International Journal of Network Security, Vol.19, No.4, PP.551-558, July 2017 (DOI: 10.6633/IJNS.201707.19(4).08)

A New Level 3 Trust Hierarchal Certificateless Public Key Cryptography Scheme in the Random Oracle Model

Mohammed Hassouna[1], Bazara Barry[2] and Eihab Bashier[2]
(Corresponding author: Eihab Bashier)

Faculty of Computer Studies, National Ribat University, P.O. Box: 55, Khartoum, Sudan[1]
Faculty of Mathematical Sciences, University of Khartoum, P.O. Box: 321, Khartoum, Sudan[2]
(Email: eihabbashier@gmail.com)

(Received Feb. 21, 2016; revised and accepted May 6 & July 19, 2016)

Abstract

Despite the fact that the traditional public key infrastructure provides Level 3 trusted authority, but its two major problems of scalability and certificate management raised the need to an alternative security infrastructure. That motivated the appearance of new technologies to replace the traditional PKI, such as the Identity based encryption, the certificateless encryption, etc. But all those new technologies are yet immature and could not introduce a trust level more than Level 2, except few trials at the level of the authority. This paper aims at introducing an integrated hierarchal certificateless scheme with a Level 3 trust authority. This is done through merging the traditional PKI hierarchy and the certificateless technology in one scheme. The new scheme employs the X509 certificate format and is free of the scalability and certificate management problems of the PKI. We also describe how our new hierarchal certificateless PKC, can be integrated with a traditional PKI through a bridge model.

Keywords: Certificateless cryptography, public key infrastructure, random oracle model, security services, trust levels

1 Introduction

Public Key Infrastructure (PKI) is a complete system to manage the public keys in any public key cryptography-based application using the concept of digital certificates. The PKI provides authentication of system users by allowing some trusted third-party to sign the public key of any entity in the system. In the context of PKI, any entity in the system can verify the authentication of any other entity by verifying its signed certificate using the trusted third-party's public key. In this way, any other cryptographic services (like confidentiality and non-repudiation) can be achieved and implemented.

Furthermore, PKI has some well established trust models that meet the organization flowchart and requirements. Examples of these trust models are hierarchal and bridge models. When the system scale gets large, the number of signed digital certificates also gets large. Therefore the overhead of the management of these certificate increases. Moreover, other issues like public key revocation and its related notification methods are raised. However, in spite of the maturity of the PKI and its wide applications and usage, the PKI has main two challenges. These challenges are scalability and certificate management [1, 10].

Some other paradigms of public key cryptography are introduced to overcome the PKI challenges and simplifying the key management. Identity-based Public Key Cryptography (ID-PKC) (which was invented by Boneh and Franklin [2]) and Certificateless Public Key Cryptography (CL-PKC) (invented in 2003 by Al-Ryami and Paterson [1]) are such examples to these paradigms. The CL-PKC addressed the key-escrow problem of the ID-PKC [1] and provided a lightweight infrastructure for managing the public keys of the users in the system without using the digital certificates. Since the original Al-Ryami and Paterson scheme [1], many certificateless encryption schemes [3, 11, 13], certificateless digital signature schemes [14, 15, 17, 18] and certificateless key agreement protocols [5, 12, 16] were appeared in the literature.

However, the existence of a trusted third party (or trusted authorities) is a common feature among all the public key infrastructure models. These trusted authorities are the certificate authority (CA) in the traditional PKI, the Key Generation Center (KGC) in the ID-PKC and CL-PKC in the certificateless infrastructure. The trusted third party in a public key infrastructure schemes is the heart of the whole security system. It controls the system components and parameters, publishes the system parameters and the users public keys, and in addition to that it might play a partial or a full role in generating the pairs of public and private keys of the users. If this third party is malicious, then the security of the whole infrastructure could be compromised. For this, Girualt [6] defined three levels of trust: At Level 1 trust, the authority knows (or can easily compute) users' secret keys and therefore, can impersonate any user at any time without being detected (the KGC of the ID-PKC). At Level 2 trust, the authority does not know users' secret keys, but it can still impersonate a user by generating false guarantees (CL-PKC). At Level 3 the authority cannot compute users' secret keys, and if it does so, it can be proven that it generates false guarantees (The CA in the traditional PKI).

