A New PKI-based Single Sign-On Protocol for a Diminutive Security Device, PANDA, in a Ubiquitous Security Environment

Ki-Woong Park, Sang-Seok Lim
CORE Lab. EECS Dept.
KAIST, 373-1 Daejeon Korea
Telephone: (+82)42-869-5425
Email: woongbak@core.kaist.ac.kr

Hosung Song
Computer Science Dept.
University of Wisconsin-Green Bay
MAC C310, Green Bay, USA
Email: songh@uwgb.edu

Kyu-Ho Park
CORE Lab. EECS Dept.
KAIST, 373-1 Daejeon Korea
Telephone: (+82)42-869-3425
Email: kpark@core.kaist.ac.kr

Abstract— **This paper describes the issues and challenges in the design of a new PKI-based security infrastructure enhanced with single sign-on and delegation technology for a diminutive security device in a ubiquitous security environment. In order to provide the PKI-based ubiquitous security infrastructure in consideration of the issues, we propose a PKI-based single sign-on protocol that provides a user with a transparent security mechanism and seamless authentication services using delegation technology. It also enables cost-effective deployment of the security services by offloading complex PKI operations from the devices to the infrastructure. Although a conventional delegation mechanism cannot support non-repudiation mechanism against malicious user's behavior, our proposed protocol and security infrastructure can provide the mechanism by devising a referee server that generates binding information between a device and authentication messages, and retains the information in its local storage for future accusation. The detailed design of the protocol and a PKI-based service infrastructure are presented and then protocol analysis is given in terms of a user authentication latency and the protocol's completeness.**

I. Introduction

We are on the threshold of realizing ubiquitous computing and communications. In a ubiquitous environment, there will be a large number of devices and sensors surrounding a user. As the user moves around, the user device will access the network that can provide ubiquitous services in a seamless manner. Behind the convenience of the ubiquitous computing, however, there has been a critical security concern that a future world which is filled with intelligent and communicable devices will pose severe risks to security of a ubiquitous environment [6]. This is because without appropriate security protection mechanisms, those devices become vulnerable to malicious scanning and eavesdropping attacks [7], which leads to fatal exposure of privacy and confidential information. Therefore, maintaining security between service devices and users becomes a issue of paramount importance in the realization of a ubiquitous environment. As a fundamental way to enable security, authentication and authorization are the two most widely used mechanisms among devices.

A full realization of a ubiquitous services in a way that security is not compromised comes at a price such that a few major changes in an existing computing environment is necessary. Most of all, mutual authentication becomes essential due to anonymity and mobility of users in a ubiquitous environment. As a consequence, the number of the authentication and authorization operations will increases drastically in proportion to the number of services and sensor devices, necessitating much higher computing power in all computing devices. Unfortunately, conventional authentication systems based on widely used RFIDs or smart cards are limited when they are used for enabling high security and dynamic authentication especially in a ubiquitous environment. They do not provide a way to measure physical proximity among devices for location-based services and do not support an ad-hoc mode communication capability between RFIDs or smart cards due to the limited complexity of circuitry. To overcome these limitations of the conventional authentication system, we developed a diminutive security device that is capable of providing ad-hoc communication with other devices, PKI-based authentication, and location based services[19], [20].

In the ubiquitous environment that makes connections between service devices and users dynamically, authentication, authorization, and accounting services should be provided by a security infrastructure[13]. To offer these services, PKI is generally considered as the most appropriate solution for the requirements. In PKI, an RSA algorithm is the key cryptographic operation that requires by far more CPU cycles than a symmetric cryptographic algorithm [4]. The computational complexity results in high deployment costs and operation overhead since those operations are performed on a diminutive device with serious battery and computing power constraints. Besides, the frequency of the users' authentication request increases rapidly along with the number of the service devices that require mutual authentication[16].

In order to conquer these disadvantages, we propose a security infrastructure which is based on PKI and an SSO [1]

[1]Single Sign-On: Single sign-on (SSO) is a mechanism whereby a single action of user authentication and authorization can permit a user to access all computers and systems, without the need to enter passwords repeatedly.[17]

protocol for a cost effective diminutive security device. The proposed SSO protocol provides users with a secure and seamless security mechanism in a ubiquitous environment using a delegation technology. It also enables the cost-effective deployment of the security services by offloading complex functionalities from the devices to the infrastructure. Although a conventional delegation mechanism cannot support a non-repudiation mechanism against malicious user's behavior, our proposed protocol and security infrastructure can provide the mechanism by devising a referee server. Also, using the proposed authentication mechanism, we can achieve the more secure mechanism in accordance with the security policy requirement without replacing devices.

Taking the aforementioned considerations, the security infrastructure and SSO protocol should meet the following requirements.

- Users aspect
  - Non-obstructive user authentication latency.
  - Minimized user intervention : Minimize user intervention but keep the same security level and functions.
- System aspect
  - Cost effective system.
  - Scalability and Interoperability.
- Security aspect
  - Confidentiality : Secure connection between users and service devices[18].
  - AAA[2] : Infrastructure must support a mutual authentication, authorization, accounting, non-repudiation and digital signature.

The remainder of the paper is organized as follows. We present relevant work in Section 2. In Section 3 we present overall system design and components of the proposed security infrastructure and in Section 4 we describe the proposed single sign-on protocol. Analysis of our system ,protocol and performance is discussed in Section 5 and Section 6 concludes this paper.

II. Related Work

A. *SFLASH*^{v3,v2}

In a ubiquitous and mobile environment, much effort has been made to facilitate the deployment of a security infrastructure required for verifying the authenticity of communicating parties and transferring trust among devices over the Internet. Among them, PKI has been considered as a promising foundation for the requirements. The organization of a traditional PKI is illustrated in the Fig.1. Even though PKI provides full security features including AAA and non-repudiation, it has a severe drawback when it is used by a diminutive device with a restricted computing power and battery lifetime in a ubiquitous environment. That is, a user service latency mainly determined by an authentication and authorization latency is exacerbated by not only the restricted resources of the device but also extremely high complexity of RSA operations for encryption, decryption, and certificate validation in PKI.

