PEPSI (Privacy-Enhanced Permanent Subject Identifier) Embedded in X.509 Certificate

Jaeil Lee,[†] *JongWook Park*[††], *Seungjoo Kim*[†††] *and JooSeok Song*[††††],

KISA(Korea Information Security Agency) Garak-Dong, Songpa-Gu, Seoul 138-803, Korea[†]
Graduate School of Information Security, Korea University, Seoul, Korea[††]
School of Information and Communication Engineering, Sungkyunkwan University, Suwon, Korea[†††]
Dept. of Computer Science, Yonsei University, Shinchon-dong, Seodaemoon-gu, Seoul, 120-749[††††]

Summary
A Certification Authority issues X.509 public key certificates to bind a public key to a subject. The subject is specified through one or more subject names in the "subject" or "subjectAltName" fields of a certificate. Where the subject is a person, the name that is specified in the subject field of the certificate may reflect the name of the individual and affiliated entities (e.g., their corporate affiliation). In reality, however, there are individuals that have the same or similar names. It may be difficult for a relying party (e.g., a person or application) to associate the certificate with a specific person based solely on the subject name. This ambiguity presents a problem for many applications. But, this ambiguity can be resolved by including a "permanent identifier" in all certificates issued to the same subject, which is unique across multiple CAs. In many cases a person's unique identifier (e.g., such as a driver license Number) is regarded as a sensitive, private or personal data. Such an identifier cannot simply be included as part of the subject field, since its disclosure may lead to misuse. This paper presents a new method for secure and accurate user authentication through the PEPSI included in the standard certificate extension of a X.509 certificate. PEPSI can be served not only for user authentication but also for the user anonymity without divulging personal information.

Key words:
PKI, X.509, Certificate, CA, User authentication, Privacy.

Introduction

Generally, the DN(Distinguished Name) in the subject field and/or the subjectAltName extension including identification information such as names and identities of a subject for each entity must be only unique for each subject certified by a CA[1,2]. However, since there are entities that have the same name and frequent change of identification information in actual environment, it is difficult to authenticate the entity in several contexts such as access control and non-repudiation services by only the certificate information itself. For preventing theses problems, the concept of PI (Permanent Identifier) including unchangeable identification information was proposed[3]. In case of including any sensitive personal information in PI, however, its structure has a potential risk to leak the user's privacy information to the public. As a result, PI can not be effective mechanism for securing personal information.

This paper introduces the "PEPSI" (Privacy-Enhanced Permanent Subject Identifier), a method to overcome the weakness of PI. In our approach, the PEPSI is transmitted to the only reliable party, and it enables the user authentication more secure and accurate. Furthermore, it can ensure the user anonymity by just proving the fact of PEPSI possession. In case of using driver license number as a PEPSI, for example, the driver license owner can only prove the fact that he/she has a driver license without divulging the license number.

2. Symbols

The following cryptography symbols are defined in this paper.

H()	Cryptographically secure hash algorithm. SHA-1[FIPS 180-1] or a more secure hash function is required.
SII	Sensitive Identification Information. (e.g., Driver License Number)
R	The random number value generated by a user.
PEPSI	Privacy-Enhanced Protected Subject Information. Calculated from the input value R, SII using two iteration of H().
E()	The encryption algorithm to encrypt the PEPSI value.
EPEPSI	Encrypted PEPSI.

D() The decryption algorithm to decrypt the EPEPSI.

3. PEPSI

3.1 Generation of PEPSI

The user generates the PEPSI as double-hashed value that hashes secret random number R with the SII(Sensitive Identification Information) such as social security number, driver license number, student ID and so forth. The PEPSI can be defined as below:

$$PEPSI = H(H(SII \,//\, R)) = H^2(SII \,//\, R)$$

Despite of the need for strong security, the SII generally has a tendency to be easily estimated such as sex, date of birth and address. Since most of the SII is around 10 digits in length and using only alphanumeric characters, the SII could be easily obtained by guessing attack. A random number R is 512 bits long and the secrete value only known for the user. It is concatenated with a SII and can be served for the purpose of increasing the complexity of the PEPSI. Besides that, it is an important value on validating PEPSI by transmitting to the reliable party through the secure channel. In order to improve processing speed of the PEPSI computation, it assumes to use cryptographically secure hash function just like SHA-1 algorithm, instead of symmetric key algorithm. As described earlier at the introduction, the sequential use of nested hash structure could be optimized to the case of noticing the fact that SII is being possessed by user, namely for the case of security environments that requires the user anonymity.