In 2013 Hassouna et al. [7] proposed an integrated Certificateless public key infrastructure model (CL-PKI). In their model, a different method for generating entity key pair has been introduced. Furthermore, Hassouna et al. [7] incorporated a different binding technique to link the entity's identity with its corresponding keys to ensure the uniqueness of the key pair. The direct security and management advantages of using this method of key generation are two-factor private key authentication, private key portability, private key recovery and private key archiving [7]. Moreover, Hassouna et al. extended their CL-PKI model by proposing a new security model for certificateless digital signature schemes. Then, they proposed a strong and efficient provable secure certificateless digital signature scheme [8] in the Random Oracle Model (ROM) without stating its security proof. Recently, Hassouna et al. [9] stated the complete security proof of the digital signature scheme in the random oracle model [8].

In this paper, we propose a Hierarchal Certificateless Public Key Cryptography Scheme (HCL-PKC) and then use it to construct a Hybrid PKI/CL-PKI scheme. These two schemes are introduced in the context of Hassouna et al.'s Cl-PKI model, hence they enjoy the security properties and key management features of Hassouna et al.'s [7] model.

The rest of this paper is organized as follows. We state Hassouna et al.'s [7] CL-PKI model in Section 2. Hassouna et al.'s [8] digital signature scheme is given in Section 3. In Section 4, we introduce the proposed Hierarchal Certificateless Public Key Cryptography Scheme (HCL-PKC). In Section 5, we give the Hybrid PKI/CL-PKI scheme. Finally, Section 6 concludes the paper.

2 Hassouna et al.'s Certificateless Public Key Infrastructure Model (CL-PKI)

As stated in [7]: to make the CL-PKC schemes suitable for practical applications, there is a need for some sort of infrastructure as the traditional PKI. Therefore, Hassouna et al. [7] proposed a CL-PKI model with three components: Registration Authority (RA), Key Generation Center (KGC) and Public Directory (PD).

The components of the proposed CL-PKI and their functions are as follows:

1) **The Registration Authority (RA):** The registration authority plays the same role as the registration authority of the traditional PKI. The user might interact with this authority and provides proofs of his personal information like names, address, national ID number and email address. After the RA verifies the information of the user, it gives the user a unique random generated password for latter authentication purposes, in addition to the system parameters, generated by the KGC server in a token or any electronic media.

2) **The Key Generation Center (KGC):** The KGC is responsible of generating its master secret and the system parameters. It has to keep it's master secret in a secure storage and publish the system parameters in a public directory. The KGC also has a database that holds the user identities with their password hashed by any strong cryptographic hash function like MD5 or SHA-1.

3) **The KGC's Public Directory (PD):** The public directory is responsible of storing the KGCs' public parameters, users identities, users partial private keys, users public key and other user parameters. It is controlled and updated by the KGC. The contents of the PD are available for only the authenticated users, who do not have the right to write in it. The typical format of the public directory records are given in Figure 1 and Figure 2, respectively.

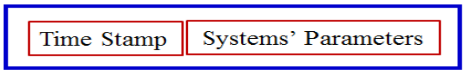

Figure 1: Systems' parameters record

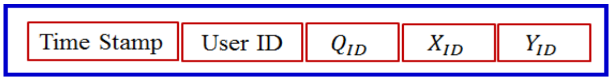

Figure 2: Contents of the public directory of a user

Typically the RA has an offline connection with the KGC. When the KGC generates the user's password at the registration time, the RA passes it to the user without knowing it.

In [7], Hassouna et al. introduced several methods of authentication between the user and the KGC/PD. The complete description of the model is as following:

- **Setup (running by the KGC):** The KGC chooses a secret parameter k to generate G_1, G_2, P, e; where $G_1$and G_2 are two groups of a prime order q, P is a generator of G_1 and $e : G_1 \times G_1 \longrightarrow G_2$ is a bilinear map. The KGC generates a random system's master key $s \in \mathbb{Z}_q^*$ and computes the system public key $P_{pub} = sP$. Then, the KGC chooses a cryptographic hash functions H_1 and H_2, where $H_1 : \{0,1\}^* \times G_1 \longrightarrow G_1$ and $H_2 : \{0,1\}^* \longrightarrow \{0,1\}^n$. Finally, the KGC publishes the system parameters *params*=$< G_1, G_2, e, P, P_{pub}, H_1, H_2, n >$, and keeps the secret master-key safe.

