A Secure Mobile Electronic Payment Architecture Platform for Wireless Mobile Networks

Phone Lin, *Senior Member, IEEE,* Hung-Yueh Chen, Yuguang Fang, *Fellow, IEEE,* Jeu-Yih Jeng, and Fang-Sun Lu

***Abstract*—When the basic functionalities of a wireless mobile network have been achieved, customers are then more interested in value-added mobile applications. In order to attract more customers to such mobile applications, a solid, secure and robust trading model is a must. This paper proposes such a secure trading model named *Mobile Electronic Payment (MEP)* for wireless mobile networks, which applies the emerging ID-based cryptography for key agreement and authentication. Our MEP attempts to alleviate the computational cost, reduce the memory space requirement in mobile devices, and meet the requirements for secure trading: avoidance of *overspending* and *double spending*, *fairness*, *user anonymity* and *privacy*. Our design is transparent to the bearer networks and is of low deployment cost. We expect that our MEP provides a viable trading architecture model for the future mobile applications.**

***Index Terms*—Mobile application, security, micropayment, bilinear pairing, identity-based cryptography, billing.**

I. Introduction

With the vast development and deployment of wireless mobile networks such as 3G UMTS [13], [22], WiMAX [18] and Wi-Fi [17], mobile networking applications enabling customers to gain network access anywhere and anytime have attracted more and more attention in our daily lives. When the basic functionalities of a wireless network have been in place, customers are now more interested in value-added mobile applications over this network. Most mobile applications come with the emergence of electronic trading (mobile commerce or m-commerce), hence good secure mobile trading model must be designed to attract more mobile users for doing business wirelessly. Thus, how to integrate the mobile applications with a secure trading model becomes an important design issue, which will significantly affect the success of any value-added mobile application. This is the major topic of this paper.

Mobile applications can be categorized into session-based applications and event-based applications. In event-based applications, user's payment is reflected by one-time events. Examples include sending a message, querying traffic information, or purchasing a song. A session-based application consists of three phases: the session-setup phase, the communication phase and the session release phase. A customer is charged for a session-based application based on either time spent or data volume transferred, e.g., VoIP-calling, video-streaming, audio-streaming, or video-conferencing.

There are a few payment models proposed in the literature [2], [21], which can be classified into two categories: the traditional payment model and the micropayment model. The examples of traditional payment models include the credit card platforms [5], [1], [24], [23] and the electronic cash platforms [6], [25], [8]. The traditional payment models allow only one payment in a *payment transaction*, which has been widely adopted for the event-based applications. Since a session-based application usually requires multiple payments during the execution of this application, with the traditional payment model, it requires multiple payment transactions to complete a session-based application. This is inefficient because heavy signaling and computational overheads are introduced into the network. On the other hand, the micropayment models allow multiple payments in a payment transaction, which is considered more efficient than the traditional payment model. Thus, the micropayment models [32], [14], [31], [27] are often adopted for most of mobile applications. To secure transactions, in [32], [27], the public-key cryptography (e.g., RSA [19]) is adopted. Unfortunately, the public-key cryptography requires heavy computation and long execution time, which may not be a good solution in wireless mobile networks. Yang et al. [31] applied the symmetric-key cryptography[1] such as Advnaced Encryption Standard (AES) [9] that is more efficient than the public-key cryptography in terms of computational cost and is more suitable for mobile devices. Unfortunately, the symmetric-key cryptography requires more frequent key establishments and updates to prevent the shared key from being compromised, and hence induces more communication cost due to key establishment and key updates. Moreover, how to establish the shared key in wireless mobile networks for the

Manuscript received January 30, 2007; revised March 21, 2007; accepted March 28, 2007. The associate editor coordinating the review of this letter and approving it for publication was S. Shen. The work of Lin was sponsored in part by the National Science Council (NSC), R.O.C., under the contract number NSC-96-2627-E-002-001-, NSC-96-2811-E-002-010, NSC- 96-2628-E-002-002-MY2, NSC-95-2221-E-002-091-MY3, and NSC 97-2218-E-002-026, Ministry of Economic Affairs (MOEA), R.O.C., under contract number 93-EC-17-A-05-S1-0017, Telcordia Applied Research Center, Taiwan Network Information Center (TWNIC), Excellent Research Projects of National Taiwan University, 95R0062-AE00-07, and Chunghwa telecom M-Taiwan program M-Taoyuan Project. The work of Fang was supported in part the US National Science Foundation under grants CNS-0626881 and CNS-0716450, and by the National Science Council (NSC), R.O.C., under the contract number NSC- 96-2811-E-002-010.

P. Lin and H.-Y. Chen are with the Dept. of Computer Science & Information Engineering, National Taiwan University, Taipei 106, R.O.C. (e-mail: {plin@, moon@pcs.}csie.ntu.edu.tw).

Y. Fang is with the Dept. of Electrical & Computer Engineering, University of Florida, Gainesville, FL 32611, USA (e-mail: fang@ece.ufl.edu).

J.-Y. Jeng and F.-S. Lu are with the Information Technology Laboratory, Telecommunication Laboratories, Chunghwa Telecom Co., Ltd., R.O.C. (e-mail: {jyjeng, fslu}@cht.com.tw).

Digital Object Identifier 10.1109/TWC.2008.070111.

[1]The sender and receiver for a message delivery use the same key to encrypt and decrypt the message and the shared key is known as a symmetric key.

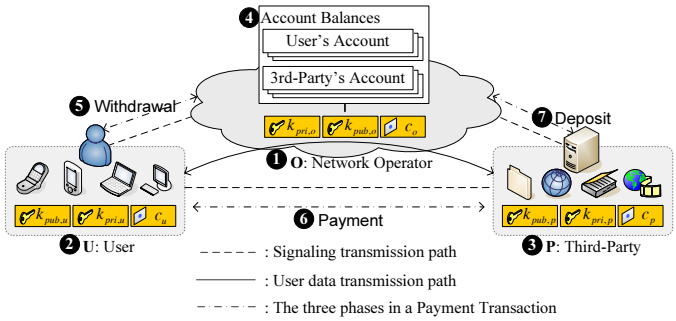


Fig. 1. The general trading model for mobile applications.

symmetric-key cryptography is very challenging.

Compared with fixed networks, mobile networks have lower bandwidth, longer transmission latency, and more unreliable connections, and mobile devices are restricted by limited memory size and low CPU computational capability [20]. The installation of mobile applications on a mobile network should be quick and of low cost. To summarize, the following requirements should be addressed when designing a suitable trading mechanism on a mobile network. First, customers expect a robust, secure, and fair trading mechanism which can be applied in different mobile networks. Second, the trading mechanism should be light-weight (i.e., with low computational complexity and low communication overhead) so that it can be easily run on mobile devices. Third, user anonymity should be achieved, that is, users' purchasing behavior or preference should not be traceable by others. Finally, a trading mechanism should be of low implementation cost.

In this paper, we design an application-level secure payment model, named Mobile Electronic Payment (MEP), for wireless mobile networks, which attempts to meet these requirements. It is based on a more general trading architecture model (cf. Section II-A), which combines both public-key cryptography and symmetric-key cryptography to overcome the disadvantages of both technologies. Specifically, we apply the emerging ID-Based Cryptography (IBC [7]; cf. Section II-B) in the MEP to generate the public-private key pairs so that the certificate overheads among the network operator (denoted as **O**), the user (denoted as **U**), and the mobile application developer or content provider (denoted as **P**) commonly required in the traditional public-key cryptography can be eliminated. Then, from these public-private key pairs, we generate three symmetric keys k_{u-o} (held by **O** and **U**), k_{o-p} (held by **O** and **P**), and k_{u-p} (held by **U** and **P**) to encrypt and decrypt the signaling messages exchanged among **O**, **U**, and **P**. An important observation is that these three symmetric keys are established without actually exchanging them among the concerned parties, a unique feature of ID-based cryptography. To prevent the symmetric keys from being compromised, in each payment transaction, the three public-private key pairs ($k_{pub,o}$, $k_{pri,o}$) held by **O**, ($k_{pub,p}$, $k_{pri,p}$) held by **P**, and ($k_{pub,u}$, $k_{pri,u}$) held by **U** are used to generate the new symmetric keys. Our design keeps the *key freshness*[2] and thus provides more robust security protection. Moreover, MEP supports both event-based and session-based applications and is suitable for the resource-constrained mobile devices because MEP attempts to alleviate the computational cost and reduce the memory space requirement in mobile devices. We expect that our MEP provides a viable trading model for the future mobile applications.

The rest of this paper is organized as follows. In Section II, we briefly illustrate the general conceptual trading model and the basics of the ID-based cryptography. Section III presents the design of MEP. In Section IV, we elaborate on the features and computational overhead of MEP. Finally, Section V concludes our work.

II. Preliminaries

A. General Conceptual Trading Model

Fig. 1 illustrates the general conceptual trading model for mobile applications [11], [32], which consists of three major components: the network operator **O** (Fig. 1 (1)), the user (customer) **U** (Fig. 1 (2)), and the mobile applications/content provider **P** (Fig. 1 (3)). The **P**s supply mobile applications to **U**s. The **O** provides network bearer services (e.g., the UMTS bearer services or the WLAN services) to **U**s, through which **U**s may use different kinds of mobile devices to access the applications. **P** and **O** may reside in different networks. For example, **O** is the operator of a cellular network, and **P** resides in the Internet.

In this trading model, **O** has to be trusted by **U** and **P**. Initially, **U** and **P** apply for accounts from **O**, and **O** maintains an account balance (Fig. 1 (4)) for each account. The public-private key pairs, ($k_{pub,o}$, $k_{pri,o}$), ($k_{pub,p}$, $k_{pri,p}$), and ($k_{pub,u}$, $k_{pri,u}$), and certificates, c_o, c_p, and c_u, which are held by **O**, **U**, and **P**, respectively, are used to address the security issues such as the confidentiality and authentication. The certificate is used to verify the owner of a public key. The certificate uses a digital signature to bind a public key with an individual's identity information (e.g., telephone number or email address). The public-private key pairs are used to encrypt and decrypt all the signaling messages exchanged among **O**, **U**, and **P**.

Before **U** purchases a mobile application from **P**, it initiates a *Payment Transaction* among **O**, **P**, and **U**. The creation process of a payment transaction consists of three phases [11]: the Withdrawal phase (Fig. 1 (5)), the Payment phase (Fig. 1 (6)), and the Deposit phase (Fig. 1 (7)). The process begins at the Withdrawal phase where **U** obtains the electronic means (e.g., the electronic tokens [6], [25] or the value-added smart card [10]) from **O**. Then, the process enters the Payment phase. In the Payment phase, **U** issues the electronic means to **P**, which is known as "*payment*". Then **P** checks the validity of the electronic means. If it is valid, **U** is permitted to purchase a mobile application. The payment may be performed either once or many times, which depends on whether the application is event-based or session-based. For an event-based application, only one payment is made in this phase. For a session-based application, multiple payments may be executed. When the mobile application ends, the process gets into the Deposit phase. In this phase, **P** uses the electronic means obtained from **U** to exchange the payment with **O**, where **O** verifies the electronic means and deposits the payment into **P**'s account.

[2]The key freshness [26] means that the key must be new at any time (i.e., old keys are not reused).

B. Basics of the ID-Based Cryptography

This section briefly discusses the fundamentals of the ID-based cryptography (IBC) [7]. The general IBC concept was proposed in [28] in 1984. Only after 2001 when Boneh and Franklin [7] successfully implemented the IBC concept by using the bilinear pairing function does IBC gain more popularity and show many more useful applications of this technique [33], [34], [35], [36], [37]. In IBC, there is no binding between the user ID and public keys. With this future, the proposed MEP adopts IBC to save transactions and cost associated with either communications and computation.

In the IBC, each user owns a key pair (k_{pub}, k_{pri}). The k_{pub} is a public key, which is derived from a user's identity information (e.g., user's telephone number or email address). The derivation of k_{pub} can be done at the user's device or at the trusted authority (e.g., a network operator). The k_{pri} is a private key, which is generated by the trusted authority by taking the k_{pub} into a function f and is passed to the user through a secure link. As mentioned in [7], [16], [37], the main advantage of the IBC is that there is no need to have certificate to bind user names with their public keys.

III. The Mobile Electronic Payment (MEP) Platform

In this section, we present the MEP platform which follows the general trading model. When a new user **U** or a mobile application/content provider **P** joins the MEP, the Key Distribution procedure (to be elaborated later) is executed to distribute **U** or **P** public-private key pairs denoted as ($k_{pub,u}$, $k_{pri,u}$) or ($k_{pub,p}$, $k_{pri,p}$), respectively. Then, **U** can purchase a mobile application from **P** by running a payment transaction. In a payment transaction, the signaling messages exchanged among **O**, **U**, and **P** are encrypted using three symmetric keys k_{u-o} (held by **O** and **U**), k_{o-p} (held by **O** and **P**), and k_{u-p} (held by **U** and **P**). The three symmetric keys are updated (by utilizing the public-private key pairs) at the beginning of every payment transaction. A payment transaction consists of three phases, the Withdrawal phase (where **U** obtains te tokens from **O**), the Payment phase (where **U** uses the tokes to purchase a mobile application from **P**), and the Deposit phase (where **P** redeems the obtained tokens from **O**).

In the following subsections, we first illustrate the key distribution procedure and then describe how a payment transaction is executed in MEP.

A. The Key Distribution Procedure

The key distribution procedure generates public-private key pairs for **O**, **U** and **P**. The design of this procedure utilizes the IBC to eliminate the certificate overhead from binding one's ID with its public key. Fig. 2 illustrates the message flow for this procedure with the following steps:

Step K1. **O** first generates a *public-params* set $(K, G_1, G_2, \hat{e}, k_{pub,o}, H_1, H_2, H_3)$ by the GENERATE-PARAMS algorithm as shown in Fig. 3. The *public-params* set contains all parameters required in MEP. The usage of the parameters is listed in Table I. Then **O** publishes the generated *public-params* set in a public place (e.g., website).

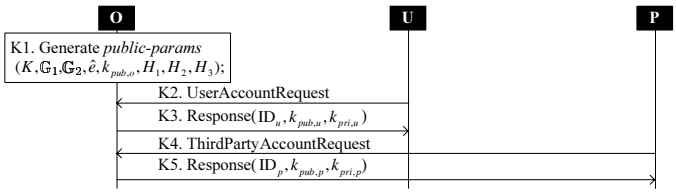


Fig. 2. Message flow for the Key Distribution procedure.

Algorithm 1 GENERATE-PARAMS

1: Generate the pairing parameters $(K, G_1, G_2, \hat{e})$;
2: Select an arbitrary generator for G_1 as the public key $k_{pub,o}$;
3: Choose a hash function $H_1 : \{0,1\}^* \rightarrow G_1$;
4: Choose a hash function $H_2 : G_2 \rightarrow \{0,1\}^N$ for some integer N;
5: Choose a one-way hash function $H_3 : \{0,1\}^* \rightarrow \{0,1\}^M$ for some integer M (e.g., H_3 can be SHA-1 and MD5);
6: **return** $(K, G_1, G_2, \hat{e}, k_{pub,o}, H_1, H_2, H_3)$;

Fig. 3. The Generate-Params algorithm.

O selects a random number $S \in Z_K^*$, and derives its private key $k_{pri,o}$ by computing

$$k_{pri,o} = S \cdot k_{pub,o} \tag{1}$$

where "$\cdot$" is defined in Property 2 in Section II-B. **O** keeps S and $k_{pri,o}$ confidential.

Step K2. **U** sends **O** the UserAccountRequest message to apply for a user account.

Step K3. Upon receiving **U**'s request, **O** selects an ID[3], ID_u, for **U** and creates an account for **U**. Then **O** generates **U**'s public key $k_{pub,u}$ and private key $k_{pri,u}$ by

$$k_{pub,u} = H_1(\text{ID}_u), \tag{2}$$

and

$$k_{pri,u} = S \cdot k_{pub,u}. \tag{3}$$

O sends $k_{pub,u}$, $k_{pri,u}$, and ID_u to **U** through the bearer network link. Since **U** is the customer of **O**, the bearer network[4] is considered secure.

Steps K4 and K5. The two steps are similar to Steps K2 and K3, respectively. **P** applies a third-party account by sending the ThirdPartyAccountRequest message to **O**. **O** selects an ID, ID_p, creates an account for **P**, and generates **P**'s public key $k_{pub,p}$ and private key $k_{pri,p}$ by

$$k_{pub,p} = H_1(\text{ID}_p) \tag{4}$$

and

$$k_{pri,p} = S \cdot k_{pub,p} \tag{5}$$

O sends $k_{pub,p}$, $k_{pri,p}$, and ID_p to **P** through the secure link between **O** and **P**.

[3]This ID is a text string and unique to **U**. For example, **U**'s telephone number can be used as its ID.

[4]During the establishment of the bearer, the authentication procedure is performed between **O** and **U**. All data exchanged between **O** and **U** is encrypted. For example, the authentication/encryption procedure in the GSM network can be found in [22].

TABLE I
THE USAGE OF THE PARAMETERS IN *public-params* SET

Parameter	Usage	Parameter	Usage
K	The order of G_1 and G_2	$k_{pub,o}$	The **O**'s public key
G_1	The cyclic group with operation "+"	H_1	The hash function used to derive one's ID to its public key
G_2	The cyclic group with operation "×"	H_2	The hash function used to derive the output of the Bilinear Pairing function $\hat{e}$ to a symmetric key
$\hat{e}$	The Bilinear Pairing function	H_3	The hash function used to generate the electronic means

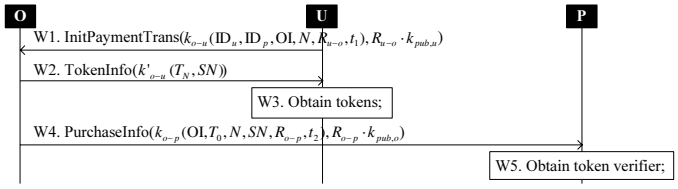


Fig. 4. Message flow for the Withdrawal phase of a payment transaction.

B. Payment Transaction in MEP

In this section, we describe the execution of a payment transaction in MEP for **U** to purchase a mobile application from **P**. Following the general trading model, a payment transaction in MEP consists of three phases: the Withdrawal phase, the Payment phase and the Deposit phase, which are described below.

1) Withdrawal Phase: In this phase, **U** obtains the electronic means (i.e., the tokens) from **O**. Fig. 4 illustrates the message flow for this phase with the following steps. To simplify our description, we use $k(D)$ to denote that the data D is encrypted by the symmetric key k with an efficient symmetric-key algorithm.

Step W1. By browsing **P**'s website, **U** selects a mobile application, gets **P**'s ID, ID$_p$, and obtains the Order Information (OI) containing the ID and the data unit price of the mobile application. Then, **U** randomly selects an integer R_{u-o} from Z_K^* and generates the symmetric key k_{u-o} by computing

$$k_{u-o} = H_2(\hat{e}(R_{u-o} \cdot k_{pub,o}, k_{pri,u})). \quad (6)$$

Then **U** sends an InitPaymentTrans message to **O** to initiate a payment transaction, where $k_{u-o}(\mathrm{ID}_u, \mathrm{ID}_p, \mathrm{OI}, N, R_{u-o}, t_1)$ and $R_{u-o} \cdot k_{pub,u}$ are carried in the message. The first parameter $k_{u-o}(\mathrm{ID}_u, \mathrm{ID}_p, \mathrm{OI}, N, R_{u-o}, t_1)$ contains the necessary information for **O** to generate the tokens for **U**. N is the amount of data units **U** will purchase, and t_1 is the current system time, which is used to prevent message replay and impersonation attacks [26]. The second parameter $R_{u-o} \cdot k_{pub,u}$ will be used by **O** to derive the symmetric key k'_{u-o} (see Step W2) and authenticate **U**. Note that k'_{u-o} is the same as k_{u-o} (Proposition III-B1), so that **O** can decrypt the $k_{u-o}(\mathrm{ID}_u, \mathrm{ID}_p, \mathrm{OI}, N, R_{u-o}, t_1)$ parameter.

Step W2. Upon receipt of the InitPaymentTrans message, **O** will perform the following tasks:

(i) **O** extracts the second parameter $R_{u-o} \cdot k_{pub,u}$ from the InitPaymentTrans message, and uses this parameter and **O**'s private key $k_{pri,o}$ to derive the symmetric key k'_{u-o} as

$$k'_{u-o} = H_2(\hat{e}(R_{u-o} \cdot k_{pub,u}, k_{pri,o})). \quad (7)$$

Then **O** uses k'_{u-o} to decrypt $k_{u-o}(\mathrm{ID}_u, \mathrm{ID}_p, \mathrm{OI}, N, R_{u-o}, t_1)$, and **O** obtains the ID$_u$, ID$_p$, OI, N, R_{u-o}, and t_1.

(ii) To authenticate the sender of the InitPaymentTrans message, **O** verifies whether $R_{u-o} \cdot H_1(\mathrm{ID}_u)$ (where R_{u-o} and ID$_u$ are obtained in (i)) is equal to the second parameter $R_{u-o} \cdot k_{pub,u}$. If they are not equal (i.e., $R_{u-o} \cdot H_1(\mathrm{ID}_u) \neq R_{u-o} \cdot k_{pub,u}$), the sender is illegal, and the phase quits without sending extra messages. If they are equal (i.e., $H_1(\mathrm{ID}_u) = k_{pub,u}$), the sender is authenticated and then **O** checks whether the difference between t_1 and the local clock time is within an *acceptance window*[5] to prevent from message replay and impersonation [26].

(iii) If the authentication is successful, **O** will then generate the tokens for **U**. Suppose that each data unit consumes one token, and N tokens are required for **U**. Let $\langle T_N, T_{N-1}, T_{N-2}, \ldots, T_1 \rangle$ denote the N tokens. Initially, **O** selects a random number as the token root T_N. Then **O** executes the GENERATE-TOKENS algorithm (see Fig. 5) with arguments T_N and N to generate N tokens. After executing the algorithm, **O** obtains the tokens $T_{N-1}, T_{N-2}, \ldots, T_1$ and a token verifier T_0. The token verifier T_0 will be used by **P** to make sure that the tokens are sent from **U** in the Payment phase. Each token indicates the data unit price of the mobile application. Then **O** deducts the cost for N tokens from **U**'s account.

(iv) The payment transaction is assigned an unique serial number SN by **O**. Then **O**

[5]The acceptance window can be a fixed-size time interval (e.g., 10 ms or 2 s).

Algorithm 2 GENERATE-TOKEN(T_N, N)

1: **for** $i \leftarrow N-1$ **downto** 0 **do**
2: $\quad T_i \leftarrow H_3(T_{i+1})$
3: **return** $\langle T_{N-1}, T_{N-2}, \ldots, T_0 \rangle$

Fig. 5. The GENERATE-TOKEN algorithm.

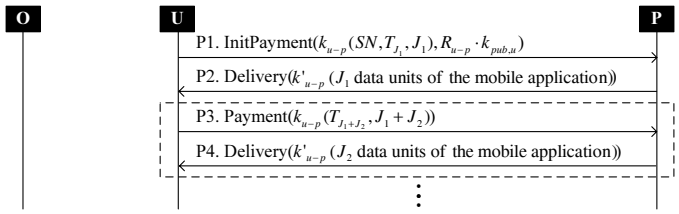


Fig. 6. Message flow for the Payment phase of a payment transaction.

sends **U** the TokenInfo message carrying $k'_{u-o}(T_N, SN)$.

Step W3. Upon receipt of the TokenInfo message, **U** uses k_{u-o} to decrypt the message and obtains T_N and SN. Then, **U** uses the token root T_N to generate N tokens by executing the GENERATE-TOKEN algorithm. Note that due to the lightweight[6] feature of the hash function H_3 [27], the tokens are generated efficiently.

Step W4. **O** selects a random integer R_{o-p} from Z_K^* and generates the symmetric key k_{o-p} as

$$k_{o-p} = H_2(\hat{e}(R_{o-p} \cdot k_{pub,p}, k_{pri,o})) \tag{8}$$

where **P**'s public key $k_{pub,p}$ is obtained by $k_{pub,p} = H_1(\mathrm{ID}_p)$. Then, **O** sends **P** a PurchaseInfo message to notify that **U** wants to purchase the mobile application. The parameters carried in the message contain $k_{o-p}(\mathrm{OI}, T_0, N, SN, R_{o-p}, t_2)$ and $R_{o-p} \cdot k_{pub,o}$, where t_2 is the current system time used to prevent from message replay and impersonation and the second parameter $R_{o-p} \cdot k_{pub,o}$ will be used by **P** to derive the symmetric key k'_{o-p} (to be elaborated in next step). Then, using SN as the index, **O** stores the information (containing ID_u, ID_p, N, T_N, and k_{o-p}) required in Deposit phase into its database.

Step W5. Upon receipt of the PurchaseInfo message, **P** extracts the second parameter $R_{o-p} \cdot k_{pub,o}$ from the message, and uses this parameter and **P**'s private key $k_{pri,p}$ to derive the symmetric key k'_{o-p} as

$$k'_{o-p} = H_2(\hat{e}(R_{o-p} \cdot k_{pub,o}, k_{pri,p})). \tag{9}$$

Note that from Proposition III-B1, we know $k'_{o-p} = k_{o-p}$. **P** uses k'_{o-p} to decrypt the first parameter $k_{o-p}(\mathrm{OI}, T_0, N, SN, R_{o-p}, t_2)$.

Similar to Step W2.(ii), **P** calculates $R_{o-p} \cdot k_{pub,o}$ (R_{o-p} is obtained from the first parameter and $k_{pub,o}$ is obtained from the *public-params* set) and checks whether the result is equal to the second parameter $R_{o-p} \cdot k_{pub,o}$ carried in the PurchaseInfo message. If they are not equal, the sender is illegal, and the phase quits without sending extra message. If they are equal, the sender is ensured to be **O**. **P** checks whether the difference between t_2 and the local clock time is within an *acceptance window* to prevent from message replay and impersonation. Then, using SN as the index, **P** stores the information (containing OI, T_0, N, and k_{o-p}) required in the payment phase and the deposit phase into its database.

[6]The hash function is about 100 times faster than RSA signature verification, and about 10,000 times faster than RSA signature generation.

The following result ensures the correctness of our MEP.

