Inter-domain usable self-certified public key

YOUNGSEOK CHUNG, SEUNGWOO LEE, DONGHO WON
School of Information & Communication Engineering
Sungkyunkwan University
300 Chunchun-dong, Jangan-gu, Suwon, Gyeonggi-do
KOREA

Abstract: Both a self-certified public key and a verifiable self-certified public key need not be accompanied with a certificate to be authenticated by other users. To validate these public keys, it is required that key generation center’s public key trusted by users. If all users trust same key generation center, public key can be validated simply. But among users in different domains, trusty relationship between two key generation centers must be accomplished. In this paper we propose the inter-domain usable self-certified public key that can be validated without certificate among users under key generation centers whose trusty relationship is accomplished. Also we present the execution of signature between users under key generation centers use different public key parameters.

Key-Words: self-certified, verifiable self-certified, trusty relationship, inter-domain, interoperability

1 Introduction

PKI(Public Key Infrastructure) is being established in each country to provide digital signature or E-commerce services safely [1,2]. There are many types of CA(Certification Authority) in PKI environment to admit a number of users and to provide efficient certification services. A user needs a certificate published by other user’s CA to execute E-commerce with other user under his own CA. The interoperability between two user’s CAs must be accomplished because each user trusts only his own CA [3,4].

There are two schemes to validate user’s public key. One is based on the certificate in PKI environment and the other is based on the ID(identification) without any certificate. And there are two kinds ID based schemes. Shamir proposed one of ID based scheme [5]. In this scheme, the ID is public key itself. And the other is the self-certified public key proposed by Girault [6]. The formal does not need separate validation procedure of a public key because the public ID plays a public key. The latter is the self-certified public key including the ID that plays certificate. In Shamir’s scheme, user’s private key is disclosed to a KGC(Key Generation Center) but not in the self-certified public key. However in the self-certified public key, a public key cannot be validated explicitly before it is used to the procedure of signature and so on. In case of happening some problem when the self-certified public key is used to a signature procedure, it is impossible to know which is wrong a signature or a public key. The verifiable self-certified public key that can be validated by public key itself is proposed in order to make up for this defect [7].

In this paper we propose the inter-domain usable self-certified public key that can be validated among users under different domains like in PKI environment but without validating certificate. A user can validate another user’s public key through his KGC. In order to use the proposed public key in practical applications, mathematical operations must be computed correctly in consideration of that each KGC uses different parameters. Also we present the method that makes users generate and validate a signature who trust different KGCs.

The organization of the paper is as follows. In section 2, we introduce concepts of self-certified public key and verifiable self-certified public key. In section 3, we propose the inter-domain usable self-certified public key and analyze it in section 4. Next we conclude and refer to future work in section 5.

2 Backgrounds

A lot of ideas related to a self-certified property (for example self-certified signature, self-certificate etc.) have been proposed since Girault proposed the self-certified public keys in 1991 [8,9].

A user chooses his private key and sends the public key that has a mathematical relationship with the private key to KGC. And then a KGC generates the self-certified public key by signing user’s public key
and ID. User’s public key in the self-certified public key is connected with user’s ID and so it becomes to have a self-certified property. Girault proposed it by two methods. One is based on the integer factorization problem using RSA/Rabin signature scheme and the other is based on the discrete logarithm problem using ElGamal signature scheme. Fig. 1 shows the method using RSA/Rabin signature scheme.

\[ n : \text{modular} \]
\[ n = p \cdot q \quad (p, q \text{ are large numbers}) \]
\[ g : \text{element of order} \quad r = p'q' \]
\[ \text{where} \quad p = 2p' + 1, \quad q = 2q' + 1 \]
\[ (g' = 1 \mod n) \]
\[ x_A : \text{user’s private key} \]
\[ y_A : \text{user’s public key} \]
\[ w_A : \text{user’s verifiable self-certified public key} \]
\[ ID_A : \text{user’s ID} \]

3 Proposed scheme
The proposed inter-domain usable self-certified public key (hereinafter referred to ISP) has a self-certified property because of using user’s ID for validating. And user’s public key and ID, KGC’s public key and ID, public key parameters are hashed and then included in the ISP. So a verifier can confirm a validity of the ISP, a user’s identity explicitly.

A trusty relationship must be accomplished between two KGCs. When a user requests another KGC’s public key or public key parameters, his KGC makes him certify the requested KGC’s public key and public key parameters by sending hashed then signed formation. So a user can trust another KGC by verifying the signature of KGC he trusts, so that he can validate another user’s public key in different certification domain.