- **Set-Secret-Value (running by the user):** A user m with an identity ID_m downloads the system parameters. He/She then generates two random secret values $x_m, x'_m \in \mathbb{Z}_q^*$. Then, it computes $X_m = x'_m P$ and sends X_m to the KGC. To provide two factor of authentication and protection for the user's private key against the device theft or compromise, the proposed scheme enforces the user to choose a strong password *pass*. The client device uses the hash function H_2 to generate $z_m = H_2(pass)$ and multiplies the base point P by the hashed password to get $z_m P$. The hash function H_2 must be capable to preserve the large size of the hashed value z_m to prevent the brute-force attack on the point $z_m P$. It then uses the hashed value z_m as key along with the MAC function to encrypt the secret value x_m as $MAC_{z_m}(x_m)$ and sends a copy to the KGC's public directory to be stored together with the point $z_m P$ locally. It is worthy to notice that there is no need to store the password *pass* or its hash value z_m.

- **Partial-Private-Key-Extract (running by the KGC):** When the KGC receives X_m from a user m with an identity ID_m, the KGC first computes $Q_m = H_1(ID_m || X_m)$, then it generates the partial private key of user m as $D_m = sQ_m$. User m can verify the correctness of his/her partial private key D_m, through testing whether $e(D_m, P) = e(Q_m, P_0)$.

- **Set-Public-Key (running by the user):** The user m whose identity is ID_m computes $Q_m = H_1(ID_m || X_m)$, $Y_m = x'_m Q_m$ and sets $< X_m, Y_m >$ as his/her long-term public key P_m. Finally, user m sends Y_m to the KGC.

- **Set-Private-Key (running by the user):** Every time a user wants to calculate and use his/her full private key, he/she enters his/her password, the system hashes it as z'_m, calculates $z'_m P$ and compares it with the stored point $z_m P$. If the comparison results in a match , then the password is correct and the user is authenticated. Then, the user uses (z_m) as a key to decrypt the stored $MAC_{z_m}(x_m)$, and uses the extracted value x_m to calculate the full private key by $(x_m + z_m)D_m$. In case of mismatch , the system aborts the process. We must note here that the private key is never stored on the client and it will be deleted after every usage.

Further issues such as the users' authentication at the first time, updates of system's parameters and users' passwords, generation of public and private key pairs, private key recovery; portability; archiving and public key revocation are discussed in details in [7].

3 Hassouna et al.'s Certificateless Digital Signature Scheme

In this section, we provide details on the certificateless digital signature scheme that was proposed by Hassouna et al. and its functionality [8].

- **Setup (running by the KGC):** The KGC chooses a secret parameter k to generate G_1, G_2, P, e where G_1 and G_2 are two groups of a prime order q, P is a generator of G_1 and $e : G_1 \times G_1 \longrightarrow G_2$ is a bilinear map. The KGC randomly generates the system's master key $s \in \mathbb{Z}_q^*$ and computes the system public key $P_{pub} = sP$. Then, the KGC chooses cryptographic hash functions H_1 and H_2, where $H_1 : \{0,1\}^* \longrightarrow G_1$ (Map-to-Point hash function), and $H_2 : \{0,1\}^n \longrightarrow \mathbb{Z}_q^*$ (any cryptographic hash function like MD5 or SHA family). Finally, the KGC publishes the system parameters *params*=$< G_1, G_2, e, P, P_{pub}, H_1, H_2, n >$, while the secret master-key is saved and secured by the KGC.

- **Set-Secret-Value (running by the user):** A user m with an identity ID_m downloads the system parameters, generates two random secret values $x_m, x'_m \in \mathbb{Z}_q^*$. Then, user m computes $X_m = x'_m P$ and sends X_m to the KGC. The proposed scheme enforces the user to choose a strong password *pass*, the system at the client side hashes the password to be $z_m = H_2(pass)$, multiplies the base point P by the hashed password to be $z_m P$, uses the hashed value z_m as key to encrypt the secret value x_m and generates the Password-based Encryption Code (PEC) as $PEC_{z_m}(x_m)$, sends a copy of it to the KGC's public directory and stores it along with the point $z_m P$ locally.

- **Partial-Private-Key-Extract (running by the KGC):** On receiving X_m computed by user m with identity ID_m, the KGC first computes $Q_m = H_1(ID_m)$, then it generates the partial private key of user m as $D_m = sQ_m$.

- **Set-Public-Key (running by the user):** The user m with identity ID_m computes $Q_m = H_1(ID_m)$, $Y_m = x'_m Q_m$ and sets $< X_m, Y_m >$ as his/her long-term public key P_m. Finally, user m sends Y_m to the KGC.

- **Set-Private-Key:** User m's private key is $S_m = (x_m + z_m)D_m = (x_m + z_m)sQ_m = (x_m + z_m)sH_1(ID_m)$. Also, the user generates the secret term $Z_m = x_m P$.

