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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 April 2011 (21.04.2011)

PCT

(10) International Publication Number
WO 2011/045723 A1

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

(21) International Application Number:
PCT/IB2010/054581

(22) International Filing Date:
11 October 2010 (11.10.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09173141.4 15 October 2009 (15.10.2009) EP

(71) Applicant *(for all designated States except US)*: **KONINKLIJKE PHILIPS ELECTRONICS N.V.** [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **ASIM, Muhammad** [PK/NL]; c/o High Tech Campus Building 44, NL-5656 AE Eindhoven (NL). **IBRAIMI, Luan** [MK/NL]; c/o High Tech Campus Building 44, NL-5656 AE Eindhoven (NL). **PETKOVIC, Milan** [NL/NL]; c/o High Tech Campus Building 44, NL-5656 AE Eindhoven (NL).

(74) Agents: **VAN VELZEN, Maaike, M.** et al.; High Tech Campus 44, NL-5656 AE Eindhoven (NL).

(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,

[Continued on next page]

(54) Title: CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION AND RE-ENCRYPTION

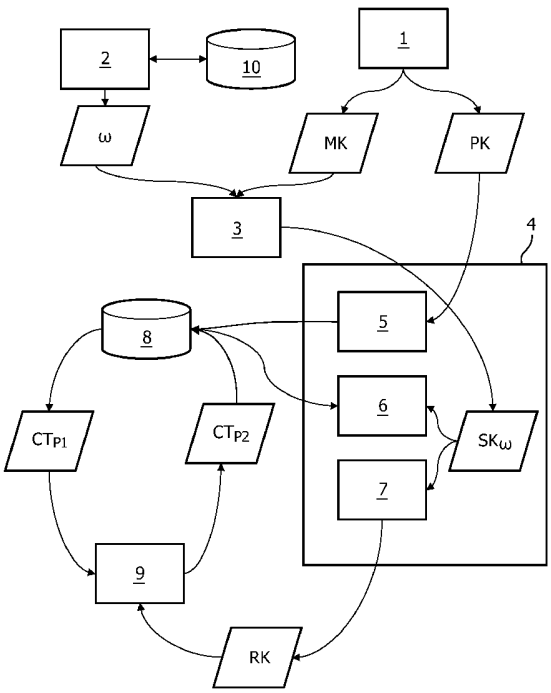

FIG. 1

(57) Abstract: A ciphertext-policy attribute-based encryption system, comprising a re-encrypter (9) for cryptographically transforming a first ciphertext (CT_{P1}) associated with a first access policy (P1) into a second ciphertext (CT_{P2}) associated with a second access policy (P2) by means of a re-encryption key (RK). The system further comprises a re-encryption key generator (7) for generating the re-encryption key (RK), wherein the re-encryption key (RK) enables the re-encrypter (9) to cryptographically transform the first ciphertext (CT_{P1}) associated with the first access policy (P1) into the second ciphertext (CT_{P2}) associated with the second access policy (P2). Said re-encryption key generator (7) comprises a subsystem for encrypting a value derived from a pseudorandom number, thereby generating a further ciphertext associated with the second access policy (P2).

WO 2011/045723 A1

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, ML, MR, NE, SN, TD, TG).

Published:

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

Ciphertext-policy attribute-based encryption and re-encryption

FIELD OF THE INVENTION

The invention relates to ciphertext-policy attribute-based encryption. The invention also relates to re-encrypting encrypted data.

BACKGROUND OF THE INVENTION

A proxy re-encryption system allows a semi-trusted proxy to transform a ciphertext computed under, for example, Alice's public key into a ciphertext that can be decrypted by using, for example, Bob's secret key. This system may work as follows: Alice or a trusted third party generates a re-encryption key and sets it in a semi-trusted proxy. On receiving Alice's ciphertexts, the semi-trusted proxy transforms the ciphertext by running the re-encryption algorithm with the re-encryption key, and sends the transformed ciphertext to Bob. Bob decrypts it with his secret key. In this way, Alice delegates her decryption rights to Bob via the semi-trusted proxy, so Alice may be called a delegator and Bob may be called a delegatee. The proxy re-encryption system may be arranged to satisfy the following criteria: 1) a semi-trusted proxy alone cannot obtain the underlying plaintext, 2) Bob cannot obtain the underlying plaintext without the semi-trusted proxy cooperating, 3) ideally, the collusion of Bob and the semi-trusted proxy does not enable the semi-trusted proxy to construct Alice's secret key.

In J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption", Proceedings of the 2007 IEEE Symposium on Security and Privacy, pages 321-334, 2007, a message is encrypted according to an access policy "P" over some descriptive attributes, while a user secret key sk_ω is associated with a set of attributes ω. The decrypter can decrypt the ciphertext if the set of attributes ω associated with the secret key sk_ω satisfies the access policy "P" associated with the ciphertext. In many situations, when a user encrypts the data, it is desirable that the user is able to establish a specific access control policy on who can decrypt this data. Traditionally, this type of expressive access control is enforced by employing a trusted server. The server is entrusted as a reference monitor that checks that a user presents proper certification before allowing him to access records or files. However, in some situations, the server might not be completely trusted. For this reason,

sensitive data may be stored in an encrypted form so that it remains private even if a server is compromised. Ciphertext-Policy Attribute-Based Encryption (CP-ABE) schemes provide a solution by encrypting the data before storing it on an untrusted server according to an access policy "P" which enforces the access control cryptographically.

For instance, Alice can upload her sensitive health data to an un-trusted server. Before uploading, the data may be encrypted according to the access policy P = (physician OR nurse), where "physician" and "nurse" are the attributes of the users who are allowed to view Alice's data. This means that only users who have at least one of the attributes "physician" or "nurse" are able to decrypt and view Alice's data with their secret key. If Alice's physician wants to view Alice's data, he downloads the data to his PC and decrypts the data using his secret key sk_{ω} associated with the attribute "physician". However, more flexible CP-ABE schemes could be desirable.

SUMMARY OF THE INVENTION

It would be advantageous to have an improved ciphertext-policy attribute-based encryption system (CP-ABE). To better address this concern, a first aspect of the invention provides a system comprising a re-encrypter for cryptographically transforming a first ciphertext associated with a first access policy into a second ciphertext associated with a second access policy by means of a re-encryption key.

In some cases, a delegator who has access to data according to the first access policy would desire to enable a delegatee to view the data. For example, the delegator could desire to delegate his task of evaluating the data to the delegatee. In such a case the delegatee needs access to the data. However, the attributes of the delegatee may not conform to the access policy with which the data was originally encrypted. Consequently, the decryption key, also referred to as secret key, of the delegatee does not allow the delegatee to decrypt the data. The re-encrypter allows changing the access policy by re-encrypting the data. Since the re-encryption is governed by a re-encryption key, it is not necessary to first decrypt the data before encrypting it with the second access policy. This way, the re-encrypter can generate the second ciphertext associated with the second access policy. However, as the re-encrypter does not have the decryption key, it cannot decrypt either the first or the second ciphertext. Consequently, the re-encrypter cannot gain access to the plaintext. This allows the re-encrypter to be implemented on a semi-trusted server or proxy. Consequently, the re-encryption does not have to be performed within the trusted environment of the delegator.

