Computer Communications 32 (2009) 602–610

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Mobile services access and payment through reusable tickets

Yaohui Lei, Alejandro Quintero *, Samuel Pierre

Mobile Computing and Networking Research Laboratory (LARIM), Department of Computer Engineering, École Polytechnique de Montré, C.P. 6079, succ. Centre-Ville, Montreal, Que., Canada H3C 3A7

ARTICLE INFO

Article history:
Received 24 January 2008
Received in revised form 17 November 2008
Accepted 17 November 2008
Available online 7 December 2008

Keywords:
Authentication
Hash chain
Mobile value-added services
Ticket-based access
Reusable ticket

ABSTRACT

Third-generation mobile systems (3G) such as Universal Mobile Telecommunications Service (UMTS) bring mobile users a broad range of new value-added services (VASs). For mobile access across multiple service domains, the traditional access mechanisms require the exchange of authentication information between the home domain and the foreign domain using roaming agreements. This requirement involves complicated and expensive authentication activities in large scale mobile networks. This paper proposes a lightweight service access mechanism in which the computation complexity is low on the mobile device and the ticket can be reused. It introduces two hash chains: an authentication chain and a payment chain. The authentication chain allows a ticket to be reused and achieves mutual authentication and non-repudiation. The payment chain is lightweight, practical and likely to avoid re-initialization. Security analysis and comparison with related works indicate that our proposal is more appropriate for mobile communication networks.

1. Introduction

The rapidly developing mobile technologies widely enhance the abilities of mobile networks to offer new services. Third-generation (3G) mobile systems such as Universal Mobile Telecommunications Service (UMTS) bring mobile users a broad range of new value-added services (VASs). To access a service, users must authenticate themselves with the value-added service provider (VASP), while the latter must also be authenticated to ensure it is not malicious. For mobile access across multiple service domains, the traditional access mechanisms [1–7] require the exchange of authentication information between the home domain and the foreign domain using roaming agreements. This may involve complicated and expensive activities over large scale networks and limit future mobile applications.

Given the low-power computing capability of mobile devices, designing of secure service access mechanisms which are suited for wireless mobile networks is a nontrivial challenge. Due to the complexity of the underlying problems, public key cryptography (PKC) or asymmetric cryptography usually involves operations such as modular multiplication and exponentiation, which are much more computationally expensive than operations in symmetric cryptography. Hence, it is best to move demanding computational requirements from the mobile device to a powerful server in protocol design.

In mobile communications, a ticket is a piece of data which is analogous to tickets we use in various social events. The ticket proves that the user has been authorized to access to VASs like tourism guide. Thus, there is no need to run long distance protocols with an on-line third credential party of the accessing user in order to validate the user's certificate. Ticket-based service access is an appealing approach in mobile communication systems where users roam and contact VASPs in foreign domains [8–13].

Users do not have to disclose personal information, when ticket authentication is required. In addition, tickets can be used in different VASPs, which is of great importance in heterogeneous mobile networks. For example, a ticket-based mechanism [8–13], together with a trusted agent which issues tickets and handles payments on users' behalf, can protect the user privacy.

Due to its flexibility, the ticket idea is also widely used for access control in dynamic and distributed networks. In [14], a ticket-based access control in mobile ad hoc networks is introduced to meet the dynamics of network topology and node membership. Mobile nodes without a valid ticket are classified as misbehaving and denied from any network access. Here, a ticket acts as a passport for a networking node. It provides a simple yet effective mechanism for controlling the access of mobile nodes.

In [15], a ticket-based secure delegation service is proposed to meet the requirements of distributed computing. It supports multiple domain models by extending the Kerberos [1] framework. By using tickets based on PKC, this mechanism realizes flexible key management and reliable session key generation between the client and the provider.

Other ticket-based applications can also be found in the literature. A ticket-based address resolution protocol (TARP) [16] implements security by distributing centrally generated MAC/IP address mapping attestations, called tickets, to clients as they join

* Corresponding author. Tel./fax: +1 514 340 3240.
E-mail address: alejandro.quintero@polymtl.ca (A. Quintero).

0140-3664/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2008.11.035

the network. In [17], the authors proposed a secure wireless LAN access based on an authentication ticket which preserves clients' privacy.

This paper proposes a novel ticket-based service access mechanism which incorporates two major contributions compared to various works. First, the proposal is lightweight on the mobile user's side. High computational complexity is transferred from the mobile user side to the ticket server and the VAS provider (VASP). Second, the ticket is reusable. Two hash chains are introduced: an authentication chain and a payment chain. The authentication chain allows a ticket to be reusable as well as achieving mutual authentication and non-repudiation. The payment chain is a staircase incremental value hash chain which is lightweight, practical and may avoid chain reinitialization.

This paper is organized as follows. Section 2 covers the background work and concepts. The proposed service access model and protocols are shown in Section 3 where ticket acquisition, service provision, payment and billing, ticket revocation and reuse issues are considered. Section 4 illustrates BAN logic [18] proofs and security analysis. In Section 5, the proposed mechanism is compared with related work including classical mechanism Kerberos and existing mobile service mechanisms. Finally, conclusions are given in Section 6.

2. Background work and motivation

2.1. Ticket types vs. service types

We encounter different types of tickets in our social life [12,13]. Some tickets bind a given user to a specific service provider. Certain tickets, however, do not restrict users and service providers. Traditionally, most mobile systems entirely rely on an implicit trust relationship between a user and a VASP. In the future, mobile systems will become larger and there may co-exist many VASPs.

While service access is controlled by a ticket, billing and charging is determined by the service characteristics. Service fees can be charged according to usage time, traffic or the number of access times. Some services (such as personal mobile banking) can be accessed free for those with the appropriate access right (such as the owner of an account). We list these types of services in Table 1.

Knowing the type of service is helpful to design the charging and billing scheme during the service provision.

2.2. Ticket usage

The main advantages of ticket-based access are the following [9]: Flexibility, scalability and privacy. Although these advantages make ticket-based access an appealing approach for service provision, several issues regarding the security of ticket design and usage must be kept in mind. One problem shown in [10] is duplication. There are two cases: a legitimate user deliberately makes copies of an acquired ticket; an eavesdropper steals a ticket. For both cases, a secure authentication associated with the ticket is required in order to prevent illegal use of a ticket. Another issue is forgery [10]. Only the ticket server can construct a valid ticket. Forgery can be prevented through encryption and digital signature from the ticket server.

Table 1

Service types	Charge metrics
S_0	Free, access right is required
S_1	The number of access times
S_2	Service provision time
S_3	Quantity of traffic

In this paper, we will also address the ticket reuse issue. The current ticket-based service access mechanisms do not allow ticket reuse. This implies that a mobile user must acquire as many tickets as the number of service requests. Thus, the gain of no long distance cross domain authentication is not well worth only one service request, although ticket have privacy and flexibility advantages. If a ticket is reusable, it will be much more interesting.

2.3. Lightweight non-repudiation techniques

We have claimed that non-repudiation is required for the service access. Because of the limited computation capability of mobile devices, the non-repudiation technique should be lightweight at the mobile user side. Digital signature is widely used for non-repudiation. However, it is usually computationally expensive. A one-way hash function chain was first proposed by Lamport [19]. Hash functions are generally faster than digital signature algorithms. Hash chains, a useful cryptographic technique, are used to provide services like authentication [19,20], micropayments [21–26], signatures [27–30], multicasting [31] and certificate revocation [32]. Let $f(x)$ be a one-way hash function and a hash chain of a length of N is constructed by applying N times of hashing:

$$f^{(N)}(x) = f(f(\cdots f(x)\cdots))$$

Hash values are called nodes and $f^{(N)}(x)$ is called the tip. We define $f^{(0)}(x) = x$. With knowledge of $f^{(N)}(x)$, $f^{(N-1)}(x)$ cannot be generated without the knowledge of x. However, given $f^{(0)}(x)$, its correctness can easily be verified through $f^{(N)}(x)$, simply checking if it satisfies the equation: $f^{(N)}(x) = f(f^{(N-1)}(x))$. This property comes directly from the property of one-way hash functions. Once the tip is used for verification, the hash chain is reduced and the new tip becomes $f^{(N-1)}(x)$.

A cryptographic message authentication code (MAC) is a short piece of information used to authenticate a message. The input of a MAC function consists of a secret key and a message to be authenticated:

$$\mathrm{MAC} = h(Key, Message)$$

where h is a one-way hash function. The MAC technique is widely used in UMTS authentication and key agreement (AKA) [4]. The MAC value protects both a message's integrity as well as its authenticity, by allowing verifiers to detect any changes to the message content.

The combination of MAC and hash chain can provide the non-repudiation purpose. A claimant randomly selects a nonce, generates a hash chain and its MAC, and sends both to the verifier. The hash chain can be used to prove the claimant for N times. At the first authentication, the claimant sends $f^{(N-1)}(x)$ to the verifier. The verifier checks if it satisfies the equation $f^{(N)}(x) = f(f^{(N-1)}(x))$. If it does, the verifier updates the stored value to $f^{(N-1)}(x)$ for the following authentication. The property of the hash chain prevents non-legitimate users from impersonating the legitimate user. Obviously, it also guarantees that the legitimate sender of a message cannot subsequently deny having sent the message.

3. Service access through tickets

3.1. Service access model

Cases where a ticket is issued and validated by the same service provider will not be covered as they do not belong to the typical models in cross domain authentication. We define the user's home domain as the ticket issuer (ticket server) and his foreign domain as the VASP. The proposed model contains four principals: a certification authority (CA), a ticket server (S), a mobile user (U) and a VASP (P).

The last three principals all subscribe to the CA which is responsible for issuing certificates to mobile users and generating key pairs (a public key and a secret key) to the ticket server and the VASP. The mobile user obtains a ticket through presenting his certificate to the ticket server. This is realized by a ticket acquisition protocol. After successfully validating the certificate, the ticket server issues a ticket to the mobile user. Here, the mobile user obtains the right to access a service and does not need to pay the ticket. Once the mobile user gets the ticket, he can use it to request the service from the VASP anytime, anywhere.

The VASP needs to access the ticket server occasionally, for example, for billing. In fact, in traditional cross domain authentication in GSM and UMTS [2,4], the home domain is always accessible for the foreign domain. For every time of service provision, the VASP generates a billing record and sends it to the ticket server.

3.2. Ticket acquisition

As we analyzed previously, in mobile networks, the users usually have limited power mobile devices to communicate with more powerful servers like VASPs. Although mobile devices become increasingly powerful today, public key cryptographic operations (encryption/decryption) remain still two or three orders of magnitude slower than symmetric key cryptography operations.

We denote U the mobile user, S the ticket server and P the VASP. We abuse the uppercase letters for principals and their identities. Before running the protocol, U first obtains a certificate Cert_U^S and his long term symmetric key K_U from the CA. The certificate is given as:

$$\mathrm{Cert}_U^S = \langle E_S(U, K_U, \mathrm{expire}, \mathrm{Udata}, \mathrm{Sig}_{\mathrm{CA}}(U, K_U, \mathrm{expire}, \mathrm{Udata}))\rangle$$

The certificate is encrypted with the ticket server's public key ($E_S(X)$ denotes the public key encryption under S's public key). This key, however, has a special property compared with ordinary secret keys. Instead of being known only by U, the key can also be known by the ticket server during the ticket acquisition phase. Aside from issuing certificates, the CA is responsible for generating symmetric keys for mobile users. In this certificate, $\mathrm{Sig}_{\mathrm{CA}}$ is the secret signing algorithm of the CA. The result after running the signing algorithm is the CA's digital signature. The digital signature of a message is a string which depends on this message and on the CA's secret key, in such a way that anyone can check the validity of the signature through the CA's signature verification algorithm. The verification algorithm takes a message and its corresponding public key to compute the signature.

Inside the certificate, U is the mobile user's identity and the field U data includes all other relevant information: version number, serial number, user ID, issuer ID and issuer name, etc. This means that U possesses a certificate specific to the ticket server S before communicating with S. A similar server specific certificate can also be found in [33].

Once the mobile user has his certificate in hand, he can start his authentication to the ticket server and obtain a ticket. The protocol is illustrated as follows:

$$U \rightarrow S: \quad \mathrm{Cert}_U^S, \{R_1, \ldots, R_T\}K_U$$

$$S \rightarrow U: \quad \{R_1 \oplus \cdots \oplus R_T, \mathrm{id}, \mathrm{ticket}, \quad C_1, \ldots, C_T, K_{UP}\}K_U$$

The protocol starts when U sends a set of random nonces $R_1,\ldots,\mathrm{R}_T$ encrypted by his long term symmetric key K_U($\{X\}_K$ denotes symmetric encryption of X under key K) and the certificate of U under S. These nonces are used to generate a set of hash chains. The length of the hash chain N is predetermined in advance for both principals.

The ticket server decrypts the certificate with its private key, then verifies the signature using the CA's public signature verification algorithm. When S checks successfully the certificate from U, it discovers the secret key K_U and stores its relevant information such as the expire date. It obtains the nonces and computes the corresponding hash chains.

Depending on the different types of service, the ticket server generates payment chains of different lengths. It chooses a set of random nonces $C_1,\ldots,C_T$ to generate payment hash chains $g^{(M)}(C_t)$ (g is a one-way hash function, $1 \leqslant t \leqslant T$) with length of M. Then, it generates an authentication key K_{UP} for future authentication between U and P, together with the ticket and its unique id. The ticket server encrypts all of these fields with K_U and sends them back to U.