- **Sign:** The user generates the signature of the message M using his secret terms $\{x_m, Z_m\}$ as follows:
  1) The signer generates a big random integer $a \in G_2^*$.
  2) The signer calculates $MP_m = H_1(m) \in G_1^*$.
  3) The signer calculates $MP_{1m} = ax_m MP_m \in G_1^*$.
  4) The signer calculates $s_m = e(MP_m, Z_m)^{ax'_m} = e(MP_m, P)^{ax_m x'_m}$.
  5) The signer sends $\sigma = (m, MP_{1m}, s_m)$ as the signature.
- **Verify:** After receiving the signature $\sigma = (m, MP_{1m}, s_m)$, the verifier uses the public key $< X_m, Y_m >$ of user m to verify the signature as follows:
  1) The verifier checks whether $e(X_m, Q_m) = e(Y_m, P)$. If it holds then user m's public key is authenticated, otherwise the signature is rejected.
  2) The verifier calculates $MP'_m = H_1(m) \in G_1^*$.
  3) If $MP_{1m} = MP'_m$ or $s_m = e(H_1(m), X_m)$ then the verifier rejects the signature. Otherwise, the verifier calculates $r_m = e(MP_{1m}, X_m)$.
  4) The verifier accepts the signature iff $r_m = s_m$, otherwise he/she rejects the signature.

3.1 Hassouna et al.'s Security Model

In Hassouna et al. [8] two types of adversaries were considered: Type I and Type II adversaries according to the term Z_m as follows:

1) **Type I Adversary A_I:** This adversary is allowed to replace the term Z_m by a valid value of his choice, but is not allowed to replace users' public keys and has not access to the master secret key s.
2) **Type II Adversary A_{II}:** This adversary has an access to the master secret key s, and is allowed to replace users public keys with valid values of his choice, but is not allowed to replace the term Z_m.

Type I adversary represents an outsider attacker and type II attacker is a malicious KGC. Two games are defined as follows.

- **Game I.** The first game is performed between a challenger C and a Type I adversary A_I as follows.
  1) Setup. The challenger C runs Setup algorithm and generates a master secret key msk and public system parameters $params$. C gives $params$ to A_I, while keeping msk secret.
  2) Queries. A_I may adaptively issue the following queries to C.
     - Partial private key queries: Upon receiving a partial private key query for an identity ID, C returns the partial private key with respect to identity ID to A_I.
     - Public key queries: Given an identity ID, C returns the corresponding public key terms $< X_A, Y_A >$ to A_I.
     - Replace public key: Given an identity ID with a pair of values (x'^1_{ID}, pk^1_{ID}) which are chosen by A_I, C updates the user ID original secret/public key (x'_{ID}, pk_{ID}) to the new (x'^1_{ID}, pk^1_{ID}).
     - $Z - key$ Extraction queries: This is a new oracle in this security model, given an identity ID, C returns the corresponding $Z - key$ value Z_{ID}.
     - Replace $Z - key$: This is a new oracle in this security model which on input (ID, x^1_{ID}, Z^1_{ID}), C replaces the user ID original term (x_{ID}, Z_{ID}) by (x^1_{ID}, Z^1_{ID}).
     - Private key queries. Upon receiving a private key query for an identity ID, C returns the corresponding private key sk_{ID} to A_I.
     - Sign queries: Proceeding adaptively, A_I can request signatures on any messages m with respect to an identity ID. C computes signature, and returns to A_I.
  3) Forgery. Eventually, A_I outputs a certificateless signature σ^* on message m^* corresponding to public key pk_{ID^*} for an identity ID^*. A_I wins the game if Verify$(params, ID^*, pk_{ID^*}, m^*, \sigma^*) = 1$ and the following conditions hold:
     - A_I has never been queried Partial private key oracle on ID^*.
     - A_I never replaced the user ID^*'s public key.
     - A_I has never been queried Private key oracle on ID^*.
     - A_I has never been queried Sign oracle on (ID^*, m^*).

  The success probability of A_I is defined as the probability that it wins in game I.
- **Game II.** This game is performed between a challenger C and a Type II adversary A_{II} as follows.
  1) Setup. The challenger C runs A_{II} on k and a special Setup, and returns a master secret key msk and public system parameters params to A_{II}.
  2) Queries. In this phase, A_{II} can adaptively access the Private key oracle, Public key oracle, Replace public key oracle, $Z - key$ oracle, Replace $Z - key$ oracle and Sign oracle, which are the same as that in Game I.