[2]AAA: Authentication, Authorization, Accounting

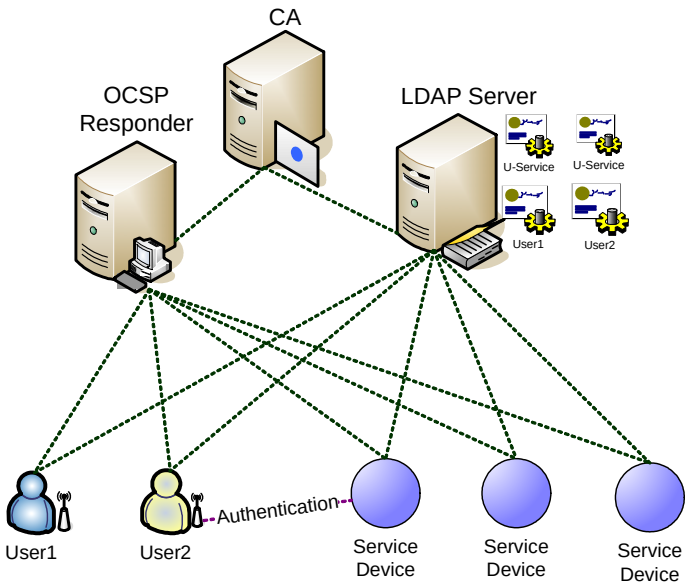

Fig. 1. The Architecture of a Conventional Public Key Infrastructure

Therefore, in order to adopt PKI in a ubiquitous environment, $SFLASH^{v2,v3}$[21] proposed a new cryptography for the purpose of reducing the complexity of the RSA algorithm. The $SFLASH$ is a signature scheme based on a trapdoor function introduced in $SFLASH^{V3}$ specification[23]. They reduced the authentication latency by using the verification mechanism based on SHA-1 algorithm[21]. It has, however, two drawbacks: that the size of its public key is much larger than that of a conventional RSA and that it is troublesome to provide the interoperability because of its deviation from the PKI standard[22].

B. *NSI*

NSI[24] introduced PKI-server that is responsible for searching and verifying certificates as well as key management on behalf of a mobile device or client. In this approach, the server provides a set of simple and abstract APIs that hide most of complex PKI operations from a client, making it possible to keep the hardware and software complexity of a client minimal. From the view point of operation complexity at client and server sides, our approach also takes a similar mechanism, called delegation[29], which has been actively studied and used in a grid computing research area. Both PKI-server and the delegation mechanism are able to minimize the PKI-related computing overhead of a client by offloading the complex operations to a powerful infrastructure server. But both mechanisms have a critical security flaw between the client and the server since they are incapable of providing non-repudiation that is essential in E-commerce. In order to provide a remedy to the problem, we extend the delegation mechanism so as to assure non-repudiation of any secure transaction between a client and a server by introducing a referee server.

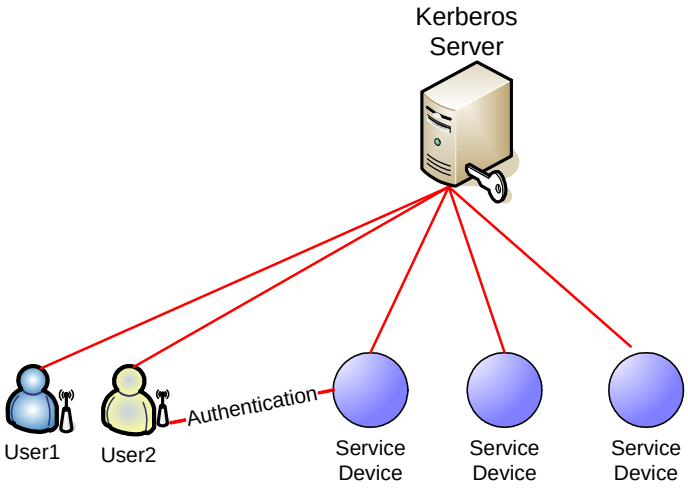

Fig. 2. The architecture of a conventional security infrastructure based on Kerberos

C. M-PKINIT

M-PKINIT[9] is lighter version of the PKINIT[8] that is an extension to the Kerberos protocol for using public key authentication between user and KDC instead of using the symmetric key based authentication. M-PKINIT was proposed in an attempt to reduce a significant overhead of authentication operation when public key protocol operations are invoked in a mobile device. It is a combination of the public key based Kerberos PKINIT and Charon, which is an authentication mechanism providing secure communication between a lightweight PDA client and a Kerberos server using an intermediary system, called a proxy. It aims at enhancing the security of the Kerberos protocol by using a minimal number of public key operations along with a proxy for load distribution. However, this scheme requires public/private key operations whenever a user moves to other Kerberos realm. Besides, three interactions are required between a mobile device and the proxy server to get a TGT[3] and a SGT[4] for each authentication.

D. Kerberos Assisted Authentication

A conventional authentication infrastructure based on symmetric cryptography named Kerberos[25] is shown in Fig.2. Kerberos can provide a shorter authentication latency than PKI in virtue of the lesser overhead of symmetric key operations. Kerberos is, however, inapplicable to our environment since it does not support a digital signature and non-repudiation mechanism that are essential functionalities in security applications. [25] studied how to deploy Kerberos into mobile ad-hoc networks. They presented a secure key exchange scheme for use in ad-hoc networks that is based on Kerberos protocol and introduced measures like replication and elections, so as to ensure maximum connectivity of the clients with servers. It cannot, however, provide an Internet-wide interoperability and AAA because Kerberos uses the authentication mechanism based on symmetric cryptography with a pre-shared secret key.

III. Design and Component of Security Infrastructure

A. Overall system design

Deliberating on the aforementioned system requirements, we design our security infrastructure as follows,

1) Our system adopts PKI as an underlying security infrastructure so as to provide a digital signature and non-repudiation mechanism for AAA.
2) In order to reduce authentication latency and user's intervention, we propose a single sign-on protocol that deploys a delegation mechanism using a proxy certificate [29]. For the protocol, we devise an intelligent delegation server which is responsible for performing prohibitively expensive PKI operations on behalf of a diminutive security device so as to minimize computational overhead at the security device. It also exploits user's context information from ubiquitous sensors for a context-aware authentication technology without compromising any security level of PKI.
3) In an attempt to provide a way to manage a large number of devices and sensors in a ubiquitous environment efficiently and cost-effectively, we employ Kerberos [27] in collaboration with the intelligent delegation server. Kerberos divides the physical ubiquitous environment into a set of sections to disperse the authentication traffic and to achieve scalability as well.

B. Internal of the proposed security infrastructure

The overall architecture of our proposed security infrastructure is illustrated in Fig.3. An organization, wherein a number of service devices and sensors are embedded, is divided into a set of sections. In each section, a Kerberos server is placed to manage all devices and sensors pertaining to the section, and their cryptographic information. The Kerberos server collaborates with the intelligent delegation servers and PKI entities such as CA, LDAP directory server, OCSP responder. In this architecture, the number of generated public/private key pairs can be reduced drastically since each device uses a symmetric key and only a public/private key pair for a Kerberos server is generated.

Seven major components of our architecture are a user, service device, CA[5], LDAP directory server, Kerberos server, delegation server, and referee server.

