ISSN 1392 – 124X, ISSN 2335 – 884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2013, Vol. 42, No. 4

Using a Stored-Value Card to Provide an Added-Value Service of Payment Protocol in VANET

Chin-Ling Chen[1], Wei-Chen Tsai[1], Yu-Yi Chen[*,2], Woei-Jiunn Tsaur[3]

[1] *Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung, Taiwan, (R.O.C.), 41349*
e-mail: clc@mail.cyut.edu.tw; s10027628@cyut.edu.tw

[2] *Department of Management Information systems, National Chung Hsing University, Taiwan, Taichung, Taiwan, (R.O.C.), 402,*
e-mail: chenyuyi@nchu.edu.tw

[3] *Department of Information Management, Da-Yeh University, Taiwan, Changhua, Taiwan, (R.O.C.), 51591*
e-mail: wjtsaur@mail.dyu.edu.tw

crossref http://dx.doi.org/10.5755/j01.itc.42.4.4471

Abstract. With the rapid development of the Internet applications, added-value service is widely used in the Internet. The added-value service provides different kind of services for users. In this paper, we propose a stored-value card to provide an added-value service of payment protocol in VANET. When user wants to enjoy the added-value service, the service provider verifies the request and sends it to the payment gateway. Then the payment gateway forwards the transaction message to the Issuer and Acquirer to process it. In our scheme, we use symmetric cryptography and digital signature to solve the security problem of payment scheme in VANET. Our scheme achieves protection against double-spending, unforgeability, non-repudiation, anonymity and the recovery issue.

Keywords: Added-value service; stored-value card; VANET; cryptanalysis; payment.

1. Introduction

With the rapid development of the Internet applications, people can use the Internet to deal with transactions. The added-value service [18] provides different kind of services for users. Added-value service refers to extra features of an item of interest that go beyond the standard expectations and provide something more while adding small cost. The benefits of the added-value service are that the consumer can easily use the various services through the service provider and the service provider can provide discount or bonus for the consumer. The stored-value card [4, 7] is a kind of smartcard that can store monetary value. It is used in a wide range of applications [5, 14] in micropayment environment. People can add value into stored-value card. The consumer has to register a stored-value card with the card issuer, and then the consumer can use stored-value to enjoy added-value services and add value anywhere. For example, when going parking, the consumer can use stored-value card to payment and obtain bonus to make more parking times. Even taking the bus or train, the consumer can pay the stored-value card to get more convenient. The consumer can obtain more benefit in life through added-value services.

Added-value applications in VANET, which improves user comfort and offers great business opportunities, attract more and more attention in daily life. Most of applications come with the emergence of electronic trade. In the Vehicular ad hoc networks (VANET), VANET is envisioned to support the development of a wide range of attractive applications such as payment services [9, 10, 11, 13, 17]. In 2010, Isaac et al. [10] proposed implementation and performance evaluation of a payment protocol for vehicular ad hoc networks. They use symmetric-key operations to design a payment protocol in VANET to achieve low computation. In 2012, Li et al. [13] proposed an efficient and secure mobile payment

* Corresponding author

protocol for restricted connectivity scenarios in vehicular ad hoc network. They use self-certified key agreement to establish symmetric keys. Thus, both the computational cost and communication cost can be reduced. In 2012, Isaac et al. [9] improve previous paper to propose a lightweight secure mobile Payment protocol for vehicular ad-hoc networks. They also use symmetric-key operations to design and implement a lightweight secure payment protocol. But, Isaac et al.'s scheme does not achieve non-repudiation, unforgeability, and recovery issue. In similar environment, we design a secure payment protocol to solve Isaac et al.'s weakness. To enable payments in VANETs, it is necessary to build payment systems that satisfy the VANET requirements [13, 15]. Therefore, we use a stored-value card to provide an added-value service of payment protocol in VANET.