3) Forgery. A_{II} outputs a certificateless signature σ^* on message m^* corresponding to public key pk_{ID^*} for an identity ID^*. A_{II} wins the game if Verify$(params, ID^*, pk_{ID^*}, m^*, \sigma^*) = 1$ and the following conditions hold:
   - A_{II} has never been queried Private key oracle on ID^*.
   - A_{II} has never been queried Replace $Z-key$ oracle on ID^*.
   - A_{II} has never been queried Signature oracle on (ID^*, m^*).

   The success probability of A_{II} is defined as the probability that it wins in Game II.

Accordingly, the security definitions of any certificateless digital signature scheme in the Random Oracle Model (ROM) can be given as follows.

Definition 1. *A certificateless signature scheme is* $(t, q_H, q_e, q_z, q_{sk}, q_{pk}, q_s, \epsilon)$*-existentially unforgeable against Type I adversary under adaptively chosen message attacks if no t-time adversary* A_I*, making at most* q_H *to the random oracles,* q_e *partial private key queries,* q_z *to the* $Z - key$ *queries,* q_{sk} *private key queries,* q_{pk} *public key queries and* q_s *signature queries, have a success probability at least* ϵ *in Game I.*

Definition 2. *A certificateless signature scheme is* $(t, q_H, q_z, q_{sk}, q_{pk}, q_s, \epsilon)$*-existentially unforgeable against Type II adversary under adaptively chosen message attacks if no t-time adversary* A_{II}*, making at most* q_H *to the random oracles,* q_z *to the* $Z - key$ *queries,* q_{sk} *private key queries,* q_{pk} *public key queries and* q_s *signature queries, have a success probability at least* ϵ *in Game II.*

Definition 3. *A certificateless signature scheme is existentially unforgeable under adaptively chosen message attack (EUF-CMA), if the success probability of any polynomially bounded adversary in the above two games is negligible.*

Theorem 1. *Hassouna et al.'s [8] digital signature scheme is secure against existential forgery under adaptively chosen message attacks in the random oracle model with the assumptions that CDHP(Computation Diffie-Hellman Problem) and BDHP (Bilinear Diffie-Hellman Problem) in* G_1 *are intractable.*

The full proof of Theorem 1 in the random oracle model is stated in [9].

4 The Proposed Hierarchal Certificateless Public Key Cryptography Scheme (HCL-PKC)

Al-Ryami and Paterson introduced a Hierarchal Certificateless Encryption scheme (HCL-PKE) in their original paper [1]. Their HCL-PKE did not provide a trust Level 3 at the sense of Girualt's definition [6]. Therefore, it was not acceptable as alternative to the traditional hierarchal PKI. In this section, we use Hassouna et al.'s [8] signature scheme as assistant technique to propose a new Hierarchal Certificateless Cryptography scheme (HCL-PKC) which is based on Hassouna et al.'s [7] CL-PKI model. The proposed HCL-PKC (See Figure 3) is straightforward and could provide a trust Level 3.

- **Root KGC Setup.** The KGC chooses a secret parameter k to generate G_1, G_2, P, e, where G_1 (additive group) and G_2 (multiplicative group) are two groups of a large prime order q, P is a generator of G_1 and $e : G_1 \times G_1 \longrightarrow G_2$ is a bilinear map. The KGC randomly generates the system's master keys $x_0, x'_0 \in \mathbb{Z}_q^*$ and computes the system public key $X_0 = x'_0 P$ and the private key term $Z_0 = x_0 P$. Then, the KGC chooses cryptographic hash functions H_1 and H_2, where $H_1 : \{0,1\}^* \times G_1 \longrightarrow G_1$ and $H_2 : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$. Finally, the KGC publishes the system parameters $params =< G_1, G_2, e, P, X_0, H_1, H_2, n >$, while the secret master-keys are saved and secured by the KGC.

- **Set-Secret-Value.** The user at level t with identity ID_t, where ID_0 is the identity of the root KGC downloads the system parameters $params$, generates two random secret numbers $x_t, x'_t \in G_2^*$. As in the signature scheme, we enforce the user to choose a strong password $pass$, the system at the client side hashes the password to be $z_m = H_2(pass)$, multiplies the base point P by the hashed password to be $z_m P$, uses the hashed value z_m as a key to encrypt the secret value x_m and generates the Password-based Encryption Code (PEC) as $PEC_{z_m}(x_m)$, sends copy of it to the KGC's public directory and stores copy of it along with the point $z_m P$ locally.