- **User** is a mobile entity receiving provided services in our ubiquitous security environment. In the environment, in order to utilize the services including authentication, authorization, and accounting, each user should carry a diminutive security device named as PANDA [6] that is a type of smart card enhanced with ZigBee [28]-based ad-hoc mode communication capability and a location

[3]TGT: Ticket Grant Ticket
[4]SGT: Session Grant Ticket
[5]CA: Certificate Authority
[6]PANDA: Personal Authentication Network Device Architecture[19], [20]

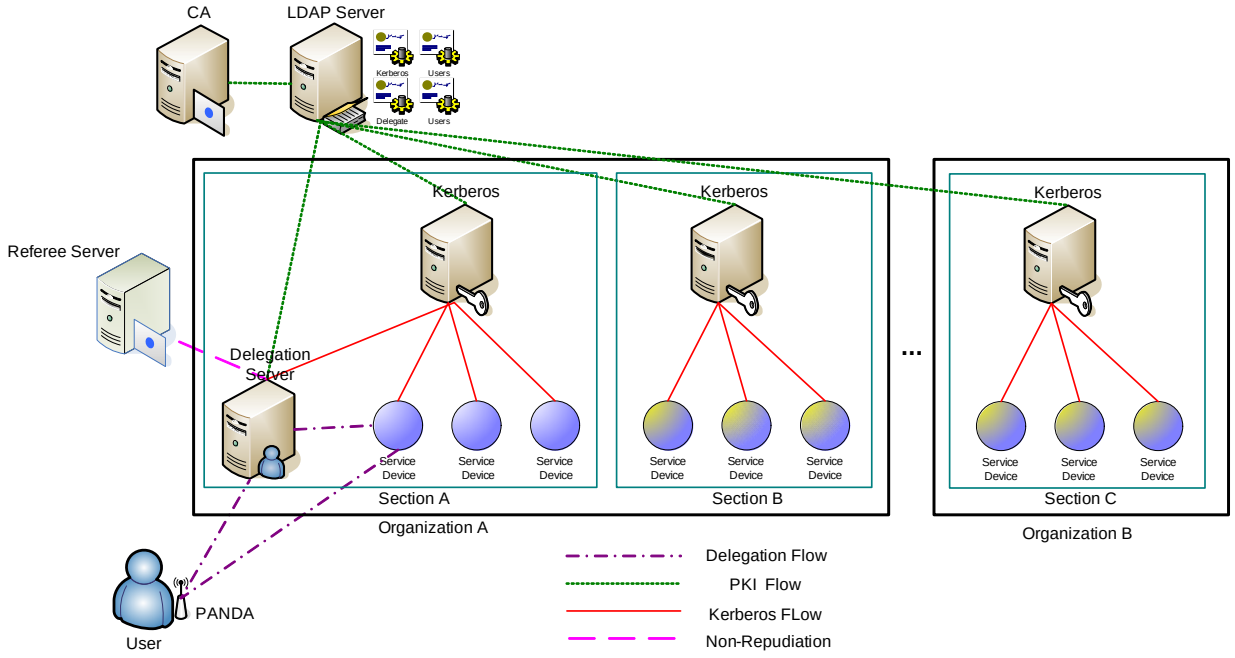

Fig. 3. The Proposed Security Infrastructure

sensing capability required for context-aware services. PANDA is shown in Fig.4.

- **Service device** is permeated in surroundings for providing services. In our environment, a service device has a ZigBee communication module in order to interact with PANDA.
- **CA** is an entity which issues digital certificates which states that the CA attests that the public key contained in the certificate belong to an entity noted in the certificate.
- **LDAP directory server** is responsible for storing and distributing certificates published by a CA.
- **Kerberos Server** is a widely-used authentication server based on a symmetric cryptography. In our environment, it is located in each section, and manages and generates TGTs and SGTs for single sign-on.
- **Delegation Server** is designed to offload complex PKI-related operations from PANDA to the infrastructure, making it possible to develop PANDA with cheap and simple hardwares. It also maintains all proxy certificates containing private keys and public keys that are delegated and signed by PANDA. When a user enters into the security infrastructure for the first time, the user will delegate his authentication operations to the delegation server by following RFC3820[29]. Afterward, the delegation server takes over all authentication operations until the users' proxy certificate is expired.
- **Referee Server** provides a non-repudiation mechanism against a malicious user's behavior. The non-repudiation mechanism will take effect on as long as PANDA uses its own private key in an authentication process. But, after PANDA delegates its operations to the delegation server, it will not use its private key any more so as not to provide a non-repudiation mechanism any more. In order to bring back the mechanism into our system even when delegating is on-going, we devised a referee server. The server investigates all authentication process, generates binding information between a device and the authentication messages on-the-fly, and retains the information in its local storage for future accusation.

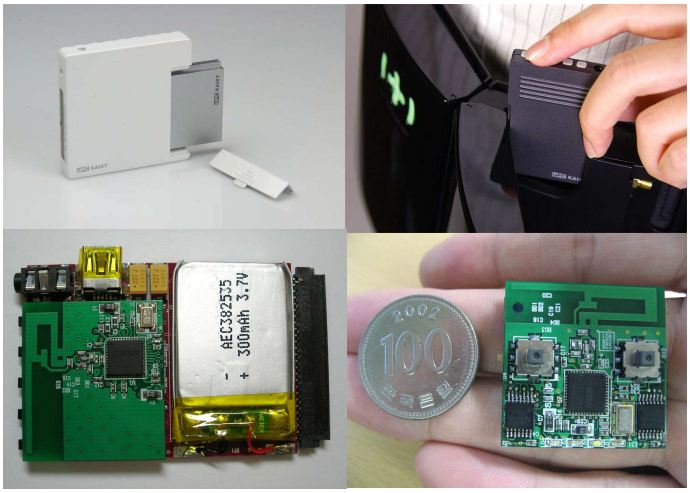

Fig. 4. PANDA(Personal Authentication Network Device Architecture) Left: PANDA Ver1.0(2005) Right : PANDA Ver2.0(2006)[19][20]

IV. Proposed SSO Protocol

A. SSO protocol flow diagram

The flow diagram of the proposed single sign-on protocol is shown in Fig.5. The protocol consists of 6 steps and the

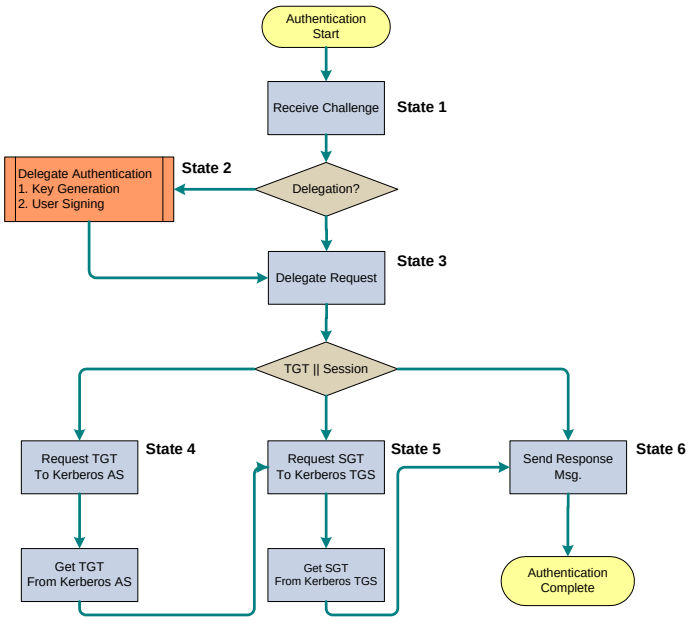

Fig. 5. The Proposed SSO Protocol Flow Diagram

protocol safety is described in the section 5.3.