A ciphertext associated with an access policy may be decrypted by means of a decryption key associated with an attribute set satisfying that access policy. In principle, the ciphertext may only be decrypted by means of a decryption key associated with an attribute set satisfying that access policy, although it is possible to have exceptions, such as a master decryption key which can decrypt independent of policy and/or attributes. However, any decryption key associated with an attribute set satisfying the access policy may be used to decrypt the message. An attribute set is a set of one or more attributes. The first ciphertext associated with the first access policy may be decrypted by means of a decryption key associated with an attribute set satisfying the first access policy, whereas the second ciphertext associated with the second access policy may be decrypted by means of a decryption key associated with an attribute set satisfying the second access policy.

The system may comprise a re-encryption key generator for generating the re-encryption key, wherein the re-encryption key enables the re-encrypter to cryptographically transform the first ciphertext associated with the first access policy into the second ciphertext associated with the second access policy. The re-encryption key generator may be arranged to use the secret key of the delegator to generate the re-encryption key. This way, only the re-encryption key needs to be generated within the trusted environment of the delegator, whereas the potentially more computationally intensive task of cryptographically transforming the first ciphertext can be performed in a semi-trusted environment. The re-encryption key may be associated with the first access policy via an attribute set which satisfies the first access policy. In such a case, the re-encryption key can be used to re-encrypt any ciphertext whose access policy is satisfied by that attribute set.

The re-encryption key generator may comprise a subsystem for encrypting a value derived from a pseudorandom number, thereby generating a further ciphertext associated with the second access policy. The re-encryption key generator may be arranged for including a representation of the further ciphertext in the re-encryption key. Using the further ciphertext, a secret (the pseudorandom number) can be communicated to a decrypter having a decryption key associated with a proper attribute set. This secret can be used as at least part of a key to decrypt a message.

The re-encrypter may be arranged for including in the second ciphertext a representation of the further ciphertext. This is a convenient way to convey the further ciphertext to the decrypter.

The re-encryption key generator may be arranged for including in the

re-encryption key an at least partly obfuscated representation of part of a decryption key associated with an attribute set satisfying the first access policy. This can be used to create an efficient encryption scheme.

The re-encrypter may be arranged for bilinear pairing of at least part of the re-encryption key and at least part of the first ciphertext. This helps to create an efficient encryption scheme.

The system may comprise a decrypter for decrypting the second ciphertext by means of a decryption key associated with an attribute set satisfying the second access policy. The decrypter performs the actual decryption of the transformed ciphertext.

The decrypter may comprise:

- a subsystem for extracting the further ciphertext from the second ciphertext;
- a subsystem for decrypting the further ciphertext by means of the decryption key to obtain the value; and/or
- a subsystem for decrypting the message stored in the second ciphertext based on the value.

This helps to make the system efficient and/or secure.

The system may further comprise:

- a key generator for receiving a set of at least one attribute and outputting a decryption key associated with an attribute set comprising at least one attribute; and/or
- an encrypter for generating the ciphertext associated with the first access policy, wherein the ciphertext comprises an encryption of a message and the ciphertext can be decrypted by means of a decryption key associated with an attribute set satisfying the first access policy.

These system parts are useful for key generation and encryption, respectively.

Another aspect of the invention provides a re-encryption key generator for use in the system set forth. The re-encryption key generator may be arranged for generating a re-encryption key (RK), wherein the re-encryption key (RK) enables a re-encrypter (9) to cryptographically transform a first ciphertext (CT_{P1}) associated with a first access policy (P1) into a second ciphertext (CT_{P2}) associated with a second access policy (P2).

The system may be implemented in one or more workstations. At least one of these workstations may be a medical workstation.

A method of ciphertext-policy attribute-based re-encryption comprises cryptographically transforming a first ciphertext associated with a first access policy into a second ciphertext associated with a second access policy by means of a re-encryption key.

The method may be implemented in form of a computer program product comprising instructions for causing a processor system to perform the method.

It will be appreciated by those skilled in the art that two or more of the above-mentioned embodiments, implementations, and/or aspects of the invention may be combined in any way deemed useful.

Modifications and variations of the image acquisition apparatus, the workstation, the system, and/or the computer program product, which correspond to the described modifications and variations of the system, can be carried out by a person skilled in the art on the basis of the present description.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other aspects of the invention are apparent from and will be elucidated with reference to the embodiments described hereinafter. In the drawings,

Fig. 1 is a block diagram of an encryption system;

Fig. 2 is a flow chart of an encryption method; and

Fig. 3 illustrates schematically an application of an encryption system.

DETAILED DESCRIPTION OF EMBODIMENTS

In this description, an example of a ciphertext-policy attribute-based proxy re-encryption (CP-ABEPRE) is described. However, modifications and alternative embodiments of the example given are within reach of the skilled person. In the exemplary system, a semi-trusted proxy can translate an original ciphertext associated with an access policy "P1" to a new ciphertext associated with an access policy "P2", without being able to access the plain data. The new ciphertext can be decrypted only by users who possess a secret key associated with a set of attributes which satisfy the associated policy "P2". CP-ABEPRE may be useful in delegation scenarios or in scenarios where the owner of the data wishes to change the access control policy. The exemplary system described herein has the advantage that even the collusion of the semi-trusted proxy and the delegatee cannot construct the secret key of the delegator. So, even if the proxy and the delegatee share their secret information, neither the proxy nor the delegatee can find out the secret key of the delegator. However, this is not a limitation. The system can be used in a number of applications such as for access control over the network storage (e.g. personal health records), secure e-mail forwarding. Other applications of the system are also possible.

From the description of the CP-ABE, it may be seen that these schemes provide advantages in certain domains where attribute-based access control is used. An example of such a domain is healthcare. However, in practice, there are scenarios where a user may want to delegate or allow access to sensitive data by another user with a different set of attributes (e.g. fitness coach, his/her subordinate, second opinion doctor), which other user is not allowed to view the data according to the original policy "P1". For example, in a scenario from the domain of healthcare, the data owner (patient) may want to allow access for a second opinion to another doctor (Dr. Bob) from a second hospital. To enable such consultation, the patient may have to change his/her consent policy to another policy "P2". The patient data may be stored at an untrusted server, for example a third party digital HealthVault provider, and encrypted according to the policy "P1". Using a CP-ABPRE scheme, a patient who wants to enable access to data for Dr. Bob, who is allowed to view the encrypted data according to the policy "P2", can compute a re-encryption key (Proxy Key) rk(P1-P2) and send the key to the proxy which is maintained by the untrusted server. The proxy, using rk(P1-P2), can transform all ciphertexts encrypted under the access policy "P1" to a ciphertext encrypted under the access policy "P2" without having access to the plain data. After that, Dr. Bob can use his key SK_{p2} to decrypt the data. Consequently, Dr. Bob can view the data and give a second opinion to the Patient, who can then ask his/her main physician for an additional examination.

Fig. 1 illustrates some aspects of an encryption system including a functionality of re-encryption. The Figure only shows an example system. Other architectures and/or modifications are also contemplated. Some of the functional blocks of the example system may be implemented on separate devices which are used by different users of the system. It is also possible to implement the whole system on a single computer. Conversely, it is also possible to distribute the functionality of a single block over a plurality of devices.