- **Set-Public-Key.** The user at level t calculates its public key (X_t, Y_t) as $X_t = x'_t P$ and $Y_t = x'_t Q_t$ where $Q_t = H_1(ID_t, X_t)$. Then, the user sends X_t to the previous user in the hierarchy ID_{t-1}.

- **Extract-Partial-Private-Key.** The user at level $t-1$ accepts the request of the users at level t (the request contains the terms Q_t and X_t) and calculates their partial private key D_t as $D_t = x_{t-1} Q_t$. Furthermore, the user at level $t-1$ signs the public term X_t of the user at level t using the proposed CL-SS scheme with the terms Z_{t-1} and the per-signature random number a_{t-1} and creates the signature as (X_t, MP_{1t}, s_t) and puts this signature along with the rest of user's public terms into the public directory $\{ID_t, Q_t, X_t, Y_t, MP_{1t}, s_t\}$.

- **Set-Private-Key.** Every time the user at level t needs to calculate and use his/her full private key, he/she enters his/her password, the system hashes it as z'_m, calculates $z'_t P$ and compares it with stored

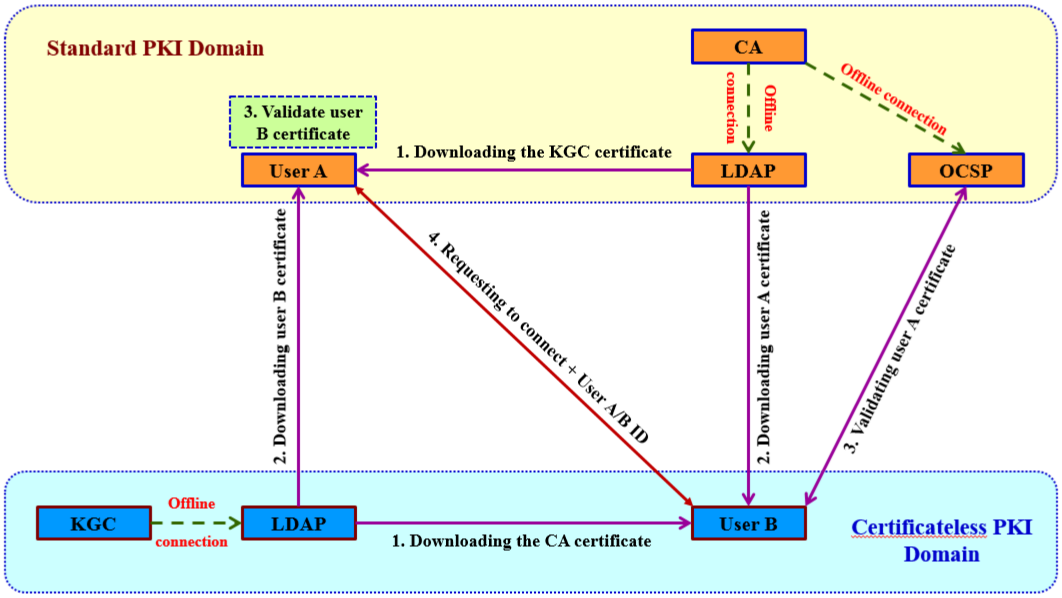

Figure 3: The proposed HCL-PKC model

point z_mP. If the comparison result in a match, then the password is correct and the user is authenticated. The user then uses (z_t) as a key to decrypt the stored encrypted value x_t, and after that uses the extracted value x_t to calculate the full private key by $(x_t+z_t)D_t$ and the term $Z_t = x_tP$. In case of a mismatch, the system aborts the process.

Every user in the system has a unique record in the Public Directory (PD) which contains the information $\{ID_t, Q_t, X_t, Y_t, PEC_{z_t}(x_t), MP_{1t}, s_t\}$. We can think about the user's record as X.509 certificate. Hence, the interoperability between the traditional PKI system and this proposed HCL-PKC scheme will be easy because the two systems will be compatible.

Furthermore, the proposed HCL-PKC scheme provides a new mechanism to authenticate the user's public key and provides a trust Level 3 as same as the hierarchal PKI does. That means if the user's public key has been replaced, then no one excepts the user's intermediate KGC can do that. This because no one can replace the signature term by a valid one except the user's intermediate KGC. Therefore, the user can detect and determine the entity that has replaced his/her public key.

