Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 10 (2012) 758 – 765

Procedia Computer Science

The 9th International Conference on Mobile Web Information Systems (MobiWIS)

An Anonymous Secure Payment Protocol in a Payment Gateway Centric Model

Jesús Téllez Isaac[a,*], Sherali Zeadally[b]

[a]Universidad de Carabobo, Computer Science Department (Facyt), Av. Salvador Allende, Sector Bárbula, Valencia, Venezuela.
[b]University of the District of Columbia, Department of Computer Science and Information Technology, Washington DC 20008, USA.

Abstract

In the last few years, a significant number of mobile payment systems have been proposed. Most of them have been based on a scenario where all the entities are directly connected to each other (formally called Full connectivity scenario). Although these scenarios offer advantages to protocol designers in terms of design simplicity and the development of payment protocols without losing security capabilities, the full connectivity scenario does not consider those situations where the client cannot directly communicate with the merchant. We present the design of an anonymous secure Payment protocol based on the Payment Gateway Centric scenario for those scenarios in mobile environments where the Client cannot communicate directly with the Merchant to process the *Payment Request*. Our proposed protocol uses symmetric-key operations which require low computational power and can be processed faster than asymmetric ones.

Keywords: Cryptography; Payment; Performance; Protocol; Electronic Commerce

1. Introduction

Since the first credit-card appeared in the 50's, the so called 'plastic money' has been widely accepted as a payment method. The exponential growth of the Internet has helped the development, proliferation and adoption of various types of electronic commerce systems which allow online payments.

A significant number of mobile payment systems have emerged in the last few years which allow payments for services and goods from mobile devices using different kinds of payments: credit-card payments, micropayments and digital coins. The relationship between the client and the merchant is quite strict for most of these mobile payment systems and does not allow the use of schemes in which direct communication among these parties is not possible. Hence, it becomes necessary to develop mobile payment systems where the client interacts with the merchant by sending messages through the payment gateway (an intermediate entity for payment clearing purposes which cannot decrypt the messages sent by the client).

Symmetric and asymmetric signature methods have been widely used to provide authentication in electronic Payment systems (including Mobile Commerce (M-commerce)) [1]. However, traditional asymmetric

*Corresponding author.
Email address: `jtellez@uc.edu.ve` (Jesús Téllez Isaac)

signature schemes make the signature computations very expensive and not suitable for those portable devices available on the market and are not based on the Texas Instruments TMS320C55x processor family (which delivers high performance, peripheral options, small packaging, and low power dissipation for the implementation of asymmetric operations in a efficient way) [2, 3]. Therefore, traditional asymmetric authentication schemes are not suitable for scenarios where an engaging party has connectivity restrictions, and consequently, communication with other parties (such a Certification Authority for verifying a certificate) is not possible during such *payment transactions*.

The mobile payment protocol proposed in this work eliminates the restriction of those mobile payment systems based on the Full Connectivity Scenario which requires direct communication between a client and a merchant for authentication purposes. Our proposed payment protocol supports both credit-card and debit-card transactions, protects the real identity of the client during the purchase and employs a symmetric-based signature scheme to satisfy the security requirements of the protocol proposed in this work. Symmetric cryptography (which employs a shared key between two parties) provides message confidentiality, message integrity, provides the authentication of participants and supports an alternative method to secure protocols for mobile payment systems. Moreover, symmetric-key operations do not require high computational power nor do they require additional communication processing steps.

The rest of the paper is organized as follows. Section 2 presents related works and the contributions of this work. In section 3, we describe the notations used in our proposed scheme, the operational model and the proposed protocol. We analyze the security and performance of the proposed protocol in Section 4. Finally, we make some concluding remarks in Section 5.

2. Related Works and Our Contributions

In the last decade, several mobile payment protocols, based on the Full Connectivity scenario (where all the entities are directly connected to each other [4]), have been proposed to improve the security of these kind of payment systems [5, 6]. Most of these proposals have considered the following methods to provide authentication: username/password, symmetric, asymmetric and elliptic curve cryptography, smart card, 2d bar code, and biometric methods [7, 8, 9] but some of them do not offer adequate security for M-commerce applications although symmetric and asymmetric signatures have been widely used for authentication purposes. Unfortunately, usage of these protocols within the Payment Gateway Centric Case mobile scenario is not possible because it restricts the communication which allows only interaction between the client and the payment gateway and between the merchant and the payment gateway. However, some protocols could be reformulated to overcome this restriction, achieving the same security and performance but for scenarios with communication restriction.

