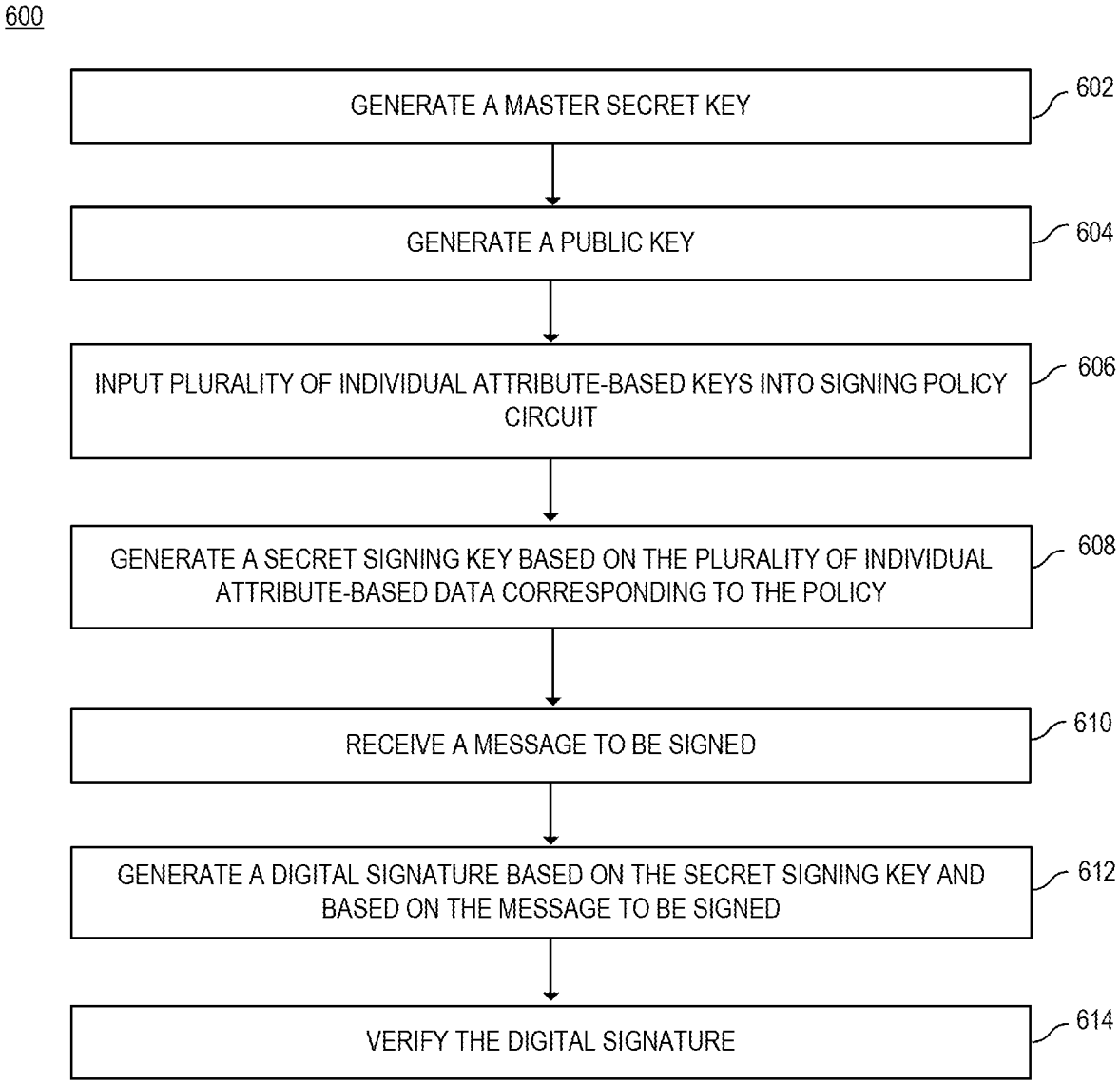
600
GENERATE A MASTER SECRET KEY
602
GENERATE A PUBLIC KEY
604
INPUT PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED KEYS INTO SIGNING POLICY CIRCUIT
606
GENERATE A SECRET SIGNING KEY BASED ON THE PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED DATA CORRESPONDING TO THE POLICY
608
RECEIVE A MESSAGE TO BE SIGNED
610
GENERATE A DIGITAL SIGNATURE BASED ON THE SECRET SIGNING KEY AND BASED ON THE MESSAGE TO BE SIGNED
612
VERIFY THE DIGITAL SIGNATURE
614