Moreover, the proposed HCL-PKC scheme inherits an attractive feature from CL-PKI model that is introduced by Hassouna et al. [7], which is stated as: Even if the KGC or the intermediate KGC replaces (temporarily) the public key (as in the traditional PKI system) in order to compromise that user for decryption or signature forgery, this attack will fail because the user's private key is calculated from another different secret value. So, replacing the user's public key is not enough for compromising that user. Therefore, the separation of public/private key generation provides strong security feature.

5 Hybrid PKI/CL-PKI Scheme

Suppose we have organization with two domains, the first domain utilizes the traditional PKI with one CA and one LDAP server for trust distribution. The other domain has the Hassouna et al.'s [7] CL-PKI which has the same structure as the traditional PKI, i.e it uses X.509 certificate format to load the certificateless user's information with the signature as Hassouna et al.'s [8] one. Then, the two domains can operate smoothly as follows:

- **Bridge Model:** Bridge trust model can be used between the CA of the PKI and the KGC of the CL-PKI. Then, the CA generates and signs the X.509 certificate (using a standard PKI and ECC-based signature scheme like ECDSA) to the KGC that includes the KGC's public parameters. Also, the KGC generates and signs the X.509 certificate (using the Hassouna et al.'s signature scheme) to the CA that includes the CA's public key. The CA stores the KGC's certificate into its local LDAP server and also the KGC stores the CA's certificate into its local LDAP server. Since the recent versions of the PKI-enabled protocols like TLS v1.2 protocol [4] have become supportive to the Elliptic Curve Cryptosystems like ECDSA signature scheme and ECDH key exchange protocol as Hassouna et al.'s CL-PKI-enabled protocols did, then it is possible to agree on using the ECDH for key exchange protocol to generate the symmetric key. The other parameters can be agreed on at the handshake phase of the transaction. Note that the users at the PKI domain needs to equipped with the pairing algorithm in order to do the signature generation/verification.

- **PKI Domain's User:** User A in the PKI domain when encrypting/signing a message to user B in the CL-PKI domain, he/she needs to do as follows:

  1) User A first request B's certificate either directly from user B or from the CL-PKI's LDAP server.
  2) After the user A gets user B's certificate, downloads the KGC's certificate from his/her local LDAP server. Then, he/she uses CA's public key to validate the KGC's certificate. If it is not valid, then user A rejects and aborts the transaction.
  3) If the KGC's certificate is valid, then user A extracts KGC's public key and uses it to verify B's certificate by verifying the signature on the user B's certificate using the Hassouna et al. signature scheme.
  4) User A also can verify the expiry/revocation of the user B's certificate using either the CRL mechanism or the OCSP protocol.
  5) After user A authenticates user B, then users A and B can start the handshake protocol to agree on the key size, generate per-session symmetric encryption key using ECDH protocol, agree on the encryption algorithm, hash function and the signature algorithm (ECDSA for PKI users and the Hassouna et al.'s one for CL-PKI users).

- **CL-PKI Domain's User:** User B in the CL-PKI domain when encrypting/signing a message to user A in the PKI domain, he/she does the following:

  1) User B first requests A's certificate either directly from user A or from the PKI's LDAP server.
  2) After user B gets user A's certificate, downloads the CA's certificate from his/her local LDAP server, then he/she uses KGC's certificate to authenticate the CA's certificate (using Hassouna et al.'s signature scheme). If it is not valid, then user B rejects and aborts the transaction.
  3) If the CA's certificate is valid, then user B extracts CA's public key and uses it to verify B's certificate (prefer to use ECDSA algorithm).
  4) User B also can verify the expiry/revocation of the user A's certificate using either the CRL mechanism or the OCSP protocol as in the traditional PKI system.
  5) After user B authenticates user A, then users A and B can start the handshake protocol to agree on the encryption key size, generate per-session symmetric encryption key using ECDH protocol, agree on the encryption algorithm, hash function and the signature scheme (ECDSA for PKI users and Hassouna et al.'s one for CL-PKI users).

6 Conclusions and Remarks

This paper used the Hassouna et al[8] signature scheme and proposed a trust Level 3 hierarchal certificateless public key cryptography scheme. The proposed hierarchal scheme is based on Hassouna et al.'s [7] CL-PKI model. Therefore, it enjoys the same security features that CL-PKI has, along with the interesting trust Level 3 satisfaction property. The paper also proposed a new Hybrid PKI/CL-PKI scheme that provides interoperability model between traditional PKI and CL-PKI systems in one organization under the X.509 certificate format.

References

[1] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 452–473, 2003.

[2] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology (CRYPTO'01)*, LNCS 2139, pp. 213–229, Springer-Verlag, 2001.