Contributions: In this work, we consider the problem of eliminating the restriction of those mobile payment systems that require direct communication between the client and the merchant for authentication purposes by providing anonymity of the origin of the data to prevent the merchant from associating the client with the messages that originated from him/her.

Inspired by the Payment Gateway Centric Model proposed by [4] and in contrast to previous proposals published for restricted scenarios [10, 11, 12, 13], we present a secure Payment Centric Model using Symmetric cryptography (PCMS) protocol for those situations where the Client cannot communicate directly with the Merchant to process the *Payment Request*.

3. Our Approach

3.1. Operational Model

The PCMS protocol was designed by taking into consideration the model suggested by Abad-Peiro et al. [14] such that the proposed protocol can be used by different payment methods. Thus, our proposed payment protocol uses the following entities: *client*, *merchant*, *issuer* (client's financial institution), and *acquirer* (merchant's financial institution). An additional entity called the *payment gateway* acts as a intermediary between acquirer/issuer (bank's private network side) and the client/vendor (the Internet side) for clearing purposes [15].

The five entities in PCMS and their interactions are shown in Figure 1. It is worth noting that is no direct interaction between the Client and the Merchant and consequently, messages between both entities should be sent through the payment gateway (which cannot decrypt this message).

The connection between the Client and the Payment Gateway and between the Merchant and the Payment Gateway is set up through the Internet, using communication technologies (wired, wireless, cellular) offered by a mobile phone operator (such as General Packet Radio Service (**GPRS**), Enhanced Data rates for Global System for Mobile Communications (**GSM**) of Evolution (**EDGE**), Evolution-Data Optimized (**EvDO**), etc.). Since the Issuer, the Acquirer, and the Payment Gateway all operate over the private networks of banks, the secure exchange of messages among them is beyond the scope of this paper.

However, before receiving Payment services, the Merchant must register with the Payment Gateway to receive the secret Sec_{M-PG} from the PG. The Client must also register with an Issuer. During the Client's registration, the following steps are performed:

1. The Client shares his/her credit- and/or debit-card information (CDCI) with the Issuer (who will not reveal it to any Merchant). The registration can be done either in person at the issuer's premises or via the issuers website.
2. The Issuer assigns several nicknames to the Client because of the trust relationship between both of them. These nicknames are known only to the Client and the Issuer and are used to prevent the Merchant from knowing the identity of the Client[1].
3. The Client registers herself/himself to the Payment Gateway. Then, the Payment Gateway shares the secret Sec_{C-PG} with the client. Besides, the issuer shares the secret Sec_{C-I} with the client.

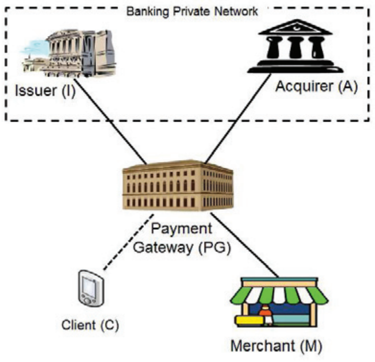


Fig. 1. Proposed Payment protocol based on the Payment Gateway Centric scenario (PCMS) Operational Model.

3.2. Notations

All the entities involved in our protocol are called parties and communicate through either wireless or wired networks or a combination of both. The symbols C, M, PG, I, and A are used to denote the names of the parties Client, Merchant, Payment Gateway, Issuer and Acquirer, respectively. The following symbols are used to represent messages used in our proposed protocol:

- **ID**$_P$;	The identity of party P that contains the contact information of P.
- **NID**$_C$:	Client's nickname, temporary identity.
- **TID**:	Identity of transaction that includes time and date of the transaction.
- **TST**$_P$:	Timestamp generated by P.
- **Stt**:	The status of transaction (Stt = {Accepted, Rejected}).
- **OD**:	Order description.
- **Price**:	Amount and currency.
- **OI**:	Order information (OI = {TID, OD, h(OD,Price)}).