The system may comprise several databases, for example a user database 10. In such a user database, users may be listed by means of user IDs or demographic information, for example. Moreover, the database 10 may store additional information for some or all of the users. For example, a user may be associated with a set of attributes ω. Such attributes may represent groups or categories to which the user belongs, or special privileges the user enjoys.

The system may further comprise a database 8 for storing encrypted data. These data may be encrypted according to an access policy. The encrypted data, or ciphertext, is then associated with that access policy. Various chunks of encrypted data, or

ciphertexts, may be associated with different access policies. Consequently, it is possible to specify in detail which users have access to which data, by encrypting the data accordingly. To decrypt a ciphertext which is associated with an access policy, the user needs to have a secret key SK_ω associated with a set of attributes ω which are acceptable for the access policy. The access policy prescribes which combination(s) of attributes are needed for decryption. To this end, the data is encrypted differently depending on the access policy.

The system further comprises one or more user environments 4. In the Figure, only one user environment 4 is depicted, however, in practice there may be more user environments. The user environment 4 may comprise secret keys and/or provide an environment for processing sensitive data.

The system may comprise a re-encrypter 9 for cryptographically transforming a first ciphertext CT_{P1} associated with a first access policy P1 into a second ciphertext CT_{P2} associated with a second access policy P2 by means of a re-encryption key RK. The re-encryption key RK may be provided from the user environment 4 to the re-encrypter 9. Moreover, the user environment 4 may send a control signal to the re-encrypter 9, indicating which ciphertext CT_{P1} from the database 8 should be re-encrypted. The re-encrypted, second ciphertext CT_{P2} may be stored in the database 8 for retrieval by any other user environments 4 which possess a secret key SK_ω associated with a set of attributes ω satisfying the new access policy P2. Optionally, the re-encrypter 9 and/or the user environment 4 may be arranged for deleting the first ciphertext CT_{P1} from the database 8. A user interface may be provided for enabling a user to select whether or not to delete the first ciphertext. However, it is possible that the second access policy P2 also allows access by all sets of attributes that were allowed access by the first access policy P1. In such a case it would be superfluous to keep the first ciphertext CT_{P1}. Also, to prevent any future access by users having a set of attributes satisfying the first access policy P1 but not the second access policy P2, the first ciphertext may be deleted from the database 8 after re-encryption.

The system may comprise a re-encryption key generator 7 which may be implemented within the user environment 4. Alternatively, the re-encryption key generator 7 may be implemented in a trusted server. The re-encryption generator 7 is arranged for generating a re-encryption key RK. This re-encryption key RK contains the information which is necessary to cryptographically transform, or re-encrypt, the ciphertext. This way, the access policy associated with a ciphertext may be changed. However, the re-encryption key RK may not comprise sufficient information to enable the re-encrypter 9 or a third party to decrypt the ciphertext into its plaintext data. The re-encryption key RK may be provided to a

re-encrypter 9, which may use the re-encryption key RK to cryptographically transform a first ciphertext CT_{P1} associated with the first access policy P1 into a second ciphertext CT_{P2} associated with the second access policy P2. The re-encryption key RK may have a given set of attributes and a given access policy associated therewith and may provide sufficient information to cryptographically transform any ciphertext associated with any access policy satisfied by this given set of attributes into a ciphertext associated with this given access policy.

The re-encryption key generator 7 may comprise a subsystem for encrypting a value derived from a pseudorandom number. The encrypted value constitutes a further ciphertext, which is associated with the second access policy P2. For example, a pseudorandom number generator is provided; the pseudorandom number, or a value derived therefrom, may be encrypted under control of the re-encrypter 7 using encrypter 5. The re-encryption key generator 7 may be arranged for including in the re-encryption key RK a representation of this further ciphertext. It is noted that the further ciphertext can only be decrypted using a secret key associated with an access policy satisfying the second access policy P2. Consequently, the re-encrypter 9 may not be able to decrypt the further ciphertext and hence may not know the pseudorandom number. The re-encrypter 7 may be arranged for including in the second ciphertext CT_{P2} a representation of the further ciphertext. Consequently, the users having a set of attributes satisfying the second access policy P2 are able to know the value or pseudorandom number.

The re-encryption key generator 7 may be arranged for including in the re-encryption key an at least partly obfuscated representation of part of a decryption key associated with an attribute set satisfying the first access policy. This part of the decryption key may be obfuscated by modifying it in dependence on the pseudorandom number.

The re-encrypter 9 may be arranged for bilinear pairing of at least part of the re-encryption key RK and at least part of the first ciphertext CP_{P1}.

The system may comprise a decrypter 6. The decrypter 6 may be arranged for decrypting a ciphertext from the database 8. The decrypter 6 may use a secret key SK_ω associated with a set of attributes ω to decrypt a ciphertext CT_P associated with an access policy P. Such a decryption may only work if the set of attributes ω satisfies the access policy P associated with the ciphertext CT_P. For example, if the set of attributes ω satisfies the second access policy P2, the decrypter 6 may be able to decrypt the ciphertext CT_{P2} which is the result of re-encryption by the re-encrypter 9.

The decrypter 6 may comprise several subsystems. For example, it may comprise a subsystem for extracting the further ciphertext from the second ciphertext CT_{P2}; a subsystem for decrypting the further ciphertext by means of the decryption key SK_ω to obtain the value derived from the pseudorandom number; a subsystem for decrypting the plaintext stored in the second ciphertext based on the value.

The system may comprise a key generator 3 for receiving a set ω of at least one attribute and outputting a decryption key or secret key SK_ω associated with an attribute set ω comprising at least one attribute. This secret key SK_ω may be provided to a user environment 4 for use by a decrypter 6 and/or a re-encryption key generator 7.

The system may further comprise a user manager 2 connected to the user database 10. The user manager 2 may be used to enter new users into the system and/or give a set of attributes to a user. The user manager 2 may be arranged for providing a set of attributes ω to the key generator 3 to produce an associated secret key SK_ω. The user manager 2 may comprise a user interface to enable a privileged user to operate the user manager.

The system may further comprise an encrypter 5. This encrypter 5 may be part of a user environment 4, although this is not necessary. In particular, it may not be necessary to have a secret key SK_ω to perform encryption operations, as these may be performed using a public key PK. The encrypter 5 may be used for generating a ciphertext CT_P associated with an access policy P. The resulting ciphertext CT_P may comprise an encryption of a message and can be decrypted by means of a decryption key SK_ω associated with an attribute set ω satisfying the access policy P. A ciphertext CT_{P1} thus generated and associated with a first access policy P1 can be changed into a second ciphertext CT_{P2} associated with a second access policy P2, by means of the re-encrypter 9 and re-encryption key generator 7.

At least part of the system described may be implemented on a computer workstation, for example a medical workstation. This may be implemented by means of a computer program.