- **State 1**: If a user wants to receive a service from a service device, the user will accept the beaconing challenge message from the service device.
- **State 2**: If it is the first time when the user accesses the security infrastructure, the user delegates his or her authentication operations to the delegation server. For this purpose, a public/private key pair generated by the delegation server and then the public key is transmitted to the user. On the arrival of the public key, the user generates a proxy certificate containing the public key and sign it using his private key. Lastly, the user sends the proxy certificate back to the delegation server.
- **State 3**: After the delegation process is completed, the user just forwards the challenge message from the service device to the delegation server.
- **State 4, 5, 6**: On the reception of the challenge message, the delegation server contacts to the Kerberos server to get a TGT ($State4$) and an SGT ($State5$). Finally, the delegation server generates and sends a response message to the service device. After the validation check of the service device, the authentication operation is terminated ($State6$).

Fig.6 shows overall interaction between entities during delegation and authentication operations described in Fig.5. Once delegation is successfully completed, $State2$ will be skipped until the corresponding proxy certificate is either expired or revoked. We summarize steps of a user authentication without $State2$ in brief as follows,

1) A service device sends a challenge message to the user.
2) The user forwards the received challenge message to a delegation server.
3) The delegation server gets TGT and SGT from the Kerberos server.
4) Using the SGT, the delegation server makes the response message and transmits it to the service device.
5) The authentication is completed with the arrival of a confirming message from the user.

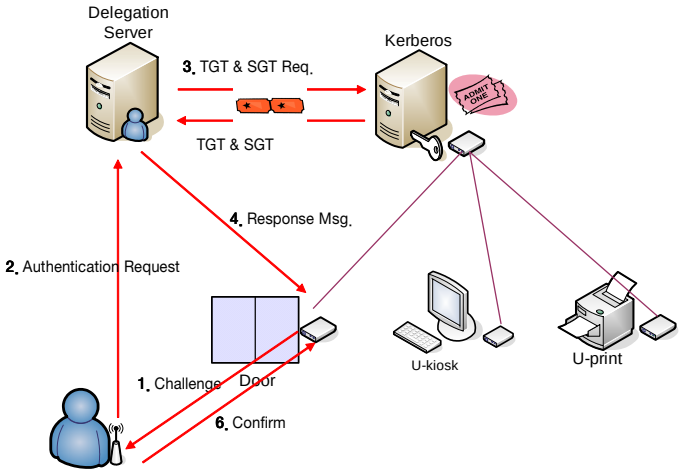

Fig. 6. The Delegated Authentication Mechanism using Proposed SSO Protocol

B. Description of the SSO protocol

In this section, we describe the authentication and the non-repudiation mechanisms of the proposed Single Sign-On protocol. More specific description of the protocol is appended in the appendix of this paper.

1) $State1$: In this state, a service device (Bob) keeps beaconing **Message 1-1** that is comprised of his service ID and $BobCapsule$ periodically until a PANDA bearer, ($Alice$), initiates an authentication process after receiving **Message 1-1**.

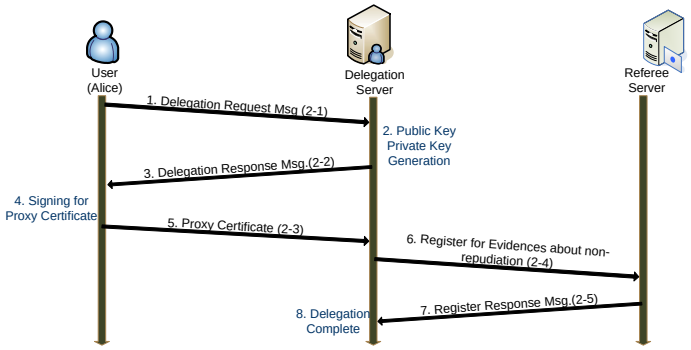

Fig. 7. $State2$: The proposed delegation mechanism

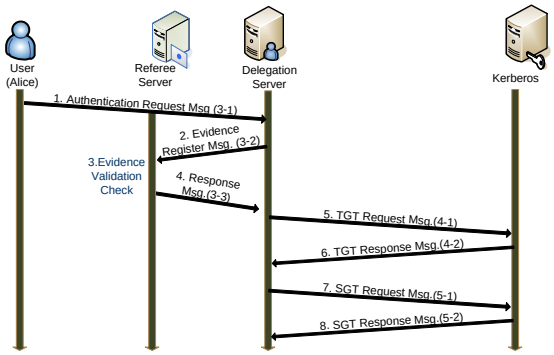

Fig. 8. Message flow diagram from the $State3$ to the $State5$

The $BobCapsule$ is the hashed value of two inputs, a unique serial number and a randomly generated nonce.

2) $State2$: The message flow of $State2$ is shown in Fig. 7. $State2$ is the phase for conducting a user delegation by generating a proxy certificate. If $Alice$ has already delegated herself, the state transits to $State3$. Otherwise, $Alice$ needs to start delegating by entering $State2$. In order to delegate authentication operations, $Alice$ sends a delegation request message, **Message 2-1**. In response, the delegation server generates a private/public key pair and sends the public key back to $Alice$. Lastly, $Alice$ generates a proxy certificate containing the public key and then signs the certificate with her private key in **Message 2-2**. In this phase, a referee server stores a secret key signed by $Alice$ and the evidence, the proxy certificate, for non-repudiation in **Message 2-3/2-4**.

3) $State3$: The $State3$ is the phase of generating and sending an authentication request to a delegation server. On the reception of a challenge message, **Message 1-1**, from the service device, $Alice$ generates **Message 3-1** by combining the challenge message with the subkey (a symmetric key) that will be secretely shared with the service device (Bob) and then sends it to the delegation server. Subsequently, the delegation server generates **Message 3-2** as a proof that states $Alice$ sends **Message 1-1** for authentication; send it to the referee server for non-repudiation. On the arrival of the message, the referee server checks the validity of the authentication request message and sends the result to the delegation server in **Message 3-3**. If the delegation server receives the OK message, the state moves to $State4$.