Proof: $k_{u-o} = k'_{u-o}$ and $k_{o-p} = k'_{o-p}$.

$$\begin{aligned}
k_{u-o} &= H_2(\hat{e}(R_{u-o} \cdot k_{pub,o}, k_{pri,u})) && \text{(from (6))} \\
&= H_2(\hat{e}(R_{u-o} \cdot k_{pub,o}, S \cdot k_{pub,u})) && \text{(from (3))} \\
&= H_2(\hat{e}(k_{pub,o}, k_{pub,u})^{R_{u-o} \cdot S}) && \text{(Bilinearity of } \hat{e}) \\
&= H_2(\hat{e}(k_{pub,u}, k_{pub,o})^{R_{u-o} \cdot S}) && \text{(Symmetry of } \hat{e}) \\
&= H_2(\hat{e}(R_{u-o} \cdot k_{pub,u}, S \cdot k_{pub,o})) && \text{(Bilinearity of } \hat{e}) \\
&= H_2(\hat{e}(R_{u-o} \cdot k_{pub,u}, k_{pri,o})) && \text{(from (1))} \\
&= k'_{u-o} && \text{(from (7))}
\end{aligned}$$

Similarly, we can prove the other identity. ■

2) Payment Phase: In the Payment phase, **U** uses the tokens to purchase a mobile application from **P**. This phase may consist of one or more payments. We assume that **U** pays J_i tokens in the ith payment. Fig. 6 illustrates the message flow for the Payment phase with the following steps.

Step P1. **U** randomly selects an integer R_{u-p} from Z_K^* and generates the symmetric key k_{u-p} by

$$k_{u-p} = H_2(\hat{e}(R_{u-p} \cdot k_{pub,p}, k_{pri,u})) \tag{10}$$

where **P**'s public key $k_{pub,p}$ is obtained from $k_{pub,p} = H_1(\mathrm{ID}_p)$. Then **U** initiates the first payment by sending a InitialPayment message, where J_1 tokens $\langle T_1, T_2, \ldots T_{J_1} \rangle$ are carried in this message. The parameters of the InitialPayment message include $k_{u-p}(SN, T_{J_1}, J_1)$ and $R_{u-p} \cdot k_{pub,u}$. The second parameter $R_{u-p} \cdot k_{pub,u}$ will be used by **P** to derive the symmetric key k'_{u-p} (see (11), Step P2). Note that in this step, **P** cannot directly extract $k_{pub,u}$ easily from the second parameter $R_{u-p} \cdot k_{pub,u}$, and the InitialPayment message does not contain any information that may leak out **U**'s identity. Therefore, "user anonymity" is well protected.

Step P2. Upon receipt of the InitialPayment message, **P** uses the second parameter $R_{u-p} \cdot k_{pub,u}$ and **P**'s private key $k_{pri,p}$ to generate the symmetric key k'_{u-p} by

$$k'_{u-p} = H_2(\hat{e}(R_{u-p} \cdot k_{pub,u}, k_{pri,p})). \tag{11}$$

From Proposition III-B1, k'_{u-p} is the same as k_{u-p}. Using the symmetric key k'_{u-p}, **P** decrypts the first parameter $k_{u-p}(SN, T_{J_1}, J_1)$ and obtains SN, T_{J_1} and J_1. **P** uses SN as the index to query its database for the token verifier T_0, N, and OI. According to the mobile application ID contained in OI, **P** identifies the mobile application that **U** wants to

purchase, and prepares N data units of the mobile application (e.g., streaming data for N seconds). To verify the token T_{J_1}, **P** checks whether the equation $\underbrace{H_3(H_3 \cdots (H_3(T_{J_1})))}_{J_1} \stackrel{?}{=} T_0$ holds. If it holds, **P** ascertains that the token T_{J_1} is legal. **P** stores T_{J_1} for verifying the token carried in the next message and discards T_0 to release the memory space. Then **P** encrypts the first unit to the J_1th unit of the mobile application using the symmetric key k'_{u-p} and responds with the J_1 data units carried in the Delivery message, to **U**.

Step P3. Upon receipt of the Delivery message, **U** decrypts the message using the symmetric key k_{u-p} and obtains the J_1 data units. Then, **U** starts the 2nd Payment to purchase the next J_2 data units by sending **P** the Payment($T_{J_1+J_2}, J_1 + J_2$) message.

Step P4. Upon receipt of the Payment message, **P** decrypts the message using the symmetric key k'_{u-p} and obtains $T_{J_1+J_2}$ and $J_1 + J_2$. **P** gets J_2 by subtracting $(J_1 + J_2) - J_1$. Then **P** checks whether the equation $\underbrace{H_3(H_3 \cdots (H_3(T_{J_1+J_2})))}_{J_2} \stackrel{?}{=} T_{J_1}$ holds. If the equality holds, **P** ascertains that the token $T_{J_1+J_2}$ is legal. **P** stores $T_{J_1+J_2}$ for verifying the token carried in next message and discards the token T_{J_1}. Then, **P** encrypts the next J_2 data units of the mobile application using the symmetric key k'_{u-p} and delivers the J_2 data units to **U**.

Repeating Steps P3 and P4, **U** sends the succeeding tokens to **P**, and **P** delivers the succeeding data units to **U**. This phase may be terminated if **U** stops paying the token or **P** stops delivering the mobile application.

3) Deposit Phase: Assume that **P** receives J ($J \leq N$) tokens after the Payment phase. In the Deposit phase, **P** redeems the J tokens from **O**. This phase consists of the following two steps.

Step D1. **P** sends **O** the deposit message carrying the parameters SN and $k_{o-p}(T_J, J)$.

Step D2. Upon receipt of the deposit message, **O** uses the first parameter SN and the index to query its database for ID_u, ID_p, N, T_N, and k_{o-p}. Using the symmetric key k_{o-p}, **O** decrypts the second parameter $k_{o-p}(T_J, J)$ and obtains T_J and J, and then checks whether the equation $\underbrace{H_3(H_3 \cdots (H_3(T_N)))}_{N-J} \stackrel{?}{=} T_J$ holds to verify the token T_J. If the equation holds, **O** deposits the credit for J tokens into **P**'s account and takes the cost for J tokens from **U**'s account. The payment transaction is completed. Otherwise (i.e., $\underbrace{H_3(H_3 \cdots (H_3(T_N)))}_{N-J} \neq T_J$), **O** treats the sender of the deposit message as an adversary, and the deposit phase will not carried through.

Note that if **P** does not exercise Step D1 after the payment phase in a predefined time period (e.g., one day), the payment transaction is considered incomplete, and **O** gives the credit for all tokens into **U**'s account, and terminates the payment transaction.

IV. Features and Overhead Analysis of MEP

A. Features of MEP

There are a few useful features of MEP including the avoidance of *overspending* and *double spending*, the *fairness*, the *user anonymity*, and the *privacy*, which are discussed next.