To request a certificate including the PEPSI, an user should generate a random number R. At this time, the random number R should be managed at the same security level with those of private key and it must be aware that different random number can be used depend on the personal security environment. For example, if an user is using local disk as a storage medium or smart card without computation power, a random number R must be stored with private key by following PKCS#8 standard in Fig.1 & 2[5]. On the other hand, in case of using compatible cryptographic token with computational power following PKCS#11, a new DATA OBJECT in Table 1 should be defined for storing random number R [7].

For this case, the PEPSI can be composed of the transformed value of private key that satisfied OWF(sk) instead of random number R. For example, the PEPSI can be formed in the form of $PEPSI=H^2(SII \,||\, H(sk))$ by using a hash value H(sk) or it can be considered as $PEPSI=H^2(SII \,||\, pk^{sk})$ that pk^{sk}, digital signature value of his/her own public key, replaces a random number R. By using this method, the user has less burden to manage a random number R and another advantage for the user to utilize his/her own private key as the other form.

Table. 1 : PKCS#11 DATA OBJECT storing the random number R

Attribute name	Attribute type	Value
CKA_CLASS	CK_OBJECT_CLASS	CKO_DATA
CKA_TOKEN	CK_BBOOL	FALSE(default)
CKA_PRIVATE	CK_BBOOL	TRUE
CKA_MODIFIABLE	CK_BBOOL	TRUE
CKA_LABEL	Local string	"Random value"
CKA_APPLICATION	Local string	-
CKA_VALUE	Byte array	R

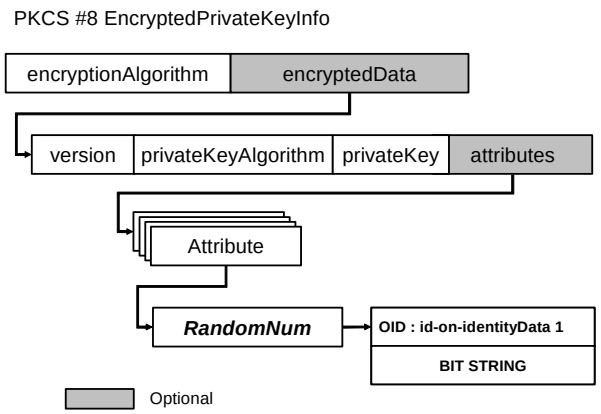


Fig. 1 PKCS#8 structure including random number R

```
-- PKCS#8
-- Private-key information syntax
PrivateKeyInfo ::= SEQUENCE {
  version                 Version,
  privateKeyAlgorithm AlgorithmIdentifier
                          {{PrivateKeyAlgorithms}},
  privateKey              PrivateKey,
  attributes [0]          Attributes OPTIONAL }
-- Random number R   Attribute
randomNum ATTRIBUTE ::= {
  WITH SYNTAX    BIT STRING,
  ID                      id-on-identityData-randomNum}
```

Fig. 2 PKCS#8 ASN.1 module including random number R

The user generating a random number R and public key pair can use PKCS#10 or RFC2511(CRMF) for requesting certificate and the PEPSI information is delivered to CA via those message syntaxes[4,6]. At this time, since CA can't validate the user's PEPSI by only the PEPSI information itself, the user should send random number R to CA. At this point, the random number R is crucial information to authenticate the user.