- **TC**:	The type of card used in the purchase process (TC={Credit, Debit}).
- **TIDReq**:	The request for TID.
- **MIDReq**:	The request for Merchant Identity (ID_M).
- **SEC**$_{A-B}$:	The master secret shared between parties A and B.
- **{M}**$_x$:	The message M symmetrically encrypted with the shared key x.
- $h(M)$:	The one-way hash function of the message M.
- **MAC(X,K)**:	Message Authentication Code of the message X with the key K.
- KS_{A-B_t}:	The session key shared between parties A and B, generated applying a hash function with t-bit cyclic shifting (either left shift or right shift) of KS_{A-B}. More details can be found in [12].
- **"PRequest"**:	Payment Request.
- **"PResponse"**:	Payment Response.
- **"VSRequest"**:	Value-Substraction Request.
- **"VSResponse"**:	Value-Substraction Response.
- **"VCRequest"**:	Value-Claim Request.
- **"VCResponse"**:	Value-Claim Response.

3.3. Our proposed Payment Gateway Centric Model Payment Protocol (PCMS)

The PCMS Protocol is composed by two sub-protocols: the *PCMS Merchant Registration Protocol* (called *MRP* and it is executed between the Client **C** and the Merchant **M**) and the *PCMS Payment Protocol* (called *PP* and is executed among the Client **C**, the Merchant **M** and the Payment Gateway **PG**).

The Client has to register with the Merchant in the *PCMS Merchant Registration Protocol* in order to send the master key KS_{C-M}. The protocol has to be executed every time the Client wants to perform transactions with a Merchant. The details of the protocol are as follows:

The Client **C** generates the master key KS_{C-M} and sends it with her/his nickname NID_C, a nonce n for the challenge-response and *MIDReq* to **PG**, encrypted with the session key w. Once the Payment Gateway **PG** receives the message, decrypts it and forwards the received message to **M**, encrypted with the session key y. After the Merchant **M** receives the message, she/he sends the Merchant's identity (ID_M) and $h(n, NID_C, ID_M, KS_{C-M})$, encrypted with the session key y. The Payment Gateway **PG** decrypts the message sent by **M** and encrypts it with the session key w before sending it to the Client **C**.

Once **C** and **M** have exchanged the necessary information, they can generate a new set of KS_{C-M_i} using the same key generation technique. The Client may then start the *PCMS Payment Protocol*.

For the *PCMS Payment Protocol*, the Client purchases goods from the Merchant and pays for them using her/his credit-card or debit-card. This protocol is formalized as follows:

1) C → PG → M: $NID_C, i, TIDReq$

M → PG → C: $\{TID, ID_M\}_{KS_{C-M_i}}$

Step 1: The Client **C** and Merchant **M** exchange (through the Payment Gateway **PG**) the information necessary to start the protocol by performing the following sub-steps.

1-1: **C** sends his/her nickname (NID_C), the index i (that will be used to generate the session key between the Client and the Merchant) and the request for the transaction identity (*TIDReq*) to **PG**.

1-2: The Payment Gateway forwards the message received from **C** to **M**.

1-3: **M** receives the request and sends back its identity (ID_M) and TID to **PG**, encrypted with KS_{C-M_i}.

1-4: The Payment Gateway forwards the message received from **M** to **C**.

2) C → PG: $PRequest$

$$PRequest = \{OI, Price, NID_C, ID_I, TST_C, z, h(KS_{C-I_z}), VSRequest\}_{KS_{C-M_i}}, \\ MAC[(OI, Price, NID_C, ID_I, TST_C, z, h(KS_{C-I_z}), KS_{C-M_{i+1}}]$$

$$VSRequest = (MAC[(Price, h(OI), TST_C, TC, ID_M), KS_{C-I_z}], TC, TST_C)$$

Step 2: Client **C** creates a *Payment Request* (referred to in the General Payment Model [14]) using the following sub-steps.

2-1: A *Value-Subtraction Request* (called *VSRequest*) is created and it includes $MAC[(Price, h(OI), TST_C, TC, ID_M), KS_{C-I_z}]$, TST_C and TC.

