Achieving Secure and Flexible M-Services Through Tickets

Hua Wang, Yanchun Zhang, Jinli Cao, *Associate Member, IEEE*, and Vijay Varadharajan, *Senior Member, IEEE*

***Abstract*—Web services via wireless technologies, mobile services (M-services), HTTP, and XML have become important for conducting business. W3C XML Protocol Working Group has been developing standard techniques such as Web Services Description Language (WSDL), simple object access protocol (SOAP), universal description discovery and integration (UDDI). However, at this stage, there is no standard technique for access control in M-services.**

This paper describes a secure and flexible access control scheme and protocol for M-services based on role based access control (RBAC). The access control architecture involves a Trusted Credential Center (TCC), a Trusted Authentication and Registration Center (TARC) and a secure ticket based mechanism for service access. Users and service providers register with the TARC and are authenticated. Based on this, tickets are issued by the TCC to users. Tickets carry authorization information needed for the requested services. In particular, we are able to specify access control polices based on roles. The protocols between the various entities in the model are protected using appropriate security mechanisms such as signatures which are used to verify correctness of the requested service, as well as to direct billing information to the appropriate user. Our architecture supports efficient authentication of users and service providers over different domains and provides a secure access model for services to users. Our model is also able to support anonymity of users. Only the TARC is able to identify misbehaving users. We believe that the proposed architecture forms a good basis for achieving a secure and flexible M-service system.

***Index Terms*—Access control architecture, anonymity, RBAC, secure M-services, ticket based access control.**

I. Introduction

A WIRELESS Web mobile service (M-service) is a Web-based application that accepts requests from different systems on the Internet and can involve a range of wireless and Web technologies such as GSM [1], XML [2], SOAP [3], and WSDL [4]. Vendors and customers can provide and obtain services without being limited by the location of an M-service. As a result, security and privacy issues in M-service systems have become more critical, especially for mobile consumers (e.g. moving from one place to another, or using wireless mobile systems). A static access control is incompatible for such dynamic mobile environments. Consumers may access services across multiple service domains and it is necessary to develop efficient cross-domain authentication and access control that can involve roaming between domains. Cross-domain authentication itself can become complicated authentication activities when the roaming path is long and dynamic. This could limit the future of M-service applications.

Furthermore, there can be different types of M-services. In some cases, there can be specific binding relationships between the participants whereas in others such as shopping, any user may be able to access a service using say some form of electronic payment. Hence, there is a need to develop a secure access control scheme that is flexible enough to capture these specific bindings and take them into account while making decisions. Also, a user may wish to access multiple M-services over a period of time. This requires that the security management is both efficient and effective when users change from one service to another. There have been several proposals relating to M-service systems [1], [5]–[7]. Probably it is accurate to say that most of them lack the required flexibility in security management. For instance, the Excellent e-service [5] provides service via different channels and manages customer communication via e-mail, text chat, and fax in the same system. However, customers have to trust the system (e.g. with their credit card numbers) and there are no mechanisms for privilege management. Another M-service system Red hat is designed to provide enterprise-class Linux for enterprise-class servers and applications [6]. It supplies the source code of some productions and also requires the private information of customers for payment. The global system for mobile communication [1] provides mechanisms for user authentication as well as integrity and confidentiality, including protection of information exchanged between mobile terminals and fixed networks. It provides only limited privacy protection for users by hiding their real identities from eavesdroppers on the radio interface [7].

There are several security issues that need to be addressed in the design and management of M-services. In particular, we believe the following are critical to developing a secure and flexible access control architecture.

Trust and Security Model. It is essential that there is an explicit representation of trust model in a M-service system. This is a need to specify the trust relationships between the users and the service providers in the system. The trust model should be supported by security services and mechanisms that can help the participants to make appropriate decisions. For instance, based

Manuscript received January 15, 2003; revised August 15, 2003. This paper was recommended by Guest Editors Z. Maamar and B. Benatallah.

H. Wang is with the Department of Mathematics and Computing, University of Southern Queensland, Toowoomba, Qld. 4350, Australia (e-mail: wang@usq.edu.au).

Y. Zhang is with the School of Computer Science and Mathematics, Victoria University of Technology, Melbourne City, MC 8001, Australia (e-mail: yzhang@csm.vu.edu.au).

J. Cao is with the Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, Vic. 3086, Australia (e-mail: jinli@cs.latrobe.edu.au).

V. Varadharajan is with the Department of Computing, Macquarie University, NSW 2109, Sydney, Australia (e-mail: vijay@ics.mq.edu.au).

Digital Object Identifier 10.1109/TSMCA.2003.819917

on the trust model and the security services such as authentication and ticket based credentials, the service provider should be able to conclude that the claimed user is who s/he claims to be and that s/he has the privileges to access the requested service. Similarly, based on the trust model and security architecture, the users in the M-service can reliably identify that a service provider has correctly charged for the service that has been provided. At present, majority of M-service systems depend entirely on an implicit trust model whereby the users and providers trust each other completely. Such a scheme is not suitable for large scale systems with numerous service providers and users over multi-domains.

Flexibility. A basic characteristic of an M-service is that a service can be provided to a user anywhere and at anytime. This in turn requires an efficient cross-domain authentication and a flexible and effective access control privilege management. Mechanisms present in current M-service systems are not adequate to fulfill these requirements. Current solutions often rely on roaming agreements for cross-domain authentication. A user, when applying services in a foreign domain, authenticates himself to the foreign service provider. This process often assumes that the foreign service provider trusts the home domain agent of the user and is based on roaming agreements between various service providers. With the rapidly growing number of service providers, such schemes will no longer be practical. There will be a need to minimize the number of interactions between the home and foreign domains.

Furthermore, the issue of privilege management across multiple domains poses a number of additional challenges. First, there is a need to represent different types of privileges for different services in different domains. Our access control architecture is able to support a range of access policies including role based access control (RBAC). RBAC has gained much popularity in access control, though the idea of partitioning privileges in terms of specific job functions and roles have been well known for several decades. Second, there is a need to verify the correctness and the validity of the privileges at the time of access to a service. This can be done by the service provider itself or some entity that is trusted by the service provider. The security management model should be flexible enough to specify a range of privileges and to evaluate them to make appropriate decisions in an efficient manner.

Efficiency. Users are susceptible to being disconnected and loosing data while accessing services in wireless environments. Also a user may need to connect to multiple services and change M-service systems to access different kinds of services on the Internet. The new scheme should provide a scalable solution for different kinds of M-services and use bandwidth efficiently, especially in wireless networks.

In this paper, we propose a security architecture for M-services addressing some of the aspects mentioned above. Our architecture involves a Trusted Credential Center (TCC), a Trusted Authentication and Registration Center (TARC) (via UDDI) and a secure ticket based mechanism for service access. Users and service providers register with the TARC and are authenticated. Services are described in the TCC and service provider by WSDL. Based on authentication, tickets are issued by the TCC to the users. Tickets carry authorization information needed for the requested services. In particular, we are able to specify access control polices based on roles. Tickets are transferred using SOAP. The protocols between the various entities in the model are protected using appropriate security mechanisms which are used to verify correctness of the requested service, the validity of the privileges, provision of service as well as secure charging users for service usage. Our architecture supports efficient authentication of users and service providers over different domains and provides a secure access model for services to users.

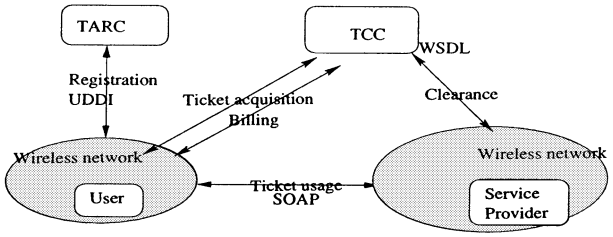

Fig. 1. M-service model.

The billing information is stored in the TCC for the users to view and access after service provision. There are lots of issues associated with this [8]–[10]. This paper will focus on the access control for M-services.

The main stages involved in the architecture are illustrated in Fig. 1.