1) Avoidance of Overspending and Double Spending: *Overspending* means that **U**'s account does not have enough credit to purchase a mobile application. *Double spending* is that **U** uses the same tokens to purchase mobile applications from different **P**s. Both *overspending* and *double spending* cause financial loss to **O** and **P**. MEP adopts the "prepaid" approach, that is, **U**'s account is deducted (see Step W2 of the withdrawal phase, Section III-B1) before **U** purchases a mobile application. If **U**'s account does not have enough credit to purchase a mobile application, **O** will not issue tokens to **U**. Hence, MEP can avoid the loss due to a user's overspending. Moreover, when **U** withdraws tokens from **O**, it has to inform **O** of **P**'s ID (see Step W1 of the withdrawal phase, Section III-B1). Then, **O** sends tokens and the corresponding token verifiers to **U** and **P** (see Steps W2 and W4 of the withdrawal phase, Section III-B1), respectively. If **U** applies the same tokens to another **P**$'$, **P**$'$ will not accept the tokens because it does not own the corresponding token verifiers. Therefore, the risk of *double spending* is avoided.

2) Fairness: After a payment transaction, **U** can get the data units of a mobile application, whose value is equivalent to the credits **U** pays for, and **P** can get the credits equivalent to the value of data units of the mobile application **P** provides [30], which is referred to as the "*fairness*".

In MEP, during the execution of the payment phase (see Section III-B2), **P** provides **U** the data units of the mobile application after **U** has paid the tokens (i.e., credits). Therefore, there is no risk for **P** to provide data units. Furthermore, **U** can terminate the token payment immediately if **P** does not send the requested data units. In this case, at most one token is lost, which is considered insignificant. The *fairness* feature can be accommodated in MEP.

3) User Anonymity: The *user anonymity* is defined in two levels: *untraceability* and *unlinkability* [2]. *Untraceability* means that **P** is not allowed to know **U**'s identity during the execution of a payment transaction. *Unlinkability* means that two different payment transactions (which involve the same **U**) cannot be linked by **P**, i.e., **P** is not allowed to identify the two payment transactions initiated by the same **U** so that any user profiling attempt fails.

In MEP, **U** and **P** negotiate only in the payment phase (see Section III-B2) for purchasing mobile applications. The information of **U** sent to **P** in this phase contains tokens, the serial number of a payment transaction, the total number of tokens paid to **P**, and the parameter $R_{u-p} \cdot k_{pub,u}$, which does not include any **U**-related information. Through the payment phase, **P** cannot identify **U**. Consequently, MEP ensures untraceability. Furthermore, the public key $k_{pub,u}$ of **U** cannot be extracted from the parameter $R_{u-p} \cdot k_{pub,u}$, which is shown in Property 2 of a bilinear pairing function

in Section II-B. Since **P** is not able to obtain either **U**'s ID or **U**'s public key, the unlinkability between any two different payment transactions can be achieved in MEP.

4) Privacy: Privacy means that the data of a mobile application exchanged between **U** and **P** cannot be revealed by any unauthorized third party except **O** who distributes the keys. In MEP, the data of a mobile application is encrypted by the symmetric key k_{u-p}, where the k_{u-p} is only known by **U** and **P** (see Steps P1 and P2 of the payment phase, Section III-B2). Hence, *privacy* is well protected in MEP.

B. Computational Overhead of MEP

This section analyzes the computational overhead for a payment transaction in MEP. The computational cost for a payment transaction can be evaluated in the following three aspects.

1) Token Generation and Verification: Let C_{h3} be the computational cost for executing the H_3 hash function for token generation and verification. Assume that **U** obtains N tokens from **O** and pays J ($J \leq N$) tokens to **P** for a mobile application (i.e., in one payment transaction). **O** and **U** generate N tokens by executing the GENERATE-TOKEN algorithm in Steps W2 and W3, respectively, where the hash function H_3 is executed N times with computational cost $2C_{h3}N$. During a payment transaction, **P** verifies J tokens sent from **U** in Steps P2 and P4 of the payment phase by executing the H_3 function J times with the computational cost $C_{h3}J$. In the deposit phase, **P** sends the last token of the received J tokens to redeem token from **O** in Step D1, and **O** runs the hash function H_3 $N - J$ times to verify the last token (see Step D2), that is, the computational cost is $C_{h3}(N - J)$. The total computational cost for token verification in a payment transaction is $C_{h3}N$. Thus, the total computational cost for token generation and verification is $3C_{h3}N$.

As mentioned in [27], H_3 is a light-weight function with low computational cost, which is about 100 times faster than the RSA signature verification and about $10,000$ times faster than the RSA signature generation. Usually, the number N of tokens required in a payment transaction is 50 to $50,000$, and the computational cost is $150C_{h3}$ to $150,000C_{h3}$, which is considered to be reasonably smaller than what RSA needs.

2) Message Encryption and Decryption: Let C_m be the computational cost of the symmetric key algorithm AES (applied in MEP) for message encryption and decryption. There are three messages (including the InitPaymentTrans, TokenInfo, and PurchaseInfo messages), encrypted in the payment phase. Suppose that there are P ($1 \leq P \leq N$) payments processed in the payment phase, where P messages (including InitPayment and Payment messages) are encrypted. The deposit message is encrypted in the deposit phase. Each message requires two operations (encryption and decryption). Thus, the total computational cost for message encryption and decryption in MEP is $2(4 + P)C_m$.

The computational cost of symmetric key cryptography is less complex than public-key cryptography [29]. Our design has the lower computational cost then what is needed than traditional public-key cryptography.

3) Symmetric-Key Update: In MEP, we update three symmetric-keys, k_{u-o}, k_{o-p}, in the withdrawal phase (see Equations (6), (7), (8) and (9)), and k_{u-p} in the payment phase (see Equations (10) and (11)) by running the $H_2(\hat{e}(a \cdot b, c))$ function, where $a \cdot b$ and c are the input parameters, and "$\cdot$" is the *scalar multiplication* operation. Let C_k be the computational cost of $H_2(\hat{e}(a \cdot b, c))$. Computing the $H_2(\hat{e}(a \cdot b, c))$ function requires to execute a hash function H_2, a bilinear pairing function $\hat{e}$ and a *scalar multiplication*. Let C_{h2} be the computational cost for the hash function H_2, $C_{\hat{e}}$ be the computational cost for the bilinear pairing function $\hat{e}$, C_{G_1} be the computational cost for the *scalar multiplication*. Then C_k can be expressed as $C_k = C_{h2} + C_{\hat{e}} + C_{G_1}$. As noted in [15], the $C_{\hat{e}}$ is much larger than the C_{h2} and the C_{G_1}, and we have $C_k = C_{h2} + C_{\hat{e}} + C_{G_1} < 3C_{\hat{e}}$. The study [4] has shown that the computation of $\hat{e}$ can be completed within 20 milliseconds on a Pentium III 1 GHz machine, and the computation of $H_2(\hat{e}(a \cdot b, c))$ can be completed within 60 milliseconds, which is reasonably low and suitable for a resource-restrained mobile device.