2-2: A new message is created which includes $C's$ nickname, $I's$ identity, *Price*, *OI* (used to inform **M** about the goods and prices requested), *VSRequest*, the index z, the timestamp TST_C read from $C's$ clock, $h(KS_{C-I_z})$ (used to prevent the approval of the payment from being modified in step 4-5) and $MAC[(OI, Price, NID_C, ID_I, TST_C, z, h(KS_{C-I_z}), KS_{C-M_{i+1}}]$.

2-3: The message created in the previous sub-step (henceforth referred to as the Payment Request) is encrypted with the session key KS_{C-M_i}.

2-4: The *Payment Request* is sent to the Payment Gateway.

3) PG → M: $PRequest$

Step 3: The Payment Gateway **PG** receives the *Payment Request* (sent from **C**) and forwards it to the Merchant **M** for its processing.

4) M → PG: $\{VCRequest, ID_M, z, h(KS_{C-I_z})\}_{KS_{M-PG_k}}, k,$
$MAC[(VCRequest, ID_M, z, h(KS_{C-I_z}), KS_{M-PG_{k+1}}]$

$$VCRequest = (VSRequest, TST_M, h(OI), TID, Price, NID_C, ID_I)$$

Step 4: The Merchant **M** generates the *Value-Claim Request* (called *VCRequest*) by performing the following sub-steps.

4-1: The message received from **PG** is decrypted to extract *OI*, TST_C and $VSRequest$.

4-2: The timeliness of the *Payment Request* is verified. If the check is successful, the following sub-steps are performed.

4-3: The *VCRequest* is prepared and contains *VCRequest*, TST_M, *h(OI)*, the identity of the transaction, order's amount, *C's* nickname and *I's* identity.

4-4: The *VCRequest*, the *M's* identity, $h(KS_{C-I_z})$ and the index z, are encrypted with KS_{M-PG_K}.

4-5: The encrypted message in sub-step 4-4 is then transmitted to the Payment Gateway **PG** with k and $MAC[(VCRequest, ID_M, z, h(KS_{C-I_z}), KS_{M-PG_{k+1}}]$.

5) Using the bank's private network,

5.1) PG → I: $NID_C, ID_M, VSRequest, TID, h(OI), z, Price, h(KS_{C-I_z})$
5.2) PG → A: $Price, ID_M$
4.3) I,A → PG: $VSResponse, Stt, h(Stt, h(OI), h(KS_{C-I_z}))$
$VSResponse = \{Stt, h(OI), h(KS_{M-PG_{k+1}})\}_{KS_{C-I_z}}$

Step 5: Using the private network of the bank, the Payment Gateway (PG) performs the following sub-steps to verify and approve the Payment.

5-1: The *VCRequest* is decrypted to retrieve *VSRequest* and the other fields, such as ID_M, NID_C.

5-2: The timeliness of *VCRequest* is verified. If the check is successful, the following steps are executed.

5-3: The *VSRequest* and other important parameters such as: $h(OI)$, TID, ID_M, $Price$, z and $h(KS_{C-Iz})$ are forwarded to the Issuer (**I**) where it is decided whether to approve or reject the transaction.

5-4: ID_M and the requested price *Price* are sent to confirm to the Acquirer **A** that the Merchant is the party to whom the requested amount *Price* will be transferred to.

5-5: The approved result (*Stt*) and the *Value-subtraction Response* (called *VSResponse* and encrypted with KS_{C-I_z}) are received from the Issuer **I**. It is worthwhile noting that the *VSResponse* is prepared by the Issuer after (a) checking the timeliness of the *VSRequest* and the validity of the Client's account, and (b) after transferring the total amount of OI to the Merchant's account.

6) PG → M: $VCResponse$

$$VCResponse = \{Stt, h(Stt, h(OI), h(KS_{C-I_z}))\}_{KS_{M-PG_{k+1}}}$$

Step 6: The Payment Gateway **PG** generates the *Value-Claim Response* (called *VCResponse*) in the following sub-steps.

6-1: The *VCResponse* is created and includes Stt and $h(Stt, h(OI), h(K_{C-I_z}))$.