This paper is organized as follows. In Section II, basic definitions and ticket types are introduced. The basic definitions include RSA, multisignature, and RBAC. There are four different kinds of tickets, they are tickets t_0, t_1, t_2, and t_3. A single signature scheme for tickets t_1, t_2 is presented in Section III and the extension of the single signature scheme to a multisignature scheme for ticket t_3 is discussed in Section IV. The security of the proposed solution with role based access control and the logical proof of the solution as well as its deployment in wireless environments are analyzed in Section V. The usage of the scheme and related works are discussed in Section VI. Finally, conclusions are presented in Section VII.

II. Security Primitives in the Proposed Architecture

A. Basic Definitions

To facilitate discussion, the following is a brief review of some well-known primitive cryptographic terminologies that will be used in this paper.

Hash function, $h(x)$ is a hash function. For a given Y it is computationally hard to find an x such that $h(x) = Y$, where x might be a vector.

Hash functions have been used in computer science for a long time. They are used in several cryptographic components including pseudo-random generators [11], digital signatures and message authentication [12].

RSA is a public key cryptosystem that offers both encryption and digital signatures (authentication) [13]. RSA works as follows: taking two large primes p and q, and computing their product $n = pq$; n is called the modulus. Choosing a number e, less than n and relatively prime to $(p-1)(q-1)$. Finding another number d such that $(ed-1)$ is divisible by $(p-1)(q-1)$.

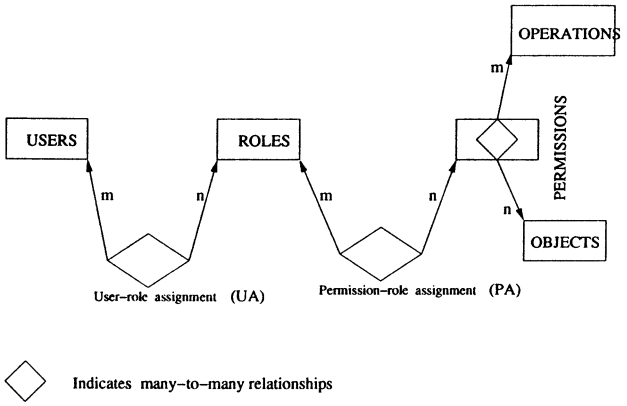

Fig. 2. RBAC relationship.

The public key is the pair (n, e), the private key is d. The factors p and q may be kept with the private key or destroyed.

It is currently difficult to obtain the private key d from the public key (n, e) for a large n (say some 2048 bit integer). RSA is often used in many different applications such as email, electronic funds transfers and remote access.

Multisignatures imply that several signatures are signed on the same document. There are at least two ways to implement multisignature. One method involves each person signing separately and the document being passed on from one person to another. Another scheme involves a message being signed simultaneously [14].

Role Based Access Control (RBAC), in general an RBAC system involves users being associated with roles and the roles being associated with permissions. A permission typically specifies target objects and the operations that can be performed on them. There are many relationships between users and roles, and between roles and permissions as shown in Fig. 2. In terms of administration, there is the management of user to role mapping (e.g. assigning users to specific roles) and the management of role to permission mapping (the privileges associated with the roles) [15]. In general the authorities that manage these two mappings can be different; furthermore the strategies for managing these mappings such as when should elements in these mappings change also varies.

In 1993, the National Institute of Standards and Technology (NIST) developed prototype implementations, sponsored external research [16], and published formal RBAC models [17]. Since then, many RBAC practical applications have been implemented [18], [19] because RBAC has many advantages such as improving the security of systems and reducing administration cost etc. However, there has been little research done on the usage of RBAC in M-service management [20].

Typically, an RBAC model has two components, MC_0 and MC_1. Model component MC_0, called the RBAC authorization database model, defines the RBAC security properties for authorization of static roles. Static properties of a RBAC authorization database include role hierarchy, inheritance, cardinality, and static separation of duty. MC_1, called the RBAC activation model, defines the RBAC security properties for dynamic activation of roles. Dynamic properties include role activation, permission execution, dynamic separation of duties, and object access. In particular, the RBAC model supports the specification of the following:

a) user/role associations; the constraints specifying user authorizations to perform roles;
b) role hierarchies; the constraints specifying which role may inherit all of the permissions of another role;
c) duty separation constraints; these are role/role associations indicating conflict of interest:
   1. Static separated duty (SSD); a constraint specifying that a user cannot be authorized for two different roles,
   2. Dynamic separated duty (DSD); a constraint specifying that a user can be authorized for two different roles but cannot act simultaneously in both,
d) Cardinality; the maximum number of users allowed, i.e. how many users can be authorized for any particular role (role cardinality), e.g., only one manager.

B. Ticket Types

There are four participants (the user, the service provider, the TARC, and the TCC) and several protocols for ticket acquisition, ticket usage, clearance, and billing in the M-service model. The user obtains tickets by running the ticket acquisition protocol. These tickets are used to access services. The user presents an appropriate ticket to the service provider who can verify the validity of the ticket. If the verification of the ticket is successful, then the service provider provides the service to the user according to the conditions on the ticket. Based on the received tickets, the TCC prepares a charging bill for each user. The exact forms of the clearance (payment to the service provider) and billing (payment to the TCC) protocols are not specified in our model. Readers may refer to [8] for details.

There are several advantages in using tickets for accessing services [21]:

Trust: Tickets may include all required information about services and service providers etc. Users can buy and use the tickets to obtain appropriate services provided by service providers. There is no contractual relationships between users and service providers.

User Privacy: Tickets can be designed to operate as pure capabilities. The service provider can make its decision whether to grant a service or not without identifying the real identities of the users.

Scalability: If tickets are designed in such a way that the information in the tickets can be verified without performing cross domain authentication in real time, then this scheme can be scalable to large systems.

Granularity of Access Control: In principle, a ticket can be used to capture a range of access control policies such as role based access and privilege based access. So one can support a fine level of access control in terms of privileges and operations and a coarse level in terms of roles.

Delegation: In general, it is possible to use the ticket mechanism to transfer privileges to other entities. If the possession of a ticket gives access then delegation can be achieved by controlling the propagation of tickets.

However a number of security issues arise such as how to prevent illegal duplication and forgery of tickets [22].

TABLE I
TICKET TYPES

Types	t_0	t_1	t_2	t_3
user	-	-	+	+
provider	-	+	-	+

Duplication. There are two types of duplication that need to be considered. The first type is that users either uses or transfers a ticket many times (similar to double spending in electronic cash systems). The second type is an eavesdropper, who listens to someone else acquiring a ticket and makes a copy for himself.

Forgery. Forgery refers to the illegal construction of a valid ticket, which can be used for accessing resources.

Modification. Users must not modify tickets. This is to prevent users from accessing resources for which the tickets have no permission, e.g. a ticket that allows travel by bus, should not be modifiable to allow travel by plane.

A ticket may bind a given user and a given service provider together. For example, a movie ticket, which usually does not specify who can use it (i.e., the user) or a travel card, which may not restrict the means of transport (i.e., the service). Based on this observation, there are four types of tickets. These are illustrated in Table I, where "+" means that the corresponding entity, user or service provider is bound by the ticket, while "−" means that it is not.

A ticket of type t_0, for instance, does not restrict the service for which it can be used, the service provider who can accept, or the user who can use it. This is very much like cash in real life. The other extreme is a ticket of type t_3, which can only be used by a given user with a given service provider. An example of this type is a flight ticket.

As shown in Table I, tickets t_1 and t_2 have only one entity bounded and ticket t_3 has two entities bounded. We will design different mechanisms related to each of the tickets. In some cases users may wish to remain anonymous during purchase and not private information of the user should be shown to service providers. Use of a ticket-based system can avoid roaming multiple service domains. A single signature can be used in tickets with only one bound entity (users or service providers). As a signer, the bound entity uses a signature to authenticate a ticket. To cope with the cases of two bound entities, it is extended to $v (v = 2)$ signers (multisignature). This means that a user can get a service if all v entities agree. The v signers case can also associate with other services provided by many cooperative providers since the number v is not limited to 2. A credential_role in the TCC is set up to issue tickets and control the user's charging bill, and a trusted_role in the TARC is also set up to judge conflicts. Each user's statement of account can be seen clearly in the TCC.