The ticket is defined as:

$$\mathrm{ticket} = \langle E_P(P, K_{UP}, \mathrm{id}, f^{(N)}(R_1), \ldots, f(N)(R_T), g^{(M)}(C_1), \ldots, g^{(M)}(C_T), \mathrm{Tdata}, \\ \mathrm{Sig}_S(P, K_{UP}, \mathrm{id}, f^{(N)}(R_1), \ldots, f^{(N)}(R_T), g^{(M)}(C_1), \ldots, g^{(M)}(C_T), \mathrm{Tdata}))\rangle$$

The ticket is specific to the VASP P. It is encrypted with P's public key, which means that only P can decrypt it and the ticket belongs to a t_3 type ticket. This requires that the ticket server knows the VASP's public key through the CA. The signature of S in the ticket is used to prevent a fake ticket. The id is the unique identity of the ticket and the T data field includes all detailed ticket information: issuer ID, issuer name, issued time, service type, service name, times of access, etc.

When the ticket is sent, the ticket server stores the information pertaining to the ticket for future billing verifications. Here is an example:

$$\mathrm{id}, P; K_U; \mathit{expire}; R_1, \cdots, R_T; C_1, \cdots, C_T; \mathit{valid}$$

where *expire* is the K_U expire date. The nonces are reserved for future authentications and payment chain validation. The field *valid* is reserved by the ticket server. When a ticket needs to be revoked, this field is set to FALSE. Otherwise, it is set to TRUE. When U receives the message, he checks the correctness of $R_1 \oplus \cdots \oplus R_T$. If it is correct, he keeps the ticket for future service request. Otherwise, he discards the ticket.

3.3. Service provision

Once U obtains a valid ticket, he can provide it to P for service provision. The protocol is illustrated as follows:

$$U \rightarrow P: \quad \mathrm{ticket}, \{f^{(N-i)}(R_t), P, N_U\}K_{UP}$$
$$P \rightarrow U: \quad \{f^{(N-i+1)}(R_t), N_U + 1, N_P, T_P\}K_{UP}$$
$$U \rightarrow P: \quad \{N_P \oplus T_P\}K$$

The protocol starts when U wants to validate his ticket to get access to the service from P. He sends the ticket, together with the hash chain i ($1 \leqslant i \leqslant N$, and the time t, $1 \leqslant t \leqslant T$), the identity of P and a nonce N_U encrypted with the authentication key K_{UP} previously generated by the ticket server S during the ticket acquisition phase. Here, the inclusion of P is to tell P that U knows P as his peer communication principal.

When P receives the service request, it decrypts the ticket with its private key and checks its validity with S's signature verification algorithm. Then, it compares this signature to the one received. If they are identical, the ticket is considered as valid. Then, it gets the id of the ticket and searches its database to see if the ticket is used for the first time. If so, it stores the hash chains and the tip $f^{(N)}(R_1)$; otherwise, it ignores the chains. It gets the key K_{UP} and decrypts the second part of the message. It checks if the hash chain $f^{(N-i)}(R_t)$ satisfies the equation:

$$f\left(f^{(N-i)}(R_t)\right) = f^{(N-i+1)}(R_t)$$

If so, it updates the hash chain by removing the current tip $f^{(N-i+1)}(R_t)$. The hash chain becomes shorter and the new tip is $f^{(N-i)}(R_t)$. Note that the mobile user uses the hash nodes between $f^{(N-1)}(R_t)$ and $f^{(0)}(R_t)$ (inclusive) for authentication while the VASP uses the hash nodes between $f^{(N)}(R_t)$ and $f^{(1)}(R_t)$ (inclusive) for verification. After that, the tth chain is abandoned and the $(t+1)$th chain is used until all the hash chains run out.

Then, it chooses a nonce N_P and generates a fresh session key is generated, for example as done in UMTS authentication and key agreement. It sends back $f^{(N-i+1)}(R_t)$ as confirmation, together with (N_U+1), N_P and the timestamp T_P encrypted by the shared key K_{UP}.

When U receives the message, he decrypts the message and checks if $f^{(N-i+1)}(R_t)$, (N_U+1) are correct. If so, he knows that mutual authentication has been realized.

He computes the fresh session key $K = f3KU(N_U \oplus N_P)$ and sends back $\{N_P \oplus T_P\}K$ for agreement of T_P as the service start time.

3.4. Ticket acquisition via VASP

In the proposed model, a mobile user acquires a ticket before a service request. It implies that the mobile user knows the VASP which he will request service from, before roaming into the service domain. This means, the ticket is type t_3. However, it is not always the case. Sometimes, the mobile user cannot predict or simply has no knowledge of the VASP, or he might need only a t_2 ticket. In this situation, the user can request a ticket from the VASP ignoring the VASP's identity. The VASP acts as a proxy to pass the request to the ticket server. This action is also helpful when the ticket server is not accessible directly by a mobile user even the user wants a t_3 ticket. The following protocol describes this action. In this case, it becomes the traditional cross domain authentication as in GSM and UMTS [2,4].

$$U \to P: \ \mathrm{Cert}_U^S, \{R_1, \ldots, R_T\}K_U$$

$$P \to S: \ E_S\Big(\mathrm{Cert}_U^S, \{R_1, \ldots, R_T\}K_U, N_P, \mathrm{Sig}_P\Big(\mathrm{Cert}_U^S, \{R_1, \ldots, R_T\}K_U, N_P\Big)\Big)$$

$$\begin{aligned} S \to P: \ & E_P(N_P+1, \mathrm{ticket}, \{R_1 \oplus \cdots \oplus R_T, \mathrm{id}, \mathrm{ticket}, C_1, \ldots, C_T, K_{UP}\}K_U, \\ & \mathrm{Sig}_s(N_P+1, \mathrm{ticket}, \{R_1 \oplus \cdots \oplus R_T, \mathrm{id}, \mathrm{ticket}, C_1, \ldots, C_T, K_{UP}\}K_U)) \end{aligned}$$

$$\begin{aligned} P \to U: \ & \{R_1 \oplus \cdots \oplus R_T, \mathrm{id}, \mathrm{ticket}, C_1, \ldots, C_T, K_{UP}\}K_U, \\ & \{\{R_1, \ldots, R_T\}K_U, N_P, T_P\}K_{UP} \end{aligned}$$

$$U \to P: \ \{N_P \oplus T_P\}K$$

The mobile user sends the certificate and a set of nonces encrypted with the secret key K_U to the VASP P. The latter appends a nonce N_P, signs the whole message and sends it to the ticket server S. When S successfully checks the validity of the signature and the certificate, it generates the hash chains based on the nonces received and billing chains $g^{(M)}(C_1), \ldots, g^{(M)}(C_T)$. Then, it creates a ticket and computes the confirmation N_P+1 and $\{R_1 \oplus \cdots \oplus R_T\}_K U$. It signs the whole message and sends it to P.

When P receives the message, it validates the signature. If it is valid, it keeps the ticket and gets the shared key K_{UP}. It forwards to U the corresponding part. It encrypts the $\{R_1, \ldots, R_T\}_K U$ with the shared key K_{UP} for confirmation, together with a nonce N_P and the timestamp T_P. The nonce and the timestamp have the same meaning as in the service provision protocol. They are used to generate fresh session key and service start time agreement.

3.5. Payment and billing

The payment chain $g^{(M)}(C_1), \ldots, g^{(M)}(C_T)$ included in the ticket are used by the VASP to charge the service. However, the usage of the payment chain is different from the usage of the authentication chain. For each time service provision, the user takes off a new node from the authentication chain while restarting the same (only one) payment chain from the beginning. This means the payment is only related to the current service provision and independent of the previous service provision.

During the service provision, payments happen several times depending on different types of service. The idea of "tick" payments based on hash chains was first introduced by Pedersen [21] and has been widely adopted in [9,22–25]. The payment in this paper is also based on ticks. The charging request is repeated several times according to the service provision requirements. The following steps are based on those in [9]. We assume that the last "tick" sent by the user is c_j.

(1) The VASP sends a request for d ticks according to the expense.
(2) The user sends back the response: the tick $c_{j-d} = g^{(j-d)}(C_t)$ with the computed $\mathrm{MAC}'_{j-d} = h(K_{UP}, (P, T_j, c_{j-d}, f^{(N-i)}(R_t)))$, $\mathrm{MAC}_{j\_}d = h(K\ _{\mathrm{U}}, (P, T_j, c_{j-d}, f^{(N-i)}(R_t)))$, where T_j is the timestamp.
(3) The VASP checks if $c_j = g^{(d)}(c_{j-d})$. If so, it continues the service provision; otherwise, it stops.

Even if the service is free (S_0 type), the VASP still asks the user to sign the payment in order to prevent the user denying the service. In this case, the value of first tick can be defined as zero.

Although the user can refuse to release a tick after a period of service, he cannot gain too much (only the value after the last tick c_j). When the service terminates, the VASP generates a billing record and subsequently sends the proof of the service provision to the ticket server.

$$\begin{aligned} P \to S: \ & E_S\Big(\mathrm{id}, P, T_P, T_j, c_j, f^{(N-i)}(R_t), \mathrm{MAC}'_j, \mathrm{MAC}_j, \\ & \mathrm{Sig}_P\Big(\mathrm{id}, P, T_P, T_j, c_j, f^{(N-i)}(R_t), \mathrm{MAC}'_j, \mathrm{MAC}_j\Big)\Big) \end{aligned}$$

This billing message contains the ticket id, the identity of the VASP, the service start time and end time, the last tick from the user c_j, the current authentication chain and MAC'_j, MAC_j. The message is authenticated by the user and signed by the VASP. Both of them cannot deny the service provision. The ticket service checks the validity of the payment chain and MAC'_j, MAC_j. If there is already a billing record corresponding to the current chain of this ticket, it is rejected by the ticket server it might be a replayed billing by the VASP. Otherwise, a billing record is stored and an aggregated bill will be sent to the user.

A problem concerning the length of the payment chain, however, should be considered. Due to its finite length, the system has to restart when it runs out the hash chain. Simply generating a longer hash chain will not help out fundamentally because the length of the hash chain cannot be set too large in practice. A simple solution is to run again the service provision protocol to use the next node of the authentication chain. Another way is to re-initialize the old hash chain. In [34], the authors proposed a method which applies public key cryptography recursively to generate infinite length hash chains. In [35,36], the authors introduced a notion of "tying" multiple finite length hash chains through one time signature. All of these methods increase resource consumption.

In this paper, we introduce a new payment method which may avoid hash chain reinitialization and keep the hash chain in a reasonable length. The idea is illustrated as follows. We do not try to prolong the hash chain while trying to use every tick efficiently. Traditionally, every tick has the same value. We define a staircase incremental value function *value*(j) associating with each tick according to its position in the hash chain (the position of the tick $g^{(j)}(C_t)$ is j).

$$value(j) = \begin{cases} a, & \text{if } (n-j) \leqslant l_0, \\ a \cdot p_1, & \text{if } l_0 < (n-j) \leqslant l_1, \\ a \cdot p_2, & \text{if} l_1 < (n-j) \leqslant l_2, \\ \cdots & \\ a \cdot p_k, & \text{if } l_{k-1} < (n-j) \leqslant l_k. \end{cases}$$

where $a \geqslant 1$ and $1 < p_1 < p_2 < \cdots < p_k$. The hash chain is cut into different segments and the ticks in different segments are associated with different values. Thus, the hash chain becomes a staircase. The ticks in the first stair ($(n-j) \leqslant l_0$) is the cheapest, while those in the last stair ($l_{k-1} < (n-j) \leqslant n$) are the most expensive. For example, the value of one single tick in the second stair can be equivalent to the value of two ticks of the first stair. For example, $p_1, p_2, p_3, \ldots$ can be $2, 3, 4, \ldots$ or $2^1, 2^2, 2^3, \ldots$. Intuitively, the ticks in the back are less frequently used than those in the front during the charging procedure.

The value function is agreed among the principals in advance. If it is possible to know the type of service in the ticket acquisition phase, the payment chain can be tailored to the service. For S_0 and S_1 service, one tick is sufficient. For S_2 and S_3 service, the length of the chain and the value function needs to be carefully designed. Through this way, we may avoid reinitializing the payment hash chain. This approach is not perfect for payment, but practical and lightweight.

3.6. Ticket revocation and reuse

The ticket revocation issue is simple. If a ticket is considered to be revoked (e.g., the mobile user's secret key K_U expires or the mobile user notices the ticket server to revoke a ticket), the ticket server sends REV OKE message to the corresponding VASP:

$$S \rightarrow P: \quad E_P(\text{id}, \text{REV OKE}, \text{Sig}_S(\text{id}, \text{REV OKE}))$$

The VASP checks if the ticket is already stored (used by the ticket holder before). If not, it means that the ticket has been issued and has not been used yet; the VASP stores the revoked ticket information. Otherwise, it updates the status of the ticket REVOKED.

When a user requests a service with a ticket, the VASP checks if the ticket has been revoked. If so, the service request is rejected.

In the solution described in this paper, a ticket can be reused, which is different from other ticket-based mechanisms. This is realized and secured through the hash chains $f^{(N)}(R_1), f^{(N)}(R_2), \ldots, f^{(N)}(R_T)$ shared between the mobile user and the VASP. This hash chain can be used $N \times T$ times. After that, this ticket becomes invalid and should be abandoned. Thus, we can say the ticket is reusable within a certain limit.

4. System security analysis

4.1. Logical proof by BAN logic

The goal of a secure access mechanism is to ensure that authentication is realized for all principals involved in a communication. The authentication protocol relies on network relationships. Any failed relationship will allow an adversary to masquerade as legitimate principals in runs of the protocol. Belief logics can be used to examine trust relationships in security protocols. We use BAN logic [18] to analyze the protocols in this paper.

This paper only presents proofs from the ticket acquisition protocol and the service provision protocol. The ticket acquisition via VASP is similar and we do not show its proof. For convenience, we put two protocols together. The idealized protocol is as follows:

$$\begin{aligned} &M1. \quad U \rightarrow S: \; ES\left(\left\{\overset{\text{Ku}}{\rightarrow} U\right\}\right), \{R\}Ku \\ &M2. \quad S \rightarrow U: \; \left\{R, Ep\left(\left\{U \overset{\text{Kup}}{\leftrightarrow} P\right\}\right), U \overset{\text{Kup}}{\leftrightarrow} P\right\}Ku \\ &M3. \quad U \rightarrow P: \; \left\{Ep\left(\left\{U \overset{\text{Kup}}{\leftrightarrow} P\right\}\right), R, Nu\right\}Kup \\ &M4. \quad P \rightarrow U: \; \{R, Nu, Np, Tp\}Kup \\ &M5. \quad U \rightarrow P: \; \{Np \oplus Tp\}K \end{aligned}$$

The $M1., M2., \ldots, M5.$ represent the message rounds. For sake of simplicity, we replace Cert_U^S and the ticket by $\{\overset{\text{Ku}}{\rightarrow} U\}K_S$ and $\{U \overset{\text{Kup}}{\leftrightarrow} P\}K_p$, respectively, and ignore signature verification fields inside. The hash chain and the ith hash node are also simplified as R. The verification of hash chain is treated independently outside of the protocol, and the returned R implies the correctness of the verification.

To analyze this protocol, we have assumed initial beliefs of the principals:

U believes fresh(R)	S believes fresh(R)
U believes $\overset{Ku}{\rightarrow} U$	S believes $\overset{Ks}{\rightarrow} S$
S believes fresh (K_{UP})	*S believes* U $\overset{Kup}{\leftrightarrow} P$
U believes fresh (N_U)	*P believes fresh(N_U)*
S believes fresh(N_U)	P believes $\overset{Kp}{\rightarrow} P$
U believes (S controls $U \overset{Kup}{\leftrightarrow} P$)	P believes (S controls $U \overset{Kup}{\leftrightarrow} P$)
P believes fresh(N_P)	*U believes fresh(N_P)*
P believes fresh(T_P)	*U believes fresh(T_P)*

Both U and P trust the shared key K_{UP} generated by S and believe its freshness. All principals believe the freshness of R, nonce N_U and timestamp T_P. Each principal trusts its own secret key.

As receiving **M1**, the principal S has S sees $(E_S\{\overset{\text{Ku}}{\rightarrow} U\})$, $\{R\}K_U$. In other words, the principal S can decrypt the message and read the secret key K_U and the random number R. With the hypothesis S **believes** $\overset{\text{Ks}}{\rightarrow} S$, the message-meaning rule yields S **believes** U **said** $(\overset{\text{Ku}}{\rightarrow} U)$. In other words, S **believes** that it is U, not anyone else, who sent the message. With the belief U **believes** $\overset{\text{Ku}}{\rightarrow} U$, we can deduce S **believes** U **said** (R). This means that S **believes** that it is U, not anyone else, who sent the random number R. The initial belief S **believes fresh**(R) and the nonce-verification rule yield S believes U **believes fresh**(R). In other words, both S and U believe that the random number R is fresh.

Similarly, as receiving **M2**, we have U **believes** $U \overset{\text{R}}{\leftrightarrow} S$, U **believes** $U \overset{\text{Ku}}{\leftrightarrow} S$. In other words, both U and S know the secrets R and K_U. Since we have hypothesis S **believes** $U \overset{\text{Kup}}{\leftrightarrow} P$, we obtain U **believes** $U \overset{\text{Kup}}{\leftrightarrow} P$. This means that U believes that K_{UP} is the shared secret key between U and P.

The deduction of **M3** is similar to M1. We do not show the details here. The final result from M3 is P **believes** U **believes fresh**(-N_U). This means that N_U is fresh and P believes this.

While U receives the **M4**, with the result U **believes** $U \overset{\text{Kup}}{\leftrightarrow} P$, the message-meaning rule for shared keys applies. As a result, we can have the following result: U **believes** P **said** $(R, \text{N}_U, N_P, T_P)$.

Moreover, there are the following hypotheses: U **believes fresh**(N_P), U **believes fresh**(T_P) The nonce-verification rule applies and yields U **believes** P **believes** $(R, \text{N}_U, N_P, T_P)$. In other words, both P and U believe the freshness of R, N_U, N_P, T_P.

At this point, U and P are mutually authenticated. The last message, **M5**, is specifically used for both the timestamp T_P and the new session key agreement. It does not influence the beliefs just established above.

We summarize the above formal analysis in natural language as follows. As the certificate is encrypted by the ticket server S and signed by CA, only S can verify it through its private key and CA's signature verification algorithm. Then, Scan read U's secret key and further decrypt the random numbers. When U receives the ticket from S, he can know that it is sent from S because the K_U is only known by U and S.

As the ticket is encrypted by service provider's public key, only P can decrypt it. When P receives this ticket, it can get the shared secret key K_{UP} stored in the ticket and thus decrypt the hash node and the random number N_U in the message. At this point, both U and P believe that K_{UP} is a shared secret key between them. They trust each other and use this key for secret communication. The next step is to verify the hash chain stored in the ticket. Of course, its security is guaranteed by K_{UP}. Finally, both U and P contribute a random number and generate alone the session key K.

4.2. Security analysis

It has been suggested that the BAN logic is unable to deal with security flaws resulting from interleaving of protocol steps [37]. We analyze the security aspects of the proposed protocols by considering possible attacks.

(1) *Forge tickets:* Outside attackers cannot fabricate or forge tickets, because they cannot generate a valid signature in the name of the CA. In the ticket acquisition phase, since the K_U is encrypted in the certificate, only S can verify U's certificate Cert_U^S and thus know U's long term secret symmetric key K_U and retrieves the nonces.
(2) *Mutual authentication:* The freshness of the ticket implies the freshness of the authentication chain and the payment chain inside. During the service provision phase, U sends a two-part message: *ticket*, the encrypted current hash node and a nonce N_U. The ticket is encrypted by P's public key and contains P's identity indicating that P has the right to validate the *ticket*. The *ticket* is signed by the ticket server and P can check if the *ticket* is valid through the signature contained in the *ticket*.
(3) *Replay and man-in-the-middle attacks:* Although an attacker can eavesdrop a valid ticket, he cannot use it later, because he cannot get the KU Pand the hash chain encrypted in the ticket. Only P can decrypt the ticket and get the hash chain then. An attacker cannot modify the ticket and the encrypted hash chain.
(4) *Key freshness:* The authentication key K_{UP} is used only during the authentication and known only by U and P. The new session key is generated for each time service provision and valid only at the current session. Both U and P contribute their nonces to establish this key. Thus, key freshness is guaranteed.
(5) *Malicious service provider:* In mobile networks, a service provider could be malicious and deliberately leak a user's ticket to someone else. In the payment and billing protocol, U signs the ticks used with his private key K_U which can only be verified by the ticket server. This signature comprises the current tips of the authentication chain and the payment chain; nobody can tamper and generate it.

4.3. Discussion

We designed a complementary protocol to help a mobile user acquire a ticket via a VASP. When U needs to acquire a ticket via a VASP, it is mandatory that the ticket server be accessible. This is similar to Kerberos [1] in which the server should always remain on-line. However, the situation of our proposal is less serious because this protocol would not happen frequently. It can be mitigated by using multiple ticket servers. It seems that the system executes cross domain authentication and loses the benefit of a ticket. However, since the ticket described in this paper is reusable, one time cross domain authentication is worth future reused local domain authentication. The authentication chain allows a ticket to be reused $N \times T$ times. Nevertheless, ticket reuse does not mean a duplicate ticket is acceptable because it would fail in authentication without a correct hash node. In order to use this chain correctly, both the user and the VASP have to synchronize its current position in the chain. If the synchronization fails (e.g., the user loses the position information), it is difficult for them to resynchronize. Consequently, the ticket should be abandoned. As for the payment chain, if it is not well designed, it can be run out before the service terminates. Experiments are helpful to determine the length of the chain and the value function.

5. Comparisons with related work

5.1. Comparison with Kerberos

Kerberos [1,38], a computer network authentication protocol developed by MIT to protect network services provided by Project Athena, allows individuals communicating over an insecure network where individuals prove their identities to one another in a secure manner.

We can specify Kerberos protocol as follows, where Alice (A) authenticates herself to Bob (B) using a server (S).

$$\begin{aligned}
&\mathrm{A} \to \mathrm{S}: && \mathrm{A}, \mathrm{B} \\
&\mathrm{S} \to \mathrm{A}: && \{T_\mathrm{S}, L, K_\mathrm{AB}, \mathrm{B}, \{T_\mathrm{S}, L, K_\mathrm{AB}, \mathrm{A}\}K_\mathrm{BS}\}K_\mathrm{AS} \\
&\mathrm{A} \to \mathrm{B}: && \{T_\mathrm{S}, L, K_\mathrm{AB}, \mathrm{A}\}K_\mathrm{BS}, \{\mathrm{A}, T_\mathrm{A}\}K_\mathrm{AB} \\
&\mathrm{B} \to \mathrm{A}: && \{T_\mathrm{A} + 1\}K_\mathrm{AB}
\end{aligned}$$

Server S stands for both AS and TGS, and K_AB stands for the session key between A and B. In this protocol, $\{T_S, L, \mathrm{K_{AB}}, \mathrm{A}\}\ K_\mathrm{BS}$ acts as a ticket that the client authenticates himself to B. In order to confirm B's identity and its recognition of A, A sends $\{A, \mathrm{T}_A\}K_\mathrm{AB}$ as an authenticator and B replies with $\{T_A + 1\}K_\mathrm{AB}$.

The protocol security relies heavily on timestamps T_S, T_A and lifespans L as reliable indicators of the freshness of a communication. Thus, one drawback is that all principals must keep local clock synchronization. Between two synchronizations with the reliable time source, local clocks may drift [40]. In Kerberos, all tickets are time-stamped and have limited lifetimes. Ideally, no authentication ticket remains valid longer than the time an expert hacker would need to crack the encryption. The value of L should be carefully set in order to prevent replay attack by an eavesdropper. Authentication tickets are session-specific to improve the security of the system by ensuring that no authentication ticket remains valid after a given session is complete. In other words, a ticket can be used only once. Whenever the user wants to access the service again, he should apply a new ticket. In other words, the ticket cannot be reused.

The security of the proposed proposal in this paper, however, relies on a shared hash chain instead of timestamps. Thus, the validity of a ticket can last long time and can be reused. This will mitigate the burden of the ticket server to issue tickets.

While the original Kerberos does not provide clients anonymity, one of the basic features of tickets is user privacy during service provision. Recently, an Internet draft [41] proposed a mechanism to allow a client to request an anonymous ticket. Moreover, Kerberos does not consider the payment and billing issue.

This issue is quite different in the mechanism of this paper. Although there is a secret authentication key shared between the ticket server and the client, this key is actually not established directly between the client and the ticket server. Instead, it is generated by the CA and transferred to the ticket server, by the client on his certificate. The ticket server does not directly have to maintain the secret key as it can obtain this key from the client's certificate while the client presents his certificate for ticket request. In addition, the key is stored with each ticket only for payment verification. Similarly, the client and the VASP do not have to maintain the authentication key either since the authentication key can be easily obtained from the ticket for each time of service request.

5.2. Comparison with extended Kerberos

As indicated by Neuman et al. [42], the traditional Kerberos presents an attractive security target in the form of the KDC which maintains a shared symmetric key with every principal. In the event of a KDC compromise, all the symmetric keys will be divulged to the attacker and will have to be revoked [43].

Various studies have been done to extend Kerberos with PKC [42–45]. These extensions integrate PKC into the initial authentication exchange and distribute most of the authentication workload away from the trusted intermediary to the communicating parties. Generally speaking, PKC can simplify the distribution of keys in the Kerberos environment because secret-key cryptology (SKC) requires a separate secret key to be shared between every pair of users. Thus, both the client and the KDC have a public-private key pair in order to prove their identities to each other over the open network.

The extended Kerberos improves the scalability and security. Nevertheless, as with the traditional Kerberos, the extended Kerberos still relies on timestamps for freshness indicators. Consequently, the timestamped ticket can only be valid in a single session. The introduction of PKC also increases the computation requirement for clients. The billing issue remains the same.

5.3. Comparisons with mobile service mechanisms

Many ticket-based mobile service access mechanisms [8–13] have been proposed to address the VAS access issue in mobile networks. We compare them with the work in this paper.

A secure billing for mobile information services conducted by Advanced Security for Personal Communications Technologies (ASPeCT) in UMTS is described in [8]. There are three principals in this pre-paid charging scheme: mobile user, service provider and UMTS service provider. This work relies on a third trusted party (TTP) to support mobile telecommunications security services which use PKC. When requiring services from the VASP, the user pays for the service by sending micropayment tokens (“ticks”) as shown in [21]. The VASP is able to check the validity of these ticks with a user’s credential certificate issued by the UMTS service provider, or an appropriate TTP acting on behalf of the UMTS service provider. The UMTS service provider will collect the billing information from VASPs and in turn forward to the user. A two phase charging protocol is illustrated. In the first phase, the user and the VASP authenticate each other and agree on a Diffie–Hellman session key. The second phase is responsible for data transfer through user’s ticks.

This work has several differences comparing to the work of this paper. First, the protocol in [8] is based on PKC and the computational complexity is more expensive than symmetric cryptography operations. This requires powerful mobile devices. In this paper, the mobile user needs only operations of hash and symmetric encryption/decryption. Second, the protocol focuses on authentication between the users and the VASP, and disregards different kinds of services in service provision. Third, the mobile user has to send his identity to the VASP. Thus, the VASP is constantly aware of the identity of users accessing services.

In [9], an accountable anonymous access to services in mobile communication systems is proposed. It is also based on Diffie–Hellman PKC and composed of three roles: user, service provider and customer care agency. The protocol demonstrates one-time use, post-paid tickets for one type of ticket. Its charging is also based on ticks [21]. There are two major differences between this mechanism and the solution proposed in this paper. First, the user has to generate a session key using Diffie–Hellman method. It involves a high level computational complexity. Second, the ticket cannot be reused. Although the user can obtain a few tickets once for later use, it is possible that a user would run out of his tickets while roaming. In this case, the user could not access a service since the customer care agency could not be reached.

In [11], the authors present a ticket model which includes three principals: the mobile user, the service provider, and the ticket server. In this model, the mobile user is assumed to register with the ticket server. This service access mechanism can be split into two phases: ticket acquisition and ticket validation. The disadvantages of this work are similar to that in [8,9]: a high-level computational complexity and non-reusable tickets. The authors did not discuss the payment and billing issue.

Another similar service access mechanism [10] is based on prepaid tickets that can only be used with the service provider issued them. Payment and billing issue is not discussed. The service provider can replay a ticket and users have no way to prove they have not used it.

In [12,13], the authors propose another ticket-based model which involves a Trusted Credential Center (TCC) and a Trusted Authentication and Registration Center (TARC). In both mechanisms, different ticket types are considered with a single or multiple signatures required for ticket validation and the prevention of ticket reuse. All tickets cannot be modified by the users. Ticket duplication, including reusing a ticket and transferring a ticket, are discussed. One of the disadvantages is the high level of computational complexity required for the signer and the verifier who could be a mobile user. The signature schemes are based on RSA which is based on large prime numbers factorization. In the single signature scheme, the signer picks a large random number and runs large exponent operations twice while the verifier has to run it three times. All of these operations require users to have powerful devices. The situation is even more serious when multiple signatures are involved. As with other ticket-based mechanisms, tickets cannot be reused.

We simulated our mechanism and the mechanisms proposed in [9,11,12]. There are four ticket protocols in this simulation: 1 for our protocol, 2 for the protocol in [12], 3 for the protocol in [11] and 4 for the protocol in [9]. We chose 128 bits for sym-

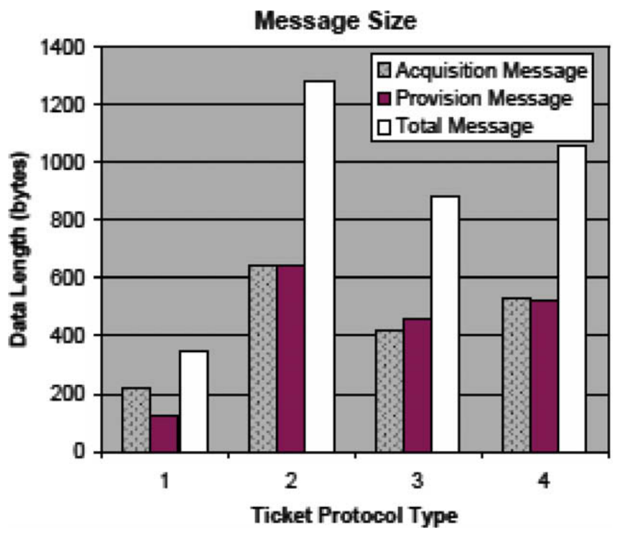

Fig. 1. Message size for all protocols.

metric key size and 1024 bits for public key size. The ticket and the certificate have the size of 64 bytes. We chose Blowfish algorithm for symmetric encryption/decryption. For all the same message fields, we used the same size for all protocols. For protocol 1, we set the hash chain size 12. In the simulations, the acquisition message means all the messages involved during the ticket acquisition phase while the provision message means all the messages involved during the service provision phase. The total message is just the sum of the acquisition message and the provision message. The message size comparison is shown in Fig. 1.

The total message size using protocol 1 is only nearly 400 bytes, much less than other protocols. This is because other protocols need to send public key or agreement key (1024 bits = 128 bytes) in the message. While in protocol 1, the public key is only used for signature generation and verification.

6. Conclusions

Mobile communication systems are bringing more and more choices of VASs to mobile users. Due to various VASPs and numerous mobile users, the guarantee of secure and suitable access is a critical concern in service provision. This paper proposed a novel mechanism to access mobile VASs through reusable tickets. Considering the limited computation capability of mobile devices, the mechanism is lightweight on the mobile user side: only symmetric encryption/decryption and hash operations are needed. We transfer high computational complexity to the ticket server and the VASP which are usually more powerful. Two hash chains are introduced to achieve mutual authentication, non-repudiation and ticket reuse. By allowing ticket reuse, a mobile user does not have to apply a new ticket for every service access. This is of high interest for two reasons: first, it is convenient for mobile users; second, it decreases the load of the ticket server. In order to avoid reinitializing the payment chain, we introduce a staircase incremental value function associating to the chain. We analyze the protocol through BAN belief logic. Comparisons with classic mechanisms (like Kerberos) and other mobile service access mechanisms, together with the experimental results, show that the proposed mechanism is lightweight, secure and more appropriate to mobile communication.

Acknowledgements

The authors appreciate the reviewers' valuable and helpful comments and suggestions.