The ISP is consists of 3 procedures. First one is that a KGC generates the ISP and sends it to user, namely issuing procedure. Second one is that a user uses the received ISP in practical applications (for example signature etc.), namely applying procedure. Last one is that another user determines which is wrong a signature or a public key if the ISP is used in signature scheme and then a verification of signature does not succeed, namely validating procedure.

3.1 Terminologies and requirements
\[ A, B : \text{users} \]
\[ KGC_{(A/B)} : \text{key generation center trusted by} \ A, B \]
- Two KGCs are in different certification domains and trustworthy relationship between them is accomplished.
- They archive each other’s public key and public key parameters.
- \( \text{ID}_{A}, \text{ID}_{\text{KGC}_A} \): IDs of \( A, \text{KGC}_A \)
- \( p_{(A/B)} \): a prime number
- It is published by \( \text{KGC}_{(A/B)} \) for user to generate private, public key and signature etc.
- It must be a large prime number in order to be protected from Pohlig-Hellman algorithm that makes possible to solve the discrete logarithm problem if \( p_{(A/B)} \) is consist of only small prime numbers’ multiplication [10]. So if \( p_{(A/B)} = 2p_{(A/B)}' + 1 \), \( p_{(A/B)}' \) must be a large prime number, too. (e.g. \( |p'| \geq 2^{160}, |p| = 2^{1024} \))
- \( n_{(A/B)} \): a public key parameter of \( \text{KGC}_A, \text{KGC}_B \)
- \( n_{(A/B)} = q_{(A/B)} \cdot r_{(A/B)} \)
- If \( q_{(A/B)} = 2q_{(A/B)}' + 1 \) and \( r_{(A/B)} = rp_{(A/B)}' + 1 \), \( q_{(A/B)}' \) and \( r_{(A/B)}' \) must be large prime numbers.
- integer 3 : \( \text{KGC} \)'s public key
- In order to improve the efficiency of computing exponentiation, \( \cdots 000011_{(2)} = 3_{(10)} \) is used between \( \cdots 000011_{(2)} \) and \( 100 \cdots 011_{(2)} \).
- \( d_{(A/B)} \): \( \text{KGC}_A, \text{KGC}_B \)'s private key
- \( 3 \cdot d_{(A/B)} = 1(\mod \phi(n_{(A/B)})) \) \( \cdots (1) \)
- \( \phi(n_{(A/B)}) = (q_{(A/B)} - 1)(r_{(A/B)} - 1) \) \( \cdots (2) \)
- \( g_{(A/B)} \): a primitive element of \( \mathbb{Z}_{p_{(A/B)}} \)
- \( h() \): a hash function
- \( \{0,1\}^t \to G_2 = \{0, \ldots , 2^t - 1\} \) (e.g. \( t = 128 \))

### 3.2 Issuing procedure

A KGC issues an ISP to user on off-line as follows.

1. \( \text{KGC}_A \) requested to issue an ISP by user \( A \) confirms \( A \)'s identity by means of interview. \( \text{KGC}_A \) keeps a list of \( A \)'s and \( 3^{rd} \) person requests to issue \( \text{ISP}_A \) using \( \text{ID}_A \) again.

2. User \( A \) chooses his private key \( x_A \in \mathbb{Z}_{\frac{p_A - 1}{2}} \), computes his public key \( y_A \) using \( \text{KGC}_A \)'s public key parameters \( p_A, g_A \). And then he sends \( \text{ID}_A, y_A \) to \( \text{KGC}_A \).

3. \( \text{KGC}_A \) generates the public information \( \text{PI}_A \) by concatenating \( \text{ID}_{\text{KGC}_A}, n_A, p_A, g_A \).

4. \( \text{KGC}_A \) generates the certification information \( CI_{\text{KGC}_A}, CI_A \) using hash function \( h() \) in order to provide an integrity and prevent a forgery of \( y_A \).

5. \( \text{KGC}_A \) generates \( \text{ISP}_A \) by signing \( CI_{\text{KGC}_A}, CI_A \) and sent it to \( A \).

6. User \( A \) generates \( CI_{\text{KGC}_A}, CI_A \) and confirms \( \text{ISP}_A \) for \( y_A \).

\[
\begin{align*}
\text{PI}_A = & \left[ \text{ID}_{\text{KGC}_A}, n_A, p_A, g_A \right] \quad \cdots (3) \\
CI_{\text{KGC}_A} = & \text{h(PI}_A) \quad \cdots (5) \\
CI_A = & \text{h(ID}_A, y_A) \quad \cdots (6) \\
CI_{\text{KGC}_A} \oplus CI_A = & \text{ISP}_A \quad \cdots (7)
\end{align*}
\]

Fig. 3 shows the procedure of issuing \( \text{ISP}_A \).