4) $State4$: The delegation server sends the TGT request message to the Kerberos server in case that the delegation server does not have previously issued the TGT for Bob **(Message 4-1)**. On the other hand, if the delegation server already holds it, the protocol moves to the $State5$ promptly without doing anything in this state. The Kerberos server responds to the TGT request message and sends the TGT to the delegation server in **Message 4-2**. Because each Kerberos server and a delegation server have their own certificates, they can authenticate each other by using an existing PKI.

5) $State5$: If the delegation server obtains TGT in $State4$, the delegation server sends a SGT (Session Grant Ticket) request message to the Kerberos server in **Message 5-1**. In response to the request message, the Kerberos server sends a new SGT for Bob to the delegation server in **Message 5-2**. If the delegation server gets SGT for Bob in advance using the prediction mechanism for the user authentication request [30], this state can be skipped and the protocol can proceed directly to the $State6$ immediately. The message flows from the $State3$ to the $State5$ are represented in Fig. 8.

6) $State6$: $State6$ is the phase where the delegation server sends the final response message to the service device, Bob, in **Message 6-1** and confirms the authentication for Bob. Then, Bob checks the validity of $Alice$ using the $BobCapsule$ and shares the subkey generated by $Alice$ in **Message 3-1**. After that, Bob sends the response message to the delegation server for the mutual authentication in **Message 6-2**. Finally, the authentication is completed by the $Alice$'s confirm message, **Message 6-3**. As a result, $Alice$ and Bob can share the subkey after the authentication. The message flow of the $State6$ is represented in Fig.9.

7) Non-repudiation mechanism: The history data that states $Alice$ authentication requests is retained in the referee server. The definition of notations and more specific messages is given in the Appendix of this paper.

- Data stored per delegation
  - ID_{Alice} : Message 2-4
  - K_{Repu_Alice} : Message 2-4
  - Seq_{Alice} : Message 2-5

- Data stored per authentication
  - $BobCapsule$: Message 3-2
  - $E\{K_{Repu\_Alice}, BobCapsule\}$:Message3-2

How to prove that $Alice$ did send a request authentication from Bob by using the history data is illustrated as follows,

1) Bob requests the referee server for the serial number that is involved in challenge message, **Message 1-1** for $Alice$.
2) The referee server has the key(K_{Repu_Alice}) and the $Bob\_Capsule$ included in the evidence message, **Message 3-2**.
3) Using the key(K_{Repu_Alice}), the referee server can decrypt $BobCapsule$ from the message.

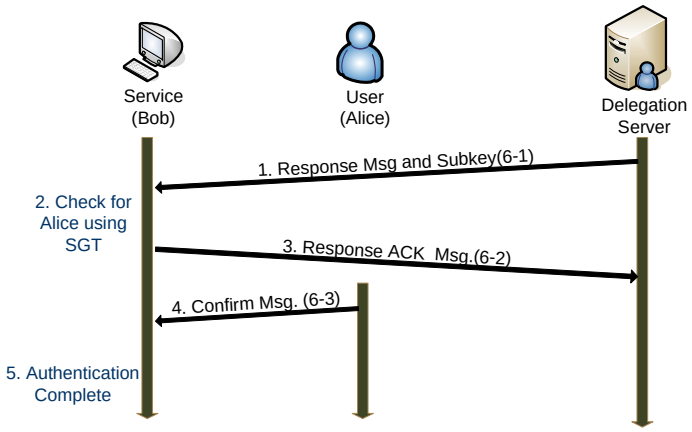

Fig. 9. Message flow diagram of the $State6$

4) Bob submits the nonce message ($Nonce_{Bob}$) that was transmitted to $Alice$ within **Message 1-1**.
5) The referee server inputs the $BobCapsule$ to the hash function. If the output of the hash function is identical with the $Nonce_{Bob}$, it proves the fact that Alice forwarded the received the $BobCapsule$ for the accused authentication. Therefore, the referee server can refute $Alice$'s repudiation.

V. Analysis of the Proposed SSO Protocol

The proposed protocol deploys PKI, the delegation mechanism using the proxy certificate and Kerberos protocol based on a symmetric key cryptography. In this section, we analyze the proposed protocol considering authentication latency, key management and safety of the protocol.

A. Aspect of the authentication latency

As we mentioned earlier, the proposed protocol can provide seamless authentication after the delegation. Fig.10 shows the message flow diagram of the proposed protocol. If a user accesses the security infrastructure for the first time, the user is required to delegate his authentication. Therefore, the initial authentication takes longer than delegated authentications. On the other hand, the authentication latency can be shortened drastically after the delegation because the user can be authenticated using a lot simpler symmetric cryptography.

Therefore, the flow of the authentication can be changed as following cases.

- The first access to the security infrastructure
  State1 - State2 - State3 - State4 - State5 - State6
- Authentication after the delegation
  State1 - State3 - State4 - State5 - State6
- If the delegation server has already obtained TGT thanks to predicted location information
  State1 - State3 - State5 - State6
- If the delegation server has already obtained TGT and SGT thanks to predicted location information
  State1 - State3 - State6

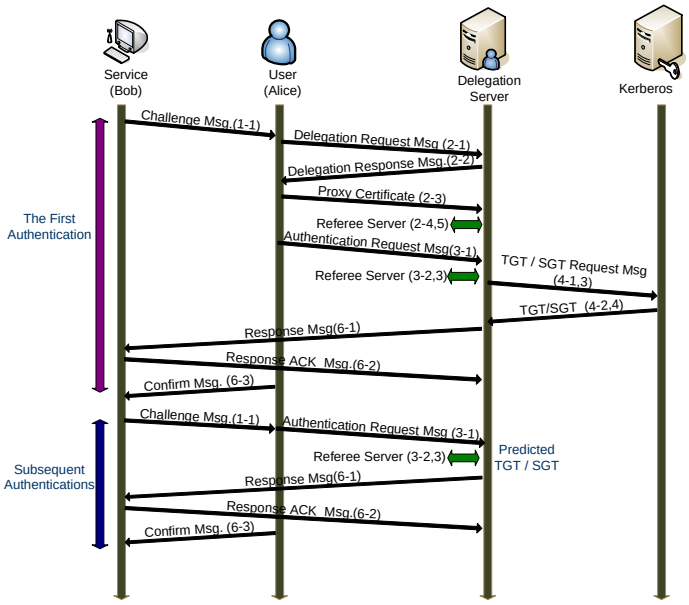

Fig. 10. The proposed SSO protocol authentication mechanism

The improvement of authentication latency with our scheme is illustrated in Fig.11, which is compared with a general PKI operation equipped with a smart card[32]. Let's assume that the user operates the authentication about 40 times per a day. In case that an authentication with a 1024-bit RSA algorithm is processed on the security device equipped with 8-bit processor[31](16Mhz), the authentication latency is on average 4.802 sec[30]. The delegation operation takes about 5.184 sec that is longer than a general PKI authentication. The latency of a contact type smart card is estimated as 4.952 sec that is slightly slower than our security device without a delegation mechanism. However, the authentication latency using our SSO protocol in PANDA can be reduced to 0.344 sec for the specified period after the delegation. As we described in the previous section, the reduction of the authentication latency of PANDA is due to offloading complex operations from the devices to the infrastructure. As a result, we can decrease the authentication time from 4.802 sec to 0.344 sec in average so as to meet our system requirements from the user's aspects.