V. CONCLUSION

This paper proposed a secure Mobile Electronic Payment (MEP) platform for the mobile commerce (m-commerce) over wireless mobile networks. In this platform, we take advantage of the emerging the ID-Based Cryptography which eliminates the necessity of certificates commonly required by other public key cryptography. Moreover, since ID-based cryptography can establish the shared key between two parties without additional message exchanges, symmetric key cryptography can be still used effectively, leading to significant computational cost. Our study shows that our MEP platform satisfies the requirements of secure trading (such as avoidance of *overspending* and *double spending*, *fairness*, *user anonymity*, and *privacy*) and has low computational cost. We expect that our MEP will provide a viable trading model for the future mobile applications and play an important role in the emerging m-commerce industries.

ACKNOWLEDGEMENT

The authors would like to thank Mr. Chien-Wei Cheng, Mr. Lu-Tsung Chang, Mr. Pei-Tang Huang, Mr. Ka-Chun Chan, and Mr. Cheng-Kai Hsieh for their assistance in the implementation of the MEP platform and Mr. Shin-Ming Cheng for his help in improving the presentation of this paper.

The authors would also thank the editor and the anonymous reviewers for their valuable comments.

REFERENCES

[1] M. M. Anderson, Financial Service Markup Language (FSML) version 1.17.1, technical report Financial Services Technology Consortium, Oct. 1998.

[2] N. Asokan, P. A. Janson, M. Steiner, and M. Waidner, "The state of the art in electronic payment systems," *IEEE Computer*, vol. 30, no. 9, pp. 28–35, Sept. 1997.

[3] P. Barreto, H. Kim, B. Bynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," in *Proc. 22nd Annual International Cryptology Conference,* Santa Barbara, CA, pp. 354–368, 2002.

[4] P. Barreto, L. Ben, and S. Michael, "Efficient implementation of pairing-based cryptosystems," *J. Cryptology*, vol. 17, no. 4, pp. 321–334, 2004.

[5] M. Bellare, J. Garay, R. Hauser, A. Herzberg, H. Krawczyk,M. Steiner, G. Tsudik, E. Herreweghen, and M. Waidner, Design, implementation and deployment of a secure account-based electronic payment system," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 611–627, Apr. 2000.

[6] J. P. Boly, A. Bosselaers, R. Cramer, R. Michelsen, S. F. Mjolsnes, F. Muller, T. P. Pedersen, B. Pfitzmann, P. Rooij, B. de, Schoenmakers, M. Schunter, L. Vallee, and M. Waidner, "The ESPRIT Project CAFE-high security digital payment systems," in *Proc. ESORICS*, pp. 217–230, 1994.

[7] D. Boneh and M. Franklin, "Identity-based encryption from Weil pairing," in *Proc. Crypto 2001*, pp. 213–229, 2001.

[8] J. Camenisch, U. Maurer, and M. Stadler, "Digital payment systems with passive anonymity-revoking trustees," in *Proc. ESORICS*, pp. 33–43, 1996.

[9] J. Daemen, S. Borg, and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer-Verlag, 2002.

[10] C. H. Fancher, "In your pocket: smartcards," *IEEE Spectrum*, vol. 34, pp. 47–53, Feb. 1997.

[11] L. Ferreira and and R. Dahab, "A scheme for analyzing electronic payment systems," in *Proc. ACSAC*, pp. 137–146, 1998.

[12] R. Grimaldi, *Discrete and Combinational Mathematics, Fifth Edition*. Addison-Wesley, 1999.

[13] K. Heikki, A. Ari, L. Lauri, N. Siamak, and N. Valtteri, *UMTS Networks-Architecture, Mobility & Services*. John Wiley & Sons, Inc., 2002.

[14] A. Herzberg and H. Yochai, "Mini-pay: charging per click on the Web," in *Proc. Sixth International World Wide Web Conference*, Santa Clara, CA, pp. 301–307, Apr. 1997.

[15] F. Hess, "Efficient identity based signature schemes based on pairings," in *Proc. Selected Areas in Cryptography: 9th Annual International Workshop*, pp. 310–324, Aug. 2002.

[16] R.-J. Chen, J.-S. Hwu, and Y.-B. Lin, "An identity-based cryptosystem for end-to-end mobile security," *IEEE Trans. Wireless Commun.*, accepted for publication, 2006.

[17] IEEE Std 802.11-1997 Information Technology-Telecommunications and Information Exchange between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless Lan Medium Access Control (MAC) and Physical Layer (PHY) Specifications, technical report IEEE Std. 802.11-1997, 1997.

[18] IEEE Standard for Local and Metropolitan Area Networks-Part 16: Air Interface for Fixed Broadband Wireless Access Systems, technical report IEEE Std. 802.16-2004, 2004.

[19] J. Jonsson and B. Kaliski, "Public-key cryptography standards (PKCS) #1: RSA cryptography specifications version 2.1, technical report RFC 3447, 2003.

[20] Y. C. Lai, P. Lin, and Y.-T. Huang, "Design and implementation of a wireless Internet remote access platform," *J. Wireless Commun. and Mobile Computing*, vol. 6, no. 4, pp. 413–429, Jan. 2006.

[21] Z.-Y. Lee, H.-C. Yu, and P.-J. Ku, "An analysis and comparison of different types of electronic payment systems," in *Proc. Portland International Conference on Management of Engineering and Technology 2001 (PICMET'01)*, vol. 2, pp. 38–45, 2001.

[22] Y.-B. Lin and I. Chlamtac, *Wireless and Mobile Network Architectures*. John Wiley & Sons, Inc., 2001.

[23] S. Low and N. Maxemchuk, "Anonymous credit cards," in *Proc. 2nd ACM Conference on Computer and Communications Security*, pp. 108–117, 1994.

[24] Mastercard and Visa, SET Secure Electronic Transactions Protocol, version 1.0 edition, Book One: Business Specifications, Book Two: Technical Specification, Book Three: Formal Protocol Definition, May 1997.

[25] F. Medvinsky and B. Neuman, "NetCash: a design for practical electronic currency on the Internet," in *Proc. First ACM Conference on Computer and Communications Security*, pp. 102–106, 1993.

[26] A. Menezes, P. V. Oorschot, and S. Vanstone, Handbook of Applied Cryptography; [Online] Available: http://citeseer.ist.psu.edu/428600.html, 1999.

[27] R. L. Rivest and A. Shamir, "PayWord and MicroMint: two simple micropayment schemes," *IEEE Trans. Veh. Technol.*, vol. 52, no. 1, pp. 132–141, 2003.