In the remaining sections, we will present schemes for the different types of tickets and discuss how the TCC issues a continuously updated account statement for users. We will also consider both single and multisignature schemes. We will not discuss the ticket t_0 further since it does not bind any entities.

III. SINGLE SIGNATURE SCHEME FOR TICKETS t_1 AND t_2

In this section, we consider a single signature scheme for tickets t_1 and t_2. There are four roles in the single signature scheme, signer, verifier, credential_role and trusted_role. Depending on tickets, the signer can be a user or service provider that signs a ticket. The verifier might be a user or service provider that verifies the signature of the signer. The credential_role in the TCC will issue tickets as well as provides information for the verifier to check the signature. Whether the signature is valid or not depends on the information. The trusted_role is a judge to solve the conflict between users and service providers. This is because only the trusted_role has the secret key of the system and can trace users and service providers. Each signer has a different but fixed identity I, which is validated once the signer is registered in the TARC and does not include any private message of the signer. Ticket t_2, for instance, is bound to a user only. A user can follow this scheme to sign a signature as a ticket, the service provider verifies it and then sends some information to the credential_role and asks for payment. Ticket t_1 is similar to ticket t_2, the signers are service providers but not users.

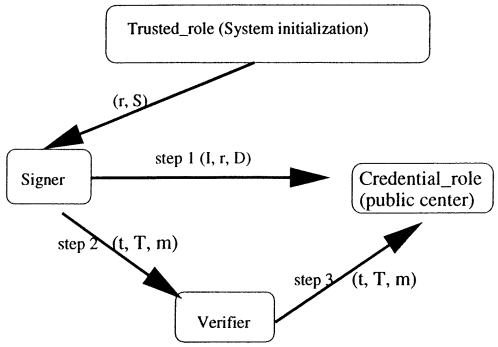

Fig. 3. Single signature scheme for tickets t_1 and t_2.

The outline of the scheme is shown in Fig. 3. In the system initialization, with SOAP methods, the trusted_role sends the private messages (r, S) to the signer when the signer I is set up, where r, S are computed by the trusted_role, r will be used in the first verification by the credential_role and S will be used as the first signature key by the signer. In the second step, the credential_role verifies if the data (I, r, D) sent by the signer are valid or not, where D is used in the ticket verification. The data (I, D) will be put on a public directory in the TCC if the data are valid. At this time, the signer can complete a message signing job.

While the signer signs a message m, the signer will send the signed message (t, T, m) as a ticket to the verifier, and the latter checks if it is true or not, where t and T are computed by the signer and m may include some service information and conditions, etc. The verifier cannot verify the message when the data (I, D) in the TCC are not correct. Then the credential_role can control the usage of the ticket, and even find who the signer is if it contacts the trusted_role. In the final step, the verifier sends a message which includes the ticket to the TCC while the ticket is true. The latter will update the data (I, D) that is used to issue a charging bill. The data (I, D) is changed while the ticket is used and the ticket is invalid if the verifier cannot get the correct data (I, D). Thus, the ticket cannot be used twice and the user can see a clear statement.

A. System Initialization

There are two elements in a signature scheme: one is the signer represented by consumers (users) or service providers; the other is the verifier. As a ticket, a signature is valid only if its verification is correct.

The trusted_role computes a public composite modulus $n = pq$ where factors are strong primes. The Trusted_role chooses also prime exponents e and d such that:

$$e * d = 1 \ (mod\, \phi(n))$$

where $\phi(n) = (p-1)(q-1)$. The pair (n, e) is made public, and d is kept secret by the TARC as the system key. The trusted_role computes when the signer with identity I signs up:

$$r = k^e \ (mod\, n), \quad S = k * I \ (mod\, n)$$

where $k \in_R Z_n$ ($a \in_R A$ means that the element a is selected randomly from the set A with uniform distribution). Then

$$S^e = r * I^e \ (mod\, n).$$

Let $D = S^e (mod\, n)$. The trusted_role secretly sends (r, S) to the signer whose public identity is I. S will be used as the first signature key to issue a ticket. Obviously, it is hard to compute S from D without system key d under the RSA assumption.

The signer with the public key I sends (I, r, D) to the credential_role, and the latter verifies the following equation:

$$D = r * I^e \ (mod\, n).$$

The data (I, r, D) are valid when the equation is successful, in which r and D are computed by the trusted_role; otherwise the (I, r, D) are invalid. The credential_role publishes in a public directory the pair (I, D) for the signer with the public key I. The initialization processes of the system are shown in Fig. 4.

B. Single Signature Scheme

The verifier can access the public values n, e and the public pair (I, D) registered in the TCC. The data D in the TCC must be correct; otherwise the signed message (the ticket) cannot be verified by the verifier.

To express the general process of the single signature scheme, it is assumed that the messages $m_1, m_2, \ldots, m_{l-1} (l \geq 1)$ have already been signed by the signer I. The messages $m_1, m_2, \ldots, m_{l-1} (l \geq 1)$ can indicate different service requirements that are included in tickets. A user can get a valid ticket if the signature is correct. The corresponding public key $(I, D_{l-1})(D_0 = D)$ of the signer is now registered in the public directory of the TCC. The message m_l for the next service will be signed by the signer using the secret key $S_{l-1} (S_0 = S)$. The signer and the verifier perform the following steps.

Input: (I, D_{l-1}, e, n),

Signer:

1) Picks $r_{l-1} \in_R Z_n$ and computes: $T_{l-1} = r_{l-1}^e (mod\, n)$.
2) Computes: $S_l = S_{l-1} * m_l (mod\, n)$, S_l will be used as the secret key by the signer I in the next signing operation.

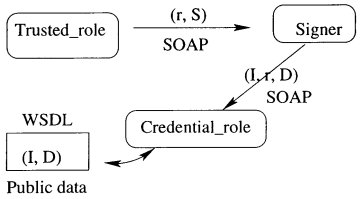

Fig. 4. Initialization for group_1.

3) Computes the Hashing value $d_{l-1} = h(T_{l-1}, m_l)(mod\, n)$.
4) Computes the final witness $t_{l-1} = r_{l-1} * (S_{l-1} * m_l)^{-d_{l-1}} (mod\, n)$.

Note: A ticket is the signature (t_{l-1}, T_{l-1}, m_l). The ticket will be recorded at the TCC; the user will send the ticket to a service provider when s/he needs the service.

Credential role:

The credential_role computes D_l for the ticket, where

$$D_l = D_{l-1} * m_l^e \ (mod\, n) = S_l^e \ (mod\, n).$$

D_l is published in the TCC. It will be used to verify the ticket by the verifier and used to issue another ticket.

Verifier:

5) The verifier gets (t_{l-1}, T_{l-1}, m_l) and knows (I, D_{l-1}), then checks that:

$$d_{l-1} = h\left(t_{l-1}^e * D_{l-1}^{d_{l-1}} * m_l^{ed_{l-1}} (mod\, n), m_l\right) (mod\, n).$$

It is easy to see that if the signer follows the protocol, the equation will be valid.

$$\begin{aligned} d_{l-1} &= h(T_{l-1}, m_l) \ (mod\, n). \\ T_{l-1} &= r_{l-1}^e \ (mod\, n) \\ &= \left(t_{l-1} * (S_{l-1} * m_l)^{d_{l-1}}\right)^e (mod\, n) \\ &= \left(t_{l-1}^e * D_{l-1}^{d_{l-1}} * m_l^{ed_{l-1}}\right) \ (mod\, n). \end{aligned}$$

Using this protocol the verifier is convinced with overwhelming probability that the signer knows the secret key S_{l-1}. This S_{l-1} is used but not revealed at the end of the protocol.

6) The verifier sends the ticket to the Credential_role. The latter updates (I, D_{l-1}) in the public director and takes a record. The ticket (t_{l-1}, T_{l-1}, m_l) cannot be used twice since it has been marked by the credential_role. ◇

Remark: The verifier must use the public data D_{l-1} in the TCC when it checks whether the signed message is true or not. The signed message will be unavailable if the data D_{l-1} are changed, then the credential_role can revoke the anonymity of the signer.