Fig.12 shows the transition of the authentication latency in accordance with the hit ratio for TGTs and SGTs. Reduction of the authentication latency is between the maximum 0.402 sec and minimum 0.150 sec. In this experiment, we deploy the prediction algorithm that is proposed in[30].

B. Aspect of the key management

From the system's perspective, it is very inefficient that every user and service device should generate their own private and public keys. Equation(1) presents the number of keys to manage in security infrastructure of the two cases. The left-hand side $(M + N)$ presents the number of the keys to manage in a conventional security infrastructure and the right-hand side represents the number of the keys to manage in the proposed security infrastructure. The right-hand side can

$$M + N \geq \frac{M}{\alpha} + N + \frac{N}{\beta}, \quad (1)$$
$$\alpha \simeq \beta \rightarrow M(\alpha - 1) > N.$$

TABLE I
PARAMETERS TO COMPARE THE NUMBER OF KEYS BETWEEN THE CONVENTIONAL APPROACH AND THE PROPOSED APPROACH

Parameter	Description
α	Number of service devices per section
β	Capacity of delegation server
M	Number of service devices
N	Number of users

be derived from the sum of the number of service devices and that of users. In the proposed mechanism, several service devices can be divided into sections. A Kerberos server is arranged into each section, and the Kerberos server generates a public/private key pair on behalf of the service devices in a section. Therefore the total number of keys can be reduced from M to $\frac{M}{\alpha}$. On the other hand, the number of user's key is increased from N to $2N + \frac{N}{\beta}$ because of the delegated keys generated in the delegation server. However, the generated proxy certificates signed by users are not managed by CA. Therefore, the number of the key managed by the CA is $N + \frac{N}{\beta}$. If we assume that the number of α is similar to the number of likewise β and $M(\alpha - 1) > N$ is satisfied, we can conclude that the number of keys to manage is reduced by $(\alpha - 1)M - N$.

C. Aspect of the protocol safety

When a user accesses the security infrastructure for the first time, the user performs a mutual authentication with a delegation server and exchanges a key pair used for the specified period using PKI. In the delegation mechanism, the user can specify the delegation period and all of the messages include the nonce data in case of replay attacks. In this section,

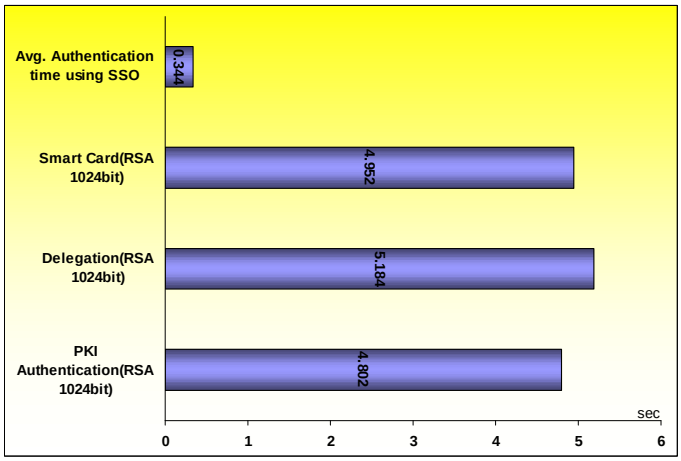

Fig. 11. Comparison of the general PKI authentication latency with the authentication latency of the proposed potocol.

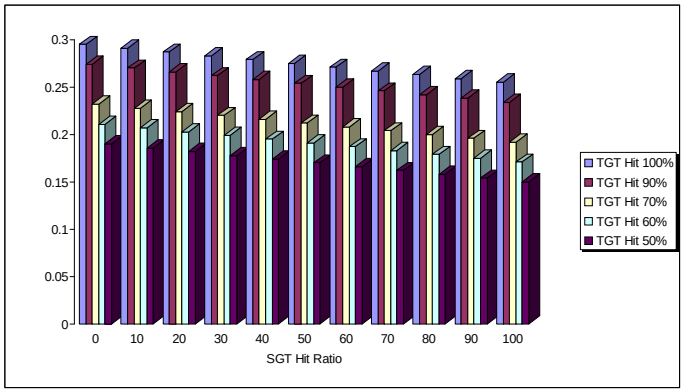

Fig. 12. Change in accordance with the hit ratio of the predicted TGT, SGT

we show the safety of our proposed protocol from replay attacks and MITM [7] attacks.

[Proposition 1] The proposed protocol is safe from the replay attacks.

Proof: Let's assume that the authentication path is $Alice$-Delegation Server-Kerberos Server-Bob. We prove the safety for the next attack types as follows.

1) Replay attack for the delegation request message(Message 2-1)
2) Replay attack for the delegation response message(Message 2-2)
3) Replay attack for the authentication request message(Message 3-1)
4) Replay attack for the response message between the delegation server and the service device(Message 6-2)

- In case 1), an intruder can try to attack by sending the captured delegation request message. However, the key to share with the delegation server cannot be read by the intruder because the key is encrypted by the public key of the delegation server. Therefore, the intruder cannot proceed the delegation mechanism any more.
- In case 2), an intruder can try to attack by sending the captured delegation response message. However, the intruder cannot succeed to attack because the delegation request message includes the nonce data enclosed in the delegation request message.
- In case 3) and 4), an intruder cannot reuse the authentication request message(Message 3-1) and the response message(Message 6-1) because the $BobCapsule$ included in the challenge message is altered per authentication.

[Proposition 2] The proposed protocol is safe from MITM attacks.

Proof: We prove the safety for the next attack types as follows.

1) MITM attack between a user and a delegation server
2) MITM attack between a user and a service device
3) MITM attack between a delegation server and a service device

[7] MITM: Man-in-the-middle

- In case of 1), each entity operates a mutual authentication over PKI before a delegation and shares the key for secure connection. Therefore, the intruder cannot forge the authentication request message of the user.
- In case of 2) and 3), each service device generates the $BobCapsule$ that is altered per an authentication, and this capsule is encrypted and transmitted using the shared key that is generated in a previous state. Therefore the intruder cannot succeed to masquerade as the user or the service device.