[28] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. CRYPTO 84 on Advances in Cryptology*, pp. 47–53, 1985.

[29] W. Stallings, *Cryptography and Network Security: Principles and Practice, Second Edition*. Prentice-Hall, 1999.

[30] H. Wang and H. Guo, "Fair payment protocols for e-commerce," in *Proc. 22th Annual Symposium on Principles of Distributed Computing*, pp. 227–245, 2004.

[31] Z. Yang, W. Lang, and Y. Tan, "A new fair micropayment system based on hash chain," in *Proc. IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, pp. 139–145, 2004.

[32] S.-M. Yen, "PayFair: a prepaid Internet micropayment scheme ensuring customer fairness," in *Proc. IEE Computers and Digital Techniques*, vol. 148,no. 6, pp. 207–213, Nov. 2001.

[33] Y. Zhang and Y. Fang, "ARSA: an attack-resilient security architecture for multi-hop wireless mesh networks," *IEEE J. Select. Areas Commun.*. vol. 24, no. 10, pp. 1916–1928, Oct. 2006.

[34] Y. Zhang and Y. Fang, "A secure authentication and billing architecture for wireless mesh networks," *ACM Wireless Networks*, 2006, accepted for publication.

[35] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "MASK: anonymous on-demand routing in mobile ad hoc networks," *IEEE Trans. Wireless Commun.*, vol. 5, no. 9, pp. 2376–2385, Sept. 2006.

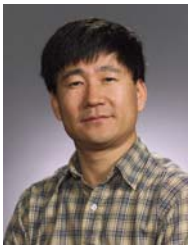
[36] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based security mechanisms in wireless sensor networks," *IEEE J. Select. Areas Commun.*, vol. 24, no. (2), pp. 247–260, Feb. 2006.

[37] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Securing mobile ad hoc networks with certificateless public keys," *IEEE Trans. Dependable and Secure Computing*, vol. 3, no. 4, pp. 386–399, Oct./Dec. 2006.

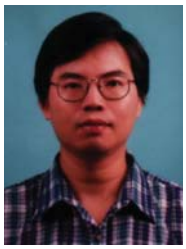
Phone Lin (M'02-SM'06) received his BSCSIE degree and Ph.D. degree from National Chiao Tung University, Taiwan, R.O.C. in 1996 and 2001, respectively. From August 2001 to July 2004, he was an Assistant Professor in Department of Computer Science and Information Engineering (CSIE), National Taiwan University, R.O.C. Since August 2004, he has been an Associate Professor in Department of CSIE and in Graduate Institute of Networking and Multimedia, National Taiwan University, R.O.C. His current research interests include personal communications services, wireless Internet, and performance modeling.

Dr. Lin has published more than twenty international SCI journal papers (most of which are IEEE Transactions and ACM papers). Dr. Lin is an Associate Editor for *IEEE Transactions on Vehicular Technology*, a Guest Editor for *IEEE Wireless Communications* special issue on Mobility and Resource Management, and a Guest Editor for ACM/Springer MONET special issue on Wireless Broad Access. He is also an Associate Editorial Member for the WCMC Journal. Dr. Lin has received many research awards. He was elected as the Best Young Researcher, the 3rd IEEE ComSoc Asia-Pacific Young Researcher Award, 2007. He was a recipient of Research Award for Young Researchers from Pan Wen-Yuan Foundation in Taiwan in 2004, a recipient of K. T. Li Young Researcher Award honored by ACM Taipei Chapter in 2004, a recipient of Wu Ta You Memorial Award of National Science Council (NSC) in Taiwan in 2005, a recipient of Fu Suu-Nien Award of NTU in 2005 for his research achievements, and a recipient of 2006 Young Electrical Engineering Award, the Chinese Institute of Electrical Engineering. Dr. Lin is listed in *Who's Who in Science and Engineering*(R) in 2006. Dr. Lin is a Senior Member, IEEE. P. Lin's email and website addresses are plin@csie.ntu.edu.tw and http://www.csie.ntu.edu.tw/∼plin, respectively.

Hung-Yueh Chen received his B.S.C.S.I.E degree from Tamkang University, Taiwan, R.O.C., in 2004 and his Master degree in Computer Science from National Taiwan University, Taiwan, R.O.C., in 2006. He won Intel Innovation Award during master program. His research interests include personal communication services, mobile computing, and network security.



Yuguang Fang (S'92-M'97-SM'99-F'08) received a Ph.D. degree in Systems Engineering from Case Western Reserve University in January 1994 and a Ph.D degree in Electrical Engineering from Boston University in May 1997. He was an assistant professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology from July 1998 to May 2000. He then joined the Department of Electrical and Computer Engineering at University of Florida in May 2000 as an assistant professor, got an early promotion to an associate professor with tenure in August 2003 and to a full professor in August 2005. He holds a University of Florida Research Foundation (UFRF) Professorship from 2006 to 2009. He has published over 200 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 is the recipient of the Best Paper Award in IEEE International Conference on Network Protocols (ICNP) in 2006 and the recipient of the IEEE TCGN Best Paper Award in the IEEE High-Speed Networks Symposium, IEEE Globecom in 2002. He has served on several editorial boards of technical journals including *IEEE Transactions on Communications*, *IEEE Transactions on Wireless Communications*, *IEEE Transactions on Mobile Computing* and *ACM Wireless Networks*. He has also been actively participating in professional conference organizations such as serving as The Steering Committee Co-Chair for QShine, the Technical Program Vice-Chair for IEEE INFOCOM'2005, Technical Program Symposium Co-Chair for IEEE Globecom'2004, and a member of Technical Program Committee for IEEE INFOCOM (1998, 2000, 2003-2008). He is a Fellow of IEEE.



Jeu-Yih Jeng received the B.S. degree in mathematics from Fu-Jen University in 1983, the M.S. degree in applied mathematics from National Chiao-Tung University in 1985, and the Ph.D. degree in computer science and information engineering from National Chiao-Tung University in 1998. Since 1985, he has been with the Information Technology Laboratory of Telecommunication Laboratories, Chunghwa Telcom Co., Ltd, where he is currently a Distinguished Researcher and a project manager. His research interests include design and analysis of personal communications services network, development of telecommunication operation support systems, and performance modeling.

Fang-Sun Lu received the B.S. degree in applied mathematics, the M.S. degree in mathematics from Fu-Jen University in 1983 and 1985, respectively. Now he is a PhD candidate in the Department of Computer Science and Information Engineering at National Chiao Tong University. Since 1985, he has been with the Information Technology Laboratory of Telecommunication Laboratories, Chunghwa Telcom Co., Ltd, where he is currently a Distinguished Researcher and a project manager. His research interests include design and analysis of personal communications services network, development of telecommunication operation support systems, and performance modeling.