However, this scheme only suits the tickets t_1 and t_2. The problem of ticket t_3 cannot be solved with this scheme. A multisignature scheme to solve the problem is explained in the next section.

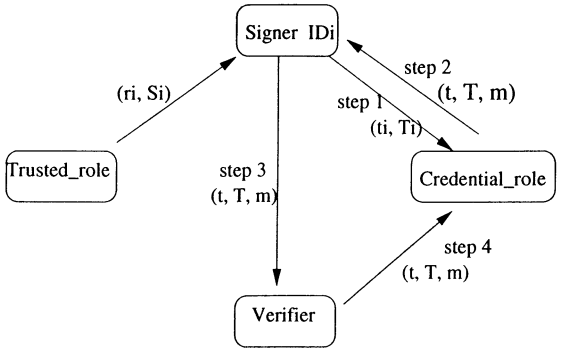

Fig. 5. Multisignature scheme for ticket t_3, using SOAP to transfer data.

IV. Multisignature Scheme for Ticket t_3

We will extend the scheme to a multisignature scheme for tickets of type t_3. Now the number of signers is not limited to two, instead we have v signers. This means that the scheme can also be used when services are provided by many cooperative providers.

Now, instead of the public key I of a signer (in the last section), we use $ID_i (i = 1, 2, \ldots v)$ as a public keys for signers U_i since there are more than one signers in a multisignature. At the beginning of the multisignature scheme, the trusted_role computes and secretly sends the messages (r_i, S_i) to signers U_i in the group when the signers are set up. This step is same as the first step in the last section. In the second step, the credential_role verifies if the data (ID_i, r_i, D_i) sent by the signers are valid or not. A vector $(ID_1, ID_2, \ldots, ID_v, g_1)$, as the group public key, will be put in the TCC, where g_1 is computed by the credential_role and will be used in the first ticket verification, then the group can sign.

In the signature process, the credential_role gets v pairs of data (t_{il}, T_{il}) from the signers with identity $ID_i (1 <= i <= v)$ when a message m is signed, where (t_{il}, T_{il}) are computed by the signer ID_i. In the next step, the credential_role sends the signed message

$$\left(t_l = \prod_{il=1}^{v} t_{il} \,(mod\, n), \quad T_l = \prod_{il=1}^{v} T_{il} \,(mod\, n), m \right)$$

to the signer as a ticket, where n is a public integer defined in the system initialization. The ticket will be sent to the verifier and the verifier checks if it is true or not. The verifier may not verify if the data g_1 in the TCC is not correct, and the signed message is invalid. Therefore the TCC can revoke the anonymity of the signers. In the final step, the verifier sends the ticket to the TCC and then the credential_role can make a record for the ticket. This process is shown in Fig. 5.

Suppose there are v signers $U_1, U_2, \ldots, U_v$ in the signature system to sign a message simultaneously, for ticket t_3, two signers are enough. Therefore the scheme can also cope with some other cases for example some services provided by many providers.

A. Scheme Initialization

Similar to the previous section, the pair (n, e) is made public, and d is kept secret by the TARC as the system key. The signer

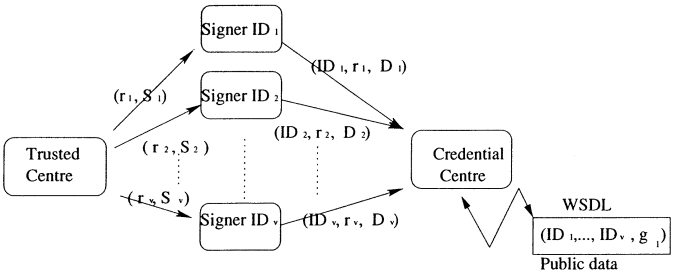

Fig. 6. Initialization of Multisignature scheme, SOAP technologies can be used in data transmission.

U_i of the system has a public key ID_i which is produced by the TCC when the signer joins the system. The trusted_role computes:

$$r_i = k_i^e \,(mod\, n), \quad S_i = k_i * ID_i \,(mod\, n)$$

$k_i \in_R Z_n$, then $S_i^e = r_i * ID_i^e \,(mod\, n)$. Let $D_i = S_i^e \,(mod\, n)$, the trusted_role secretly sends (r_i, S_i) to the signer with the public key ID_i. S_i will be used by U_i as the first signature key. It is hard to compute S_i from ID_i without the system key d under the RSA assumption.

The signer U_i sends (ID_i, r_i, D_i) to the credential_role, and the latter verifies the following:

$$D_i = r_i * ID_i^e \,(mod\, n). \tag{1}$$

The data (ID_i, r_i, D_i) are valid if the (1) is successful, which means all v signers agree to issue a ticket. Otherwise the data (ID_i, r_i, D_i) are invalid. While the equation is successful for $i = 1, 2, \ldots, v$, the credential_role computes a system public key:

$$g_1 = \prod_{i=1}^{v} D_i \,(mod\, n) = \prod_{i=1}^{v} S_i^e \,(mod\, n).$$

The credential_role registers in a public directory a vector $(ID_1, ID_2, \ldots, ID_v, g_1)$ for signers $U_1, U_2, \ldots, U_v$. The data g_1 will be used and changed when a valid signature is made. The processes are shown in Fig. 6.

B. Multisignature Scheme

When the verifier accesses the system public key n, e and the public vector $(ID_1, ID_2, \ldots, ID_v, g_1)$ in the TCC, the data g_1 must be correct, otherwise the signature is unavailable since the verifier cannot verify the signed message.

Assuming that a message $m_l (l = 1, 2, 3, \ldots)$ including service information and users requirements will be signed by the signers $U_1, U_2, \ldots, U_v$. S_{il-1}, the secret key of signer U_i is changed when the message m_l has been signed ($(i = 1, 2, \ldots, v)$ and $S_{i0} = S_i$). This means S_{il-1} is a once-a-time secret key and it will improve the security of the system. z is a public prime number which is known to v signers and it will be used in the new multi-signature scheme. The processes of the multi-signature scheme are below.

Input: (ID_i, D_i, e, n),
Signer U_i:
Step 1.

1.1 Picks $r_{il} \in_R Z_n$ and computes: $T_{il} = r_{il}^e (mod\, n)$.

1.2 Computes: $S_{il} = S_{il-1} * m_l \ (mod\, n)$.

S_{il} will be used as the secret key by U_i in the next signing operation.

1.3 Computes: $t_{il} = r_{il} * (S_{il-1} * m_l)^z \ (mod\, n)$.

1.4 Sends the pair (t_{il}, T_{il}) to the credential_role.

The credential_role can now produce a ticket but it is not able to get the secret key S_{il-1} from the data (t_{il}, T_{il}).

Credential_role:

Step 2. The credential_role computes

$$g_{l+1} = g_l * m_l^{ve} \ (mod\, n).$$

and

$$t_l = \prod_{il=1}^{v} t_{il} \ (mod\, n), \quad T_l = \prod_{il=1}^{v} T_{il} \ (mod\, n)$$

g_{l+1} is published in the public directory, it will be required to issue another ticket. (t_l, T_l, m_l) is a ticket which will be used for requesting services.

It should be noted when using ticket t_3, both the user and the service provider are signers, however, the ticket (t_l, T_l, m_l) is only sent by the credential_role to the user. The user will send the ticket to a service provider to ask for a purchase. The service provider, as a verifier, will verify the ticket. The verifier will follow the next steps when the ticket is received.

Verifier:

Step 3. The verifier knows the public data $(ID_1, ID_2, \ldots, ID_v, g_l)$ in the TCC and data (t_l, T_l, m_l), checks that:

$$T_l = t_l^e * g_l^{-z} * m_l^{-zve} \ (mod\, n) \tag{2}$$

It is easy to see that if the signer and the credential_role follow the steps, the (2) will be valid:

$$\begin{aligned} T_l &= \prod_{il=1}^{v} T_{il} \ (mod\, n) \\ &= \prod_{il=1}^{v} t_{il}^e * (S_{i\,l-1} * m_l)^{-ze} \ (mod\, n) \\ &= t_1^e * g_1^{-z} * m_l^{-zve} \ (mod\, n). \end{aligned}$$

Step 4. The verifier sends the ticket to the TCC. The latter will update the data g_l and prepare a charging bill for the user.