Fig. 2 shows a flow chart illustrating a method of ciphertext-policy attribute-based data re-encryption. To perform the method, suitable components of the system illustrated in Fig. 1 may be used. In step 21, data is encrypted according to a first access policy. This step may result in a first ciphertext associated with the first access policy. In step 22, it is considered if the access policy needs to be changed. If so, in step 27, the first ciphertext is cryptographically transformed into a second ciphertext associated with a second access policy. This is done by means of a re-encryption key which may be provided by a user. After re-encryption, the method returns to step 22 to consider if the access policy needs

to be changed again. If the access policy does not need to be changed in step 22, the method proceeds to step 23. In step 23 it is considered if the ciphertext needs to be decrypted. If so, in step 24 it is checked whether a secret key associated with a set of attributes satisfying the access policy of the ciphertext is available. This access policy associated with the ciphertext can be the first access policy or the second access policy, for example. If the necessary secret key is available, the secret key is used to decrypt the ciphertext in step 25. After that the process terminates in step 26. However, the process can also return to step 22 for example, to enable other users to decrypt the data or to change the access policy (again). If the ciphertext does not need to be decrypted in step 23, the method may return to step 22. If the needed secret key is not available in step 24, an error signal is produced and the process may terminate in step 28 or return to step 22. The method or parts thereof may be implemented as one or more computer programs.

A CP-ABE scheme may comprise four main algorithms which may be executed by different actors in the system. An example system has been described with reference to Fig. 1 and Fig. 2. In particular, the algorithms Setup, KeyGen, Encrypt, and Decrypt may be distinguished, wherein KeyGen stands for key generation. The CP-ABPRE scheme may extend CP-ABE schemes by adding a proxy component to the existing actors of CP-ABE (which include a trusted authority (TA) and users) and the algorithms RKGen and Re-Encrypt, wherein RKGen stands for re-encryption key generation.

-Setup(): run by the trusted authority (TA), the algorithm on input of a security parameter, outputs a master secret key "MK" which may be kept private, and the master public key "PK" which may be distributed to users.

-KeyGen (ω, MK): run by the trusted authority (TA), the algorithm may take as input a set of attributes ω which represent properties of a user, and the master secret key MK, and it may output a user secret key sk_ω associated with the set of attributes ω. Such a user secret key sk_ω may be used later on for decrypting ciphertexts which have an access policy which is satisfied by the set of attributes ω.

-Encrypt (m, P1, PK): run by the encrypter, the algorithm may take as input a message "m" to be encrypted, an access policy P1, and the master public key "PK". The access policy P1 prescribes which combination of attributes the decrypter needs to have in order to be allowed access to "m". The algorithm may output the ciphertext "c_{p1}".

-RKGen (sk_ω, P1, P2): run by the delegator, this algorithm may take as input the secret key sk_ω and the access policies (P1,P2) and may output a unidirectional

re-encryption key rk(P1-P2) if sk_ω satisfies P1, or an error symbol $\perp$ (or, alternatively, an unusable re-encryption key) if ω does not satisfy P1.

-Re-Encrypt (cp, rk(P1-P2)): run by the proxy (or re-encrypter), this algorithm may take as input the ciphertext "c_{p1}" associated with the access policy P1, and the re-encryption key rk(P1-P2), and may output the ciphertext "c_{p2}" associated with the access policy P2.

-Decrypt (c_{pi}, sk_ω): run by the decrypter, the algorithm may take as input the ciphertext c_{pi} and the secret key sk_ω , and may output a message m if the set of attributes ω satisfies the policy P_i, or an error symbol if ω does not satisfy P_i. Herein, i may be 1 or 2.

For example, RKGen may comprise the step of selection of random values l, x' from Z_p such that $fx'=x$. RKGen may comprise the step of generating a random value, i.e. g^l. RKGen may comprise the step of modifying the secret key (of delegator) associated with the attribute set that satisfies the first access policy by multiplying it with g^l. RKGen may comprise the step of re-arranging the secret key (of delegator) associated with the attribute set that satisfies the first access policy for inclusion in the re-encryption key. RKGen may comprise generating a random component for inclusion in the re-encryption key. RKGen may comprise deriving a pseudorandom number that is encrypted from a value generated during the setup phase based on a random number "f" from Z_p, wherein this random number "f" is part of a Master Secret Key MK.

For example, Re-Encrypt may comprise the step of bilinear pairing of $\hat{D}^{(4)}$ from the re-encryption key and $C^{(4)}$ from the first ciphertext to generate $I^{(1)}$. Re-Encrypt may comprise the step of bilinear pairing of $\hat{D}^{(1)}$ from the re-encryption key and $C^{(1)}$ from the first ciphertext and multiplication of the resultant value with $I^{(1)}$ to generate $I^{(2)}$. Re-Encrypt may comprise the step of division of $C^{(2)}$ from the first ciphertext by the output $I^{(2)}$ to generate $I^{(3)}$. Re-Encrypt may comprise the step of bilinear pairing of $\hat{D}^{(3)}$ from the re-encryption key and $C^{(3)}$ of the first ciphertext and multiplication of the resultant value with output $I^{(3)}$ to generate $\hat{C}^{(2)}$. Re-Encrypt may comprise the step of rearrangement of the values for the output as second ciphertext, i.e. $\left(\hat{C}^{(1)}, \hat{C}^{(2)}, \hat{C}^{(3)}\right)$. The symbols used in this paragraph are explained hereinafter.

As mentioned before, in practice there are scenarios where a user would like to delegate his/her access rights to data to another user or may want to enable access for the users with some different set of attributes. For example, patient (Alice) may want to allow another user (e.g. Dr. Bob) who has a secret key sk_ω associated with attribute set ω=(c, d) to

access her files encrypted according to a policy, say P1=(a AND b), where "a" and "b" are the attributes necessary to decrypt the message, and "c" and "d" are the attributes of the prospective user, Dr. Bob in this case. Therefore, Alice has to update P1=(a AND b), where "a" and "b" are the attributes of the user, to another access policy say P2= (a AND b) OR (c AND d), which can be satisfied by Dr. Bob in this case. A straightforward approach to achieve this would be that Alice sends to the access control server her secret key sk_{Alice} which satisfies the access policy "P1". The access control server decrypts the encrypted data using sk_{Alice} and re-encrypts the data according to the new policy P2. After that, Dr. Bob would be able to decrypt the data using his secret key sk_{ω} associated with attribute set ω=(c, d).

However, a drawback of this approach is that the server may gain access to the plain data and to the secret key of Alice. Consequently, the server should be a trusted entity. In practice, the server might not be trusted. To avoid this drawback, Alice could perform by herself the re-encryption process by downloading the ciphertexts, decrypting the ciphertexts using her keys that correspond to P1 and re-encrypting the data according to P2. However, the main disadvantage of this approach is that Alice has to be involved in each re-encryption. In both of these approaches, the process is also computationally intensive as the data is first decrypted and then encrypted again.

Using a re-encryption key rk(P1-P2), it is possible to enable a proxy to transform, using that key, ciphertext associated with the access policy P1 into ciphertext associated with access policy P2, without having access to the plain data.

A ciphertext-policy attribute-based proxy re-encryption scheme may support efficient outsourced policy updates. It allows a proxy maintained by an untrusted server (or untrusted system) to transform a ciphertext associated with an access policy "P1" into a ciphertext associated with an access policy "P2". In this transformation process, the untrusted server (or system or proxy) does not get access to the plain data.

A ciphertext-policy attribute-based proxy re-encryption scheme may be useful for the dynamic environments where a person wants to delegate the access rights to a second person (delegate) related to a data encrypted according to the access policy P1, where the delegate is only allowed to view the data encrypted according to an access policy P2.

A ciphertext-policy attribute-based proxy re-encryption scheme may be useful for the dynamic environments where the access policy is changed frequently, e.g. in the healthcare domain, a patient may want to enable access for another doctor (e.g. Dr. Bob), or another category of healthcare professionals, in order to get a second opinion.