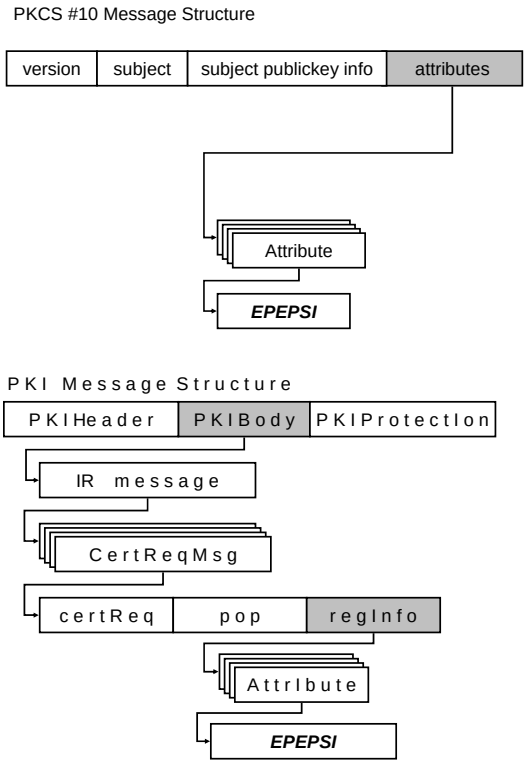


Fig. 3 Certificate Request Format including EPEPSI

EPEPSI=E_{CA}(PEPSI || R) which is encrypted by the public key of a CA with PEPSI and R can be differently located in according to certificate request format being currently used. In case of PKCS#10, for example, it can be included in the attribute field of CertificateRequestInfo for PKCS#10 as Fig. 3. For RFC 2511, it is stored in the regInfo field of CertReqMsg. The structure of EPEPSI can be defined as Fig. 4.

In order to issue certificate, CA must implement validation process about the EPEPSI extracted from certificate request format. Namely,

$$PEPSI, R = D_{CA}(EPEPSI)$$
$$PEPSI' = H^2(SII \,//\, R)$$
$$PEPSI \stackrel{?}{=} PEPSI'$$

```
-- Encrypted PII
id-on-identityData-encryptedPII OBJECT IDENTIFIER ::=
                { id-on-identityData 3 }
EncryptedPII ::= SEQUENCE {
  version          [0]    INTEGER {v1(0)},
  pIIHashAlg       [1]    AlgorithmIdentifier OPTIONAL,
  pIIEncAlg        [2]    AlgorithmIdentifier OPTIONAL,
  certID           [3]    CertId         OPTIONAL,
  encryptedPII     [4]    EncryptedContent
}
EncryptedContent ::= SEQUENCE {
  pII                ProtectedIDInfo,
  randomNum          BIT STRING
}
```

Fig. 4 EPEPSI ASN.1 Module

A CA obtains the PEPSI and random number R by decrypting EPEPSI from its own private key. Afterward, CA generate new PEPSI' and verify if it is identical with the decrypted PEPSI. After successful validation of PEPSI, CA issues certificate including PEPSI described in Fig. 5 as a form of otherName from the GeneralName structure in SubjectAltName extension. Particularly, it is noticeable that the PEPSI can be acceptable to PI structure because realName field of IdentityData can represent real name of the user and the field of userInfo can be defined as extension field that stores various identification information.

```
-- IdentityData Attribute
-- Stored as a form of otherName from the GeneralName
-- structure in SubjectAltName extension
id-on-identityData   OBJECT IDENTIFIER ::= { id-on ? }
IdentityData ::= SEQUENCE {
  realName  UTF8String (SIZE (1..100)),
            -- real name of the user
  userInfo  SET SIZE (1..MAX) OF AttributeTypeAndValue
      OPTIONAL -- could be used for including various
               -- identification information
}

-- PEPSI
 id-on-identityData-protectedIDInfo
           OBJECT IDENTIFIER ::= {id-on-identityData 2}

 ProtectedIDInfo ::= SEQUENCE {
  hashAlg                  AlgorithmIdentifier,
  protectedIDInfo  [0]     OCTET STRING -- PEPSI
 }

 HashContent ::= SEQUENCE {
  identifier          PrintableString, -- SII
  randomNum       BIT STRING       -- Random number
 }
```

Fig. 5 IdentityData and PEPSI ASN.1 Module

3.2. Verification of PEPSI

The reliable party which offers an user access control and non-repudiation service by adapting PEPSI can use a validation process considering the following user requirements and service environment. As a normal case, the user sends random number R, SII, and his/her own certificate including the PEPSI. At this time, the reliable party generates PEPSI' by using SII and R from the user and then authenticates the user by comparing it with the PEPSI extracted from the user certificate. Particularly, if the user doesn't want to disclose SII and random number R for ensuring anonymity, the procedure is a little bit different. For this case, the user only sends the intermediate value of PEPSI as a form of H(SII ‖ R) to reliable party. The reliable party can generate PEPSI' by hashing H(SII ‖ R) one more time. Eventually, as satisfying all requirements of the user, the reliable party is able to authenticate the user whether the PEPSI in certificate is identical with the PEPSI' generated by oneself.