The signed message in the multisignature scheme will be invalid if the data g_l is changed. Then the credential_role can revoke the ability to sign messages of the signers.

V. System Security

A. Threat Analysis

First we analyze the threats to the system, including threats from outsiders and consider how to address the security problems of duplication, forgery and modification. Recall that there are four roles in the scheme. They are the signer, the verifier, the credential_role and the trusted_role.

Outsider Threat: An outsider knows the public data (I, D_l) and $(ID_1, \ldots, ID_v, g_l)$. It is hard to compute the secret key S_l from D and S_{il} from g_l without system key d under the RSA assumption.

Verifier: knows (I, D_l) and ticket (t_{l-1}, T_{l-1}, m_l) in the first sub-scheme and $(ID_1, \ldots, ID_v, g_l)$ and ticket (t_l, T_l, m_l) in the second subscheme. But no useful message can be obtained from these public data. The verifier knows no more information about the key than the outsider.

Credential_role: can revoke the anonymity of the users since it can control the ability to sign messages by the signers. It knows only as much as the outsider does and hence it cannot get the secret key either.

Signer: knows the secret key S_l of tickets t_1, t_2, but cannot use the secret key S_l and the ticket twice. Using the secret key S_l a second time, to produce another ticket, requires a second verification. If the previous verifier was honest, the public data in the TCC would be updated and the second ticket would be rejected. There are similar cases for the signers in the second subscheme.

Trusted_role: knows the system key d, and can get the signer's key S_l. So the TARC must be trusted. Here the trusted_role can be a judge.

The secret keys S_l and S_{il} are not revealed at the end of the process and no secret information is revealed during the running of the system. They are only dependent on the trusted_role, and does not depend on the credential_role. The security is also improved since the secret keys are changed once a message is signed.

Duplication is prevented since using a ticket twice requires that the ticket be verified twice, and the second verification cannot succeed as the data in TCC are changed after the first verification. In the multi-signature scheme, for instance, the TCC issues tickets and sends them to users. The other four, even the trusted_role, cannot forge a ticket because the messages of (t_{il}, T_{il}) are only sent to the TCC which is not able to get the secret key S_{il-1} from the data. To protect from eavesdropping or sending the ticket to other users, SOAP technology can be used between users and the TCC. The user cannot modify the service information since it is needed in the ticket verification.

There is no limitation on the service providers with our approach. Hence this scheme can be used by wireless service providers. The PKI technologies [23] could be used in the processing of the scheme. For example, in the initialization of the system for the tickets t_1, t_2, the trusted_role may use PKI approach to secretly send (r, S) to a signer.

The data transferred in current wireless environments is prone to loss. The ticket scheme can preserve the integrity of exchanged data in the lossy wireless environments. This means either the system can find the lost data or users cannot obtain services. For instance, tickets t_1, t_2 need to be sent to the TCC and the verifier. Tickets are invalid if data is lost in these two processes. When this occurs, users have to send tickets again until they are received. The verifier will send tickets to the credential_role. The system can find the lost data when they are missing, and users can still get services since tickets are valid through verifications. Users may use tickets twice since the data (I, D_{l-1}) in the TCC are not updated in time. However, the system will double charge the users if it receives the same ticket twice.

B. Logical Proof for Security

We have demonstrated that the secret key of a ticket does not appear during a business process. This section examines

the ticket issue and use stages using logical proof [24]. Logical proof allows analysis of trust between principals involved in authentication. The ticket approach is about access control and not about authentication, hence, the analysis is different from analysis of authentication protocols. The goal is not to prove an indentity of principals but to prove that a user has been granted privilege to access a service. The use of logical proof requires the transformation of the ticket approach into an idealised protocol and then refinement until the required trust is obtained. We do not repeat the ticket approach and do not show the logical proof for tickets t_2 and t_3 but for ticket t_1, because the logical proofs are similar. The idealized protocol associated to ticket t_1 for logical proof is as follows:

M1: $TARC \rightarrow User : (r, s)$;
M2: $User \rightarrow TCC : (I, r, D)$;
M3: $User \rightarrow TCC : (t_{l-1}, T_{l-1}, m_l)$;
M4: $User \rightarrow service\ provider : (t_{l-1}, T_{l-1}, m_l)$;
M5: $TCC \rightarrow service\ provider : (I, D_{l-1})$;
M6: $service\ Pprovider \rightarrow TCC : (I, D_l)$.

We have the following assumptions with PKI technlogy where K_{UT}, K_{TS} and K_{US} means shared keys between User and TCC, TCC and service provider, User and service provider respectively.

A1: User $\rightarrow$ TCC: User $\overset{K_{UT}}{\longleftrightarrow}$ TCC;
A2: TCC believes

$$\text{TCC} \quad \overset{K_{TS}}{\longleftrightarrow} \quad \text{Service Provider}\, x;$$

A3: $D_l = D_{l-1} * m_l^e \ (mod\ n)$;
A4: User believes;

$$\text{User} \quad \overset{K_{US}}{\longleftrightarrow} \quad \text{Service Provider};$$

A5: TCC believes D_l is fresh where $l = 0, 1, \ldots$;
A6: TCC believes m_l is fresh;
A7:Service provider believes m_l is fresh.

To verify the approach we need to reach the following states:

P1: TCC believes User believes (I, r, D);
P2: TCC believes User believes (t_{l-1}, T_{l-1}, m_l);
P3: Service provider believes (t_{l-1}, T_{l-1}, m_l);
P4: TCC believes service provider believes (I, D_l);

*P*1 relates to the end of the system initialization phase and the other three relate to the ticket use process. We prove these states as follows:

Proof: After *M*2 and *M*3:
R1: TCC sees (I, r, D) and (t_{l-1}, T_{l-1}, m_l)
From *R*1, *A*1, and the *message meaning rule* [24];
R2: TCC believes the User once said (I, r, D) and (t_{l-1}, T_{l-1}, m_l);
Since TCC believes D, m_l are fresh (*A*5, *A*6) and using the *nonce verification rule* [24];
R3: TCC believes user believes (I, r, D) and (t_{l-1}, T_{l-1}, m_l);
(*P*1) ans (*P*2) have been proved.
After *M*4;
R4: Service provider sees (t_{l-1}, T_{l-1}, m_l)
From *R*4 and *A*4 and the *message meaning rule*;
R5: Service provider believes User once said (t_{l-1}, T_{l-1}, m_l)
Since User believes m_l is fresh and the *nonce verification rule*;
R6: Service provider believes user believes (t_{l-1}, T_{l-1}, m_l)
From *M*4;
R7: Service provider believes user controls (t_{l-1}, T_{l-1}, m_l)
From the *jurisdiction rule* [24], *R*6 and R_7;
R8: Service provider believes (t_{l-1}, T_{l-1}, m_l)
(*P*3) has been proved.
After *M*6
TCC sees (I, D_l),
From *A*2 and the *message meaning rule*;
R9: TCC believes service provider said (I, D_l)
Using the *nonce verification rule*, *A*3 and *A*7;
R10: TCC believes service provider believes (I, D_l)
*P*4 has been proved. ■.

C. RBAC Management

In this subsection, we analyze the security of the system from the management viewpoint, and then we discuss how to use RBAC for the management of the M-service system.

With RBAC, users cannot associate with permissions directly. Permissions must be authorized for roles, and roles must be authorized for users. In RBAC administration, two different types of associations must be managed, i.e. associations between users and roles, and associations between roles and permissions. When a user's job changes, only the user/role associations change. If the job is represented by a single role, then when a user's job changes, only two user/role associations need to be changed: remove the association between the user and the user's current role, and add an association between the user and the user's new role.

As we mentioned before, relationships of roles such as Role hierarchies, SSD, and DSD have to be decided when a system is designed. Some relationships like cardinality, the maximum number of users etc, can be decided when the system is in operation.

Now we consider the RBAC management of the new payment scheme. There are four major roles in the system, the TARC, the user, the TCC and the service provider. The TARC, the TCC and the service provider, should be companies comprising many participants. We will not discuss the relationships of all the participants in these companies since they are beyond this paper. We will only consider a manager role in the TCC and the service provider separately. In Fig. 7, there are some dotted lines from the TARC, TCC, and service provider to the staff since these components are staff members. Therefore they are senior to the role of staff. In TCC, the work of an operator can be decided by the manager, so the manager inherits the operator, e.g. $operator \prec manager$. The relationship of the operator and the manager in the service provider is similar to that in the TCC.

The TARC, TCC, and service provider have DSD relationships with the user. This is because everyone in these three companies can be a user, but cannot act at the same time. The staff in these three companies must first log out if they want to register as users. For example, a person, who is a staff member of the TARC, can not work as a staff member and be a user in the same time. This is because a staff member in the TARC knows the private information of service providers. The people in the TCC and the service provider need to verify tickets, for instance, ticket t_2, the TCC

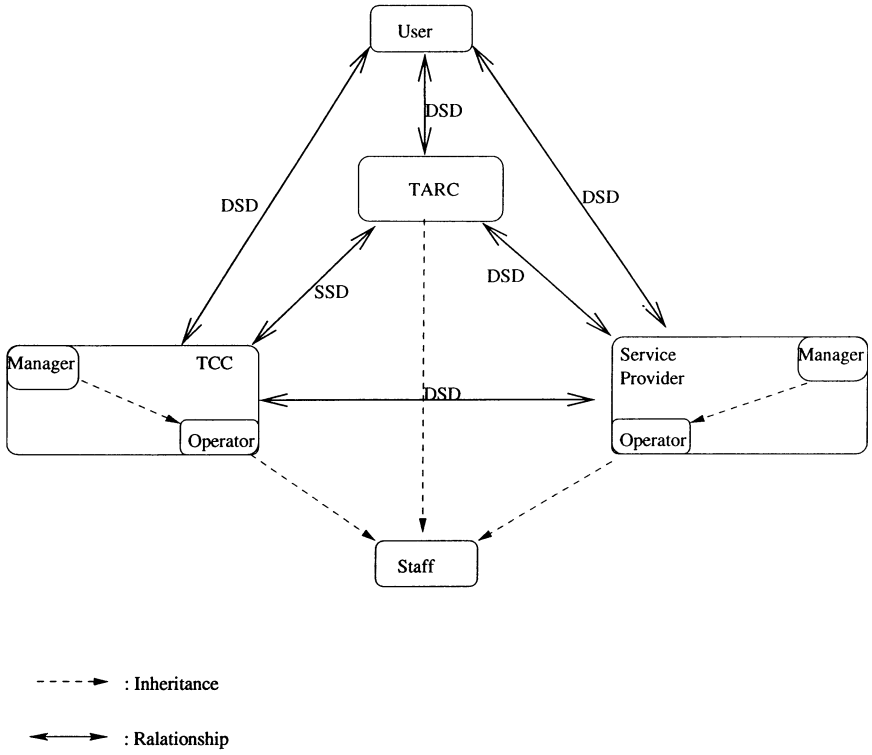

Fig. 7. Relationships of the roles in the M-service system.

computes D_l and the service provider has to check whether $d_{l-1} = h(t_{l-1}^e * D_{l-1}^{d_{l-1}} * m_l^{ed_{l-1}}(mod\, n), m_l)(mod\, n)$ or not. These tasks cannot be done by one person.

The TARC has an SSD relationship with the TCC. This is because the TARC knows the system key d that can be used to trace the personal message of users and the duties of the TCC are to issue tickets for users, to verify tickets and to prepare bills. The latter never needs to know any private information of users. The TARC has an DSD relationship with the service provider for a similar reason to the relationship between the TARC and the users.

The TCC has DSD relationship with the service provider. The service provider knows a ticket (t_{l-1}, T_{l-1}, m_l) that will be sent to the TCC. The latter will verify the ticket and update data (I, D_{l-1}). Based on the verification, the TCC will send a bill to users. Users may receive a wrong bill if a person is authorized by both of the TCC and the service provider at the same time.

VI. Analysis of Ticket Usage and Related Works

A. Ticket Usage and An Example

First let us consider the tickets t_1 and t_2. Since the signature of t_1 and t_2 are similar, we will only consider ticket t_2.

Let us suppose that users, service providers are registered by UDDI in the TCC. A ticket will be obtained by a user who requests the service in the ticket. When requiring a service, the user goes to the TCC for a ticket. The credential_role will use SOAP to send a message m_l including the service information, current time and user's requirements to the user. As a signer, the user signs the message and makes a ticket (t_{l-1}, T_{l-1}, m_l). The ticket (t_{l-1}, T_{l-1}, m_l) is acceptable to the service provider. As a verifier, the service provider verifies if the ticket is valid or not, using the data (I, D_{l-1}) in the TCC. Neither the service provider nor the credential_role knows who the user is. Only the trusted_role can trace the user from the public key I. When the ticket (t_{l-1}, T_{l-1}, m_l) is used the credential_role will make a record for the data D_{l-1}. The record will be used to prevent the ticket from being duplicated and to issue a charging bill. Then users can see the bill at any time. Finally, the credential_role can send a bill to the user.

In the mechanism presented here, a user can issue many tickets which can be used at any time, because a ticket's validity depends only on the data in the TCC. The data $D_0, D_1, \ldots, D_{l-1}, D_l, \ldots$ are published in the public directory. Thus there is no order of tickets. The user can also lend the ticket (t_{l-1}, T_{l-1}, m_l) to others. This is very convenient for users. Furthermore, most computing in this scheme is done on the terminal side (the user or the service provider) which can reduce the resources needed for the M-service system.

Let us now consider t_3. Ticket t_3 binds a user and service providers and it should be an agreement between the user and the service providers.

When a user requires a ticket t_3 from the TCC, the credential_role will send the user's requirement to the service providers. The credential_role will issue a public key for the user and the service providers if the service providers agree to provide the service. The credential_role sends a message including the service information, current time, requirement and agreements of the service providers to the user and the service providers. As signers, the user and the service providers use their secret key to sign this message, and then return the data (t_{il}, T_{il}) to the TCC. The credential_role makes a ticket (t_l, T_l, m_l) and sends it to the user. The ticket (t_l, T_l, m_l)

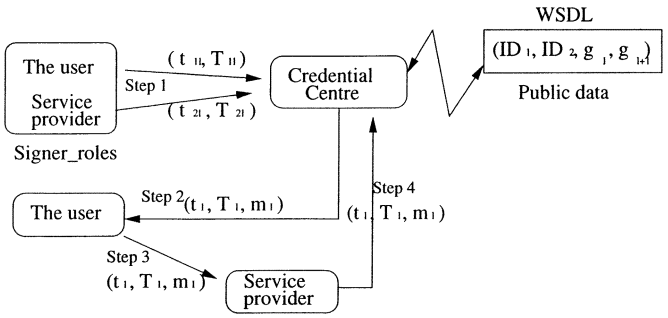

Fig. 8. Usage of ticket t_3, SOAP is used in data transmission.

can be used by the service provider. As a verifier, the service provider uses the public data $(ID_1, \ldots, ID_v, g_l)$ in the TCC to verify if the ticket is valid or not. Neither the service provider nor the credential_role knows who the user is. Only the TCC can trace the user's identity from the public key ID_i. After the data g_l is updated, the user can see a clear charging bill in the TCC. Finally, the credential_role can send a bill to the user. This can be seen in Fig. 8.

As the tickets t_1, t_2, and t_3 have no fixed order, this means no ticket needs to be used earlier or later. This is because the data for a ticket verification are $g_1, \ldots, g_l, g_{l+1}$ in the public directory. In addition, the datum g_l is changed and marked while the ticket (t_l, T_l, m_l) is used. Therefore, a ticket cannot be used twice.

Based on the two sub-schemes, the overall solution has the following features:

1) It is anonymous for the user.
2) The ticket can be transferred to others.
3) The security of the system is improved since the secret keys S_l and S_{il} are used only once.

We give one example to explain how the schemes work. The example related to the single signature scheme is for ticket t_1.

Suppose the system initialization is: $p = 11$, $q = 23$ and $n = 253$, $e = 7$, $d = 63$ such that $e * d = 1(mod\ 220)$. Here $220 = (p-1)(q-1)$. For simplicity, we suppose the hash function is $H(x, y) = 3^x * 5^y$.

Let us assume that a user I is $I = 25$. Randomly selecting $k = 4$ then $(r, S) = (192\,100)(mod\ 253)$, computes $D = 163(mod\ 253)$. The trusted_role sends $(r, S) = (192\,100)$ to the user with $I = 25$.

Suppose the first time the user needs to sign the message $m_1 = 9$ which includes the service information, etc. The user sends $(I, r, D) = (25\,192\,163)$ to the credential_role, the credential_role verifies whether $D = r * I^e$ or not. $(I, D) = (25, 163)$ is published by TCC.

Input: $(I, D, e, n) = (25, 163, 7, 253)$.

User: Randomly chooses $r_0 = 5$ and computes $T_0 = 201(mod\ 253)$, $S_1 = S_0 * m_1 = 141(mod\ 253)$, $d_0 = h(201, 9) = 50(mod\ 253)$ and $t_0 = r_0 * (S * m_1)^{-d_0} = 15(mod\ 253)$. The user sends to the verifier_role the signature (15, 201, 9) as a ticket.

Verifier_role: At first the verifier_role checks

$$d_0 = h\left(t_0^e * D_0^{d_0} * m_1^{ed_0}\ (mod\, n), m_1\right)\ (mod\, 253),$$

and then computes $D_1 = D_0 * m_1^e = 113(mod\ 253)$ if the above equation is succeed.

The verifier_role must get the public pair $(I, D_0) = (25, 163)$ in the TCC when s/he verifies whether the ticket is available or not. The ticket is unavailable if the public pair (25, 163) is changed. After the verifier_role checked the availabiblity of the ticket, he/she sends $(I, D_1) = (25, 113)$ to the TCC to update (25, 163).

B. Comparisons

Related work has been done on secure billing for E-services [9], securing XML Web services [25] and accountable anonymous access to services [21].

A secure billing scheme for M-service has been proposed in [9]. It demonstrates how a micro payment scheme can be integrated into a prepaid charging protocol and users obtain tickets from the universal mobile telecommunication systems (UMTS) service providers, who act as brokers. When requiring services from service providers, the tickets are then sent by the users to the service providers. The settlements between the service providers and the brokers are then accomplished offline. The UMTS service providers will collect the billing information from all the service providers accessed by given users and integrate them in a single bill addressed to the users. This proposal is different from ours in two aspects. First, it focuses on authentication between users and service providers to billing by using smart card technology and elliptic curve cryptography. Therefore, there is no facility for various services and no protocols for different kinds of tickets. By contrast, our work provides a rich variety of options that can deal with all documents of services. Second, users in the secure billing scheme have to send their identities to service providers. The identities are encrypted on the way to the service providers and are protected from eavesdroppers. However, the service providers know the identities. Hence, it has does not providing anonymity for the users with respect to service providers. In our scheme, users are anonymous to service providers since tickets sent by the users to the service providers include all required information for services.

Securing XML Web services is described by Damiani, Vimercati, and Samarati in 2002 [25]. Two experiments are discussed. One is that restricting access to an XML Web service to authorized users. Another one is that protecting the integrity and confidentiality of XML messages exchanged in a Web service environment. The authors introduce SOAP highlights, how to use SOAP headers for credential transfer and access control. The main difference between our scheme and the work in [25] is that we focus on a trusted model for users and service providers in a wireless environment and consider a solution for different kinds of M-services, whereas the latter is a discussion of providing a secure infrastructure to XML Web services.

Finally, anonymous access to services in mobile environments is presented in [21]. It illustrates a ticket based mechanism for service access and proposed how agencies and tickets work together through a ticket based protocol between users, customer care agencies and service providers. The protocol accomplishes authentication of service providers to mobile users, establishment of a shared session key between users and service providers, and correct and undeniable

charging. However, our work substantially differs from that proposal. Differences arise in the following three aspects. First, their protocol does not provide an overall solution for various services but only a special mobile service of type t_2. By comparison, we have analyzed the characteristics of various services and presented a detailed scheme for different kinds of services. Second, the protocol addresses the problems of lack of trust and scalability in mobile systems. We have also discussed in addition a possible charging scheme for M-services. Finally, the tickets in their work have to follow some specific models such as the Outlet model, the Kiosk model or the Agency model. Therefore, the main processing in the protocol is authentication between users, service providers and customer care agencies. By contrast, users in our scheme obtain the required tickets and use them when a service is required; we have also provided authentication between users, trusted authorities and the service providers.

VII. Concluding Remarks

M-service systems are becoming increasingly popular in business. They can be regarded as a special form of electronic commerce, where users buy services instead of products from service providers over the network. In this paper, a secure and flexible scheme for accessing M-services is proposed. In our secure M-service system, the TCC issues tickets for the users. The ticket can specify a range of access control policies for different types of services. The user presents the ticket to the service provider who can then verify its validity. Based on the privileges in the ticket the access to the service is provided to the user. The scheme is scalable to large systems involving multiple domains. It is also able to support anonymity and user privacy if required. New users and service providers can trust each other and join the system at anytime. We have considered a possible approach to charging the users for the service provided.

Acknowledgment

The authors would like to thank the reviewers for their good comments and suggestions.

References

[1] A. Mehrotra, *GSM System Engineering*. Norwood, MA: Artech House, 1997.

[2] T. Bray, J. Paoli, C. Sperberg-McQeen, and E. Maler, *Extensible Markup Language (XML) 1.1 (Second Edition)*. Cambridge, MA: World Wide Web Consortium (W3C), 2000.

[3] D. Box, *Simple Object Access Protocol (SOAP) 1.1*. Cambridge, MA: World Wide Web Consortium (W3C), 2000.

[4] R. Chinnici, M. Gudgin, J. Moreau, and S. Weerawarana, *Web Services Description Language (WSDL) 1.2*. Cambridge, MA: World Wide Web Consortium (W3C), 2002.

[5] (2002) Excellent e-Service. [Online] Available: http://www.excellente-service.com/

[6] C. Paul. (2002) Migrate With Red Hat Linux Advanced Server. [Online] Available: http://www.redhat.com/solutions/migration/

[7] A. Mehrotra and L. Golding, "Mobility and security management in the GSM system and some proposed future improvements," *Proc. IEEE*, vol. 86, pp. 1480–1496, July 1998.

[8] H. Wang and Y. Zhang, "Untraceable off-line electronic cash flow in e-commerce," in *Proc. 24th Australian Computer Science Conf.,*. Gold-Coast, Australia, Feb. 2001, pp. 191–198.

[9] K. Martin, B. Preneel, C. Mitchell, H. Hitz, A. Poliakova, and P. Howard, "Secure billing for mobile information services in UMTS," in *Proc. 5th Int. Conf. Intelligence Services Networks Technology for Ubiquitous Telecom Services*. Antwerp, Belgium, May 1998, pp. 535–548.

[10] H. Wang, J. Cao, and Y. Zhang, "A consumer anonymity scalable payment scheme with role based access control," in *Proc. 2nd Int. Conf. Web Information Systems Engineering*, Kyoto, Japan, Dec. 2001, pp. 53–62.

[11] M. Bellare, R. Canetti, and H. Krawczyk, "Pseudorandom functions revisited: the cascade construction and its concrete security. Extended abstract," in *Proc. 37th Annu. Symp. Foundations Computer Science*, Burlington, VT, Oct. 1996, pp. 514–523.

[12] D. Waleffe and J. J. Quisquater, "Better login protocols for computer networks," in *Proc. Eur. Symp. Research Computer Security*, Toulouse, France, Oct. 1990, pp. 163–172.

[13] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21,, no. 2, pp. 120–126, Feb. 1978.

[14] D. R. Stinson, *Cryptography: Theory and Practice*. Boca Raton, FL: CRC, 1995.