VI. Conclusion

This paper presented our effort in designing a new PKI-based security infrastructure that offers an efficient authentication technology for an ubiquitous environment, wherein a large number of devices and sensors are scattered for providing various services. Our security infrastructure features two main achievements: 1) PKI-based single sign-on protocol especially tailored for managing efficiently a large number of devices and sensors in the ubiquitous environment, 2) an intelligent delegation server with a newly devised referee server that ensures non-repudiation of any transaction between a delegator and delegatee. As a consequence, our infrastructure enables a cost-effective but uncompromisingly secure development of a diminutive security device. Furthermore, our delegation mechanism significantly improves an authentication latency as well. According to the performance evaluation, the authentication latency(Avg. 0.34sec) is much shorter than a contact type smart card(Avg. 4.59sec) and a general PKI authentication latency(Avg. 4.80sec).

As processing power speeds up, computing capability of a malicious user also is enhanced, misleading to the development of various attack patterns. To cope with this problem, a security administrator has to devise a series of new security policy and upgrade a security infrastructure accordingly and timely. In this aspect, the delegation approach used in our solution turns out to be very useful since the security infrastructure can be ameliorated only by upgrading the components of the security infrastructure without the substitution of the users' security device. As a result, our security infrastructure and protocol can be applied to the ubiquitous security environment. Based on our design and performance evaluation, we developed the PANDA and are currently implementing a security infrastructure and services for a ubiquitous campus.

Acknowledgment

This material is based upon the Next Generation PC Project that is supported by Institute of Information Technology Assessment.

References

[1] W. Keith Edwards and Rebecca E. Grinter "At Home with Ubiquitous Computing: Seven Challenges",Ubicomp 2001, LNCS 2201, pp. 256-272.

[2] Kyu Ho Park, UFC Project Group* "UFC: A Ubiquitous Fashionable Computer", Next Generation PC 2005,October 2005.

[3] "http://core.kaist.ac.kr/UFC.html" KAIST Computer Engineering Research Laboratory, IT-839 Project Home Page.

[4] Willian Stallings, "Cryptography and Network Security Principles and Practices" 4th Edition Pearson Education.

[5] M. Satyanarayanan, Carnegie Mellon University and Intel Research Pittsburgh "A Catalyst for Mobile and Ubiquitous Computing" IEEE Pervasive Computing 2002, Feb 2002

[6] Marc Langheinrich, "A Privacy Awareness System for Ubiquitous Computing Environments", ACM CSCW 2002

[7] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks",Lixia Zhang International Conference on Network Protocols (ICNP) 2001

[8] Tung, B., et al., Public Key Cryptography for Initial Authentication in Kerberos, 2001: http://www.ietf.org/internet-drafts/draft-ietf-cat-kerberos-pk-init-12.txt.

[9] Harbitter, A. and Menasce, D. A. : "The performance of public key enabled Kerberos authentication in mobile computing applications", Proc. of the 8th ACM conference on Computer and Communications Security 2001,78-85.

[10] Fox, A. and Gribble, S. D. : "Security on the move: Indirect Authentication using Kerberos", Proc. of the Second Annual International Conference on Mobile Computing and Networking 1996, 155-164.

[11] Gregory D. Abowd and Elizabeth D. Mynatt, "Charting past, present and future research in ubiquitous computing. ACM Transactions on Computer-Human Interaction", Special issue on HCI in the new Millenium, 7 (1):29?58, March 2000.

[12] Stephen S. Intille, "Change Blind Information Display for Ubiquitous Computing Environments", Ubicomp 2000

[13] Ian F. Akyildiz and Shantidev Mohanty, "A Ubiquitous Mobile Communication Architecture for Next-Generation Heterogeneous Wireless Systems" IEEE Radio Communications 2005

[14] R. Morales-Salcedo, H. Ogata, and Y. Yano, "Using RFID and Dynamic metadata in an Educational Digital Library", IASTED International Conference on WEB-BASED EDUCATION, pp.323Security and privacy rights management for mobile and ubiquitous computing-331

[15] Wei Zhouand Christoph Meinel, "Implement role based access control with attribute certificates" Proceedings of the 6th International Conference on Advanced Communication Technology (ICACT2004)

[16] Mike Fraser, "Mobile and Ubiquitous Computing" COMSM0106 Mobile and Ubiquitous Computing

[17] The Open Group, "SSO Definition: http://www.opengroup.org"

[18] P. Horster, M, Michels, "Hidden signature schemes based on the descrete logarithm problem and related concepts", Proc. of Communications and Multimedia Security 1995 Chapman & Hall 1995

[19] Ki-Woong Park, Sang-Seok Lim and Kyu-Ho Park, "PANDA: An Interoperable Mobile Security Card for biquitous Services", KAIST CORE Lab. Technical Report,2006.

[20] Ki-Woong Park, H.-J. Choi, and K. H. Park, "An interoperable authentication system using zigbee-enabled tiny portable device and pki," in Internation Conference on Next Generation PC, October 2005.

[21] Nicolas T. Courtois, Louis Goubin1 and Jacques Patarin , "SFLASHv3, a fast asymmetric signature scheme, Proceedings of ASIACRYPT" 1998, LNCS n1514, Springer, 1998, pp. 35-49.

[22] Jintai Ding, Dieter Schmidt, "CRYPTANALYSIS OF $SFLASH^{v3}$" International Association for Cryptologic Research Technical Report, 2003.

[23] J. Patarin, L. Goubin, N. Courtois, "$C^{*\pm}$ and HM: Variations around two schemes of T. Matsumoto and H. Imai, in Advances in Cryptology", Proceedings of ASIACRYPT'98, LNCS n1514, Springer, 1998, pp. 35-49.

[24] Mehrdad Jalali-Sohi and Peter Ebinger, "Towards Efficient PKIs for Restricted Mobile Devices" International Conference on Communications and Computer Networks (CCN) 2002, Cambridge

[25] Asad Amir Pirzada and Chris McDonald, "Kerberos Assisted Authentication in Mobile Ad-hoc Networks" ACM International Conference Proceeding Series; Vol. 56 , 2004

[26] Shashi Kiran,Patricia Lareau,Steve Lloyd " PKI Basics - A Technical Perspective" PKI Forum 2002

[27] J. Linn "The Kerberos Version 5 GSS-API Mechanism" RFC1964, 1996

[28] Z. A. B. of Directors, "ZigBee Specification v1.0." ZigBee Alliance,2005.

