Server-Aided Delegation in Ubiquitous Computing Environment

Mi Sun Shim* , Jong-Phil Yang** and Kyung Hyune Rhee***
*Dept. of Information Security,
**Dept. of Computer Science,
***Div. of Electronic, Computer and Telecommunication Engineering,
Pukyong National Univ., Busan, 608-737, Korea
E-mail: *ssssblue@yahoo.co.kr, ** bogus@mail1.pknu.ac.kr, *** khrhee@pknu.ac.kr
Tel : +82-51-620-6395 Fax : +82-51-626-4887

Abstract: Computing today is becoming ubiquitous. In such ubiquitous computing environments, entities expect to access resources and services at any time from anywhere. Therefore, the question rises of how to establish trust relationship in previously unknown devices or resources with such environment. After reviewing a model to delegate trust to communicating entities in ubiquitous environment and its security problems, we present a new model for secure delegation over communication entities. We adopt two-party signature scheme as cryptographic primitives. Additionally, we apply threshold cryptosystems to our model for more secure enhancement.

Keywords: Trust delegation, Ubiquitous Security, two-party signature

1. INTRODUCTION

Computing today is moving away from the desktop, becoming diffused into our surrounding and onto our personal digital devices. The ubiquitous computing paradigm is exploding in popularity as one of the main applications taking advantage of advanced mobile and wireless communication technologies. Imagine a scenario where a user, with a portable device, walking through a building, switches on the lights in the corridor and lowers the temperature of the room that he/she enters. This is an example of ubiquitous environments that will soon be a reality. In these ubiquitous computing environments users expect to access resources and services anytime and anywhere, leading to serious security risks and problems with access control as these resources can now be accessed by almost anyone with a mobile device. Therefore, the question rises of how to establish trust relationship in previously unknown devices or resources with such environment, to be delegated services and resources. Accordingly, trust management is one of the most important research issues in ubiquitous computing environment [8].

In this paper, we are concerned about trust delegation in a restricted space as a part of trust management in ubiquitous computing environments. Firstly, we review a model to delegate trust to communication entities in ubiquitous environment and its security problems. We propose a new model for secure delegation over communication entities in ubiquitous environments. In our model, we adopt two-party signature schemes as cryptographic primitives. For real implementation, we consider two types of two-party signature schemes based on factoring and discrete logarithm; mediated RSA(mRSA) and User-controllable two-party Schnorr signature(UCTPS)[4][7], respectively. Additionally, we apply threshold cryptosystems to our model with fault-tolerant property. Because of the inherent character of threshold signature schemes, it can be easily adapted and modeled in the distributed network such like ubiquitous environment.

This paper is organized as follows: Section 2 presents related works and motivations of our work. Section 3 explains system model. Section 4 details a protocol to provide an enhanced trust delegation with our model and security properties. Section 5 focuses on some considerations and system performance. Finally, We discusses out ongoing work and presents a brief summary of our work in section 6.

2. RELATED WORKS

2.1. Distributed Trust

In [8][9], they use distributed trust approach to provide access control to a foreign user. In the distributed trust, they present a flexible mechanism that provides sufficient restrictions on delegation of right in ubiquitous environment. So, the notion of distributed trust is similar to Simple Public Key Infrastructure(SPKI) and Pretty Good Privacy(PGP). There is an example of distributed trust scenario called as Smart Office:

John is a printer repairman of one of the office's partners, but service manager is unable to understand his role in the organization, so he is denied access to use the services. John approaches one of the managers, Susan, and asks for permission to use the services in the Smart Office. According to the policy, Susan has the right to delegate those rights to anyone she trusts. Susan delegates to John, the right to use the lights and the printer but no the fax machine, for a short period of time. Susan's laptop sends a short lived signed delegation to John's hand-held device. When John enters the room, the client on his hand-held device sends his identity certificate and the delegation to

the service manager. As Susan is trusted and has the ability to delegate, the delegation conforms to the policy and John now has access to the light and the printer in the office. Once the delegation expires, John is denied access to any services in the office.

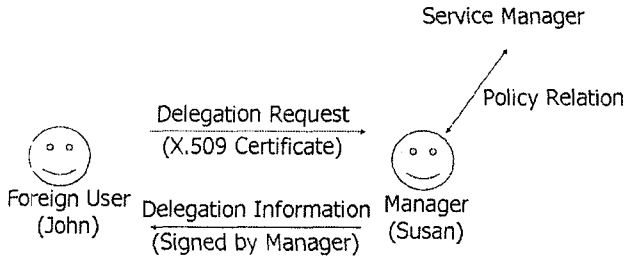

Fig. 1. Issuing delegation information in Distributed Trust

Fig.1 shows the procedure to issue delegation information for a foreign user in the distributed turst. We point out two security concerned problems in the upper example.

- *Is it possible to restrict fine-grained control for the key which is used for signing a delegation by Susan? Specially, on the valid period of the key.*
- *In case that Susan is malicious, is it possible to prohibit Susan from signing a delegation for a John who conspires with her?*

In this paper, we use two-party signature schemes as cryptographic primitives to overcome the above mentioned problems. By using two-party signature schemes, we can successfully restrict Susan's signing ability within limited time period and prevent Susan from signing maliciously. Practical solution of our work will be introduced in the latter.

2.2. Two-Party Signature Schemes

Mediated RSA : mRSA [4]
It is a variant of RSA which splits a private key of a user into two parts. As in standard RSA, each user has a public key (n_u, e_u) and a private key d_u, where n is the product of two large primes, $\gcd(e_u, \phi(n_u)) = 1$, and $d_u \times e_u = 1 \bmod \phi(n_u)$. The public key of a user u is the same as in standard RSA, as is the public-key operation. The two parts of a user's private key are $d_{1,u}$ and $d_{2,u}$, where d_u is the standard private key and $d_u = d_{1,u} + d_{2,u} \bmod \phi(n_u)$. $d_{1,u}$ is the part held by the user and $d_{2,u}$ is the part held by a server. This division of the private key requires changes to the standard RSA key setup because the server must not know $d_{1,u}$ and a user must not know $d_{2,u}$. So, a trusted party(e.g., a CA) performs key setup by generating $\{p_u, q_u, e_u, d_{1,u}, d_{2,u}\}$ for a user u. The private key d_u is generated in the standard manner, but is communicated to neither the server nor the user. Instead, $d_{2,u}$ is chosen as a random integer in $[0, n_u - 1]$, and $d_{1,u}$ is then calculated as

$$d_{1,u} = d_u - d_{2,u} \bmod \phi(n_u).$$

Because the private key d_u is split into two "halves", private key operations require the participation of both the user and the server: e.g., each party raises the message to its half-exponent, modulo n, and the results are then multiplied, also modulo n. Thus the full private key never needs to be reconstructed.

User Controllable Two Party Schnorr Signature : UCTPS [7]
It collaboratively generates a digital signature via Schnorr signature scheme between a user and a server[1]. Let p and q be large primes with $q \mid p-1$. Let g be a generator of a multiplicative subgroup of Z_p^* with order q. Then, $H()$ denotes a collision resistant hash function. To initialize the scheme, a user generates private key x and public key y, where $1 \le x \le q$ and $y = g^x \bmod p$. Then, the user generates a random control parameter δ and computes virtual private key $SK = x + \delta$. The user securely sends SK to a server and deletes the private key from system memory. To sign a message M, the user generates a value $\tilde{k} \in {}_R Z_q$, computes $\tilde{r} = g^{\tilde{k}} \bmod p$ and send $\tilde{r}$, M to the server. The server generates a random $k \in {}_R Z_q$, and computes $r = g^k \bmod p$ and $R = r \cdot \tilde{r} = g^{\tilde{k}+k} \bmod p$. Then, the server computes virtual signature $vs = k + H(M, R) \cdot SK \bmod q$ and sends (r, vs) to the user. Then, the user computes $R = r \cdot \tilde{r}$ and derives real signature $s = vs + (\tilde{k} - H(M,R) \cdot \delta) \bmod q$.
Finally, the user's valid signature on M is (R,s), where $s = (k + \tilde{k}) + H(M,R) \cdot x \bmod q$.

To adapt UCTPS to our model, we assume that a trusted party generates a private key and computes a corresponding virtual private key instead of the user same as the key generation procedure of mRSA.

3. SYSTEM MODEL

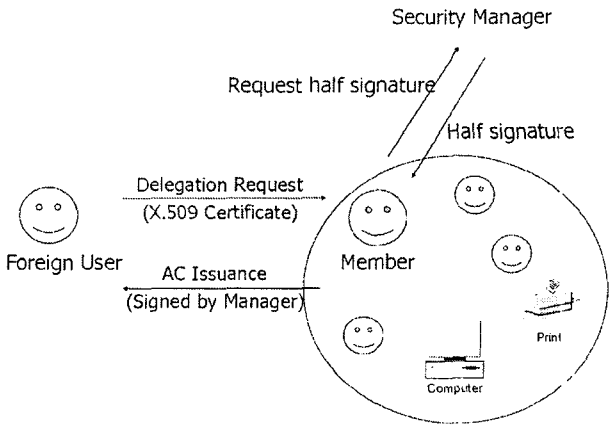

Fig. 2. System Model

We consider a modified new model for smart office scenario in the Distributed Trust of Section 21. Fig. 2 shows a system model for our work. In our model, there is a unique trusted party called as security manager(SM) which owns a partial secret information for each member's private key and collaboratively sign a message together with each member. We assume that our smart office consists of lots of members and system resources such as several lights, a fax machine and a printer, etc. Someone who owns appropriate right can access system resources. When a foreign user wants to have permission to access resources in the smart office, he approaches one of members in the smart office. Then, the member acts as a delegator for the foreign user. The delegator issues an access credential based on the system policy by collaborating with security manager. In our model, access credential acts as delegation information and includes rights to access resources, valid period and issuer identification, etc. On receiving an access credential, the foreign user can access resources in the smart office under the rights in the access credential. That is, when a resource receives an access credential, it checks the validity of the access credential based on the system policy. Once the access credential expires, the foreign user is denied to access any resources. To design our system, we must consider the following requirements.

- A member cannot issue an admission credential to a foreign user arbitrarily. That is, the system must present an elegance solution to restrict malicious issuances of access credentials
- The foreign user is allowed to access certain services without creating a new identity for him in the smart office or assigning a temporary role to him.
- The system should be realistic for implementation in the ubiquitous environment.

4. PROTOCOL DESCRIPTIONS

4.1 Initial Setting

For the rest of this paper, we use the following notations and assumptions.

- SM : Security manager in the smart office. It has a partial secret information for each member's private key. It also has a key-pair for itself and X.509 certificate which is issued by a certificate authority(CA) in PKI.
- M_i : The i -th member in the smart office. Conceptionally, his private key is split into two parts. One is held by the user and the other is held by security manager. When he receives a request from a foreign user, he can act as a delegator for the foreign user. To issue an access credential for the foreign user, it collaboratively signs the access credential with security manager by using two-party signature schemes.
- FU : A foreign user. He has X.509 certificate in PKI.
- AC_{FU} : Access credential for FU. Through AC_{FU}, FU has the right to access resources in the smart office.

In this paper, since we use mRSA and UCTPS as cryptographic primitives, we separately describe our model according to the underlying cryptographic primitive. As a matter of convenience, when we use mRSA for our model, we call it as CASE-1. Otherwise if we use UCTPS, we call it as CASE-2. To initialize the system, a certificate authority(CA) performs the followings:

Step 1.
CA generates a private key and a corresponding public key based on the underlying cryptographic primitives for a user in the smart office

- CASE-1: CA generates a private key d_{M_i} and a corresponding public key e_{M_i} for M_i based on key generation rule of RSA.
- CASE-2: CA generates a private key x_{M_i} and a corresponding public key y_{M_i} for M_i based on key generation rule of Schnorr signature scheme.

Step 2.
Generally, CA splits M_i's private key into two parts . Then, CA securely sends one of two parts to M_i and the other to security manager.

- CASE-1: d_{M_i} is split to d_{1,M_i} and d_{2,M_i} same as section 2.2 Then, CA securely sends d_{1,M_i} to M_i and d_{2,M_i} to security manager.
- CASE-2: CA generates a random control parameter δ and computes $SK = x + \delta \pmod q$. Then, CA securely sends δ to M_i and SK to security manager.

Step 3.
CA issues X.509 certificate to M_i .

Above Step 1 through Step 3 can be performed as many times as the number of members in the smart office.

4.2 Access Credential Issuance Protocol

When a foreign user(FU) enters the smart office, the following steps are performed:

Step 1: Delegation Request
FU sends dele-request to M_i, who will be a delegator for FU . This request consists of FU 's X.509 certificate and additional information.

Step 2: Signing
Upon receipt of dele-request, the delegater (M_i) make right information(R-Info) for FU based on the system policy.

(a) (*Local partial signing*): To issue an access credential(AC_{FU}) for FU .

- CASE-1: The delegator computes partial signature

$$mps = (\text{R-Info})^{d_{1,M_i}} \bmod n_{M_i}$$

and sends to security manager(SM).

- CASE-2: The delegator generates a value $\tilde{k} \in {}_R Z_q$ and computes $\tilde{r} = g^{\tilde{k}} \bmod p$. Then, it computes

$$mps = \tilde{k} - H(\text{R-Info}, R) \cdot \delta \bmod q$$

and sends $\tilde{r}$, R-Info to SM .

(b) (*SM's partial signing*) : Upon receipt of the message from the delegator,

- CASE-1: SM computes a partial signature for the delegator

$$smps = (\text{R-Info})^{d_{2,M_i}} \bmod n_{M_i}$$

and sends it to the delegator.

- CASE-2: SM generates a random $k \in {}_R Z_q$ and computes $r = g^k \bmod p$ and $R = r \cdot \tilde{r} = g^{k+\tilde{k}} \bmod p$. Then, SM computes

$$smps = k + H(\text{R-Info}, R) \cdot SK \bmod q$$

and sends ($r, smps$) to the delegator.

(c) (*Combining*) : Upon receipt of the message from SM , The delegator combines mps with $smps$ to derive the delegator AC_{FU} .

- CASE-1: The delegator multiplies mps by $smps$ as the following:

$$AC_{FU} = mps \times smps \bmod n_{M_i}$$

- CASE-2: The delegator computes R and derives AC_{FU} which consists of (R, s) , where s is computed by the following equation:

$$s = mps + smps \bmod q$$
$$= (k + \tilde{k}) + H(\text{R-Info}, R) \cdot x \bmod q$$

Step 3: AC Issuance

The delegator(M_i) sends his X.509 certificate and AC_{FU} to FU . Upon receipt of it, FU verifies M_i's X.509 certificate and obtains e_{M_i} . Then, FU verifies AC_{FU} . If verification is satisfied, FU can use it to access resources in the smart office.

4.3 Security Properties

The smart office model in this paper provides the following security properties:

- *Immediate revocation for delegation*
  In case that the security manager wants to revoke the delegation right of a member, the security manager can simply revoke the signing ability of the member by means of without computing with a part of the member's private key.

- *Strong protection of private keys*
  By using two-party signature schemes, the private keys for members in the smart office is kept in two different positions. Therefore, an adversary who wants to know a private key for a user must compromise both the user's machine and security manager.

- *No additional confidential channel*
  Because of inherent character of mRSA and UCTPS, an additional confidential channel is not necessary between security manager and a member. However, if the message authentication is required, we must consider a method to perform key agreement for Message Authentication Code(MAC).

- *No now identity for the smart office*
  It is necessary for a foreign user to create a new identity to access system resources in the smart office. That is, from the viewpoint of the foreign user, he becomes a participant in a ubiquitous computing service without concerning about any specific actions which are required in the smart office.

5. SOME CONSIDERATIONS AND SYSTEM PERFORMANCE

5.1 Enhancing the security of our system

In our model, we restrict a signing ability of members in the smart office by using two party signature scheme. However, since the security manager possesses lots of secret information related to members' private keys as many as the number of members, it should be a main target of a malicious adversary. Therefore, we can enhance the security of our model by using threshold signature schemes. The simplified approach to make our model more secure, we can implement the security manager in the smart office as a distributed system. That is, the distributed system which consists of n servers acts as the security manager(SM) in the smart office. Then, the half of private key of each member which is kept In SM is securely shared among n servers in the distributed system by using secret sharing scheme such as [2] . To compute SM's partial signature on step2-(b) in access credential issuance protocol, at least k servers collaboratively compute it by using threshold signature schemes such as [5],[7],[10],[11].

When we implement the security manager in a distributed manner for threshold protection, the system performance is getting worse than original model. However, we can expect to build a more secure system than the original model.

5.2 System Performance

We use JDK 1.3.1 and Bouncy Castle package to simulate the proposed model[3][6]. And we perform these tests with Pentium IV processors and measured computational costs for participation.

In fact, there must be many kinds of methods which we can hybridize. However, in this paper, we present a hybrid approach of mRSA and Threshold RSA in [5] for the limitation of the paper size. In this hybrid approach, the signing on the user side is performed as mRSA. But, the signing on the security manager side is performed by threshold RSA.

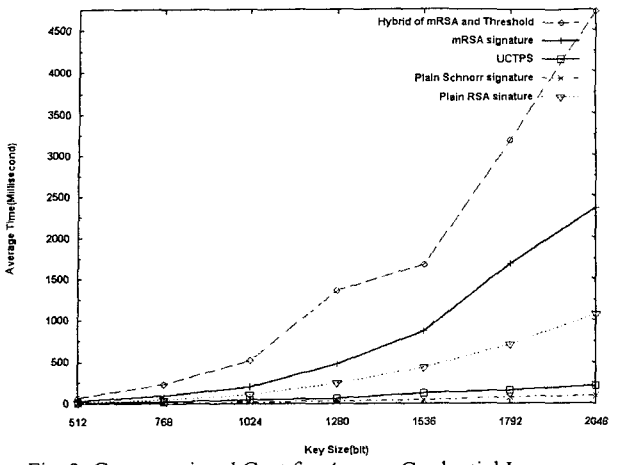

Fig. 3. Computational Cost for Access Credential Issuance

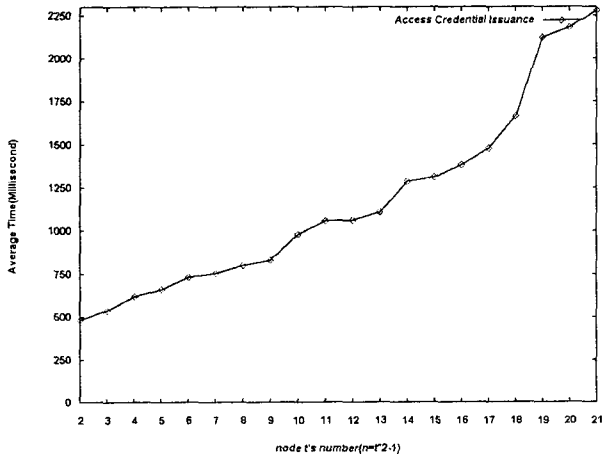

Fig. 4. Hybrid scheme of mRSA and Threshold in setting of 1024-bits key size

Inputs:

$smps'$: the candidate delegation information on the SM

mps : the user's local partial signature

$state\_smps$: statement of the Delegation information

Output:

$smps$: the Delegation Information on SM

1: $Z : (state\_smps)^{-n} \bmod n$

2: $j = 0, Y := smps'$

3: while $j < k$ do

4: $Y := Y \cdot Z \bmod n,\ j := j + 1$

5: if $(state\_smps = (Y \cdot mps)^{PK} \bmod n)$ then

6: break while

7: end if

8: end while

9: output $Y = smps$

Fig. 5. k -bounded coalition offsetting algorithm for hybrid scheme of mRSA and Threshold RSA

Fig. 3 shows computational cost to issue an access credential for a foreign user according to the various key size. The values of 'Hybrid of mRSA and Threshold' are computed in setting of (4, 7) threshold RSA. To apply threshold RSA to our scheme, it is necessary for us to modify 'k -bounded coalition offsetting algorithm' in [5] to Fig.5.

Generally, the key generation for Schnorr scheme is the same as DSS key generation without no restriction of the sizes p and q. But, We implemented Schnorr signature and UCTPS by using the same method to DSS. That is, since we use 160-bits p and q in our simulation. Plain Schnorr signature is the best performer in term of computation time. Note that UCTPS is better efficient than mRSA from the viewpoint of communication time. As expected, the hybrid approach of mRSA and threshold RSA is the worst performer in term of computation time. Fig. 4 shows computation cost to issue an access credential in setting of 1024-bits key size according to the various number of members.

6. CONCLUSION

In this paper, we have considered security requirements for trust delegation in ubiquitous environment, and designed a new security model for secure delegation over communication entities. To guarantee security requirements, we adopt two-party signature schemes as cryptographic primitives. Additionally, we applied the threshold cryptosystems to our model with fault-tolerant property because it can be easily adapted and modeled in the distributed network suck like ubiquitous environment. To evaluate the availability of our model, we implemented some cryptographic primitive and system parameters, and evaluated the performance by simulation.

Acknowledgements

This research was supported by the Program for the Training of Graduate Students in Regional Innovation which was conducted by the Ministry of Commerce, Industry and Energy of the Korean Government.

References

[1] Alfred J. Menezes, Paul C. van Ocrshot, Scoot A. Vanstone, *Handbook of Applied Cryptography*, CRC press, 1997.

[2] A. Shamir, "How to Share a Secret", C.ACM,22(1):612-613, 1979.

[3] Bouncy Castle 1.23, "https://www.bouncycastle.org."

[4] D. Boneh, X. Ding and G. Tsudik, "Identity-Based Mediated RSA", In Proceedings of International Workshop on Information and Security Applications, Jeju Island, Korea, 2002.

[5] Haiyun Luo, Songwu Lu, "Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks", UCLA Computer Science Technical Report 200030, Oct. 2000.

[6] Jess Garms, and Daniel Somerfield, Professional Java Security, Wrox Press Ltd., 2001.

[7] Jong-Phil Yang, Sang Uk Shin and Kyung-Hyune Rhee, "A Simplified Approach to User Controllable Threshold Signatures", IEEE, July 2004.

[8] L. Kagal et al., "Vigil: Providing Trust for Enhanced Security in Pervasive Systems", TechReport, University of Maryland, Baltimore County, August 2002.

[9] L. Kagal, T. Finin and A. joshi, "Moving from Security to Distributed Trust in Ubiquitous Computing Environments", K. Kidong and E. F. Smith, IEEE Computer, December 2001.

[10] R.Gennaro, S.Jarecki, and H Krawczyk, "Revisiting the Distributed Key Generation for Discrete-Log Based Cryptosystems", RSA Security'03, April 2003.

[11] Victor Shoup, "Practical threshold signatures", in Proc. Eurocrypto 2000, LNCS 1807, p.207-220, 2000.