In our scheme, we exploit symmetric cryptography to satisfy the security of the communication. In each transaction, sender must sign the transaction message, and receiver will verify if the signature is true. Symmetric cryptography can provide message confidentiality, and provide the authentication of participants for e-payment scheme. Moreover, symmetric-key operations neither need high computational cost nor do they require additional communication steps. In this paper, the Services Provider (SP) provides the added-value services for user. The user can use a stored-value card and select added-value services through On-Board Unit (OBU). Then the Services Provider processes the transaction message and sends it to the Payment Gateway (PG). The Payment Gateway will forward the transaction message between the Services Provider and the Issuer.

The proposed scheme must be able to achieve the following requirements [3, 8, 11, 12, 15] in order the proposed scheme could be applied in real world.

1. Double spending: The digital cash cannot be copied and reused. Then we have to minimize the risk of forgery and establish an authenticity system.
2. Unforgeability: Only authorized parties (i.e. the bank or the Issuer) can produce the stored-value card and added value.
3. Non-repudiation: Everyone should sign each transaction as proof of the integrity and origin of data.
4. Anonymity: The user must remain anonymous in each transaction.
5. Recoverable: If the user loses his or her stored-value card, he or she can recover his or her card.

The remainder of this paper is organized as follows. Section 2 presents our proposed scheme. In Section 3 we make a security analysis. Finally, the conclusion is offered in Section 4.

2. The proposed scheme

2.1. Notations

The following notations are used in the proposed scheme:

ID_x	: the identity of *x*
U	: the user
V	: the vehicular
SP	: the server provider
I	: the issuer bank
OBU	: the on-board unit
AVM	: the added-value machine
A	: the services provider's financial institution
PG	: the e-commerce application service provider between the acquirer and the issuer bank
PW	: the password of user
s	: the permanent secret parameter of Issuer
$Cert_x$	: the certificate generated by *x*
N_x	: the nonce generated by *x*
T_x	: the timestamp generated by *x*
$SK_{\text{A\_B}}$	: the session key between A and B
$value_C$	: the current stored value of the stored-value card
$value_{add}$	: the amount which the passenger wants to add into the stored-value card
msg_{conf}	: the confirmation message
TN	: the transaction number
OI	: the order information (*OI*=(*service*, *price*, *time*)), where *service* is a kind of service from Service Provider, *price* is the fee of the service, and *time* is the current time
Pub_x / Prv_x	: the public key / private key of *x*
$E_K[M]/D_K[M]$	: using the symmetric key K to encrypt/ decrypt the message M
$S_X(M)/V_X(M)$	: using the X's private/public key to sign/ verify the message M
$h(\cdot)$	: a one way hash function
$\oplus$	: bitwise exclusive operator
$A \stackrel{?}{=} B$	: determines if A is equal to B

2.2. Architecture

There are nine participants involved in the proposed scheme. The Overview of the proposed architecture is presented in Figure 1.

1. Services Provider (SP): The services provider is an organization that provides some kind of value-added services.
2. User (U): The user makes a request to services provider.

3. Vehicular (V): The vehicular denotes a car or transportation.
4. Issuer (I): The issuer is the user's financial institution and issues stored-value card.
5. On-Board Unit (OBU): An OBU is a device which is installed into a seat in car.
6. Key Generation Centre (KGC): A Key Generation Centre generates public key and private key to each participant.
7. Added-Value Machine (AVM): The AVM is an intelligent equipment to provide self-served added value services.
8. Payment Gateway (PG): The payment gateway is an e-commerce application service provider between the acquirer and the issuer bank on the private network.
9. Acquirer (A): The acquirer is the services provider's financial institution.

Step 1: KGC→I, U,V, SP, A: The KGC allocates pubic key and private key to the Issuer, User, Vehicular, Services Provider, and Acquirer.

Step 2: U↔I: The User registers a new stored-value card with the Issuer. The Issuer creates an account for the user and returns stored-value card to user. After receiving the card, the User will write the private key in the stored-value card.
SP↔A: The Services Provider registers an account with the Acquirer. The Acquirer creates an account and returns a certificate to the services provider.