<table>
<thead>
<tr>
<th>user A</th>
<th>\text{ID}_A, y_A</th>
<th>\text{PI}<em>A = [\text{ID}</em>{\text{KGC}_A}, n_A, p_A, g_A]</th>
<th>CI_{\text{KGC}_A} = \text{h(PI}_A)</th>
<th>CI_A = \text{h(ID}_A, y_A)</th>
<th>CI_{\text{KGC}_A} \oplus CI_A = \text{ISP}_A \quad \cdots (8)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.3 Applying procedure
An example of applying an ISP to a practical application is as follows. In this example user A generates a signature using ElGamal signature scheme and then user B verifies it.

1. User A chooses an element $k$ randomly and computes intermediate value $R$.
   $$k \in \mathbb{Z}_{p_A^{-1}} \cdot \frac{1}{2}$$
   $$R = g_A^k \pmod{p_A} \cdot \cdots \quad (12)$$

2. User A generates his signature $S$ for the hash value of $M$, $H = h(M)$ and then sends $M, R, S, ISP_A, PL_A', y_A$ to user B.
   $$S = (H - x_A \cdot R)k^{-1} \pmod{p_A - 1} \cdot \cdots \quad (14)$$

3. User B verifies A’s signature.
   $$y_A^R \cdot R^S = g_A^H \pmod{p_A} \cdot \cdots \quad (13)$$

Fig. 4 shows the procedure of generating and verifying a signature.

<table>
<thead>
<tr>
<th>user A</th>
<th>user B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k \in \mathbb{Z}_{p_A^{-1}} \cdot \frac{1}{2}$</td>
<td>$R = g_A^k \pmod{p_A}$</td>
</tr>
<tr>
<td>$H = h(M)$</td>
<td>$S = (H - x_A \cdot R)k^{-1} \pmod{p_A - 1}$</td>
</tr>
<tr>
<td>$y_A^R \cdot R^S = g_A^H \pmod{p_A}$</td>
<td>$H = h(M)$</td>
</tr>
</tbody>
</table>

3.4 Validating procedure
If verification does not succeed in subsection 3.3, the procedure of validating an ISP is accomplished as follows.

1. User B requests $CI_{KGC_A}$ to $KGC_B$ in order to verify A’s signature.
2. $KGC_B$ generates $PL_A$ using $KGC_A$’s public key parameters validated and archived, and generates the certification information $CI_{KGC_A}$.
   $$PL_A' = [ID_{KGC_A}, || n_A || p_A || g_A] \cdot \cdots \quad (16)$$

3. $KGC_B$ generates the signature for $CI_{KGC_A}$" and sends it to user B.
   $$S_B = CI_{KGC_A}^{\nu_d} \pmod{n_B} \cdot \cdots \quad (17)$$

4. User B becomes to trust $KGC_A$’s public key parameters by comparing the hashed $PL'_A$ with the verified $S_B$.
   $$h(PL_A') = S_B \pmod{n_B} \cdot \cdots \quad (18)$$

5. User B validates $y_A$ using $KGC_A$’s public key parameters and $ISP_A$ received from A. B can confirm that a signature is wrong if a validation succeeds, otherwise A’s public key $y_A$ is wrong.
   $$CI_A'' = h(ID_A || y_A) \cdot \cdots \quad (19)$$
   $$h(PL_A') \oplus CI_A'' = ISP_A^{3} \pmod{n_A} \cdot \cdots \quad (20)$$

Fig. 5 shows the procedure of validating $ISP_A$.

4 Security analysis and properties
Security analysis of an ISP and its properties are as follows.

4.1 Security against the 3rd user’s substitution attack for user’s public key
The 3rd user C cannot substitute his public key $y_C$ for $y_A$ included in $ISP_A$. Because $d_A$ is needed to substitute $y_C$ for $y_A$. It is equal to factoring $n_A$ in
formula (2) to compute \( d_A \) (in formula (1)), the inverse of 3 on modulo \( n_A \). So the 3\(^{rd}\) user’s attack like this is computationally infeasible.