A ciphertext-policy attribute-based proxy re-encryption scheme may support multiuser decryption and multiuser delegation.

Fig. 3 shows an example of an architecture of an encryption system. In the system, the data owner 30 (patient) encrypts the health data according to the access policy P1, say P1= (GP AND Hospital 1) OR (Owner of data (Patient)), and uploads the encrypted data "c_{P1}" to an untrusted storage server 31, as indicated by arrow 41. For example, the Internet may be used as a means of communication. The general practitioner (GP) 34 from the Hospital 1 downloads the encrypted data from the un-trusted storage server 31, as indicated by arrow 42, and decrypts them locally. Note that besides the owner of the data (i.e., the Patient himself), only users who have the attributes GP and Hospital 1 can decrypt the ciphertext. The patient may want a second opinion from a GP 35 from Hospital 2, who is not allowed to view the data encrypted according to access policy P1= [(GP AND Hospital 1) OR (Owner of data (Patient))]. Consequently, in order to enable access to patient's data for GP 35 from Hospital 2 for the purpose of second opinion, the data owner 30 may compute the re-encryption key rk(P1-P2) that may be used by the semi-trusted proxy 33 to transform the ciphertext encrypted according to the policy P1= [GP AND Hospital 1] into a ciphertext encrypted according to the policy P2= [GP AND (Hospital 1 OR Hospital 2) OR (Owner of data (Patient))]. The re-encryption key rk(P1-P2) may be sent to a re-encryption key storage server 32, as indicated by arrow 43. The proxy 33, upon receiving the re-encryption key from the patient and original ciphertext associated with P1 from the server, may re-encrypt the ciphertext associated with "P1" into a ciphertext associated with "P2" using the re-encryption key rk(P1-P2). This is depicted with arrows 44 and 45. Note that in practice the semi-trusted proxy 33 could also be integrated in the re-encryption key storage server 32. After the re-encryption, the GP 35 from Hospital 2 can decrypt the data using his/her secret key, as depicted by arrow 46.

In the following, an example CP-ABPRE scheme is described. This example scheme comprises a number of algorithms which may be implemented on computer servers, for example using a computer program which implements the algorithm. Some of these algorithms may be omitted or implemented only partly, as appropriate. Moreover, different algorithms may be arranged to be executed on different computer devices. It is also possible to distribute the operations involved in a single algorithm over a plurality of devices and/or processors. The algorithms described below include a setup algorithm, a key generation algorithm, an encryption algorithm, a re-encryption key generator (RKGen), a re-encryption

algorithm, and a decryption algorithm. Modifications of these algorithms are possible, the specific examples described below are not limiting.

- Setup. The setup algorithm selects a bilinear group G_0 of prime order p and generator g, and a bilinear map $\hat{e}: G_0 \times G_0 \rightarrow G_1$. Next to this, the setup generates the list of attributes in the system $\Omega = \{a_1, a_2, ..., a_k\}$, picks randomly $\alpha, \beta, f, x_1, x_2, \cdots, x_k \in Z_p$, and sets $T_j = g^{x_j}$ $(1 \leq j \leq k)$. The public key is published as:

$$PK = (\hat{e}(g,g)^{(\alpha+\beta)}, g^f, \{T_j\}_{j=1}^k).$$

The master secret key consists of the following components:

$$MK = (\alpha, \beta, f, \{x_j\}_{j=1}^k).$$

- KeyGeneration(MK,ω). The key generation algorithm takes as input the master secret key MK and an attribute set ω, wherein $\omega \subseteq \Omega$. For each user the algorithm picks at random $r \in Z_p$ and computes a secret key SK_ω which comprises the following components:

$$SK_\omega = (D^{(1)} = g^{\alpha - r},$$

$$\{D_j^{(2)} = g^{\frac{r+\beta}{x_j}}\}_{a_j \in \omega}).$$

- Encryption(m, p_1, PK). To encrypt a message $m \in G_1$, under the access policy p_1 over the set of available attributes Ω, the encryption algorithm picks a random value $s \in Z_p$, and assigns s_i values (which are shares of s) to attributes in p_1 in the following fashion:

1. The encrypter transforms the access policy into an access tree where the interior nodes represent AND or OR boolean operators, and leaf nodes represent the actual attributes appearing in the policy. Note that the policy may have the form of an expression including AND and/or OR operators, to indicate valid combinations of attributes which are sufficient to be allowed access.
2. It assigns the value s to the root node.
3. Use, for example, the Benaloh and Leichter (Josh Benaloh and Jerry Leichter, "Generalized Secret Sharing and Monotone Functions, Advances in Cryptology - CRYPT0 '88, LNCS 403, pp. 27-35, 1990) secret sharing scheme to assign values S_i to leaf nodes (attributes) in the following fashion. Recursively, for each un-assigned non-leaf node, it does the following:

a) If the node is AND, it assigns a share to each child node, such that the sum of all shares is s. Mark this node as assigned.

b) If the node is OR, it assigns the same value s to each child. Mark this node as assigned.

The resulting ciphertext may comprise the following components:

$$\mathrm{CT}_{p_1} = (C^{(1)} = g^s$$
$$C^{(2)} = m \cdot \hat{e}(g,g)^{(\alpha+\beta)s}, C^{(3)} = g^{fs},$$
$$\{C_{j,i}^{(4)} = g^{x_j S_i}\}_{a_{j,i} \in p_1}).$$

- $\mathrm{RKGen}(\mathrm{SK}_\omega, p_1, p_2, \mathrm{PK})$: The algorithm outputs a re-encryption key which is used by the re-encryption algorithm to transform the ciphertext associated with p_1 into a ciphertext associated with p_2. The algorithm first parses SK_ω as $(\mathrm{D}^{(1)}, \{\mathrm{D}_j^{(2)}\}_{a_j \in \omega})$, picks at random $l, x' \in Z_p$, sets $fx' = x$ and computes the re-encryption key $\mathrm{RK}_{p_1 \to p_2}$ which may comprise the following components:

$$\mathrm{RK}_{p_1 \to p_2} = (\hat{D}^{(1)} = D^{(1)} \cdot g^l,$$
$$\hat{D}^{(2)} = Encryption(g^{x-l}, p_2, PK),$$
$$\hat{D}^{(3)} = g^{x'} = g^{\frac{x}{f}},$$
$$\hat{D}_j^{(4)} = \{D_j^{(2)}\}_{a_j \in \omega}.$$

- $\mathrm{Re-Encrypt}(\mathrm{CT}_{p_1}, \mathrm{RK}_{p_1 \to p_2})$. The algorithm parses CT_{p_1} as $(\mathrm{C}^{(1)}, \mathrm{C}^{(2)}, \mathrm{C}^{(3)}, \{\mathrm{C}_{j,i}^{(4)}\}_{a_{j,i} \in p_1})$, and $\mathrm{RK}_{p_1 \to p_2}$ as $(\hat{\mathrm{D}}^{(1)}, \hat{\mathrm{D}}^{(2)}, \hat{\mathrm{D}}^{(3)}, \{\hat{\mathrm{D}}_j^{(4)}\}_{a_j \in \omega})$, and computes the following:

- In a first step, for every attribute $a_j \in \omega$, it computes the following:

$$I^{(1)} = \prod_{a_j \in \omega} \hat{e}(\hat{D}_j^{(4)}, C_{j,i}^{(4)}) = \prod_{a_j \in \omega} \hat{e}(g^{\frac{r+\beta}{x_j}}, g^{x_j S_i}) = \hat{e}(g^{r+\beta}, g^s).$$

- In a second step, it computes the following:

$$I^{(2)} = \hat{e}(C^{(1)}, \hat{D}^{(1)}) \cdot I^{(1)} = \hat{e}(g^s, g^{\alpha - r} \cdot g^l) \cdot \hat{e}(g,g)^{(r+\beta)s} = \hat{e}(g^s, g^{\alpha+\beta} \cdot g^l).$$

- In a third step, it computes the following:

$$I^{(3)} = \frac{C^{(2)}}{I^{(2)}} = \frac{m \cdot \hat{e}(g^s, g^{\alpha+\beta})}{\hat{e}(g^s, g^{\alpha+\beta} \cdot g^l)} = \frac{m}{\hat{e}(g^s, g^l)}, \text{ and}$$

$$\hat{C}^{(2)} = \hat{e}(C^{(3)}, \hat{D}^{(3)}) \cdot I^{(3)} = \hat{e}(g^{sf}, g^{\frac{x}{f}}) \cdot \frac{m}{\hat{e}(g^s, g^l)} = m \cdot \hat{e}(g^s, g^{x-l}).$$

- In a fourth step, it computes the following:

$\hat{C}^{(1)} = C^{(1)}$.

$\hat{C}^{(3)} = \hat{D}^{(2)}$.

The algorithm outputs the re-encrypted ciphertext, which may comprise the following components:

$$\mathrm{CT}_{p_2} = (\hat{C}^{(1)}, \hat{C}^{(2)}, \hat{C}^{(3)}).$$

- Decryption$(C_{p_i}, \mathrm{SK}_\omega)$: The decryption algorithm takes as input the ciphertext C_{p_i} and decryption key SK_ω. It checks if the secret key SK_ω associated with the attribute set ω satisfies the access policy p_i. If not, it may output an error symbol $\perp$, or unusable output. If ω satisfies the access policy p_i and C_{p_i} is a regular (not re-encrypted) ciphertext, then the decryption algorithm performs the following:

- In a first step, the algorithm chooses the smallest subset $\omega' \subseteq \omega$ which satisfies the access policy p_i and parses C_{p_i} as $(C^{(1)}, C^{(2)}, \{C^{(4)}_{j,i}\}_{a_j \in p_i})$, and SK_ω as $(D^{(1)}, \{\mathrm{D}^{(2)}_j\}_{a_j \in \omega})$.

- In a second step, for every attribute $a_j \in \omega'$, it computes

$$Z^{(1)} = \prod_{a_j \in \omega'} \hat{e}(D^{(2)}_j, C^{(4)}_{j,i}) = \prod_{a_j \in \omega'} \hat{e}(g^{\frac{r+\beta}{x_j}}, g^{x_j S_i}) = \hat{e}(g^{r+\beta}, g^s).$$

- In a third step, it computes

$$Z^{(2)} = \hat{e}(D^{(1)}, C^{(1)}) \cdot Z^{(1)} = \hat{e}(g^{\alpha - r}, g^s) \cdot \hat{e}(g^{r+\beta}, g^s) = \hat{e}(g, g)^{(\alpha+\beta)s}.$$

- In a fourth step, the message is obtained by computing

$$m = \frac{C^{(2)}}{Z^{(2)}}.$$

If ω satisfies the access policy p_i and C_{p_i} is a re-encrypted ciphertext, then the decryption algorithm performs the following:

* In a first step, it parses C_{p_i} as $(\hat{C}^{(1)}, \hat{C}^{(2)}, \hat{C}^{(3)})$

* In a second step, it recovers the message in the following way:

$$m = \frac{\hat{C}^{(2)}}{\hat{e}(\hat{C}^{(1)}, \mathrm{Decrypt}(\hat{C}^{(3)}, \mathrm{SK}_{\omega}))}.$$

It will be appreciated that the invention also applies to computer programs, particularly computer programs on or in a carrier, adapted to put the invention into practice. The program may be in the form of a source code, an object code, a code intermediate source and object code such as in a partially compiled form, or in any other form suitable for use in the implementation of the method according to the invention. It will also be appreciated that such a program may have many different architectural designs. For example, a program code implementing the functionality of the method or system according to the invention may be sub-divided into one or more sub-routines. Many different ways of distributing the functionality among these sub-routines will be apparent to the skilled person. The sub-routines may be stored together in one executable file to form a self-contained program. Such an executable file may comprise computer-executable instructions, for example, processor instructions and/or interpreter instructions (e.g. Java interpreter instructions). Alternatively, one or more or all of the sub-routines may be stored in at least one external library file and linked with a main program either statically or dynamically, e.g. at run-time. The main program contains at least one call to at least one of the sub-routines. The sub-routines may also comprise function calls to each other. An embodiment relating to a computer program product comprises computer-executable instructions corresponding to each processing step of at least one of the methods set forth herein. These instructions may be sub-divided into sub-routines and/or stored in one or more files that may be linked statically or dynamically. Another embodiment relating to a computer program product comprises computer-executable instructions corresponding to each means of at least one of the systems and/or products set forth herein. These instructions may be sub-divided into sub-routines and/or stored in one or more files that may be linked statically or dynamically.

The carrier of a computer program may be any entity or device capable of carrying the program. For example, the carrier may include a storage medium, such as a ROM, for example, a CD ROM or a semiconductor ROM, or a magnetic recording medium, for example, a floppy disc or a hard disk. Furthermore, the carrier may be a transmissible carrier such as an electric or optical signal, which may be conveyed via electric or optical cable or by radio or other means. When the program is embodied in such a signal, the carrier may be constituted by such a cable or other device or means. Alternatively, the carrier may be an integrated circuit in which the program is embedded, the integrated circuit being adapted to perform, or used in the performance of, the relevant method.

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 alternative embodiments without departing from the scope of the appended claims. In the claims, any reference signs placed between parentheses shall not be construed as limiting the claim. Use of the verb "comprise" and its conjugations does not exclude the presence of elements or steps other than those stated in a claim. The article "a" or "an" preceding an element does not exclude the presence of a plurality of such elements. The invention may be implemented by means of hardware comprising several distinct elements, and by means of a suitably programmed computer. In the device claim enumerating several means, several of these means may be embodied by one and the same item of hardware. The mere fact that certain measures are recited in mutually different dependent claims does not indicate that a combination of these measures cannot be used to advantage.

CLAIMS:

1. A ciphertext-policy attribute-based encryption system, comprising a re-encrypter (9) for cryptographically transforming a first ciphertext (CT_{P1}) associated with a first access policy (P1) into a second ciphertext (CT_{P2}) associated with a second access policy (P2) by means of a re-encryption key (RK).

2. The system according to claim 1, wherein a ciphertext associated with an access policy can be decrypted by means of a decryption key associated with an attribute set satisfying that access policy.

3. The system according to claim 1, further comprising a re-encryption key generator (7) for generating the re-encryption key (RK), wherein the re-encryption key (RK) enables the re-encrypter (9) to cryptographically transform the first ciphertext (CT_{P1}) associated with the first access policy (P1) into the second ciphertext (CT_{P2}) associated with the second access policy (P2).