FIG. 6

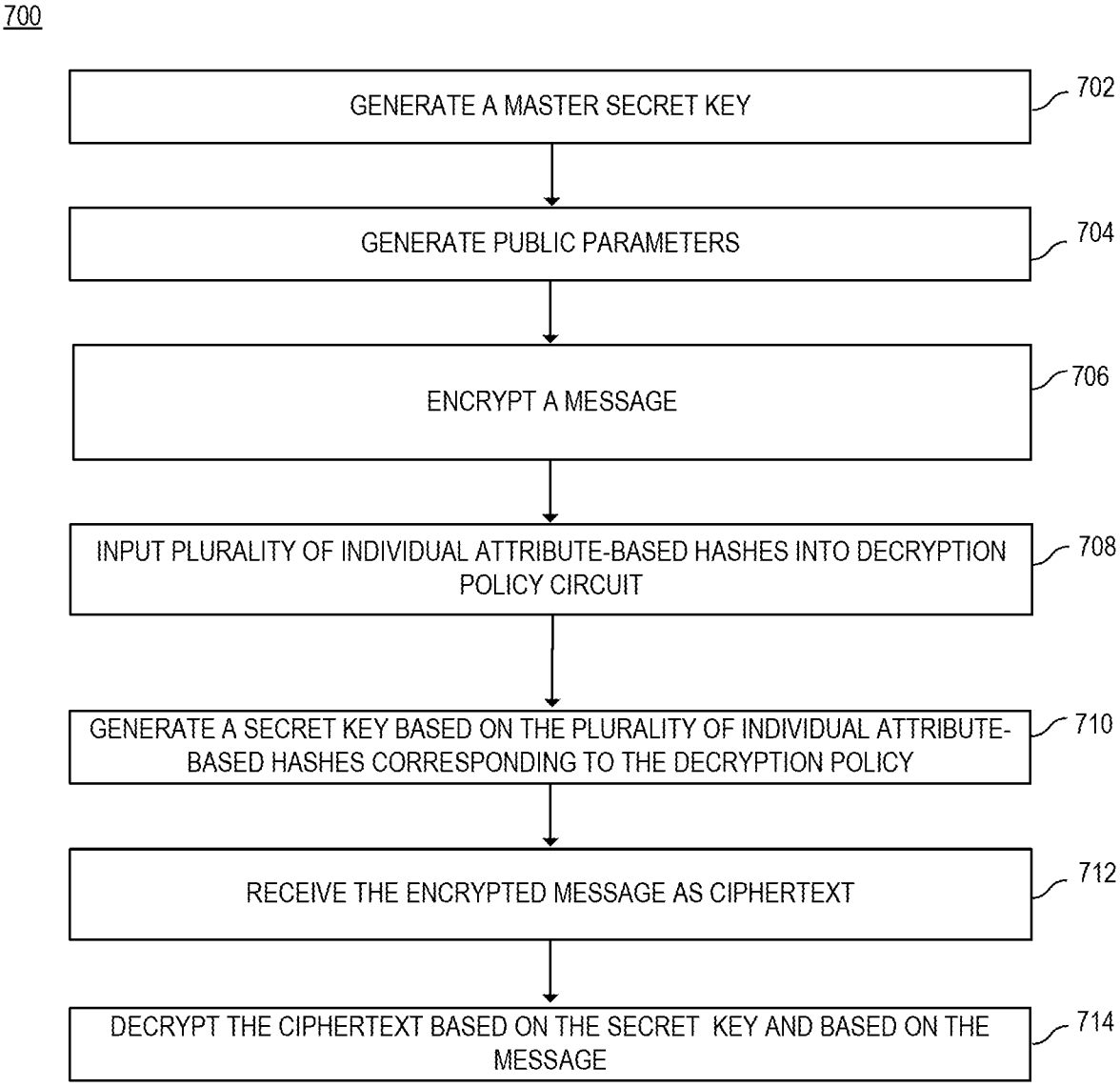
700
GENERATE A MASTER SECRET KEY
702
GENERATE PUBLIC PARAMETERS
704
ENCRYPT A MESSAGE
706
INPUT PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED HASHES INTO DECRYPTION POLICY CIRCUIT
708
GENERATE A SECRET KEY BASED ON THE PLURALITY OF INDIVIDUAL ATTRIBUTE-BASED HASHES CORRESPONDING TO THE DECRYPTION POLICY
710
RECEIVE THE ENCRYPTED MESSAGE AS CIPHERTEXT
712
DECRYPT THE CIPHERTEXT BASED ON THE SECRET KEY AND BASED ON THE MESSAGE
714