6-2: The *VCResponse* is encrypted with $KS_{M-PG_{k+1}}$ and is sent to **M**.

7) PG → C: $PResponse$

$$PResponse = \{VSResponse\}_{KS_{C-PP_{j+1}}}$$

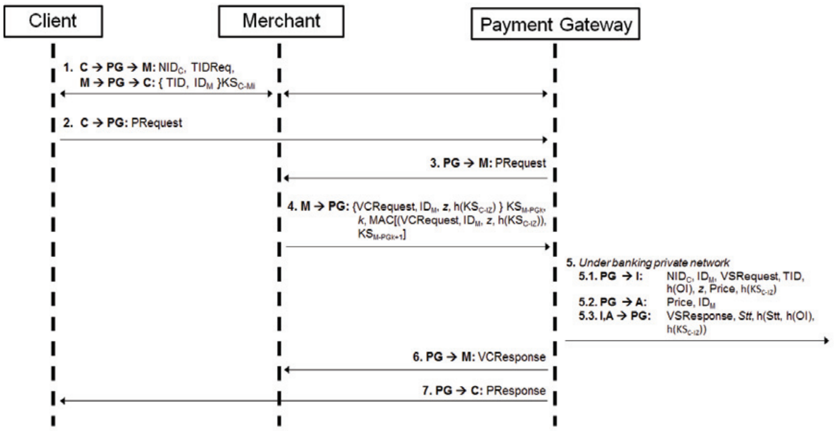
Step 7: The Payment Gateway **PG** generates the *Payment Response* (which represents the result of the client's request and is called *PResponse*) in the following sub-steps.

7-1: The *PResponse* is created and includes the *VSResponse*.

7-2: The *PResponse* is encrypted with $KS_{C-PG_{j+1}}$ and is sent to **C**.

7-3: Once the client recieves the message, it decrypts it to retrieve *VSResponse*. Then, the Client's own *OI* is compared with the received *h(OI)* to check whether or not the message is the response of his request . If they do not match, a message is sent to the Payment Gateway to notify it of the response failure. The Payment Gateway then starts a recovery procedure or resends the message.

After a transaction is completed, KS_{C-M_i}, KS_{C-I_z}, KS_{M-PG_k} y KS_{C-PG_j} are put in the revocations list of every entity of the system to prevent their replay between the Client and the Merchant. Fig. 2 shows the transmitted messages among the parties of the system during the execution of our proposed *PCMS Payment Protocol*.


Client
Merchant
Payment Gateway
1. C → PG → M: NID_C, TIDReq,
M → PG → C: { TID, ID_M }KS_C-Mi
2. C → PG: PRequest
3. PG → M: PRequest
4. M → PG: {VCRequest, ID_M, z, h(KS_C-IZ) } KS_M-PGK, k, MAC[(VCRequest, ID_M, z, h(KS_C-IZ)), KS_M-PGK+1]
5. Under banking private network
5.1. PG → I: NID_C, ID_M, VSRequest, TID, h(OI), z, Price, h(KS_C-IZ)
5.2. PG → A: Price, ID_M
5.3. I,A → PG: VSResponse, Stt, h(Stt, h(OI), h(KS_C-IZ))
6. PG → M: VCResponse
7. PG → C: PResponse

NID_C	: *Client's nickname, temporary identity.*
ID_M	: *Merchant's identity.*
TIDReq	: *Transaction Identity Request.*
MIDReq	: *Merchant's identity Request.*
TID	: *Identity of transaction.*
TST_C	: *Client's Timestamp.*
Stt	: *Transaction's status.*
KSA-Bt	: The session shared between **A** and **B**
MAC[x,k]	: Message Authentication Code of the message **x** with the key **k**.

OI	: *Order information.*
Price	: *Order's amount.*
PRequest	: *Payment Request.*
PResponse	: *Payment Response.*
VSRequest	: *Value-Substraction Request.*
VSResponse	: *Value-Substraction Response.*
VCRequest	: *Value-Claim Request.*
VCResponse	: *Value-Claim Response.*
h(x)	: *A one-way hash function* of the message **x**
I, k, z, j	: *Indexes.*

Fig. 2. Message exchange during the execution of the *PCMS Payment Protocol*.