[3] A. W. Dent, B. Libert, and K. G. Paterson, "Certificateless encryption schemes strongly secure in the standard model," in *Public Key Cryptography*, pp. 344–359, 2008.

[4] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol*, IETF RFC 4346, 2006.

[5] N. A. Farah, M. Hassouna, M. Hashim, E. B. M. Bashier, "A secure and efficient key agreement protocol based on certificateless cryptography," *University of Khartoum*, 2016. `http://khartoumspace.uofk.edu/handle/123456789/21502`)

[6] M. Girault, "Self-certified public keys," in *Advances in Cryptology (EUROCRYPT'91)*, LNCS 547, pp. 490–497, Springer, 1992.

[7] M. Hassouna, B. Barri, N. Mohamed, and E. Bashier, "An integrated public key infrastructure model based on certificateless cryptography," *International Journal of Computer Science and Information Security*, vol. 11, no. 11, pp. 1–10, 2013.

[8] M. Hassouna, E. Bashier, and B. Barry, "A short certificateless digital signature scheme," in *International Conference of Digital Information Processing, Data Mining and Wireless Communications*, pp. 120–127, 2015.

[9] M. Hassouna, E. Bashier, and B. Barry, "A strongly secure certificateless digital signature scheme in the random oracle model," *International Journal of Network Security*, vol. 18, no. 5, pp. 938–945, 2016.

[10] A. V. N. Krishna, A. H. Narayana, K. M. Vani, "Window method based cubic spline curve public key cryptography," *International Journal of Electronics and Information Engineering*, vol. 4, no. 2, pp. 94–102, 2016.

[11] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model," in *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pp. 273–283, 2007.

[12] R. Mokhtarnameh, S. Ho, and N. Muthuvelu, "An enhanced certificateless authenticated key agreement protocol," in *Proceedings of the 13th International Conference on Advanced Communication Technology*, pp. 802–806, 2011.

[13] S. S. Vivek, S. S. D. Selvi, C. P. Rangan, "CCA2 secure certificateless encryption schemes based on RSA," in *Proceedings of the International Conference on Security and Cryptography (SECRYPT'10)*, pp. 208–217, 2011.

[14] C. Wang, D Long, and Y. Tang, "An efficient certificateless signature from pairing," *International Journal of Network Security*, vol. 8, no. 1, pp. 96–100, 2009.

[15] H. Xiong, Z. Qin, and F. Li, "An improved certificateless signature scheme secure in the standard model," *Fundamenta Informaticae*, vol. 88, no. 1-2, pp. 193–206, 2008.

[16] H. Yang, Y. Zhang, and Y. Zhou, "An improved certificateless authenticated key agreement protocol," in *IEEE 14th International Conference on Communication Technology (ICCT'12)*, pp. 26–30, 2012.

[17] Y. Yuan and C. Wang, "A secure certificateless signature scheme in the standard model," *Journal of Computational Information Systems*, vol. 9, no. 11, pp. 4353–4362, 2013.

[18] L. Zhang and F Zhang, "A new provably secure certificateless signature scheme," in *IEEE International Conference on Communications*, pp. 1685–1689, 2008.

Biography

Mohammed Alfateh Hassouna is Assistant Professor at Department of Computer Science - Faculty of Computer Studies - The National Ribat University - Sudan. He was gain his PhD in cryptography from the University of Khartoum - Sudan. Currently he is working as ICT Manger at the National Ribat University. He has many published papers in several international journals and conferences related to the information security and cryptography. (Email: m.fateh@ribat.edu.sd)

Bazara Barry is an associate professor at the department of Computer Science - University of Khartoum and formerly the head of the same department. He was director of research at the Faculty of Mathematical Sciences. Bazara is a reviewer and TPC head/member of many international journals/conferences and a member of the IEEE. He has won several best paper and research awards at the international level. (Email: baazobarry@hotmail.com)

Eihab Bashier obtained his PhD in 2009 from the University of the Western Cape in South Africa. He is an associate professor of applied mathematics at University of Khartoum, since 2013 and recently, he joined the department of Mathematics, Physics and Statistics of Qatar University. The research interests of Dr. Bashier are mainly in numerical methods for differential equations with applications to biology and in information and computer security. In 2011, Dr. Bashier won the African Union and the Third World Academy of Science (AU-TWAS) young scientists national award in basic sciences, technology and Innovation. Dr. Bashier is a reviewer for many international journals and an IEEE member. (Email: eihabbashier@gmail.com)