[15] H. Wang, J. Cao, and Y. Zhang, "Formal authorization allocation approaches for role-based access control based on relational algebra operations," in *Proc. 3nd Int. Conf. Web Information Systems Engineering*, Singapore, Dec. 2002, pp. 301–312.

[16] H. Feinstein, "Final Report: NIST Small Business Innovative Research (SBIR) Grant: Role Based Access Control: Phase 1. Tech. Rep.," SETA Corp., McLean, VA, Jan 1995.

[17] D. Ferraiolo and D. Kuhn, "Role based access control," in *Proc. 15th Natl.Computer Security Conf.*, Washington, DC, Oct. 1992, pp. 544–563.

[18] J. Barkley, K. Beznosov, and J. Uppal, "Supporting relationships in access control using role based access control," in *Proc. 3rd ACM Workshop RoleBased Access Control*, Fairfax, VA, Oct. 1998, pp. 55–65.

[19] D. Ferraiolo, J. Barkley, and D. Kuhn, "Role-based access control model and reference implementation within a corporate intranet," *ACM Trans. Inform. Syst. Security*, vol. 2, no. 1, pp. 34–64, 1999.

[20] R. Sandhu, "Future directions in role-based access control models," in *Proc. Information Assurance in Computer Networks: Methods, Models, and Architectures for Network Security, International Workshop*. St. Petersburg, Russia,: Lecture Notes in Computer Science, May 2001, vol. 2052, pp. 67–73.

[21] L. Buttyan and J. Hubaux, "Accountable anonymous access to services in mobile communication systems," in *Proc. 18th Symp. Reliable Distributed Systems*. Lausanne, Switzerland, Oct. 1999, pp. 384–389.

[22] B. Pratel and J. Crowcroft, "Ticket based service access for the mobile user," in *Proc. MobiCom: Int. Conf. Mobile Computing Networking*, Budapest, Hungary, Sept. 1997, pp. 223–232.

[23] R. Housley, W. Ford, W. Polk, and D. Solo. (1999, Jan) Internet X.509 Public Key Infrastructure Certificate and CRL Profile. [Online] Available: http://www.ietf.org/rfc/rfc2459.txt.

[24] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication, from proceedings of the royal society, volume 426, number 1871, 1989," in *William Stallings, Practical Cryptography for Data Internetworks*. Los Alamitos, CA: IEEE Comput. Soc. Press, 1996.

[25] E. Damiani, S. D. Capitani, and P. Samarati, "Toward securing xml web services," in *Proc. ACM Workshop XML Security*, Washington, DC, Nov. 2002, pp. 90–96.

[26] A. Lubinski, "Security issues in mobile database access," in *Proc. IFIP WG 11.3 12th Int. Conf. Database Security*. Chalkidiki, Greece, 1999, pp. 223–234.

[27] ——, "Database security meets mobile requirements," in *Proc. Int. Symp. Database Technology Software Engineering. WEB Cooperative Systems*, Baden, Germany, Aug. 2000, pp. 110–121.

[28] Y. Frankel, A. Herzberg, P. Karger, H. Krawczyk, C. Kunzinger, and M. Yung, "Security issues in a CDPD wireless network," *IEEE Pers. Commun.*, vol. 2, pp. 16–27, Aug. 1995.

[29] H. Wang, J. Cao, and Y. Kambayashi, "Building a consumer anonymity scalable payment protocol for the internet purchases," in *Proc. 12th Int. Workshop Research Issues Data Engineering: Engineering E-Commerce/E-Business Systems*, San Jose, CA, Feb. 2002, pp. 159–168.

[30] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–88, 1981.

[31] E. Damiani, S. D. Capitani, S. Paraboschi, and P. Samarati, "Design and implementation of an access control processor for XML documents," *Comput. Netw.*, vol. 33, no. 1–6, pp. 59–75, Jun. 2000.

[32] ——, "Securing SOAP e-services," *Int. J. Inform. Security*, vol. 1, no. 2, pp. 100–115, Feb. 2002.

[33] ——, "A fine-grained access control system for XML documents," *ACM Trans. Inform. Syst. Security*, vol. 5, no. 2, pp. 169–202, May 2002.
[34] U. Wilhelm, S. Staamann, and L. Buttyan, "On the problem of trust in mobile agent systems," in *Proc. IEEE Network Distributed Systems Security Symp.*, San Diego, CA, 1998, pp. 11–13.

Hua Wang is a Lecturer with the University of Southern Queensland, Toowoomba, Australia. He has been active in the areas of information systems management, distributed database management systems, access control, software engineering, and electronic commerce. He has participated in research projects on mobile electronic system, Web service, and role-based access control for electronic service system.

Yanchun Zhang received the Ph.D. degree in computer science from The University of Queensland, Brisbane, Australia, in 1991

He is Associate Professor in the School of Computer Science and Mathematics and Director of Internet Technologies and Applications Research Lab (ITArl) at Victoria University of Technology, Melbourne, Australia. He is a founding Editor and Co-Editor-In-Chief of *World Wide Web: Internet and Web Information Systems (WWW Journal)* and founding Book Series Editor on (Norwell, MA: Kluwer) Web Information Systems Engineering and Internet Technologies, and a Co-Chairman of Web Information Systems Engineering (WISE) Society and Steering Committee Chair of WISE Conference Series. His research areas cover database and information systems, distributed databases and multidatabase systems, CSCW, database support for cooperative work, electronic commerce, internet/web information systems, web data management, Web mining, Web search, and Web services. He has published over 100 research papers in refereed international journals and conference proceedings, and has edited over ten books/proceedings and journal special issues.

Dr. Zhang has been a key organizer of several international conferences such as APWeb'05 PC Co-Chair, APWeb'03 and APWeb'04 Publication Chair, RIDE'02 PC Co-Chair, WISE'01 Publication Chair, WISE'00 General Co-Chair, CODAS'99 Co-Chair, etc. He is a member of IFIP Working Group WG 6.4 on Internet Applications Engineering.

Jinli Cao (A'01) received the Ph.D. degree in computer science from The University of Southern Queensland (USQ), Toowoomba, Australia, in 1997.

He is a Senior Lecturer in the Department of Computer Science and Computer Engineering at La Trobe University, Melbourne, Australia. She was a Lecturer in The James Cook University, from 1998 to 2002, respectively. Her research interests include transaction management, distributed databases, mobile database management, internet and Web information systems, electronic commerce and Web services. She served a number of international conference as a PC member and is a member of the editorial reviewer board of the *International Journal of Distance Education Technologies*.

Vijay Varadharajan (SM'02) received the B.E.E. degree from the University of Sussex, Sussex, U.K., in 1981 and the Ph.D. degree in computer and communication security from Plymouth and Exeter University, U.K., in 1984.

He was sponsored by British Telecom Research Laboratories, London, U.K. and is the Microsoft Chair and Professor of Computing at Macquarie University, Sydney, Australia, the Director of Information and Networked System Security Research, Sydney, Australia, the Technical Board Director of Computer Science for the Australian Computer Society, and also Adjunct Professor at the University of Western Sydney, Sydney, Australia. Previously, he was the Foundation Professor and Head of School of Computing and IT at the University of Western Sydney Nepean, Australia and was responsible for worldwide security research at corporate Hewlett-Packard Laboratories, U.K. He has also worked with various HP Divisions in the U.S., U.K., Germany, France, and Italy and has been with HP since 1988. He is a member of the editorial board of the *Journal of Information Security*, *ACM Transactions on Information Systems Security*, on board for *Computer Abstracts*, as well as the editorial board for the Trusted Computing Platform Association and the Microsoft Board of Trustworthy Computing. He has published over 200 papers in international journals and conferences and has co-authored and edited over eight books on security, networks, and distributed systems.

Prof. Varadharajan is a Fellow of the British Computer Society (FBCS), a Fellow of the IEE (FIEE), a Fellow of the IMA (FIMA), a Fellow of the Australian Insitute of Engineers (FIEAust), and a Fellow of the Australian Computer Society. He received an award from the Council of National Academic Awards and is also listed in the *Marquis' Who is Who in the World*.