4. Proposed Payment Protocol Analysis and Discussions

4.1. Security Analysis

In this section, we analyze the security of our proposed Payment Protocol. In the case of PCMS protocol, the client uses a nickname NID_C (a temporary identity known only to the client and the issuer) instead of his/her real identity. As a result neither the merchant nor the payment gateway can map the nickname to the client's true identity. This anonymity protects relevant information from third parties but not unrestrained anonymity (because if it is necessary, the anonymity of a transaction can be undone at a later time) [16].

The *Confidentiality* of messages transmitted in each transaction while in transit with the proposed protocol is protected by employing symmetric cryptography. Moreover, the encryption key also allows the receiver and the sender to authenticate each other. In addition, to maintain the *Integrity* of important messages, the proposed protocol uses the Message Authentication Code (MAC).

Since the session key KS_{C-I_Z} could be generated only by the Client or Issuer but not by the Merchant, the *Non-repudiation* of a transaction is ensured in the proposed protocol PCMS. Thus, the Merchant can provide a non-repudiable evidence to prove to other parties that the Client has sent a message or requested the Merchant to perform a transaction.

The Timestamp included in the transmitted message ensures the freshness of the message and prevents *replay attacks*. Moreover, the usage of different session keys from one master secret for every transaction makes the proposed protocol secure against *key guessing attacks*.

4.2. Performance Analysis

In this section, we present a performance analysis (in terms of the number of cryptographic operations of the involved parties and computation costs) of the proposed mobile payment protocol PCMS in order to demonstrate the superior performance of our proposed protocol over the payment protocols KSL (proposed by [15]) and LMPP (proposed by [17]) in terms of efficiency even using a scenario with communication restrictions. Table 1 shows the number of cryptographic operations performed by each party of KSL, LMPP and PCMS during a transaction.

We compare PCMS protocol with KSL and LMPP because they employ symmetric-key operations, the same way to exchange messages between the entities and similar session key generation technique. Hence, KSL and LMPP are good recently proposed protocols to demonstrate the efficiency of the proposed protocol PCMS.

Cryptographic Operations		**Number of Cryptographic Operations**		
		PCMS	**KSL**	**LMPP**
Symmetric-key encryptions/decryptions	**C**	3	4	5
	M	4	5	6
	PG	3	2	-
Hash Functions	**C**	2	2	2
	M	1	-	1
	PG	-	-	-
Keyed-hash functions	**C**	2	2	-
	M	1	2	-
	PG	-	1	-
Key generation	**C**	2	2	-
	M	4	1	-
	PG	3	1	-

Table 1. Number of cryptographic operations performed by the protocols PCMS, KSL and LMPP at the client (**C**), merchant (**M**) and payment gateway (**PG**). Note that the dash character ("-") means zero value.

The notation used in the computation cost analysis is as follows: a) T_{sym} represents the time required to execute an encryption/decryption symmetric operation, b) T_h represents the time needed to execute a one-way hash function, and c) T_{kh} represents the time required to execute a keyed-hash functions (or MAC).

The client needs $T_{sym} + 2T_h + 2T_{kh}$ to generate a *Payment Request* whilst the time required by the client to confirm the payment is $2T_{sym}$. Finally, the total time required by the client to perform a transaction is $4T_{sym} + 2T_h + 2T_{kh}$ (including a time T_{sym} required for the initial information exchange between **C** and **M**).

From the results in Table 1, it can be seen that PCMS requires a lower number of cryptographic operations for symmetric key encryptions/decryptions than KSL and LMPP. For the keyed-hash functions, PCMS needs a smaller number of cryptographic operations than KSL but more than LMPP. In the case of Hash functions, PCMS requires the same number of cryptographic operations as LMPP but one more than KSL.

Despite that in our proposed protocol the key generation process (needed to update keys regularly) requires more cryptographic operations than KSL and LMPP, this would not cause increased delays because the key generation processes can be done offline.

5. Conclusion

We have proposed a lightweight protocol for secure on-line payments in a restricted scenario where direct communication between the client and the merchant is not possible. Our protocol employs symmetric cryptographic techniques which has low computation requirements (since no public-key operation is required) for all engaging parties. Moreover, the Payment Gateway takes an active role in the Payment process because it acts as a proxy that allows communications between the Client and the Merchant.