Step 3: U↔I: Before each transaction, the User must communicate with the Issuer to coordinate a session key.
V↔SP: Before each transaction, the Vehicular must communicate with the Services Provider to coordinate a session key.
SP↔I: Before each transaction, the Services Provider must communicate with the Issuer to coordinate a session key.

Step 4: U↔I: When the User wants to add value into the stored-value card via the AVM, the AVM will forward the encrypted transaction message to the Issuer. The Issuer will decrypt the message and verify whether the stored-value card is legal or not. Finally, the AVM adds value in the card.

Step 5: U→SP: When the User wants to use the added-value service, the User has to use OBU to select an added-value service. And, OBU sends the request to the Services Provider.

Step 6: SP→PG: Upon receiving the request message from OBU, the Services Provider will check the transaction information and forward the payment message to the Payment Gateway.

Step 7: PG→I: After receiving the payment message from Services Provider, the Payment Gateway will forward the payment message to the Issuer to authenticate through private network.

Step 8: I→PG: The Issuer checks the transaction information and deducts the price from the User's account. Then the Issuer sends the confirmation message to the Payment Gateway.

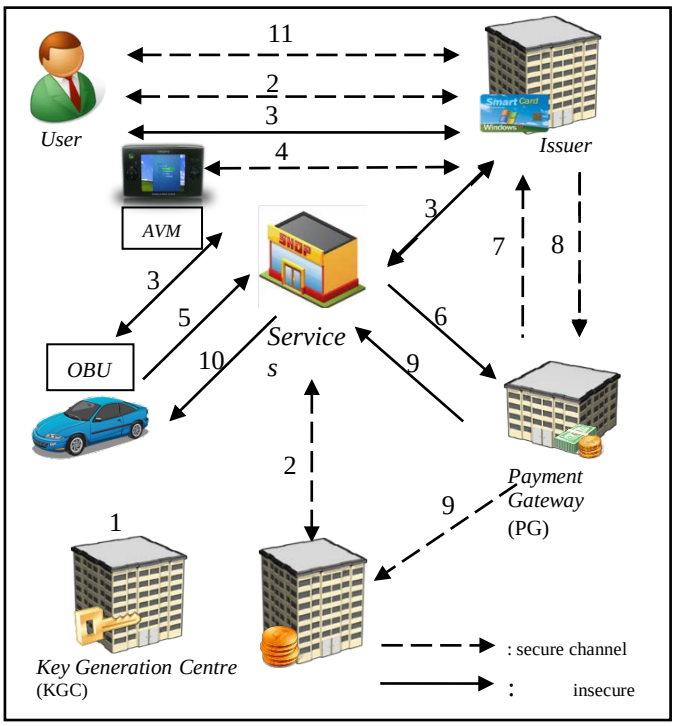

Figure 1. Overview of the proposed architecture

Step 9: PG→SP: When the Payment Gateway receives the confirmation message from the Issuer, the Payment Gateway will forward the confirmation message to the Services Provider.
PG→A: The Payment gateway sends the transaction information to the Acquirer. Upon receiving the message, the Acquirer will transfer the transaction fee to the Services Provider's account.

Step 10: SP→V: After receiving the confirmation message, the Services Provider supplies the added-value service to the Vehicular.

Step 11: U↔I: If the User lost his/her stored-value card, the User can send the recovering request to the Issuer. After receiving the request, the Issuer verifies if the User is legal. If the user is legal, the Issuer will send the new stored-value card to the user according to the User's account.

2.3. The initialization phase

In this phase, the Key Generation Centre (KGC) allocates public key and private key to each participant. When receiving the key pair, they will store private key and publish pubic key. Then they can use private key to make a signature and use the public key to verify if the signature is valid or not.