References

[1] B.C. Neuman, T. Ts'o, Kerberos: an authentication service for computer networks, IEEE Communications 32 (9) (1994) 33–38.
[2] M. Rahnema, Overview of the GSM system and protocol architecture, IEEE Communications Magazine 31 (1993) 92–100.
[3] C.H. Lee, M.S. Hwang, W.P. Yang, Enhanced privacy and authentication for the global system for mobile communications, Wireless Networks 5 (1999) 231–243.
[4] Third Generation Partnership Project, Technical Specification Group Services and System Aspects, 3G Security, Security Architecture (Release 7), 3GPP Std. TS 33.102 V7.0.0, December 2005.
[5] M. Zhang, Y. Fang, Security analysis and enhancements of 3GPP authentication and key agreement protocol, IEEE Transactions on Wireless Communications 4 (2) (2005) 734–742.
[6] C. Huang, J. Li, Authentication and key agreement protocol for UMTS with low bandwidth consumption, in: Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA 2005), Taipei, Taiwan, IEEE Computer Society, March 28–30, 2005, pp. 392–397.
[7] L. Harn, W. Hsin, On the security of wireless network access with enhancements, in: Proceedings of the 2003 ACM workshop on Wireless Security, San Diego, USA, September 19, 2003, pp. 88–95.
[8] K. Martin, B. Preneel, C. Mitchell, H. Hitz, A. Poliakova, P. Howard, Secure billing for mobile information services in UMTS, in: Proceedings of the Fifth International Conference on Intelligence Services Networks Technology for Ubiquitous Telecom Services, Antwerp, Belgium, May 1998, pp. 535–548.
[9] L. Butty'an, J. Hubaux, Accountable anonymous service usage in mobile communication systems, in: Proceedings of the 18th Symposium on Reliable Distributed Systems, Lausanne, Switzerland, October 1999, pp. 384–389.
[10] B. Patel, J. Crowcroft, Ticket based service access for the mobile user, in: Proceedings of the Third Annual ACM/IEEE International Conference on Mobile computing and networking, Budapest, Hungary, September 26–30, 1997, pp. 223–233.
[11] B. Lee, T. Kim, S. Kan, Ticket based authentication and payment protocol for mobile telecommunications systems, in: Proceedings of the Pacific Rim International Symposium, Dependable Computing, December 17–19, 2001, pp. 218–221.
[12] H. Wang, J. Cao, Y. Zhang, Ticket-based service access scheme for mobile users, Australian Computer Science Communications 24 (1) (2002) 285–292.
[13] H. Wang, Y. Zhang, J. Cao, V. Varadharajan, Achieving secure and flexible m-services through tickets, IEEE Transactions on Systems, Man, and Cybernetics 33 (6) (2003) 697–708.
[14] H. Luo, J. Kong, P. Zerfos, S. Lu, L. Zhang, URSA: ubiquitous and robust access control for mobile ad hoc networks, IEEE/ACM Transactions on Networking 12 (6) (2004) 1049–1063.
[15] K. Chang, T. Lee, B. Chun, T. Kim, Ticket-based secure delegation service supporting multiple domain models, in: Proceedings of the 2001 Pacific Rim International Symposium on Dependable Computing, December 17–19, 2001, pp. 289–292.
[16] W. Lootah, W. Enck, P. McDaniel, TARP: ticket-based address resolution protocol, in: Proceedings of the 21st Annual Computer Security Applications Conference, December 5–9, 2005, pp. 106–116.
[17] T. Komori, T. Saito, A secure wireless LAN system retaining privacy, in: Proceedings of the 18th International Conference on Advanced Information Networking and Applications, vol. 2, 2004, pp. 370–375.
[18] M. Burrows, M. Abadi, R. Needham, A logic of authentication, ACM Transactions on Computer Systems 8 (1) (1990) 18–36.
[19] H. Lamport, Password authentication with insecure communication, Communications of ACM 24 (11) (1981) 770–772.
[20] N. Haller, The S/KEY one-time password system, in: Proceedings of the ISOC Symposium on Network and Distributed System Security, February 1994, pp. 151–157.
[21] T.P. Pedersen, Electronic payments of small amounts, in: Proceedings of the Fourth Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science, vol. 1189, Cambridge, UK, 1996, pp. 59–68.
[22] R. Anderson, C. Manifavas, C. Southerland, Netcard – a practical electronic cash system, in: Proceedings of the International Workshop on Security Protocols, Cambridge, UK, April 10–12, 1996, pp. 49–57.
[23] G. Horn, B. Preneel, Authentication and payment in future mobile systems, in: Proceedings of the Fifth European Symposium on Research in Computer Security, vol. 1485, 1998, pp. 277–293.
[24] R. Rivest, A. Shamir, Payword and micromint: two simple micropayment schemes, in: Proceedings of the Fourth Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science, vol. 1189, Cambridge, UK, 1996, pp. 69–87.
[25] J. Zhou, K. Lam, Undeniable billing in mobile communication, in: Proceedings of the Fourth ACM/IEEE International Conference on Mobile Computing and Networking, Dallas, Texas, October 1998, pp. 284–290.
[26] R. Hauser, M. Steiner, M. Waidner, Micro-payments based on iKP, in: Proceedings of the 14th Worldwide Congress on Computer and Communications Security Protection, Paris, 1996, pp. 67–82.
[27] N. Asokan, G. Tsudik, M. Waidners, Server-supported signatures, Journal of Computer Security (1997).
[28] L. Harn, H. Lin, A non-repudiation metering scheme, Communications Letters (IEEE) 5 (12) (2001) 486–487.
[29] N. Perrig, The BiBa one-time signature and broadcast authentication protocol, in: Proceedings of the Annual Conference on Computer and Communications Security (ACM), 2001, pp. 28–37.
[30] X. Ding, D. Mazzocchi, G. Tsudik, Experimenting with server-aided signatures, in: Network and Distributed Systems Security Symposium (NDSS'02), 2002.
[31] A. Perrig, R. Canetti, D. Song, D. Tygar1, Efficient and secure source authentication for multicast, in: Proceedings of the Network and Distributed System Security Symposium NDSS 2001, February 2001.
[32] W. Aiello, S. Lodha, R. Ostrovsky, Fast digital identity revocation, in: Proceedings of the Advances in Cryptography-Crypto'98, Lecture Notes in Computer Science, vol. 1462, Berlin, Germany, 1998, pp. 137–152.
[33] D. Wong, A. Chan, Mutual authentication and key exchange for low power wireless communications, in: Proceedings of the IEEE MILCOM, vol. 1, 2001, pp. 39–43.
[34] K. Bicakci, N. Baykal, Infinite length hash chains and their applications, in: Proceedings of the 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative 7 Enterprises (WETICE'02), Pittsburgh, USA, June 2002, pp. 57–61.
[35] V. Goyal, How to re-initialize a hash chain. Available from: <http://eprint.iacr.org/2004/097.pdf>.
[36] Y. Zhao, D. Li. An improved elegant method to re-initialize hash chains. Available from: <http://eprint.iacr.org/2005/011.pdf>.

[37] C. Boyd, W. Mao, On a limitation of BAN logic, Advances in Cryptology: Eurocrypt'93, Springer, Berlin, 1993. pp. 240–247.
[38] C. Neuman, T. Yu, S. Hartman, K. Raeburn. The Kerberos network authentication service (v5). Available from: <http://www.ietf.org/rfc/rfc4120.txt>.
[40] W. Diffie, P.C. van Oorschot, M.J. Wiener, Authentication and authenticated key exchanges, Designs, Codes and Cryptography 2 (2) (1992) 107–125.
[41] L. Zhu, P. Leach, K. Jaganathan, Anonymity support for Kerberos, July 2006. Available from: <http://www.ietf.org/internet-drafts/draft-ietf-krb-wg-anon-01.txt>.
[42] C. Neuman, B. Tung, J. Wray, J. Trostle. Public key cryptography for initial authentication in Kerberos. Available from: <ftp://ietf.org/internetdrafts/draft-ietf-cat-kerberos-pk-init-02.txt>.
[43] M. Sirbu, J.C. Chuang, Distributed authentication in Kerberos using public key cryptography, Symposium on Network and Distributed System Security, IEEE Computer Society Press, San Diego, California, 1997.
[44] L. Zhu, B. Tung. Public key cryptography for initial authentication in Kerberos (PKINIT). Available from: <http://www.ietf.org/rfc/rfc4556.txt>.
[45] I. Downnard, Public-key cryptography extensions into Kerberos, IEEE Potentials 21 (5) (2002) 30–34.