Although the proposed protocol was designed for a mobile payment system based on a restricted scenario, the security properties are still preserved (as if we were working in a scenario with full connectivity among the different entities).

Our performance analysis shows that, in general terms, the proposed mobile payment protocol (PCMS) requires less computation than KSL and LMPP which leads to improved end-to-end performance and could be deployed on mobile devices with limited computational resources.

References

[1] Z.-Y. Hu, Y.-W. Liu, X. Hu, J.-H. Li, Anonymous micropayments authentication (ama) in mobile data network, in: 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04), 2004, pp. 46–53.

[2] R.-J. Hwang, S.-H. Shiau, D.-F. Jan, A new mobile payment scheme for roaming services, Electronic Commerce Research and Applications 6 (2) (2007) 184–191.

[3] R. Martinez-Pelaez, F. J. Rico-Novella, C. Satizabal, Study of mobile payment protocols and its performance evaluation on mobile devices, International Journal of Information Technology and Management 9 (3) (2010) 337–356.

[4] S. Chari, P. Kermani, S. Smith, L. Tassiulas, Security issues in m-commerce: A usage-based taxonomy, in: E-Commerce Agents, Marketplace Solutions, Security Issues, and Supply and Demand, 2001, pp. 264–282.

[5] M. Hassinen, K. Hyppönen, K. Haataja, An open, pki-based mobile payment system, in: Emerging Trends in Information and Communication Security, International Conference (ETRICS'06), 2006, pp. 86–100.

[6] F. Buccafurri, G. Lax, Implementing disposable credit card numbers by mobile phones, Electronic Commerce Research 11 (3) (2011) 271–296.

[7] N. K. Ratha, J. H. Connell, R. M. Bolle, Enhancing security and privacy in biometrics-based authenticationsystems, IBM Systems Journal 40 (3) (2001) 614–634.

[8] J. Gao, V. Kulkarni, H. Ranavat, L. Chang, A 2d barcode-based mobile payment system, in: Third International Conference on Multimedia and Ubiquitous Engineering (MUE 2009), 2009, pp. 320–329.

[9] O. R. Vincent, O. Folorunso, A. Akinde:, Improving e-payment security using elliptic curve cryptosystem, Electronic Commerce Research 10 (1) (2010) 27–41.

[10] J. T. Isaac, J. S. Camara, A. I. Manzanares, J. T. Márquez:, Anonymous payment in a kiosk centric model using digital signature scheme with message recovery and low computational power devices, Journal of Theoretical and Applied Electronic Commerce Research 1 (2) (2006) 1–11.

[11] J. T. Isaac, J. S. Cámara, A secure payment protocol for restricted connectivity scenarios in m-commerce, in: E-Commerce and Web Technologies, 8th International Conference (EC-Web'07), 2007, pp. 1–10.

[12] J. T. Isaac, S. Zeadally, J. S. Camara, Implementation and performance evaluation of a payment protocol for vehicular ad hoc networks, Electronic Commerce Research 10 (2) (2010) 209–233.

[13] W. Li, Q. Wen, Q. Su, Zhengping, An efficient and secure mobile payment protocol for restricted connectivity scenarios in vehicular ad hoc network, Computer Communications 35 (2) (2012) 188–195.

[14] J. L. Abad-peiro, N. Asokan, M. Steiner, M. Waidner, Designing a generic payment service, IBM Systems Journal 37 (1) (1997) 72–88.

[15] S. Kungpisdan, B. Srinivasan, P. D. Le, Lightweight mobile credit-card payment protocol, in: 4th International Conference on Cryptology in India (Progress in Cryptology - INDOCRYPT 2003), 2003, pp. 295–308.

[16] N. Asokan, Anonymity in mobile computing environment, in: First Workshop on Mobile Computing Systems and Applications (wmcsa 1994), 1994, pp. 200–204.

[17] T. S. Fun, L. Y. Beng, J. Likoh, R. Roslan, A lightweight and private mobile payment protocol by using mobile network operator, in: International Conference on Computer and Communication Engineering, 2008, pp. 162–166.