4.2 Security against a valid user’s denial attacks
It is impossible for user \( A \) to deny his public key \( y_A \) by insisting that his public key is \( y_A' \) not \( y_A \) and his certification information is \( CI_A'' = h(ID_A \parallel y_A'') \) not \( CI_A \). Because \( d_A \) is needed to generate \( ISP_A'' \) for \( CI_A'' \), namely \( y_A'' \). To compute \( d_A \) is equal to factoring \( n_A \) in formulas (1), (2). So a valid user’s attack like this is computationally infeasible.

4.3 Security against the 3\(^{rd}\) user’s substitution attack for KGC’s public information
The 3\(^{rd}\) user \( C \) cannot substitute his public information \( PI_C = [ID_{KGC_c} \parallel n_c \parallel p_c \parallel g_c] \) for \( PI_A \), namely \( CI_{KGC_c} = h(PI_c) \) for \( CI_{KGC_A} \). Because user \( A \) can confirm that \( CI_{KGC_c} \neq S_B^3 (mod n_B) \) is true easily. And it is impossible for \( C \) to generate \( S_C = CI_{KGC_c} \cdot d_B^{-1} \) (mod \( n_B \)) without knowing \( KGC_B \)’s private key \( d_B \). Knowing \( d_B \) is equal to factoring \( n_A \) in formula (2). So the 3\(^{rd}\) user’s attack like this is computationally infeasible.

4.4 Properties
First, a user can confirm the validity of an ISP for his public key. For example user \( A \) can confirm the validity of \( ISP_A \) for \( y_A \). This is why he generates his public key \( y_A \) on modulo \( p_A \) first in formula (3). After generating \( y_A \) and sending it to \( KGC_A \), he receives \( ISP_A \) from \( KGC_A \). \( p_A \) used in generating \( y_A \) is included in formula (7) through formulas (8), (9) and (10). And then \( A \) confirms if \( ISP_A \) is valid and ISP or not in formula (11).

Second, a user can confirm the validity of other user’s public key from that user’s ISP explicitly. For example user \( B \) can confirm the validity of \( y_A \) from \( ISP_A \). Because \( y_A \) is inputted to one-way hash function that doesn’t permit an inverse operation in formula (20), and then included in formula (21) to be validated.

Third, a user can validate the ISP of user’s under different certification domain. For instance user \( B \) can validate \( ISP_A \). Because \( B \) receives \( KGC_A \)’s public information (formula (16)) signed by \( KGC_B \) (formulas (17), (18)) \( B \) trusts. As \( B \) verifies \( KGC_B \)’s signature in formula (19), \( B \) becomes to trust \( KGC_A \)’s public information.

Forth, a user can execute a cryptographic application like signature together with user under different certification domain. For instance user \( B \) can verify user \( A \)’s signature. This is why \( A \) generates the signature using \( p_A \) in formula (12), (13), (14) and \( B \) verifies \( A \)’s signature using \( p_A \) that is validated in formula (15).

After all an ISP doesn’t provides only a self-certified property, a verifiability but also a usability and an applicability to practical application among users under different certification domains. Table 1 shows an ISP’s properties in comparison with some existing schemes.

<table>
<thead>
<tr>
<th>Properties</th>
<th>SP</th>
<th>VSP</th>
<th>ISP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-certification</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Verifiability</td>
<td>×</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Usability</td>
<td>×</td>
<td>×</td>
<td>☐</td>
</tr>
<tr>
<td>Applicability</td>
<td>×</td>
<td>×</td>
<td>☐</td>
</tr>
</tbody>
</table>

SP : Self-certified Public key
VSP : Verifiable Self-certified Public key
ISP : Inter-domain usable Self-certified Public key

5 Conclusions
Recently according as users are being increased and PKI appropriate to wired and wireless environment is being built, the concepts of interoperability in PKI environment are being studied actively in order to establish a trusty relationship among users under different certification domains.

There are many methods that users under different certification domains can validate other user’s public keys in certificate based schemes. On the other hand
only same certification domain’s users can validate existing self-certified public keys.

In this paper we propose inter-domain usable self-certified public key that can be validated among users under key generation centers whose trusty relationship is accomplished. Also we present the method that makes users generate and verify a signature who under key generation centers use different public key parameters.

We show that the proposed ISP is secure against attacks that the 3rd user forges a valid user’s public key or substitutes his own public information for KGC’s public information, and a valid user denies his public key or insists other user’s public key is his.

Self-certified public keys need not any certificate because of self-certified property. But it is necessary for self-certified public keys to be connected with the wired and wireless PKI environments in order to provide efficient cryptographic communications using least certificates. So from now on we’ll study efficient methods of connecting various PKI environments with self-certified public keys, and interoperability, too

References: