Wireless Netw (2007) 13:663–678
DOI 10.1007/s11276-006-8148-z

A secure authentication and billing architecture for wireless mesh networks

Yanchao Zhang · Yuguang Fang

Published online: 9 June 2006
© Springer Science + Business Media, LLC 2006

Abstract Wireless mesh networks (WMNs) are gaining growing interest as a promising technology for ubiquitous high-speed network access. While much effort has been made to address issues at physical, data link, and network layers, little attention has been paid to the security aspect central to the realistic deployment of WMNs. We propose *UPASS*, the first known secure authentication and billing architecture for large-scale WMNs. UPASS features a novel user-broker-operator trust model built upon the conventional certificate-based cryptography and the emerging ID-based cryptography. Based on the trust model, each user is furnished with a universal *pass* whereby to realize seamless roaming across WMN domains and get ubiquitous network access. In UPASS, the incontestable billing of mobile users is fulfilled through a lightweight realtime micropayment protocol built on the combination of digital signature and one-way hash-chain techniques. Compared to conventional solutions relying on a home-foreign-domain concept, UPASS eliminates the need for establishing bilateral roaming agreements and having realtime interactions between potentially numerous WMN operators. Our UPASS is shown to be secure and lightweight, and thus can be a practical and effective solution for future large-scale WMNs.

Keywords Wireless mesh networks (WMNs) · Roaming · Security · Authentication · Billing

Y. Zhang (✉) · Y. Fang
Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
e-mail:{yczhang@, fang@ece.}ufl.edu

1. Introduction

Wireless mesh networks (WMNs) have been gaining momentum recently as a promising technology for ubiquitous high-speed network access [1]. Figure 1 depicts a logical WMN architecture where stationary mesh routers form a wireless multihop backbone with long-range high-speed wireless techniques such as WiMAX [2]. The backbone is connected to the Internet via high-speed wireless or wired links. End users, while at rest or in motion, can assess the network by either a direct wireless link to a nearby mesh router or a chain of intermediate users to a distant mesh router. A review of the advantages of WMNs yields a long list: low upfront investment, self-organization and self-maintenance, incremental deployment, high robustness, good scalability, increased coverage, and so on [1]. These attractive features have inspired numerous research, experiment and deployment efforts to advance the ubiquitous deployment of WMNs [1].

It is envisaged that the future large-scale WMN will consist of a huge number of WMN domains, each administrated by an independent operator. Unlike a cellular network domain often of a country scale, a WMN domain may be on a community, section, metro or larger scale. Therefore, the number of WMN operators is expected to be much larger than that of current cellular network operators. At the same time, users desire single sign-on (SSO) and seamless roaming across WMN domains. To enable this, entity authentication must be conducted between a serving domain and a mobile user for two reasons. First, the serving domain should authenticate the user to avert fraudulent use of network resources. Second, the user must authenticate the serving domain to prevent an attacker from impersonating an operator for various wicked motives [3]. The principal reason for requiring entity authentication is to bill a mobile user for enjoying network access. Billing in WMNs, however, faces new challenges—not

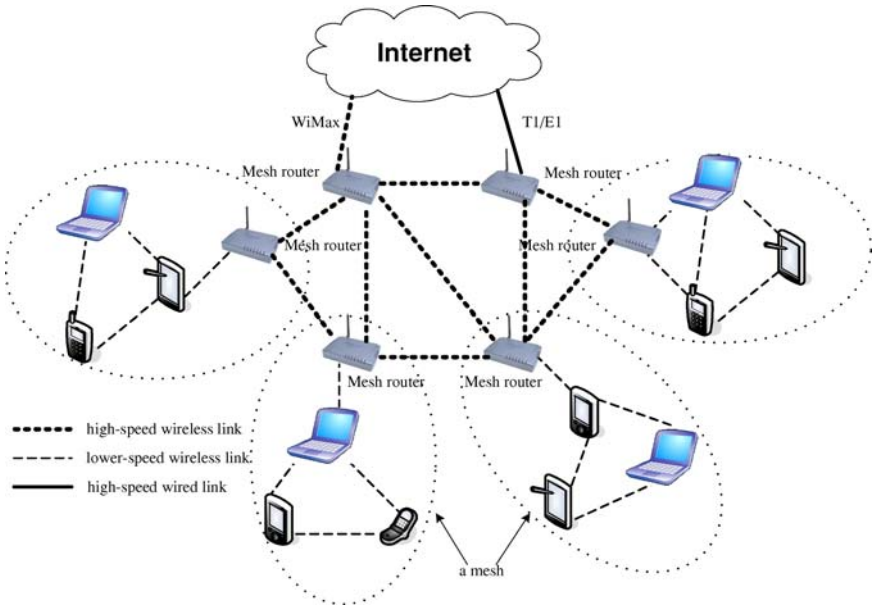


Fig. 1 A logical wireless mesh network architecture

only should the serving domain be paid for providing network access, but also intermediate users must be remunerated for relaying others' traffic to and from the mesh router. Otherwise, users with individual interests will be reluctant to serve others in order to save their own resources such as energy [4–7].

Authentication and billing of mobile users is a traditional research topic. A number of elegant solutions have been proposed in the contexts of Global System for Mobile Communications (GSM) [8], Personal Communication Systems (PCSs) [9], Universal Mobile Telecommunication System (UMTS) [10, 11], Mobile IP networks [12], among many others. Despite the difference in specifics, these schemes all depend on a home-foreign-domain model. Specifically, each user has a *home domain* where he[1] is registered on a long-term basis and billing information is accumulated. When the user roams into a *foreign domain*, his home domain is contacted for his credentials to authenticate him. Subsequently, the foreign domain reports the amount of services accessed by the user to his home domain which, in turn, pays the foreign domain and charges the user an amount commensurate with his usage.

The conventional solution above has four main drawbacks making it less suitable for WMNs. First, it often involves a potentially time-consuming and expensive execution of an authentication protocol among a user, his home domain and the foreign domain. As the user base grows large, the overall network authentication signaling overhead would be significant. Second, a bilateral service level agreement (SLA) has to be established between each pair of WMN domains to permit user roaming between them. Such SLAs may be relatively easy to establish between relatively few cellular network operators, but will be very difficult to set up between potentially numerous WMN operators. Third, users have to trust both home and foreign network operators to make correct charges over the services they receive. There is often lack of evidence to resolve possible disputes over the number of network access requests and the duration of each request. Last, the conventional solution does not consider how to reward intermediate users who relay traffic for others, which is crucial for stimulating cooperation in packet forwarding in WMNs.

In this paper, we present a secure authentication and billing architecture, called *UPASS*, to enable seamless roaming and ubiquitous network access in future large-scale WMNs. Our UPASS stems from an all-too-familiar real life scenario. A user first applies for a credit card with a bank whereby to purchase goods at any supermarket accepting credit cards. Supermarkets needn't have prior relationships with each other, but just need to establish a trust relationship with one or a few banks that accept payments from users and pays supermarkets. If we view each supermarket as a distinct WMN domain, the consumption of a user at different supermarkets can be regarded as his roaming across various WMN domains. This analogy motivates us to adopt the sophisticated credit card-based business model in the UPASS design.

The players in our UPASS are brokers, users and WMN operators. Brokers issue a *universal pass* to each user by which the user can enjoy ubiquitous WMN access. Once validating a pass, an operator can grant network access to the pass holder without fear of not being paid later. The relationship between a pass holder (user), a WMN operator and the broker is analogous to that between a credit card user, a supermarket and the card-issuing bank. However, a WMN operator

[1] No gender implication.

in UPASS does not need to perform realtime checking with a broker about the authenticity of a user pass, different from what a supermarket does for a presented credit card. This is desirable for reducing communication overhead as well as service response delay. In contrast to the conventional solution, our UPASS eliminates the need for establishing bilateral SLAs between WMN operators. Instead, each WMN operator merely needs to have a prior relationship with one or a few brokers whose quantity is considered much smaller than that of WMN operators. In addition, entity authentication in UPASS just involves the local interaction between a user and a serving domain without requiring the on-line involvement of the corresponding broker.

A crucial issue in UPASS is the design of passes. Since passes of short sizes are beneficial for the resource-constrained wireless arena, we leverage the emerging ID-based cryptography (IBC) (cf. Section 2.1) to enable a pass size of at most a few tens of bytes. The use of IBC-based passes facilitates very efficient mutual authentication and shared-key establishment between a user and a serving WMN domain and between users visiting the same WMN domain. To permit the universal verifiability of passes, our UPASS features a hybrid trust model that harnesses the advantages of both IBC and the conventional certificate-based cryptography (CBC), while averting their respective disadvantages.

In UPASS, billing of mobile users is achieved through a realtime micropayment approach as a combination of digital signature and one-way hash-chain techniques [13]. Our approach ensures *billing incontestability*: the user just pays what he ought to pay, while the WMN operator, as well as intermediate users participating in packet forwarding, receives the amount commensurate with the offered service. It is also lightweight regarding the storage requirement, the communication and computation overhead on users and operators.

As far as we know, our UPASS is the first work along this line in the context of WMNs. In addition to providing entity authentication and undeniable billing, UPASS can serve as a solution base for other security issues in WMNs such as secure routing, DoS attacks and worms. Since the research and development of WMNs are still in their infancy, we believe that our UPASS has a high potential of becoming an important component of future large-scale WMNs.

The rest of this paper is organized as follows. Section 2 briefly introduces IBC and defines the paper scope. Next we present the network architecture under consideration and the system models. This is followed by a detailed illustration of pass-based entity authentication in Section 4. Section 5 dwells on the billing approach of UPASS. We then analyze the overhead of UPASS in Section 6, survey related work in Section 7, and end with conclusions and future work.

2. Preliminaries

2.1. Introduction to IBC

IBC is receiving extensive attention as a powerful alternative to the traditional CBC. Its main idea is to make an entity's public key directly derivable from its publicly known identity information such as its email address. IBC thus eliminates the need for public-key distribution realized via certificates. The recent prosperity of IBC has taken place due to the application of the following *pairing* technique.

Let p, q be two large primes and $\mathbb{E}/\mathbb{Z}_p$ indicate an elliptic curve $y^2 = x^3 + ax + b$ over $\mathbb{Z}_p = \{i | 0 \leq i \leq p-1\}$. We denote by $\mathbb{G}_1$ a q-order subgroup of the additive group of points on $\mathbb{E}/\mathbb{Z}_p$, and by $\mathbb{G}_2$ a q-order subgroup of the multiplicative group of the finite field $\mathbb{F}^*_{p^2}$. The Discrete Logarithm Problem (DLP) is required to be hard[2] in both $\mathbb{G}_1$ and $\mathbb{G}_2$. A pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ with the following properties:

1. *Bilinear*: For all $P, Q \in \mathbb{G}_1$ and all $c, d \in \mathbb{Z}^*_q$,

$$\hat{e}(cP, dQ) = \hat{e}(cP, Q)^d = \hat{e}(P, dQ)^c = \hat{e}(P, Q)^{cd} \text{ etc.} \tag{1}$$

2. *Non-degenerate*: If P is a generator of $\mathbb{G}_1$, then $\hat{e}(P, P)$ is a generator of $\mathbb{G}_2$.
3. *Computable*: There is an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

Note that $\hat{e}$ is also *symmetric*, i.e., $\hat{e}(P, Q) = \hat{e}(Q, P)$ for all $P, Q \in \mathbb{G}_1$, which follows immediately from the bilinearity of $\hat{e}$ and the fact that $\mathbb{G}_1$ is a cyclic group. Modified Weil [14] and Tate [15] pairings are examples of such bilinear maps for which the *Bilinear Diffie-Hellman Problem* (BDHP) is believed to be hard[3]. We refer to [14, 15] for a more comprehensive description of how the pairing parameters should be chosen in practice for efficiency and security. How to bootstrap a pairing-based IBC cryptosystem is left for discussion in Section 3.3.1.

2.2. Scope of the paper

As the first paper on authentication and billing in WMNs, we do not have the ambition to solve all related problems. In particular, we just consider security attacks aimed at

[2] It is computationally infeasible to extract the integer $x \in \mathbb{Z}^*_q = \{i | 1 \leq i \leq q-1\}$, given $P, Q \in \mathbb{G}_1$ (respectively, $P, Q \in \mathbb{G}_2$) such that $Q = xP$ (respectively, $Q = P^x$).

[3] It is believed that, given $\langle P, xP, yP, zP \rangle$ for random $x, y, z \in \mathbb{Z}^*_q$ and $P \in \mathbb{G}_1$, there is no algorithm running in expected polynomial time, which can compute $\hat{e}(P, P)^{xyz} \in \mathbb{G}_2$ with non-negligible probability.

authentication and billing. How to deal with denial-of-service (DoS) type attacks, such as physical-layer jamming, MAC-layer misbehavior [16] or routing disruption [17], though important, is not addressed in this paper. In addition, we do not intend to study efficient MAC and routing schemes, but merely assume the existence of such schemes. Moreover, we will not investigate mobility management [18], another important issue to support global seamless roaming. Our conjecture is that mobility management can be realized via some location service providers to and from which current locations of mobile users are reported and acquired.

3. Network architecture and system models

In this section, we present the network architecture under consideration, and the user-broker-operator relationship model, the trust model, and the pass model used in our UPASS.

3.1. Network architecture

The large-scale WMN architecture in our mind consists of a number of WMN domains, each operated by a different WMN operator. We refer to a subnet comprising a mesh router and mobile users within its coverage area as a *mesh* (cf. Fig. 1). A WMN domain is composed of a certain number of meshes, either physically adjacent or non-adjacent. For example, a WMN operator may own meshes in multiple cities or only in one city section. WMN domains may overlap with each other, and whether or not neighboring domains are connected solely depends on policy issues out of our consideration.

Generally speaking, a mesh router has much more powerful computation and communication capacities than a mobile user. Similar to [4], we assume that a mesh router sends packets in one hop to all users in its coverage. By contrast, a mobile user may transmit packets in one hop or multiple hops to a mesh router within or beyond his transmission range. There are two main reasons motivating us to assume a single-hop downlink. First of all, mobile users can save their scarce energy resources, as there is no need to relay downlink packets. Secondly, the single-hop downlink can greatly facilitate the transmissions of control packets such as the *Beacons* from a mesh router to mobile users to announce its existence. Note that, however, our UPASS can be easily extended for use in symmetric WMNs with both multihop uplinks and downlinks.

As [5], we require *all* communications to pass through a mesh router. We note that this assumption may lead to suboptimal routes when the source and destination are not neighbors but are close to each other. However, it is expected that communications to and from a mesh router will constitute the majority of traffic in a mesh whose main use is to relay users' traffic to and from the wired Internet. Therefore, such suboptimal cases should happen rarely. In addition, this assumption would significantly reduce the routing complexity from the users' point of view. The reason is that they only need to maintain a route to the mesh router instead of one route per potential destination in the same mesh.

In this paper, we do not specify the underlying MAC protocol and any existing scheme such as the IEEE 802.11 or its variant can be applied. Likewise, any of the established ad hoc routing protocols such as AODV [19] or DSR [20] can be used for route discovery and packet forwarding.

3.2. User-broker-operator relationship model

In our UPASS the players are brokers, users and WMN operators. Brokers issue universal passes to users to authorize them to make payments to WMN operators in return for network access services. Brokers also redeem the user payments collected by operators. Different from what they are in a conventional home-foreign-domain solution, users in UPASS are not bound to any specific operator so that user-operator relationships are transient. By comparison, both user-broker and broker-operator relationships are long-term. In fact, one may view brokers as regular banks with which both users and operators have opened accounts. We assume that brokers are fully trustable by both users and operators, but a user and an operator usually do not play full trust on each other.

The above relationship model is well-suited for ubiquitous high-speed network access via WMNs. The users see the advantage of being able to get network access by any WMN operator on demand. The WMN operators might initially view this as an undesirable situation because users are no longer tied into any long-term revenue-generating plan. In the long run, however, an operator will have potentially many more customers available to it—all the users are potentially available to all WMN operators with our model, while, under the traditional home-foreign-domain model, a user is locked to a specific operator once signing an agreement. In addition, the operators are relieved from the heavy burden of establishing bilateral SLAs with potentially many other operators. Instead, each of them just needs to have a trust relationship (like opening an account) with certain broker(s), the number of which is considered much smaller than that of WMN operators. The brokers can make profits by deducting fees from an operator's credit or adding fees to a user's charge. They may also impose entry or subscription fees to users and operators for participation in their payment systems.

3.3. Trust model

In our UPASS, the trust model is a hybrid one which merges the traditional CBC and the newly emerging IBC. It consists of a number of *trust domains*, each managed by a broker or a WMN operator. IBC is used in each trust domain, while CBC is adopted for certification of trust-domain parameters.

1) *Trust domain setup*: The administrator of each trust domain bootstraps its domain as follows:

1. Generate the pairing parameters $(p, q, \mathbb{E}/\mathbb{Z}_p, \mathbb{G}_1, \mathbb{G}_2, \hat{e})$, as described in Section 2.1.
2. Select an arbitrary generator P of $\mathbb{G}_1$.
3. Choose a cryptographic hash function H_1, mapping arbitrary strings to non-zero elements in $\mathbb{G}_1$.
4. Pick a random $\kappa \in \mathbb{Z}_q^*$ as the domain-master-secret and derive a public domain-public-key as $P_{pub} = \kappa P$.

The resulting trust-domain parameters are defined as follows.

$$\begin{aligned}\text{domain-params} &:= (\text{group-params}, \text{domain-public-key})\\ &:= ((p, q, \mathbb{E}/\mathbb{Z}_p, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, H_1, P), P_{pub})\end{aligned}$$

The administrator must keep κ confidential to itself, while makes domain-params publicly known. Like Diffie-Hellman group parameters used in IPSEC [21], group-params can be predefined by standard bodies (e.g., IETF) for general use and be shared by many domains. Using standard parameters will make it possible to use a well-known short index in place of group-params to shorten the representation of domain-params. In contrast, κ and P_{pub} are unique to each trust domain.

2) *Certification of domain parameters*: In an IBC cryptosystem, two communicating parties are required to use the same public system (domain) parameters. Therefore, there is a need for certification of trust-domain parameters, which is realized through conventional certificates in our UPASS. In particular, we view the domain-params of a trust domain as a conventional public key. The domain administrator gets its domain-params certified by a trusted third-party Certification Authority (CA). Such domain-params certificates can be stored at some public directory from which they can be retrieved as needed. An alternative way is to harness the Domain Name System (DNS), as suggested in [22]. That is, the domain-params certificate of each trust domain is stored and distributed as part of its DNS record.

What does the hybrid IBC/CBC trust model bring us? At a first glance, it appears to have created a level of complexity, but we believe that this makes the system more efficient and scalable. Compared to a pure CBC trust model or a Public Key Infrastructure (PKI), our hybrid trust model enables very short-sized passes and efficient global verification of passes without reliance on conventional long certificates, as shown shortly. In addition, a pure IBC trust model requires that all trust domains share the same domain-master-secret and thus domain-params. In practice, it is almost impossible to establish such a system of global trust. By comparison, our hybrid trust model is much more practical because each trust domain generates its own domain-master-secret and domain-params. Moreover, since trust domains are relatively much fewer than mobile users, it is much more feasible and manageable to use CBC in certificating domain-params rather than individual users' public keys as in a conventional PKI.

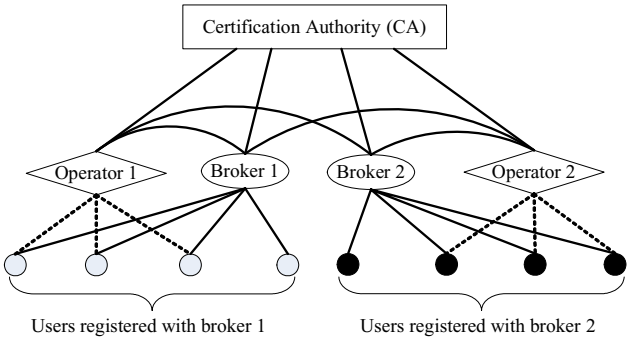


Fig. 2 The abstract trust model of UPASS, where solid and dashed lines indicate long-term and transient trust relationships, respectively

For clarity, we show the abstract hybrid trust model in Fig. 2, where a CA serves as the root responsible for certificating domain-params. The second level consists of trust domains administrated by brokers and WMN operators which have enduring trust relationships with the root CA. An operator may have a long-term trust relationship with one or a few brokers, as depicted in Fig. 2. The third level comprises mobile users who have long-term trust relationships with associated brokers and transient trust relationships with operators of the visited WMN domains. In practice, the root CA may be replaced by a hierarchical PKI, in which case conventional certificate chains [23] can be used for verification of domain-params certificates generated by different CAs. For simplicity, we shall focus on the single CA case hereafter.

3.4. Pass model

We now introduce the pass model used in UPASS. There are two types of passes, *user* and *router* passes, whereby a user and a mesh router of the serving WMN domain can authenticate each other. We assume that each router in an operator domain is uniquely identifiable by a network access identifier [24] (R-NAI) of format routerID@operator_domain. For simplicity, we also assume that each user has a unique NAI (U-NAI) of format userID@broker_domain obtained from his enrolled broker. Note that, however, our UPASS can be directly applied to the more general case that a user has multiple U-NAIs from distinct brokers without modification.

1) *Router pass acquisition*: Prior to network deployment, a WMN operator furnishes each domain-inside mesh router

with a router pass R-pass := (R-NAI, expiry-date) and a pass-based key R-key := κH_1(R-pass). Here κ is the operator's domain-master-secret and H_1 is the hash function specified in its public domain-params. An (R-pass, R-key) pair is nothing but a standard ID-based public and private key pair in an IBC cryptosystem [14]. The expiry-date field is introduced to guarantee the freshness of an R-pass. Before an R-pass expires, the operator should transmit to the mesh router a new (R-pass, R-key) pair via a secure channel in time. Depending on different security policies, an (R-pass, R-key) pair may be updated hourly, daily, weekly, or even monthly, and can be sent along with other domain-related control signaling traffic. An R-pass will be made publicly known, while the corresponding R-key should be kept secret to a mesh router itself. Also note that, it is computationally infeasible to deduce κ from the ⟨R-pass, R-key⟩ pair because of the difficulty of solving the DLP in $\mathbb{G}_1$ (cf. Section 2.1).

Alternatively, an R-pass can be implemented as a conventional public-key certificate and the R-key as the corresponding private key. In contrast to a typical X.509 certificate [25] of about 1 KB, our ID-based R-pass has at most only a few tens of bytes in size. The main reason is that our R-pass retains the R-NAI and expiry-date parts of a certificate, while dumps the most space-consuming parts, namely, a public key and the digital signature of a CA. Such ID-based passes can enable much more efficient entity authentication, shared-key establishment and billing, as will be seen later.

2) *User pass acquisition*: Before joining the network, each user has to register with a desired broker, similar to applying for a credit card. Upon a registration request, the broker usually needs to verify the user's personal data such as his driver's licence or social security number (SSN) and check his credit status. Depending on registration policies in place, the broker may also require a security deposit. The broker then issues a *user pass* to the user of format:

U-pass := (U-NAI, expiry-date, otherTerms).

There are several points we want to clarify. (1) The userID part of U-NAI can be decided by the user himself or the broker, as long as it is unique in the broker domain. (2) Expiry-date specifies the expiry date of a U-pass and the user has to renew it in time if desiring to stay with the same broker. The validity period of a U-pass relies on different registration policies or user plans of the broker. (3) The broker may use the otherTerms field to specify other terms and conditions enforced on the U-pass holder. For example, the broker may limit the amount that the user can spend per day at any WMN operator, or name the list of WMN domains the user is allowed to visit, with which the broker has cooperative agreements.

In addition to the U-pass, the broker issues to the user a pass-based key, U-key := κH_1(U-pass), where κ is the broker's domain-master-secret. A (U-pass, U-key) pair is a standard ID-based public and private key pair in an IBC cryptosystem as well. Similar to an R-pass, a U-pass is also much shorter than a conventional certificate implementing the same functionalities or having the same otherTerms field.

3) *User pass protection and revocation*: The U-pass can be made publicly known, but the U-key should be well safeguarded and kept confidential to the user himself. The user may store his (U-pass, U-key) pair in his often-used mobile device or in a USB drive so that he can use it on multiple devices if any. There are many possible means to protect his U-key. One usual way is to require the user to input a personal identification number (PIN) preset and memorized by himself for per access to his U-key.

It is possible that a careless user loses his (U-pass, U-key) pair which is unprotected using the PIN method. This occurs, for instance, when the user loses the mobile device or the USB drive that stores his secret pair. In that case, the user should report it immediately to the broker and his liability should be limited accordingly, as it is for credit-card loss. However, it should be noted that the loss of a (U-pass, U-key) pair would cause much less severe consequences or financial losses than that of a credit card. The reason is that U-passes are specifically designed for buying network access services whose rates are becoming more and more lower.

The broker can take several measures to minimize its financial risk. For example, if a user repeatedly reports a (U-pass, U-key) loss, the broker can refuse to issue him new passes. In addition, the broker may specify a carefully-designed spending-limit in a pass. It may also use a short U-pass validity period, say one day, and send to the user (e.g., via email) a new (U-pass, U-key) pair at the early morning of each day that is only valid for that day. Or, the broker can maintain a *revocation list* of U-passes whose holders have reported losses, or which are otherwise problematic. The WMN operators can download the revocation lists from the brokers each morning and refuse to serve users whose presented U-passes appear on the revocation lists. Although the last method requires certain interactions between operators and brokers, it is still considered to be much simpler and more lightweight than the conventional home-foreign-domain method, in which an operator performs real-time checking with each roaming user's home domain about his account status.

4. Entity authentication

In this section, we elaborate on how to leverage user and router passes to achieve entity authentication. We consider both user-router authentication and user-user authentication which occurs between users visiting the same WMN domain. For user-router authentication, we further distinguish between *inter-domain* authentication, which occurs when a

user migrates from one WMN domain to another, and *intra-domain* authentication, which happens when a user makes his way from one mesh to another of the same WMN domain. We also make the usual assumption that inter-domain migrations happen less frequently than intra-domain ones. So does inter-domain authentication than intra-domain authentication. As a byproduct, our authentication schemes also facilitate efficient user-router and user-user shared-key establishment. The shared keys are important to prevent from unauthorized access to and modification of subsequent messages transmitted in the air.

The following cryptographic primitives are used throughout the remainder of this paper. $h_k(M)$ refers to the message integrity code (MIC) of message M under a symmetric key k, where h can be any fast hash function such as SHA-1 [26]; $\mathcal{E}_{pk}(M)$ means an IBC-based encryption operation on message M with public key pk; $\mathcal{S}_{sk}(M)$ denotes message M with its IBC-based signature under private key sk. Please refer to [27] for a number of elegant IBC encryption and signature schemes.

4.1. Inter-domain authentication

Each mesh router is required to periodically broadcast *Beacon* messages that can be received by all users within its coverage area. A *Beacon* should include the router's R-pass and other information such as the current network access fee. Upon receipt of a *Beacon* from a router of a domain different from where he currently stays, a user executes the inter-domain authentication protocol if deciding to join the new domain. For example, the user may switch to the new domain with a stronger signal strength or a lower access fee than the old one. Complete specification of conditions impacting a user's switching-domain decision is beyond the paper scope.

We take user U_1 with ($\mathsf{U_1}$-pass, $\mathsf{U_1}$-key) and router R_1 with ($\mathsf{R_1}$-pass, $\mathsf{R_1}$-key) as an example to illustrate the inter-domain authentication protocol. As mentioned before, packet transmissions from R_1 to U_1 are in one hop, while from U_1 to R_1 may take multiple hops. For simplicity, we assume that there is always an uplink path from U_1 to R_1 discovered through the underlying routing protocol. We further assume that U_1 and R_1 have been in possession of each other's authentic domain-params. We want to stress that, for each WMN domain, U_1 needs to retrieve and verify its domain-params certificate for only once. Then U_1 can perform inter-domain authentication with any router in that domain, directly using their R-passes as their public keys. Likewise, knowing the authentic domain-params of a broker would allow a router to authenticate all users holding U-passes issued by that broker. This is one of the beauty of IBC!

The mutual authentication between R_1 and U_1 can be accomplished through the following three-way protocol.

(1) $R_1 \rightarrow *$: $\mathsf{R_1}$-pass, $\mathcal{S}_{\mathsf{R_1}\text{-key}}(t_1)$
(2) $U_1 \rightarrow R_1$: $\mathsf{U_1}$-pass, $\mathcal{S}_{\mathsf{U_1}\text{-key}}(t_2)$
(3) $R_1 \rightarrow U_1$: $\overline{\mathsf{U_1}\text{-pass}}$, $\mathcal{E}_{\mathsf{U_1}\text{-pass}}(\overline{\mathsf{U_1}\text{-key}})$

R_1 transmits message (1) as part of *Beacon* messages that are periodically broadcasted to its coverage area. Here t_1 is a timestamp commonly used to prevent message replay and impersonation attacks [23].

Upon receipt of (1), U_1 does the following in sequence:

1. Check whether the difference between t_1 and his local clock time is within an *acceptance window*[4].
2. Make sure that $\mathsf{R_1}$-pass has not expired by examining the embedded expiry-date.
3. Verify $\mathcal{S}_{\mathsf{R_1}\text{-key}}(t_1)$ with $\mathsf{R_1}$-pass as the public key.

If all the checks succeed, U_1 regards R_1 as a legitimate router. It then unicasts back to R_1 message (2), including $\mathsf{U_1}$-pass, a timestamp t_2 and his signature over t_2, $\mathcal{S}_{\mathsf{U_1}\text{-key}}(t_2)$. Upon receiving (2), R_1 carries out actions analogous[5] to those by U_1. If all the inspections are successful, R_1 determines that U_1 is a legitimate user of the corresponding broker domain.

After authenticating U_1, R_1 contacts its domain administrator for a temporary ($\overline{\mathsf{U_1}\text{-pass}}$, $\overline{\mathsf{U_1}\text{-key}}$) pair,

$$\begin{cases} \overline{\mathsf{U_1}\text{-pass}} := (\overline{\mathsf{U_1ID}}@\mathsf{operator\_domain}, \mathsf{expiry\text{-}date}) \\ \overline{\mathsf{U_1}\text{-key}} := \kappa H_1(\overline{\mathsf{U_1}\text{-pass}}) \,. \end{cases}$$

Here, $\overline{\mathsf{U_1ID}}@\mathsf{operator\_domain}$ is the temporary NAI of U_1 in that WMN domain, expiry-date indicates the expiry date of this temporary user pass, and κ is that WMN domain's domain-master-secret. Subsequently, R_1 sends $\overline{\mathsf{U_1}\text{-pass}}$ in plaintext and $\overline{\mathsf{U_1}\text{-key}}$ encrypted with public key $\mathsf{U_1}$-pass to U_1 in message (3). Upon receiving (3), U_1 decrypts $\overline{\mathsf{U_1}\text{-key}}$ using his private key $\mathsf{U_1}$-key and then checks whether the equation $\hat{e}(\overline{\mathsf{U_1}\text{-key}}, P) = \hat{e}(H_1(\overline{\mathsf{U_1}\text{-pass}}), P_{pub})$ holds, where $\hat{e}$, P and P_{pub} are extracted from the domain-params of the WMN domain. The check should succeed for a valid ($\overline{\mathsf{U_1}\text{-pass}}$, $\overline{\mathsf{U_1}\text{-key}}$) pair due to the following equations:

$$\begin{aligned} \hat{e}(\overline{\mathsf{U_1}\text{-key}}, P) &= \hat{e}(\kappa H_1(\overline{\mathsf{U_1}\text{-pass}}), P) \\ &= \hat{e}(H_1(\overline{\mathsf{U_1}\text{-pass}}), P)^{\kappa} \ (\hat{e} \text{ is bilinear}) \\ &= \hat{e}(H_1(\overline{\mathsf{U_1}\text{-pass}}), \kappa P) \ (\hat{e} \text{ is bilinear}) \\ &= \hat{e}(H_1(\overline{\mathsf{U_1}\text{-pass}}), P_{pub}) \ (P_{pub} = \kappa P). \end{aligned}$$

[4] This can be a fixed-size time interval, e.g., 10 ms or 20 s, preset to account for the maximum message transit and processing time, plus clock skew.

[5] If the aforementioned revocation-list method is used, R_1 also needs to check that $\mathsf{U_1}$-pass is not on the revocation list of U_1's enrolled broker.

After a successful check, U_1 saves ($\overline{\mathsf{U_1\text{-}pass}}$, $\overline{\mathsf{U_1\text{-}key}}$) for subsequent use as his temporary credential in that WMN domain. Router R_1 and its domain administrator may record the mapping between $\mathsf{U_1}$-pass and $\overline{\mathsf{U_1\text{-}pass}}$ if needed. The usefulness of such temporary credentials in intra-domain and user-user authentication will be shown shortly.

After a successful three-way handshake, R_1 and U_1 have implicitly established a shared key

$$\begin{aligned} K_{R_1,U_1} &= \hat{e}(\mathsf{R_1\text{-}key}, H_1(\overline{\mathsf{U_1\text{-}pass}})) \\ &= \hat{e}(H_1(\mathsf{R_1\text{-}pass}), H_1(\overline{\mathsf{U_1\text{-}pass}}))^{\kappa} \\ &= \hat{e}(H_1(\overline{\mathsf{U_1\text{-}pass}}), H_1(\mathsf{R_1\text{-}pass}))^{\kappa} \\ &= \hat{e}(\overline{\mathsf{U_1\text{-}key}}, H_1(\mathsf{R_1\text{-}pass})) = K_{U_1,R_1}. \end{aligned} \tag{2}$$

The above equations hold by the bilinearity and symmetry of $\hat{e}$ (cf. Section 2.1). Here, R_1 (respectively, U_1) derives the shared key using the first line (respectively, fourth line) pairing computation. Due to the difficulty of solving the BDHP, K_{R_1,U_1} is exclusively available to R_1 and U_1 without counting the trustworthy administrator of that WMN domain. Subsequent traffic encryption and authentication between R_1 and U_1 can then realized via K_{R_1,U_1} along with efficient symmetric-key algorithms.

4.2. Intra-domain authentication

Intra-domain authentication occurs when user U_1 moves out of the coverage area of R_1 into that of another mesh router, say R_2 with ($\mathsf{R_2}$-pass, $\mathsf{R_2}$-key), of the same WMN domain. The naive reuse of the inter-domain authentication protocol is less efficient because the existing trust relationship between R_1 and U_1 is not exploited. Another option would be to let R_1 hand over K_{R_1,U_1} to R_2 through a secure channel so that R_2 and U_1 can authenticate each other through a classical symmetric-key challenge-response technique based on K_{R_1,U_1} [23]. Such an approach would cause non-negligible processing burden and communication overhead on mesh routers, especially when the user base is growing large. It is also obviously insecure to constantly employ K_{R_1,U_1} or session keys derived from it to secure the communications between U_1 and multiple or even all routers of the same WMN domain.

Fortunately, U_1 can achieve efficient mutual authentication with R_2 through his temporary credential ($\overline{\mathsf{U_1\text{-}pass}}$, $\overline{\mathsf{U_1\text{-}key}}$) before its expiry date. Also assume that an uplink route from U_1 to R_2 is available. The intra-domain authentication protocol works in three steps as well:

(1) $R_2 \rightarrow *$: $\mathsf{R_2}$-pass, $\mathcal{S}_{\mathsf{R_2\text{-}key}}(t_1)$
(2) $U_1 \rightarrow R_2$: $\overline{\mathsf{U_1\text{-}pass}}$, t_2, $h_{K_{U_1,R_2}}(t_1 \parallel t_2 \parallel \mathsf{R_2\text{-}pass})$
(3) $R_2 \rightarrow U_1$: $h_{K_{R_2,U_1}}(t_1 \parallel t_2 \parallel \overline{\mathsf{U_1\text{-}pass}})$

Message (1) is similar to that of the inter-domain authentication protocol and is broadcasted by R_2 periodically as part of its *Beacon* messages. Up receipt of (1), U_1 knows that R_2 belongs to the same domain as R_1 by inspecting the $\mathsf{R_2}$-NAI of $\mathsf{R_2}$-pass. He then performs the analogous operations he does in the inter-domain authentication protocol. If all the checks succeed, he derives a shared key $K_{U_1,R_2} = \hat{e}(\overline{\mathsf{U_1\text{-}key}}, H_1(\mathsf{R_2\text{-}pass}))$. Then he computes a MIC $h_{K_{U_1,R_2}}(t_1 \parallel t_2 \parallel \overline{\mathsf{U_1\text{-}pass}})$ sent to R_2 in message (2), where t_2 is a timestamp and $\parallel$ denotes concatenation.

Upon receiving (2), router R_2 makes sure that $\overline{\mathsf{U_1\text{-}pass}}$ has not expired and then computes a shared key $K_{R_2,U_1} = \hat{e}(\mathsf{R_2\text{-}key}, H_1(\overline{\mathsf{U_1\text{-}pass}}))$. According to Eq. (2), $K_{R_2,U_1} = \hat{e}(H_1(\overline{\mathsf{U_1\text{-}pass}}), H_1(\mathsf{R_2\text{-}pass}))^{\kappa} = K_{U_1,R_2}$ if and only if both U_1 and R_2 are legitimate. R_2 then recalculates the MIC and compares it with what U_1 sent. If they are equal, R_2 knows that user U_1 must have been authenticated by a peer router in the same domain because U_1 has a valid temporary credential. After that, R_2 computes a new MIC $h_{K_{R_2,U_1}}(t_1 \parallel t_2 \parallel \overline{\mathsf{U_1\text{-}pass}})$ and unicasts it to U_1 in message (3) to prove its knowledge of K_{R_2,U_1}. Upon receiving (3), U_1 regenerates the MIC and checks if the result matches with what R_2 sent. If so, U_1 considers R_2 as an authentic router of the present domain, as it has a valid ($\mathsf{R_2}$-pass, $\mathsf{R_2}$-key) pair.

The intra-domain authentication protocol is computationally more efficient than the inter-domain one due to its replacement of moderately expensive IBC signature and encryption operations with fast hash operations. This is desirable because intra-domain authentication happens much more frequently. Note that, if his $\overline{\mathsf{U_1\text{-}pass}}$ has expired, U_1 has to execute the inter-domain authentication protocol with R_2.

4.3. User-user authentication

User-user authentication in the same mesh is also important because each user should only forward data packets to and from the mesh router for those who are legitimate. Otherwise, he might get unpaid for his packet forwarding service. Users might as well need to establish pairwise shared keys to secure traffic between them. Such shared keys are also the foundation of many secure ad hoc routing protocols [28].

User-user authentication and shared-key establishment can be easily fulfilled based on their ($\overline{\mathsf{U\text{-}pass}}$, $\overline{\mathsf{U\text{-}key}}$) pairs. The reason is that possession of an authentic temporary credential can serve as the proof that the holder has been authenticated by the current WMN domain. Suppose users U_1 and U_2 both have finished mutual inter-domain authentication with the same or different mesh routers of the same domain and received their respective ($\overline{\mathsf{U_1\text{-}pass}}$, $\overline{\mathsf{U_1\text{-}key}}$) and ($\overline{\mathsf{U_2\text{-}pass}}$, $\overline{\mathsf{U_2\text{-}key}}$). Once actively exchanging or passively learning (e.g., from routing messages) the $\overline{\mathsf{U\text{-}pass}}$ of each other, U_1 and U_2 can calculate the same shared key $K_{U_1,U_2} = \hat{e}(H_1(\overline{\mathsf{U_1\text{-}pass}}), H_1(\overline{\mathsf{U_2\text{-}pass}}))^{\kappa}$, similar to what U_1 and R_1 do

in Eq. (2). Subsequently, U_1 and U_2 can authenticate each other with classical symmetric-key challenge-response authentication techniques [23]. For instance, U_1 can send to U_2 a challenge r_1 encrypted with K_{U_1,U_2}. If U_2 can report a correct response, say $(r_1 + 1)$, U_1 declares the authentication of U_2 successful. Similarly, U_2 can authenticate U_1.

Owning an authentic temporary credential permits a user to achieve mutual authentication and establish a pairwise shared key with all the other users present in the same WMN domain. It is also worth noting that user-user authentication and shared-key establishment can be done in an on-demand manner, e.g., when U_1 and U_2 become neighbors, or U_1 is helping U_2 transmit traffic to and from the mesh router.

User-user authentication is expected to occur much more frequently than intra-domain authentication. This is due to the dynamic user join and leave, and the frequent uplink route changes caused by user mobility inside a mesh. In light of this, the advantages of our IBC-based temporary user passes over their longer certificate-based alternatives are more substantial.

4.4. Discussion

In our UPASS, to allow freedom of involvement, a mobile user is not punished for refusing to relay others' authentication and subsequent data packets to a mesh router. For those intending to be packet forwarders in return for monetary earnings, we make an important assumption that they should directly relay message (2) of both the inter-domain and intra-domain authentication protocols to a mesh router without performing message authenticity checking. We note that this assumption might introduce room for a special DoS attack, in which attackers continuously send a number of faked authentication messages via innocent intermediate users to a mesh router. This attack bears similarity to the physical-layer jamming DoS attack, and is a potential threat against any feasible entity authentication scheme for wireless networks with multi-hop uplinks. From our point of view, it is very difficult, if not impossible, to figure out an effective countermeasure. In addition, the aftermath of this attack is rather localized to the attacked mesh and attackers generally gain nothing. Therefore, this attack is less likely to be launched by rational attackers who only attempt to misbehave if benefiting from doing so. We leave the investigation of possible solutions or at least damage-limiting measures as our future work.

5. Incontestable billing of mobile users

Once finishing mutual inter- or intra-domain authentication with a mesh router, a user can start to access the network through it. In this section, we present a realtime micropayment approach to realize incontestable billing of mobile users for receiving network access services.

5.1. Billing basics

We assume that each WMN operator has two network access rates, λ and γ monetary units (*m-unit*s) per traffic unit (*t-unit*), say 0.05 and 0.01 cents/KB. In particular, a user needs to pay the network operator and each intermediate user λ and γ m-units, respectively, for each t-unit received or transmitted through them. Different WMN operators may have diverse access rates and each operator may also dynamically adjust its access rates. For example, λ and γ can be set higher during busy hours, while lower during idle hours. An operator even can enforce various charging rates for mesh routers deployed in different locations. All we require is that each mesh router should include its current λ, γ values in periodically broadcasted *Beacon* messages. These two are usually important inputs to a user's decision-making process as to whether to join a WMN domain. Also note that our UPASS can be easily extended to adopt a time-based rather than traffic-based charging method, which is omitted for brevity.

In what follows, we take user U_1 and router R_1 as an example to illustrate our *session*-based billing scheme. A session begins when a new uplink route from U_1 to R_1 is established and terminates when the route breaks due to reasons such as user mobility. We also assume the existence of a secure routing protocol that finds a *valid* uplink route. Many existing secure ad hoc routing protocols such as Ariadne [17] or ARAN [29] can serve this purpose after minor modifications. We further postulate that router R_1 can reliably verify that each intermediate user indeed participates in forwarding each packet from U_1. This can be fulfilled, for example, by asking each intermediate user to attach to each forwarded packet a MIC calculated under its pairwise shared key with R_1 established during mutual authentication. After verification of the received MICs, R_1 can ascertain that the corresponding intermediate users indeed participated in forwarding the packet for U_1. Due to space limitations, we will not dwell on this point hereafter.

In concurrent on-demand ad hoc routing protocols such as AODV [19] or its secure version ARAN [29], a multihop route is finally chosen by the intended destination, which is router R_1 in our case. Suppose R_1 selects an uplink route with n intermediate users and informs U_1 about it. Then U_1 can decide that he totally needs to pay $\mathsf{rate}_{\mathsf{up}} := \lambda + n\gamma$ m-units per t-unit transmitted via the multihop uplink and λ m-units per t-unit received via the single-hop downlink. The uplink charging rate $\mathsf{rate}_{\mathsf{up}}$ varies across sessions with different uplink route lengths. Whenever a new session begins due to a newly discovered uplink route, R_1 should inform U_1 about this. Here, we assume that the WMN operator does not collude with intermediate users to cheat U_1 in the sense that R_1

always selects the cheapest route for U_1 allowed by the underlying routing metric. For instance, if the hop count is the routing metric, R_1 will always pick the shortest (i.e., cheapest) route for U_1. This assumption is reasonable because the operator is always paid with a constant rate of λ m-units/t-unit for both uplink and downlink traffic, independent of the route length.

There is a possible attack launched by collusive users. In particular, collusive users within the same mesh first exchange certain cryptographic materials such as permanent or temporary passes, pass-based keys and the pairwise keys shared with router R_1. The purpose is to make each of them able to *emulate* all the other conspirators, i.e., to act as several consecutive users but only incurring the communication cost of a single user. If successfully performed, this emulation attack may cause U_1 to pay more than what he ought to pay. We note that this attack may be possible only when an emulator resides on the uplink route discovered via the underlying secure routing protocol. For example, if the emulator acts as too many conspirators, leading to a long uplink route length, the trustworthy R_1 will select other routes with shorter lengths. This is very likely to happen because of the usual availability of multiple candidate routes from U_1 to R_1. Therefore, the damage of the emulation attack might be rather limited. To deal with the case that the emulator is on the uplink route, the best known countermeasure is through statistical approaches proposed by Jakobsson et al. [4] and Salem et al. [5]. For lack of space, we refer interested readers to [4] and [5] for details.

5.2. Payment structures

We now define an important data structure called a *payment structure* used in our billing process. Let $\mathsf{D}_{\mathsf{U}_1\to\mathsf{R}_1} := \langle \mathsf{R}_1\text{-NAI}, \text{expiry-date}, L, a_1, t, m\rangle$. A payment structure is defined as follows:

$$\langle \mathcal{S}_{\mathsf{U}_1\text{-key}}(\mathsf{D}_{\mathsf{U}_1\to\mathsf{R}_1}), \langle a_m\rangle, \langle w_{1,t}\rangle, \langle w_{2,t}\rangle, \ldots, \langle w_{m,t}\rangle\rangle.$$

Expiry-date specifies the expiry date of this payment structure before which it is redeemable at U_1's enrolled broker. Figure 3 depicts an exemplary payment structure for $m \geq 3$ and $t \geq 2$.

We write $\langle a_m\rangle$ for m hash values $\{a_i | 1 \leq i \leq m\}$ generated as follows: U_1 first picks a random number a_m and then recursively computes $a_i = h(a_{i+1})$ for $i = m-1, m-2, \ldots, 1$. Due to the one-way feature of the hash function h, if a_m is chosen randomly, given a_{i-1} it is computationally infeasible to find a_i, while given a_i it is computationally efficient to derive a_{i-1}. Each $\langle w_{i,t}\rangle$ $(1 \leq i \leq m)$ denotes t hash values $\{w_{i,j} | 1 \leq j \leq t\}$ generated by U_1 in the similar way, where each $w_{i,t}$ is chosen at random. The chain-length parameters m, t are selected at U_1's convenience, the choice of which

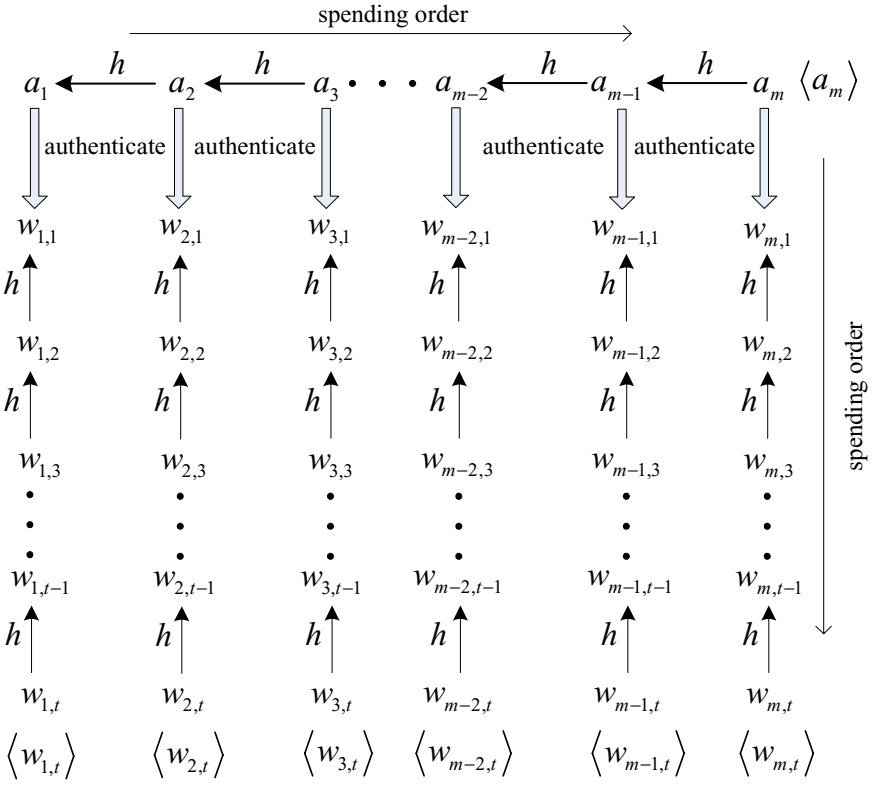


Fig. 3 An exemplary payment structure ($m \geq 3, t \geq 2$)

will be discussed shortly. We also refer to a_m and $w_{i,t}$ to as the *root*s of $\langle a_m \rangle$ and $\langle w_{i,t} \rangle$, respectively.

$\mathcal{S}_{\mathsf{U_1\text{-}key}}(\mathsf{D}_{\mathsf{U_1}\to\mathsf{R_1}})$ is U_1's signed commitment to his payment structure for R_1, and should be sent to R_1 before starting any session. For example, U_1 can send it as part of its authentication message to R_1. Upon recept of it, R_1 first verifies the signature using $\mathsf{U_1\text{-}pass}$ as U_1's public key and, if successful, saves it for subsequent verification of payments from U_1. We require R_1 to acknowledge the recept of $\mathcal{S}_{\mathsf{U_1\text{-}key}}(\mathsf{D}_{\mathsf{U_1}\to\mathsf{R_1}})$.

Each $\langle w_{i,t} \rangle$ is called a *payment chain*, of which each $w_{i,j}$ is termed a *payment token* and worth L m-units. The payment tokens are spent in order, but not necessarily consecutively. In other words, once U_1 spends $w_{i,j}$, he cannot spend $w_{i,k}$ for $k < j$. The m payment chains do not need to be generated simultaneously at the beginning. Instead, U_1 can defer the generation of $\langle w_{i+1,t} \rangle$ until payment tokens of $\langle w_{i,t} \rangle$ are used up. By comparison, $\langle a_m \rangle$ is referred to as a *proof chain* and used to provide efficient authentication of payment-chain roots. Elements of $\langle a_m \rangle$ are called *proof token*s, and are not only used in order but also consecutively: a_1 first, then a_2, and so forth. Note that, once used, a payment or proof token can be dumped by U_1 to save storage space.

We take a concrete example to explain how a proof token a_i is used to authenticate root $w_{i,1}$ of $\langle w_{i,t} \rangle$. Recall that user U_1 has sent the authenticated a_1 to router U_1. To spend payment tokens of $\langle w_{1,t} \rangle$, U_1 first sends $(w_{1,1}, h_{a_1}(w_{1,1}))$ to R_1. We view a_1 as a one-time password of U_1 and thus $h_{a_1}(w_{1,1})$ as a MIC. Upon receipt of the message, R_1 recalculates the MIC and checks the result against what U_1 sent. If the two are equal, R_1 knows that $w_{1,1}$ indeed came from U_1 and then saves it for subsequent verification of payment tokens of $\langle w_{1,t} \rangle$. Suppose U_1 has used up payment tokens of $\langle w_{i,t} \rangle$ and wants to use $\langle w_{i+1,t} \rangle$ for $i \geq 1$. To do so, he sends to router R_1 a triplet $(a_{i+1}, w_{i+1,1}, h_{a_{i+1}}(w_{i+1,1}))$ as a commitment to $\langle w_{i+1,m} \rangle$. Upon receiving it, R_1 first checks whether $a_{i+1} = h(a_i)$. If so, R_1 determines that a_{i+1} was sent by U_1 because nobody else is able to forge a_{i+1} that can pass the check, due to the one-way feature of $\langle a_m \rangle$. Subsequently, R_1 recomputes the MIC $h_{a_{i+1}}(w_{i+1,1})$. If the result matches with what U_1 sent, R_1 knows that $w_{i+1,1}$ is a valid root which can be used to verify subsequent payment tokens from $\langle w_{i+1,1} \rangle$. It is worth point out that R_1 just needs to memorize the highest-indexed proof token from $\langle a_m \rangle$. In addition, R_1 is required to acknowledge the receipt of $(a_{i+1}, w_{i+1,1}, h_{a_{i+1}}(w_{i+1,1}))$.

Here may come a question: why should we use m payment chains of size t instead of a single one of size tm? The reason is that doing so imposes a much smaller storage requirement on U_1. In particular, the single-chain approach requires U_1 to store about $tm/2$ payment tokens on average during the payment process. Suppose SHA-1 [26] is used as h and each of payment and proof tokens is a SHA-1's 20-byte output. Also assume that L, m and t are equal to 1, 50 and 100, respectively. This means that a single payment chain provides a total worth of 5000 m-units, while requiring an average space of about 50 KB. In contrast, using our payment structure allows U_1 to store just $m/2$ proof and $t/2$ payment tokens on average, representing an average storage overhead of only about 1.5 KB. In addition, employing shorter payment chains can minimize the waste coming from unspent hash tokens. Such storage savings come at the cost of some service delay caused by generating a new payment chain in realtime. However, since the hash operation is very fast and a hash chain with 1000 tokens can be derived in less than one second [3] even in low-end devices, such a delay is believed to be affordable. Also notice that a new payment-chain commitment (triplet) can be transmitted along with regular data packets so that the extra communication overhead can be minimized.

A payment structure is both *user-specific* and *router-specific* and thus is of no value to another user or router. It is also *session-independent* in that U_1 can use it across different sessions with R_1. A payment structure supports the generation of up to m payment chains of size t. Once all m payment chains are used up, a new payment structure needs to be generated if needed. Since generating a new payment structure involves a signature generation on U_1 and a signature verification on R_1, respectively, we suggest using a slightly larger m to reduce moderately expensive signature operations.

5.3. Making payments

In what follows, we first discuss how user U_1 pays router R_1 and then intermediate users along the uplink route.

Paying routers

To make payments to R_1, U_1 maintains a *debt counter* DC_{U_1} recording the amount in m-units he owes to R_1. DC_{U_1} is increased by λ for each downlink t-unit and by $\mathsf{rate_{up}}$ for each uplink t-unit. Accordingly, R_1 maintains for U_1 a *profit counter* PC_{U_1} which is increased by λ and $\mathsf{rate_{up}}$ for each t-unit sent to and received from U_1, respectively.

We require that R_1 specify in its periodically broadcasted *Beacon* messages a parameter θ_{R_1}, indicating the maximum amount in m-units that each user is allowed to owe it. Whenever $DC_{U_1} \geq \theta_{R_1}$, U_1 should make a payment to clear its debt at R_1 in due time to avoid service cutoff by R_1. Without loss of generality, suppose U_1 is spending payment tokens of $\langle w_{i,t} \rangle$. For ease of presentation, we temporarily assume that $\langle w_{i,t} \rangle$ still has enough unspent payment tokens. If the lowest-indexed unspent token is $w_{i,u}$, U_1 sends to R_1 a payment of format $(w_{i,j}, j)$, where $u \leq j \leq t$ is the minimum integer such that $(j - u + 1)L \geq \theta_{R_1}$. He then decreases DC_{U_1} by $(j - u + 1)L$ and thus DC_{U_1} may be a negative value sometimes. Since the worth L of each payment token is usually of a small amount, say several cents, we refer to each payment like $(w_{i,j}, j)$ as a *micropayment*.

For each payment chain $\langle w_{i,t}\rangle$, router R_1 merely needs to store the payment token with the highest index, say $(w_{i,k}, k)$ $(1 \le k \le t)$. This means that R_1 has been paid kL m-units by U_1 using $\langle w_{i,t}\rangle$ and $((i-1)t+k)L$ m-units in all. Upon receipt of $(w_{i,j}, j)$, R_1 first verifies that $j > k$ and then $w_{i,k} = h^{j-k}(w_{i,j})$, where h^{j-k} means applying the hash function h iteratively to $w_{i,j}$ for $(j-k)$ times. If both checks succeed, R_1 knows that U_1 indeed made a payment because nobody else can generate a valid payment token passing the checks, due to the one-way feature of $\langle w_{i,t}\rangle$. Subsequently, R_1 replaces $(w_{i,k}, k)$ with $(w_{i,j}, j)$ and decreases PC_{U_1} by $(j-k)L$.

Assume that R_1 sets a threshold $\theta^*_{R_1}$ and stops serving U_1 if it does not receive a payment in the first data packet from U_1 once $PC_{U_1} \ge \theta^*_{R_1}$. This may happen either because U_1 does not make a payment at all, or because a payment gets lost on its way to R_1, for example, due to a route break. Fortunately, the hash-chain technique can well tolerate payment losses. For instance, suppose R_1 does not receive $(w_{i,j}, j)$ but a later payment $(w_{i,l}, l)$ for $l > j$. If $l > k$ and $w_{i,k} = h^{l-k}(w_{i,l})$, R_1 can change $(w_{i,k}, k)$ to $(w_{i,l}, l)$ and decrease PC_{U_1} by $(l-k)L$. Obviously, this is equivalent to R_1 having correctly received both $(w_{i,j}, j)$ and $(w_{i,l}, l)$. To leverage this loss-tolerance feature, however, $\theta^*_{R_1}$ should be set larger than θ_{R_1}. The difference between $\theta^*_{R_1}$ and θ_{R_1} determines the trade-off between payment-loss tolerance and the financial risk of the operator. The larger the difference, the more payment losses R_1 can tolerate, the higher financial risk the operator runs because R_1 may not make a payment at all, and vice versa.

If the remaining tokens of $\langle w_{i,t}\rangle$ are not enough to cover DC_{U_1}, U_1 should generate a new payment chain $\langle w_{i+1,t}\rangle$. It then sends the new chain commitment $(a_{i+1}, w_{i+1,1}, h_{a_{i+1}}(w_{i+1,1}))$ to R_1 which, in turn, verifies the commitment as described in Section 5.2. Subsequently, U_1 can delete unspent payment tokens of $\langle w_{i,t}\rangle$ if any and start to pay R_1 with payment tokens of $\langle w_{i+1,t}\rangle$.

At last, R_1 is required to store a payment record for U_1 of format

$$\langle \mathcal{S}_{\mathsf{U_1\text{-}key}}(\mathsf{D}_{\mathsf{U_1}\to\mathsf{R_1}}), a_k, \{(w_{i,1}, h_{a_i}(w_{i,1}), w_{i,k_i}, k_i | 1 \le i \le k\}\rangle.$$

Here, a_k $(1 \le k \le m)$ refers to the highest-indexed proof token and w_{i,k_i} $(1 \le k_i \le t)$ is the highest-indexed payment token from $\langle w_{i,t}\rangle$. In rare cases, if U_1 has generated and used multiple payment structures, R_1 should maintain such a record for each of them.

Paying intermediate users

We now discuss how to pay intermediate users using the hash-chain technique. A naive way is for U_1 to generate a payment structure for each intermediate user and release payment tokens at pre-defined intervals, as he does for R_1. Such an approach has three significant drawbacks. First of all, it is computationally inefficient. For U_1, he has to generate multiple payment structures and thus perform multiple signature generations. Once the uplink route breaks, he has to redo these operations for newly-joined intermediate users on the new route. Each intermediate user has to first verify a signature and then each subsequent proof or payment token. Since a user may act as packet forwarders for multiple users simultaneously, he has to do these operations for each of them. Secondly, it is communicationally inefficient in that U_1 must release multiple hash tokens at one time according to pre-defined intervals. Lastly, it is space inefficient because U_1 has to maintain multiple payment structures at the same time, and each user needs to maintain at least one payment record for all the other users with him as a packet relay.

To minimize the burden of mobile users, we propose to let R_1 pay intermediate users on behalf of U_1. This is the reason why a payment from U_1 to R_1 covers all what R_1 and all the intermediate users should get. Consider an intermediate user U_2 as an example. After authenticating U_2, R_1 generates a payment structure for U_2 and sends to him the signed commitment to the payment structure. Once verifying R_1's signature, U_2 saves the commitment for later verification of payment and proof tokens sent by R_1. The payment structure is also both user-specific and router-specific, and is used by R_1 to pay U_2 for all the traffic he forwards for all the other users in R_1's coverage area. The detailed payment process is similar to that of U_1 and omitted here due to space constraints.

5.4. Redemption of payment structures

All payment structures should be redeemed at the users' enrolled brokers before their expiry dates. At the end of each day (or other suitable period), R_1 reports all the stored payment records to its domain operator who, in turn, assembles the records related to a same broker and sends them in bulk.

For each submitted payment record as $\langle \mathcal{S}_{\mathsf{U_1\text{-}key}}(\mathsf{D}_{\mathsf{U_1}\to\mathsf{R_1}}), a_k, \{(w_{i,1}, h_{a_i}(w_{i,1}), w_{i,k_i}, k_i | 1 \le i \le k\}\rangle$, a broker does the following in sequence:

(1) Examine $\mathcal{S}_{\mathsf{U_1\text{-}key}}(\mathsf{D}_{\mathsf{U_1}\to\mathsf{R_1}})$, including verifying the user's signature, checking the expiry-date, and so on.
(2) Check that $a_1 = h^{k-1}(a_k)$ and saves the intermediary values $a_{k-1}, \ldots, a_2$. For each $i \in [1, k]$,
(3) Calculate a MIC $h_{a_i}(w_{i,1})$. If the result matches the corresponding value in the submitted record,
(4) Check that $w_{i,1} = h^{k_i-1}(w_{i,k_i})$ and, if successful, credit the operator's account with $k_i L$ m-units.

If the operator has no account at the broker corresponding to a payment record, it can redeem the payment record at its own enrolled broker that will interact with the corresponding

broker on behalf of it. Then there would be some money transfer between the two brokers, analogous to what happens in daily life when one deposits some checks issued by banks other than his enrolled bank. Likewise, mobile users can redeem their payment records stored for operators at the brokers.

5.5. Security analysis

Our micropayment approach ensures incontestable billing. For a user, he must digitally-sign a payment structure before using it to pay a WMN operator, so he cannot deny the payments he makes later. In addition, the user cannot obtain more services than he will actually be billed for, as he is required to release payment tokens in realtime at predefined intervals to avoid service cutoff by the operator. For an operator, it cannot overcharge the user who releases valid payment tokens commensurate with the amount of received services. Since a payment structure is both user-specific and router-specific, it also prevents from both *double-spending* and *double-redemption* of a payment structure. In particular, the user cannot use the same payment structure to pay different routers; the operator can redeem the same payment structure of a user only once via that user's registered broker.

Note that our billing scheme cannot completely prevent from cheating by a user or an operator, which might happen only at the end of each service duration. For example, in one case, user U_1 does not pay for the last few t-units received or transmitted via router R_1, e.g., by leveraging the difference between θ_{R_1} and $\theta^*_{R_1}$. In the other case, R_1 does not serve U_1 for the last payment he made, if U_1 is asked to prepay payment tokens. In both cases, the financial loss (or gain) of the user or the broker is less significant, say several m-units. Considering the similar situation in cellular networks where an operator usually enforces a basic charging unit, e.g., 6 seconds, we believe that such rare cheating situations should be tolerable.

Regarding the payment process from an operator (through a router like R_1) to a user, say U_2, we argue that the operator would have the right incentive to behave honestly. The reason is that, if not receiving payments from R_1 in due time, U_2 will stop forwarding packets for other users within R_1's coverage. If this happens frequently, the affected users who experience frequent service disruptions will heap all blames on the operator. Both those users and U_2 will choose to shun that operator in the future. Since the operator's reputation is worth much more than what it can earn from cheating, it would rather not to do so. Other security analysis is similar to that of the payment process from a user to an operator, which is omitted here for lack of space.

6. Overhead analysis of UPASS

In this section, we analyze the computation, storage and communication overhead of the proposed UPASS.

6.1. Computation overhead

In our UPASS, users, mesh routers and brokers are required to occasionally perform a few public-key operations, including CBC signature verifications[6], IBC signature and encryption operations, and pairing computations[7]. A few years ago, this computational requirement was significant, especially on the user's side. But with the rapid progress in both CBC and IBC, public-key encryption and signature schemes that are both more secure and significantly faster are currently available. Moreover, the computational costs of public-key operations have continued to decrease due to the rapid development of hardware implementations. For example, researchers have recently announced FPGA implementations of both RSA [30] and the pairing [31] in several milliseconds. We are also aware of efficient implementations of the pairing on smartcards [32, 33]. It is important to note that such public-key operations are executed relatively rarely. Once finishing mutual authentication, two users or a user and a router can secure subsequent traffic between them using the established shared key along with efficient symmetric-key algorithms. Moreover, billing of mobile users involves only fast hash operations, except the few IBC signature operations for generating and verifying payment-structure commitments. Therefore, we believe that the computation overhead of our UPASS is rather affordable even on the possibly resource-constrained user's side.

6.2. Storage overhead

Our UPASS requires a user and a router to allocate space for the implementations of IBC signature and encryption primitives, a CBC signature primitive such as RSA [34] and a hash function such as SHA-1 [26]. With modern technology, all those can be implemented within a few tens of KB. Regarding the billing scheme, it takes about 1.5 KB on average to store a payment structure for $m = 50$ and $t = 100$, as analyzed in Section 5.2. A similar memory size is required for storing a payment record. Such small storage overhead is affordable even for modern low-end mobile devices like PocketPCs or PDAs with a usual RAM (Random Access Memory) size of several tens of MB, not to mention for a user's laptop and a powerful mesh router.

[6] These are needed when a router (or a user) wishes to verify the domain-params certificate of a broker domain (or an operator domain).

[7] The pairing computation by far takes the most running time of an IBC cryptographic primitive.

6.3. Communication overhead

In our UPASS, entity authentication and billing involve only local interactions between users and mesh routers, without realtime involvement of any third party such as a broker. Therefore, our UPASS is communicationally much more efficient than conventional schemes based on the home-foreign-domain model. The release of hash tokens in the billing scheme incurs certain communication overhead, but such overhead is proportional to the traffic volume supported. For example, a 20-byte hash token can be released per 25 or 50 KB data traffic, representing an overhead of about 0.08 or 0.04 percent that is considered to be acceptable. In addition, the transmission of accumulated payment records at a WMN operator or a user to a broker can be done at pre-scheduled intervals such as per week or month, instead of in realtime. The resulting communication overhead is also very small.

7. Related work

In this section, we review prior work that is closely related to our UPASS. Patel and Crowcroft removes the reliance on a home domain and proposes a ticket-based service access approach for a mobile user [35]. Although their ticket concept is similar to our pass notion, our UPASS differs significantly from [35] in the pass design, the trust model, the authentication and payment approaches, and the application context.

Some efforts have also been made in recent years to foster cooperation in packet forwarding in infrastructure-supported multihop wireless networks. Zhong et al. propose a credit-based scheme, called Sprite, for mobile ad hoc networks with access to a backbone [6]. While eliminating the conventional need for tamper-resistant hardware, Sprite has several drawbacks regarding its overhead, security and topology requirements, as noted in [4]. Jakobsson et al. [4] and Salem et al. [5] propose different payment-based schemes to encourage packet forwarding in multihop cellular networks. Both work still depends on the home-foreign-domain model and the realtime collaboration of network operators.

One-way hash chains have been adopted previously to make electronic payments of small amounts, called *micropayments*. The main purpose is to avoid high transaction overhead (in comparison with the value of payment) such as bank processing fees associated with traditional macropayment (e.g., credit-card based) approaches [36]. Due to its lightweight nature, such a micropayment technique has been applied by a few researchers to the mobile wireless setting [3, 37, 38]. Since network access fees are of growingly small amount (e.g., 0.05 cents/KB), our UPASS takes a similar approach to bill mobile users. However, the consideration of efficiently paying both network operators and packet forwarders distinguishes our billing scheme significantly from previous schemes.

8. Conclusion

In this paper, we present UPASS, the first known secure authentication and billing architecture for large-scale WMNs. Unlike conventional home-foreign-domain proposals, our UPASS is a *homeless* solution and eliminates the need for establishing bilateral SLAs and having realtime interactions between a potentially huge number of WMN operators. With our UPASS in place, each user is no longer bound to a specific network operator. Instead, he can achieve efficient mutual authentication with any visited WMN domain and thus get ubiquitous network access by a universal pass designed under a novel hybrid IBC/CBC trust model. In addition, UPASS features a lightweight realtime micropayment approach to realize incontestable billing of mobile users. Our UPASS is secure and lightweight, and can serve as a practical and effective solution for future large-scale WMNs.

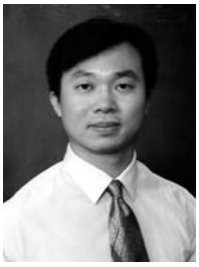
As the future research, we will study faster inter-domain and intra-domain authentication methods using a cross-layer design paradigm. In addition, we plan to explore mobility management issues under our UPASS architecture. Finally, we will seek efficient solutions based on UPASS to other security issues such as secure routing, DoS attacks and worms.

Acknowledgments This work was supported in part by the U.S. Office of Naval Research under Young Investigator Award N000140210464 and the U.S. National Science Foundaion under grants ANI-0093241 (CAREER Award and DBI-0529012).

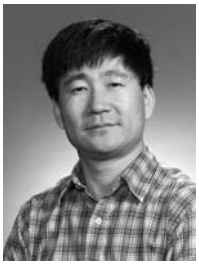
References

1. I. Akyildiz, X. Wang and W. Wang, "Wireless mesh networks: A survey," *Computer Networks* (March 2005).
2. The WiMAX Forum. http://www.wimaxforum.org.
3. J. Zhou and K. Lam, "Undeniable billing in mobile communication," in: *ACM MobiCom'98*, Dallas, TX (Oct. 1998).
4. M. Jakobsson, J.-P. Hubaux and L. Buttyan, "A micro-payment scheme encouraging collaboration in multi-hop cellular networks," in: *7th Int. Conf. Financial Cryptography (FC'03)*, Gosier, Guadeloupe (Jan. 2003).
5. N. Salem, L. Buttyan, J. Hubaux and M. Jakobsson, "A charging and rewarding scheme for packet forwarding in multi-hop cellular networks," in: *ACM MobiHoc*, Annapolis, Maryland (June 2003).
6. S. Zhong, J. Chen and Y. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in: *IEEE INFOCOM*, San Francisco, CA (April 2003).
7. Y. Zhang, W. Lou and Y. Fang, "SIP: A secure incentive protocol against selfishness in mobile ad hoc networks," in: *IEEE WCNC*, Atlanta, GA (March 2004).
8. European Telecommunications Standards Institute (ETSI), "GSM 2.09: Security aspects" (June 1993).

9. H. Lin and L. Harn, "Authentication protocols for personal communication systems," in: *ACM SIGCOMM'95*, Cambridge, MA (Aug./Sept. 1995).
10. 3GPP TS 21.102, 3rd Generation Partnership Project (3GPP); Technical Specification Group (TSG) SA; 3G Security; Security Architecture, version 4.2.0, Release 4 (2001).
11. Y. Lin and Y. Chen, "'Reducing authentication signalling traffic in third-generation mobile network," in: *IEEE Trans. Wireless Commun.*, Vol. 2, No. 3 (May 2003) pp. 493–501.
12. C. Perkins, "IP mobility support for IPv4," RFC 3344 (Aug. 2002).
13. L. Lamport, "Password authentication with insecure communication," in: *Comm. of the ACM*, Vol. 24, No. 11 (Nov. 1981) pp. 770–772.
14. D. Boneh and M. Franklin, "Identify-based encryption from the weil pairing," in: *Proc. CRYPTO'01*, ser. LNCS, Vol. 2139. Springer-Verlag (2001) pp. 213–229.
15. P. Barreto, H. Kim, B. Bynn and M. Scott, "Efficient algorithms for pairing-based cryptosystems," in: *Proc. CRYPTO'02*, ser. LNCS, Vol. 2442. Springer-Verlag (2002) pp. 354–368.
16. M. Cagalj, S. Ganeriwal, I. Aad and J.-P. Hubaux, "On selfish behavior in csma/ca networks," in: *IEEE INFOCOM'05*, Miami, FL (March 2005).
17. Y.-C. Hu, A. Perrig and D.B. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," in: *ACM MobiCom*, Atlanta, GA (Sept. 2002).
18. W. Ma and Y. Fang, "Dynamic hierarchical mobility management strategy for mobile ip networks," in: *IEEE J. Select. Areas Commun.*, Vol. 22, No. 4 (May 2004) pp. 664–676.
19. C. Perkins, E. Belding-Royer and S. Das, "Ad hoc on-demand distance vector (AODV) routing," RFC 3561 (July 2003).
20. D. Johnson and D. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Networks*. Kluwer Academic Publishers, Vol. 353 (1996) pp. 153–181.
21. D. Harkins and D. Carrel, "The Internet key exchange (IKE)," RFC 2409 (Nov. 2003).
22. D. Smetters and G. Durfee, "Domain-based administration of identity-based cryptosystems for secure email and ipsec," in: *Proc. 12th USENIX Security Symposium*, Washington, DC (Aug. 2003).
23. A. Menezes, P. van Oorschot and S. Vanston, *Handbook of Applied Cryptography*. CRC Press (1996).
24. B. Aboda and M. Beadles, "The network acces identifier," RFC 2486 (Jan. 1999).
25. ITU-T Recommendations X.509, "Authentication framework," Geneva (1989).
26. NIST, "Digital hash standard," Federal Information Processing Standards PUBlication 180-1 (April 1995).
27. R. Dutta, R. Barua and P. Sarkar, "Pairing-based cryptography : A survey," Cryptology ePrint Archive Report 2004/064 (2004).
28. Hu and A. Perrig, "A survey of secure wireless ad hoc routing," in: *IEEE Security & Privacy*, Vol. 2, No. 3 (May–June 2004) pp. 28–39.
29. K. Sanzgiri, D. LaFlamme, B. Dahill, B. Levine, C. Shields and E. Belding-Royer, "'Authenticated routing for ad hoc networks," in: *IEEE J. Select. Areas Commun.*, Vol. 23, No. 3 (March 2005) pp. 598–610.
30. O. Nibouche, M. Nibouche, A. Bouridane and A. Belatreche, "Fast architectures for fpga-based implementation of rsa encryption algorithm," in: *IEEE Int. Conf. Field-Programmable Technology*, Brisbane, Australia (Dec. 2004).
31. T. Kerins, W. Marnane, E. Popovici and P. Barreto, "Efficient hardware for the tate pairing calculation in characteristic three," in: *Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)*, Edinburgh, Scotland (Aug./Sept. 2005).
32. Gemplus, http://www.gemplus.com/press/archives/2004/id_security/02-11-2004-Identity-Based_Encryption.html.
33. G. Bertoni, L. Chen, P. Fragneto, K. Harrison and G. Pelosi1, Computing tate pairing on smartcards, White Paper, STMicroelectronics, (2005). [Online]. Available: http://www.st.com/stonline/products/families/smartcard/ast_ibe.htm
34. R. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," in: *Communications of the ACM*, Vol. 21, No. 2 (Feb. 1978) pp. 120–126.
35. B. Patel and J. Crowcroft, "Ticket based service access for the mobile user," in: *ACM MobiCom'97*, Budapest, Hungary (Sept. 1997).
36. R. Rivest and A. Shamir, "Payword and MicroMint: Two simple micropayment schemes," in: *Proc. Int. Workshop on Security Protocols*, ser. LNCS, Vol. 1189. Springer-Verlag (1996) pp. 69–87.
37. H. Tewari and D. O'Mahony, "Real-time payments for mobile IP," in: *IEEE Commun. Mag.*, Vol. 41, No. 2 (Feb. 2003) pp. 126–136.
38. ——, "Multiparty micropayments for ad-hoc networks," in: *IEEE WCNC'03*, New Orleans, LA (March 2003).



Yanchao Zhang received the B.E. degree in Computer Communications from Nanjing University of Posts and Telecommunications, Nanjing, China, in July 1999, and the M.E. degree in Computer Applications from Beijing University of Posts and Telecommunications, Beijing, China, in April 2002. Since September 2002, he has been working towards the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, Florida, USA. His research interests are network and distributed system security, wireless networking, and mobile computing, with emphasis on mobile ad hoc networks, wireless sensor networks, wireless mesh networks, and heterogeneous wired/wireless networks.



Yuguang Fang received the BS and MS degrees in Mathematics from Qufu Normal University, Qufu, Shandong, China, in 1984 and 1987, respectively, a Ph.D degree in Systems and Control Engineering from Department of Systems, Control and Industrial Engineering at Case Western Reserve University, Cleveland, Ohio, in January 1994, and a Ph.D degree in Electrical Engineering from Department of Electrical and Computer Engineering at Boston University, Massachusetts, in May 1997.

From 1987 to 1988, he held research and teaching position in both Department of Mathematics and the Institute of Automation at Qufu Normal University. From September 1989 to December 1993, he was a teaching/research assistant in Department of Systems, Control and Industrial Engineering at Case Western Reserve University, where he held a research associate position from January 1994 to May 1994. He held a post-doctoral position in Department of Electrical and Computer Engineering at Boston University from June 1994 to August 1995. From September 1995 to May 1997, he was a research assistant in Department of Electrical and Computer Engineering at Boston University. From June 1997 to July 1998, he was a Visiting Assistant Professor in Department of Electrical Engineering at the University of Texas at Dallas. From July 1998 to May 2000, he was an Assistant Professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology, Newark, New Jersey. In May 2000, he joined the Department of Electrical and Computer Engineering at University of Florida, Gainesville, Florida, where he got early promotion to Associate Professor with tenure in August 2003, and to Full Professor in August 2005. His research interests span many areas including wireless networks, mobile computing, mobile communications, wireless security, automatic control, and neural networks. He has published over one hundred and fifty (150) papers in refereed

professional journals and conferences. He received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002. He also received the 2001 CAST Academic Award. He is listed in Marquis Who's Who in Science and Engineering, Who's Who in America and Who's Who in World.

Dr. Fang has actively engaged in many professional activities. He is a senior member of the IEEE and a member of the ACM. He is an Editor for IEEE Transactions on Communications, an Editor for IEEE Transactions on Wireless Communications, an Editor for IEEE Transactions on Mobile Computing, an Editor for ACM Wireless Networks, and an Editor for IEEE Wireless Communications. He was an Editor for IEEE Journal on Selected Areas in Communications: Wireless Communications Series, an Area Editor for ACM Mobile Computing and Communications Review, an Editor for Wiley International Journal on Wireless Communications and Mobile Computing, and Feature Editor for Scanning the Literature in IEEE Personal Communications. He has also actively involved with many professional conferences such as ACM MobiCom'02 (Committee Co-Chair for Student Travel Award), MobiCom'01, IEEE INFOCOM'06, INFOCOM'05 (Vice-Chair for Technical Program Committee), INFOCOM'04, INFOCOM'03, INFOCOM'00, INFOCOM'98, IEEE WCNC'04, WCNC'02, WCNC'00 (Technical Program Vice-Chair), WCNC'99, IEEE Globecom'04 (Symposium Co-Chair), Globecom'02, and International Conference on Computer Communications and Networking (IC3N) (Technical Program Vice-Chair).