4. The system according to claim 3, wherein the re-encryption key generator (7) comprises a subsystem for encrypting a value derived from a pseudorandom number, thereby generating a further ciphertext associated with the second access policy (P2), the re-encryption key generator (7) being arranged for including a representation of the further ciphertext in the re-encryption key.

5. The system according to claim 4, wherein the re-encrypter (9) is arranged for including a representation of the further ciphertext in the second ciphertext (CT_{P2}).

6. The system according to claim 3, wherein the re-encryption key generator (7) is arranged for including in the re-encryption key (RK) an at least partly obfuscated representation of part of a decryption key (SK_{ω}) associated with an attribute set (ω) satisfying the first access policy (P1).

7. The system according to claim 1, wherein the re-encrypter (9) is arranged for bilinear pairing of at least part of the re-encryption key (RK) and at least part of the first ciphertext (CT_{P1}).

8. The system according to claim 1, further comprising a decrypter (6) for decrypting the second ciphertext (CT_{P2}) by means of a decryption key (SK_ω) associated with an attribute set (ω) satisfying the second access policy (P2).

9. The system according to claim 8, wherein the decrypter (6) comprises:
- a subsystem for extracting the further ciphertext from the second ciphertext (CT_{P2});
- a subsystem for decrypting the further ciphertext by means of the decryption key (SK_ω) to obtain the value; and
- a subsystem for decrypting the message stored in the second ciphertext (CT_{P2}) based on the value.

10. The system according to claim 1, further comprising:
- a key generator (3) for receiving an attribute set (ω) of at least one attribute and outputting a decryption key (SK_ω) associated with the attribute set (ω); and
- an encrypter (5) for generating the ciphertext (CT_{P1}) associated with the first access policy (P1), wherein the ciphertext (CT_{P1}) comprises an encryption of a message and the ciphertext (CT_{P1}) can be decrypted by means of a decryption key (SK_ω) associated with an attribute set (ω) satisfying the first access policy (P1).

11. A re-encryption key generator (7) for use in the system according to claim 1, the re-encryption key generator being arranged for generating a re-encryption key (RK), wherein the re-encryption key (RK) enables a re-encrypter (9) to cryptographically transform a first ciphertext (CT_{P1}) associated with a first access policy (P1) into a second ciphertext (CT_{P2}) associated with a second access policy (P2).

12. A workstation comprising the system according to claim 1 or the re-encryption key generator according to claim 11.

13. The workstation according to claim 12, wherein the workstation is a medical workstation.

14. A method of ciphertext-policy attribute-based re-encryption, comprising cryptographically transforming (27) a first ciphertext (CT_{P1}) associated with a first access policy (P1) into a second ciphertext (CT_{P2}) associated with a second access policy (P2) by means of a re-encryption key (RK).

15. A computer program product comprising instructions for causing a processor system to perform the method according to claim 14.

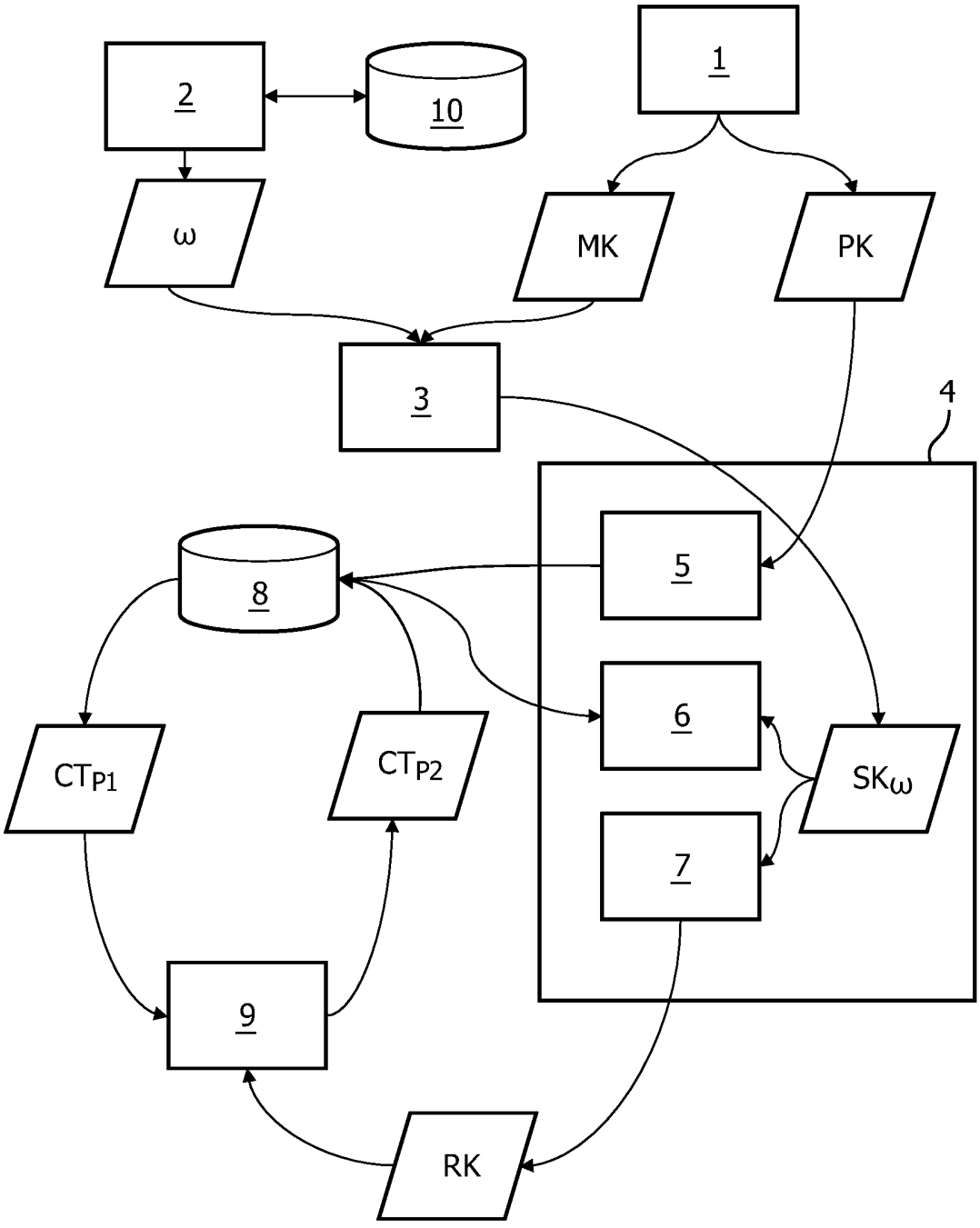

FIG. 1

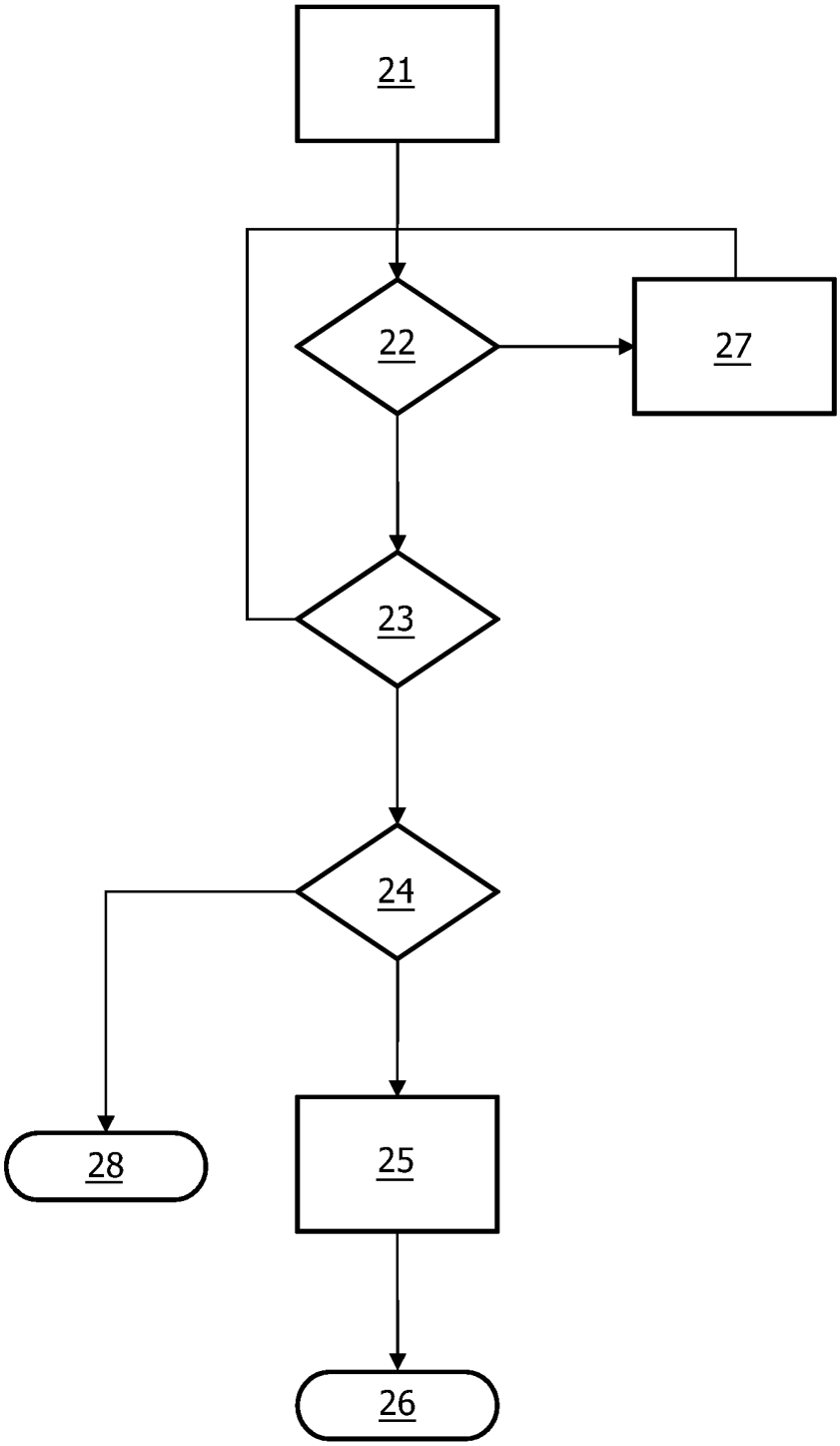

FIG. 2

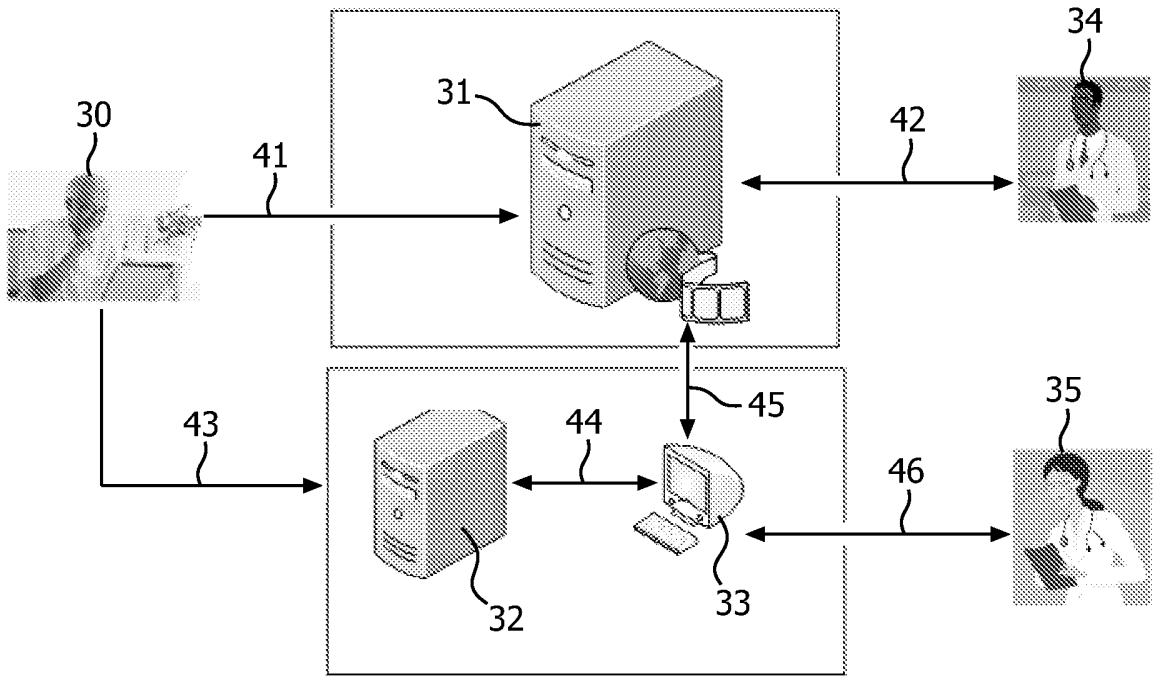

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2010/054581

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/30
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, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	LIANG ET AL: "Attribute Based Proxy Re-encryption with Delegating Capabilities", ACM SYMPOSIUM ON INFORMATION, COMPUTER AND COMMUNICATIONS SECURITY, ASIACCS'09, 10 March 2009 (2009-03-10), - 12 March 2009 (2009-03-12), pages 276-286, XP040465600, Sydney, Autralia DOI: 10.1145/1533057.1533094 the whole document sections 2.1, 4.2, 2.3	1-15

-/--

[X] 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	Date of mailing of the international search report
19 January 2011	25/01/2011

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Billet, Olivier

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2010/054581

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	SHUCHENG YU ET AL: "Attribute-based content distribution with hidden policy", SECURE NETWORK PROTOCOLS, 2008. NPSEC 2008. 4TH WORKSHOP ON, IEEE, PISCATAWAY, NJ, USA, 19 October 2008 (2008-10-19), pages 39-44, XP031356491, DOI: DOI:10.1109/NPSEC.2008.4664879 ISBN: 978-1-4244-2651-5 the whole document	1-15
A	GOYAL ET AL: "Attribute-based encryption for fine-grained access control of encrypted data", ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY, CCS 2006, 30 October 2006 (2006-10-30), - 3 November 2006 (2006-11-03), XP040050996, the whole document	1-15