FIG. 7

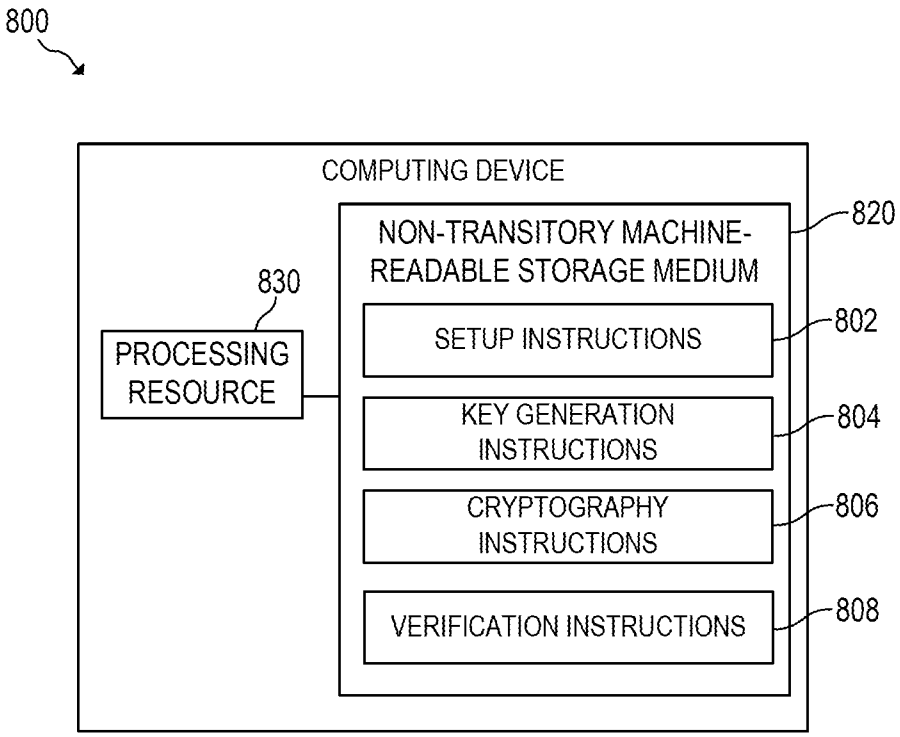
800
COMPUTING DEVICE
NON-TRANSITORY MACHINE-READABLE STORAGE MEDIUM
820
830
PROCESSING RESOURCE
SETUP INSTRUCTIONS
802
KEY GENERATION INSTRUCTIONS
804
CRYPTOGRAPHY INSTRUCTIONS
806
VERIFICATION INSTRUCTIONS
808

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/047773

A. CLASSIFICATION OF SUBJECT MATTER

H04L 9/08(2006.01)i, H04L 9/30(2006.01)i

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 9/08; H04L 9/22; H04L 9/32; G06F 12/14; G06F 21/00; H04L 9/30

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & keywords: attribute-based cryptography system, digital signature, cryptography policy

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2012-0260094 A1 (MUHAMMAD ASIM et al.) 11 October 2012 See paragraphs [0047]-[0088]; and figures 1-7.	1-15
Y	US 2013-0073860 A1 (LUAN IBRAIMI et al.) 21 March 2013 See paragraphs [0033]-[0089]; and figures 1-4.	1-15
Y	US 2012-0144210 A1 (YACOV YACOBI) 07 June 2012 See paragraphs [0048]-[0058]; and figures 4-8.	2-5,9,10,12-15
A	US 2013-0322627 A1 (KATSUYUKI TAKASHIMA et al.) 05 December 2013 See paragraphs [0223]-[0530]; and figures 5-23.	1-15
A	US 2012-0063593 A1 (JAN LEONHARD CAMENISCH et al.) 15 March 2012 See paragraphs [0028]-[0082]; and figures 1-13.	1-15

☐ 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 application or patent 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	Date of mailing of the international search report
07 May 2015 (07.05.2015)	**08 May 2015 (08.05.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea
Facsimile No. ++82 42 472 7140

Authorized officer
KIM, Do Weon
Telephone No. +82-42-481-5560

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/047773

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2012-0260094 A1	11/10/2012	CN 102656591 A	05/09/2012
		EP 2513832 A1	24/10/2012
		JP 2013-514577 A	25/04/2013
		RU 2012130355 A	27/01/2014
		WO 2011-073894 A1	23/06/2011
US 2013-0073860 A1	21/03/2013	CN 102907038 A	30/01/2013
		EP 2572471 A1	27/03/2013
		JP 2013-527712 A	27/06/2013
		WO 2011-145011 A1	24/11/2011
US 2012-0144210 A1	07/06/2012	US 8635464 B2	21/01/2014
		WO 2012-075446 A2	07/06/2012
		WO 2012-075446 A3	30/08/2012
US 2013-0322627 A1	05/12/2013	CN 103348624 A	09/10/2013
		EP 2670081 A1	04/12/2013
		JP 05606344 B2	15/10/2014
		JP 2012-155088 A	16/08/2012
		KR 10-1431412 B1	19/08/2014
		KR 10-2013-0130810 A	02/12/2013
		WO 2012-101913 A1	02/08/2012
US 2012-0063593 A1	15/03/2012	US 2014-0059345 A1	27/02/2014
		US 8577029 B2	05/11/2013