[29] S. Tuecke, V. Welch "Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile" RFC3820, 2004

[30] Woo-Min Hwang, S. Lim, K. Park "A Context-Aware Replacement Page Cache for Wearable Computer" Next Generation PC, 2006

[31] ATMEL, "ATmega128(L) Datasheet: 8-bit Microcontroller with 128K Bytes In-System Programmable Flash." ATMEL., 2005.

TABLE II

NOTATIONS OF THE ENTITIES AND MESSAGES

Definition of the Entity Symbols
User: $Alice$ Delegation Server: $Delg$ Kerberos Authentication Server(AS): $Kerb\_AS$ Kerberos Ticket Grant Server(TGS): $Kerb\_TG$ Service Device: Bob
Definition of the Message Symbols
ID_x: ID of X PU_x: Public Key of X PR_x: Private Key of X $PU_{x,y}$: Delegated Public Key of X by Y $PR_{x,y}$: Delegated Private Key of X by Y $CR_{x,y}$: Proxy Certificate of X by signing of Y $Nonce$: Random number against replay attacks $K_{x,y}$: Symmetric key between X and Y $Delg_X$: Delegation process of X SN: Serial Number
Denotation
Example: $A \rightarrow B$: Message1 Message1 = Message2 || Message3.
A: Sender, B: Receiver Message1 contains two contexts that are Message2 and Message3.

[32] Infinion, "infinion SLE66 DataSheet. Infinion.", 2006.

APPENDIX

In this section we specify our proposed single sign-on protocol for implementation. Notations of the entities and messages to describe the proposed protocol are described in table II.

1) State1
Message1-1. $Bob \rightarrow Alice$:
ID_{Bob}||Bob Capsule
Bob Capsule $= SN||Hash(SN||Nonce_{Bob})$

2) State2
Message2-1. $Alice \rightarrow Delg$:
$E\{PU_{Delg}, DelgRequest\}$
$DelgRequest = ID_{Delg}||ID_{Alice}||K_{Alice,Delg}||$
$Nonce_{Alice}||E\{PR_{Alice}, K_{Alice,Delg\_A}||$
$E\{PU_{Referee}, K_{Repu\_Alice}||ID_{Alice}||ID_{Delg}\}\}$
Message2-2. $Delg \rightarrow Alice$:
$E\{K_{Alice,Delg\_A}, PU_{Alice,Delg\_A}||Nonce_{Alice}\}$
Message2-3. $Alice \rightarrow Delg$:
$E\{K_{Alice,Delg\_A}, CR_{Delg\_A,Alice}\}$
Message2-4. $Delg \rightarrow Referee$:
$E\{PU_{Referee}, K_{Delg\_A,Referee}||CR_{Delg\_A,Alice}||$
$Nonce_{Referee}||E\{PR_{Delg}, CR_{Delg\_A,Alice}||$
$E\{PU_{Referee}, K_{Repu\_Alice}||ID_{Alice}||ID_{Delg\_A}\}\}$
Message2-5. $Referee \rightarrow Delg$:
$E\{PU_{Delg}, Seq_{Alice}||Nonce_{Referee}\}$

3) State3
Message3-1. $Alice \rightarrow Delg$:
$ID_{Alice}||E\{K_{Alice,Delg\_A}, ID_{Alice}||ID_{Bob}||BobCapsule$
$||Subkey_{Alice,Bob}||E\{K_{Repu\_Alice}, BobCapsule\}\}$
Message3-2. $Delg \rightarrow Referee$:
$ID_{Delg\_A}||E\{K_{Delg\_A,Referee}, ID_{Bob}||Seq_{Alice}||$
$Nonce_{Referee}||BobCapsule||E\{PR_{Alice,Delg\_A},$
$E\{K_{Repu\_Alice}, BobCapsule\}\}\}$
Message3-3. $Referee \rightarrow Delg$:
$E\{K_{Delg\_A,Referee}, OKorBAD||Nonce_{Referee}\}$

4) State4
Message4-1. $Delg \rightarrow Kerb\_AS$:
$E\{PU_{Kerb}, TGT-REQ||CR_{Alice,Delg\_A}||$
$E\{PR_{Delg}, CR_{Alice,Delg\_A}||K_{Kerb\_AS,Delg\_A}\}\}$
TGT-REQ=Ticket Grant Ticket Request
$=$PA_Data$||ID_{Alice}||ID_{TGS}||Time||Nonce_{TGT}$
PA_Data=Pre-Authentication Data
$= E\{K_{Kerb\_AS,Delg\_A}$,Delegation System Time $\}$
Time=Key-Expire$||Time_{Auth}||Time_{Start}||Time_{End}$
Message4-2. $Kerb\_AS \rightarrow Delg$:
$ID_{Alice}||TGT_{Alice,TGS}||$
$E\{K_{Delg\_A}, K_{Kerb\_AS,Delg\_A}||Nonce_{TGT}||Times\}$
$TGT = ID_{Kerb\_TGS}||E\{K_{AS,TGS}, K_{Delg\_A,Kerb\_TGS}$
$||Times||ID_{Alice}\}$

5) State5
Message5-1.$Delg \rightarrow Kerb\_TGS : TGS_{Alice}$-$Req$
TGS_{Alice}-$Req = ID_{Bob}||Times||Nonce_{TGS}||$
$TGT_{Alice}||Authenticator(Alice, Kerb\_TGS)$
$TGT_{Alice} = ID_{Kerb\_TGS}||E\{K_{AS}, TGS_{Alice},$
$K_{Delg\_A,Kerb\_TGS}||Times||ID_{Delg\_A}\}$
$Authenticator(Alice, Kerb\_TGS) =$
$E\{K_{Delg\_A,Kerb\_TGS}||ID_{Alice}||K_{Delg\_A,Ser}\}$
Message5-2.$Kerb\_TGS \rightarrow Delg : TGS_{Alice}$-$Rep$
TGS_{Alice}-$Rep = ID_{Alice}||SGT_{Alice}||$
$E\{K_{Delg\_A,TGS}, K_{Delg\_A,Bob}||Nonce_{TGS}||Times||ID_{Bob}\}$
$SGT_{Alice} = ID_{Kerb\_TGS}||E\{K_{Delg\_A,Bob}||$
$ID_{Alice}||Times\}$

6) State6
Message6-1. $Delg \rightarrow Bob$:
$SGT_{Alice}||Nonce_{Bob}||E\{K_{Delg\_A,Bob}, ID_{Alice}||$
$Subkey_{Alice,Bob}||BobCapsule\}$
Message6-2. $Bob \rightarrow Delg$:
$E\{K_{Delg\_A,Bob}, Subkey_{Alice,Bob}||Nonce_{Bob}\}$
Message6-3. $Alice \rightarrow Bob$:
$ID_{Alice}||E\{Subkey_{Alice,Bob}, BobCapsule\}$