4. Analysis

PEPSI is secure against brute-force attack by using a 512 bits long random number R. Although an attacker is accurately able to guess a SII with partial information about sex, age and date of birth due to short length of the SII, there is no way to verify whether the guessed SII is accurate unless the attacker can obtain a random number R. Also, since the PEPSI is based on using SHA-1 that has cryptographically secure characteristic, it is difficult to find another SII and R that will generate same hash value. In case of birthday problem, the PEPSI can be cryptographically secure since it requires SQRT{2^n} complexity where n is the bit-length of the modulus size. Although a random number R is occasionally delivered to the reliable party by network, there is no problem for disclosing the random number R due to transmitting by secure channel such as SSL/TLS. In addition to that, it strengthens the security of random number R by periodically regeneration of random number R.

5. Conclusion

This paper presents a strong user authentication scheme. PEPSI included in X.509 certificate extension can be served not only for user authentication but also for the user anonymity without divulging personal information.

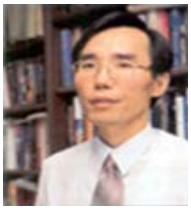
References

[1] ITU-T Recommendation X.509 : "Information Technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", 2000
[2] R. Housley, W. Polk, W. Ford and D. Solo.: IETF RFC3280, "Internet X.509 Public Key Infrastructure: Certificate and CRL Profile", 2002.
[3] D. Pinkas, T. Gindin, RFC 4043 "Internet X.509 Public Key Infrastructure Permanent Identifier", March, 2005
[4] M. Myers, C. Adams, D. Solo and D. Kemp.: IETF RFC 2511, "Internet X.509 Certificate Request Message Format", 1999
[5] RSA Security.: PKCS#8 v1.2, "Private Key Information Syntax Standard", 1993
[6] RSA Security.: PKCS#10 v1.7, "Certification Request Syntax Standard", 2000
[7] RSA Security.: PKCS#11 v2.11, "Cryptographic Token Interface Standard Revision 1", 2001

Jae-Il Lee received his B.S. and M.S. degrees in Computer Science and Statistics from Seoul National University, Seoul, Korea, in 1986 and 1988, respectively. He is pursuing his Ph.D. degree in Computer science at Yonsei University, Seoul, Korea. He was a Software Engineer at IBM Korea, Inc., from 1991 to 1996. He is currently a Vice President of the Korea Information Security Agency, Seoul, Korea. His research interests include information security, PKI, mobile internet security.

JongWook Park received his B.S. degrees in college of information technology from Ajou University, Suwon, Korea, in 1998, and M.S. degrees in Graduate School of Information Security from Korea University, Seoul, Korea, in 2004. He was a software engineer at SamSung SDS, Inc., from 1998 to 2000 and he was a security engineer at Korea Information Security Agency, from 2000 to 2004. He is currently a Ph.D. student in Graduate School of Information Security at Korea University, Seoul, Korea. His research interests include PKI, mobile internet security.

SeungJoo Kim received his B.S. degrees, M.S. degrees and Ph. D. degree in Dept. of Information Engineering, Sungkyunkwan University, Suwon, Korea, in 1994, 1996, and 1999 respectively. He was a director at Korea Information Security Agency, from 1998 to 2004. He is currently a Assistant Professor of School of Information and Communication Engineering, Sungkyunkwan University. His research interests include Attacks on Cryptosystems and PET.



JooSeok Song received the B.S degree in electrical engineering form Seoul National University, Seoul, Korea, in 1976, and the M.S. degree in electrical engineering from KAIST, Korea, in 1979. In 1988, he received the Ph.D. degree in computer science from University of California at Berkeley. He had been an Assistant Professor of Naval Postgraduate School from 1988 to 1989. He is currently a Professor of Computer Science at Yonsei University, Seoul. Korea. His research interests include cryptography, information security, wireless communication, and mobile security.