2.4. The registration phase

2.4.1. The User registers with the Issuer

In this phase, the User registers a stored-value card from the Issuer. The Issuer creates an account for User. The overview of this phase is presented in Figure 2. The steps are described as follows:

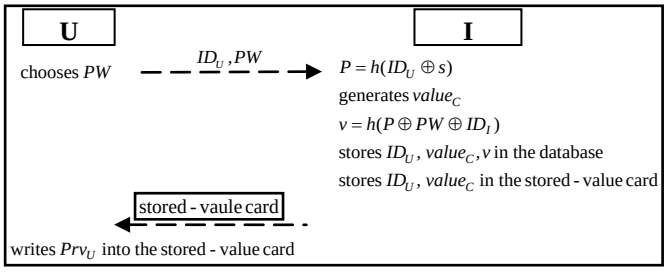

Figure 2. Overview of the registration phase (the U registers with the I)

Step 1: The User chooses a password PW and sends ID_U, PW to register with Issuer via a secure channel.

Step 2: After receiving the registration message, the Issuer computes

$$P = h(ID_U \oplus s) \quad (1)$$

and generates $value_C$ and PW. Then computes

$$v = h(P \oplus PW \oplus ID_I) \quad (2)$$

stores $ID_U, value_C$ and v in the database; and stores $ID_U, value_C$ in the stored-valued card. Finally, the I sends the stored-value card to the User.

Step 3: Upon receiving the stored-value card, the User writes Prv_U into the stored-value card.

2.4.2. The Services Provider registers with the Acquirer

In this phase, the Services Provider registers with the Acquirer, and the Acquirer will store each transaction data to the Services Provider's account. The Acquirer creates an account for the Services Provider and sends a registration certificate to the Services Provider. The overview of this phase is presented in Figure 3. The steps are described as follows:

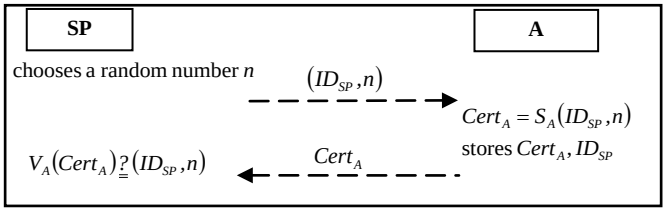

Figure 3. Overview of the registration phase (the SP registers with the A)

Step 1: The Services Provider chooses a random number n and sends (ID_{SP}, n) to Acquirer via a secure channel.

Step 2: After receiving the registration message, the Acquirer computes

$$Cert_A = S_A(ID_{SP}, n) \quad (3)$$

Finally, the Acquirer sends the registration certificate $Cert_A$ to the Services Provider and stores $Cert_A, ID_{SP}$ in the database.

Step 3: Once receiving the registration certificate, the Services Provider verifies the certificate using the Acquirer's public key

$$V_A(Cert_A) \stackrel{?}{=} (ID_{SP}, n) \quad (4)$$

If Eq. (4) holds, the Services Provider stores the certificate in the database.

2.5. The key agreement phase

In 1976, Diffie and Hellman [6] proposed a key agreement protocol. The RFC 2631 was drawn up for this key agreement protocol in 1999 by the IETF (Internet Engineering Task Force) [16]. Our scheme uses the RFC 2631 protocol to construct a session key according to Chen and Liu's scheme [1]. For each transaction, the sender and receiver should negotiate a session key before communication. For example, before the User communicates with the Issuer, they have to establish a session key SK_{U_I} to authenticate the message.

2.6. The added-value phase

In this phase, when the User wants to add value to stored-value card, the User must store value into stored-value card through AVM. In our scheme, we assume the communication channel is secure between AVM and the Issuer [2]. The User sends the added-value message to the Issuer via AVM, and then the AVM forwards the encrypted message to the Issuer. After receiving the message, the Issuer verifies the User and stores the amount in the stored-value card. Finally, the Issuer returns the confirmation message to the User. The overview of this phase is presented in Figure 4. The scenario is described as follows:

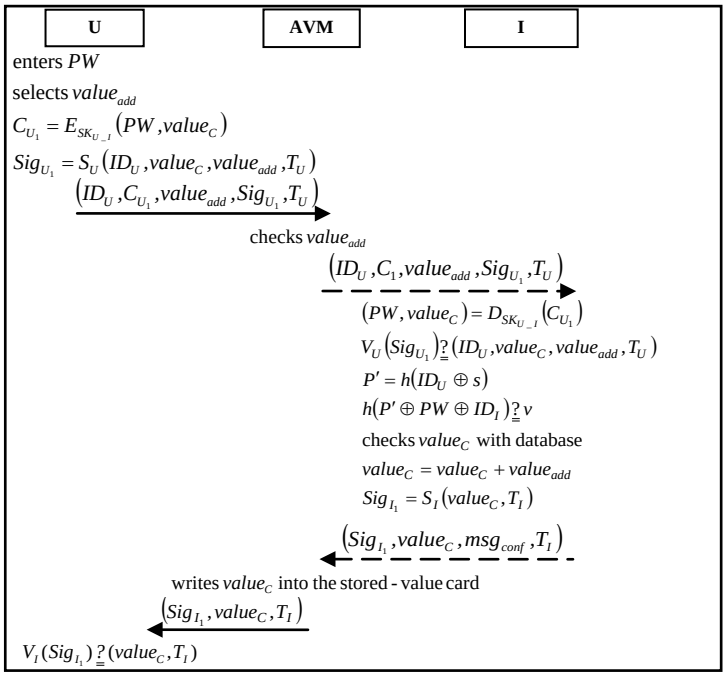

Figure 4. Overview of the added-value phase

Step 1: The User enters PW and $value_{add}$, then encrypts the message $(PW, value_C)$

$$C_{U_1} = E_{SK_{U\_I}}(PW, value_C) \quad (5)$$

and makes a signature Sig_{U_1}

$$Sig_{U_1} = S_U(ID_U, value_C, value_{add}, T_U). \quad (6)$$

Finally, the User sends $(ID_U, C_{U_1}, value_{add}, Sig_{U_1}, T_U)$ to the AVM.

Step 2: After receiving the transaction message, the AVM checks if the User's cash is valid or not and checks if $value_{add}$ is correct. Finally, the AVM forwards the message $(ID_U, C_{U_1}, value_{add}, Sig_{U_1}, T_U)$ to Issuer.

Step 3: Once receiving the message, the Issuer checks if the timestamp T_U is in a valid time. Then, the Issuer decrypts C_{U_1}

$$(PW, value_C) = D_{SK_{U\_I}}(C_{U_1}) \quad (7)$$

and verifies the signature Sig_{U_1}

$$V_U(Sig_{U_1}) \stackrel{?}{=} (ID_U, value_C, value_{add}, T_U). \quad (8)$$

If Eq. (8) holds, then it computes

$$P' = h(ID_U \oplus s) \quad (9)$$

$$h(P' \oplus PW \oplus ID_I) \stackrel{?}{=} v \quad (10)$$

If Eq. (10) holds, the User is a legal user. Next, the Issuer checks $value_C$. If $value_C$ is right in the database, it proofs $value_C$ is not tampered. Then the Issuer computes

$$value_C = value_C + value_{add} \quad (11)$$

and then makes a confirmation message msg_{conf} and a signature Sig_{I_1}

$$Sig_{I_1} = S_I(value_C, T_I). \quad (12)$$

Finally, the Issuer sends (Sig_{I_1}, $value_C$, msg_{conf}) to the AVM.

Step 4: Upon receiving the confirmation message, the AVM writes $value_C$ into the stored-value card and forwards the message ($Sig_{I_1}, value_C, T_1$) to the User.

Step 5: The User verifies if the signature Sig_{I_1} is valid or not:

$$V_I(Sig_{I_1}) \stackrel{?}{=} (value_C, T_I). \quad (13)$$

If Eq. (13) does not hold, the User can request the Issuer to check and resend the message.

2.7. The payment phase

In this phase, when the User wants to use added-value services at anywhere, the User can communicate with services provider through OBU and the Issuer will check the transaction in time. The User selects the added-value services and sends order information to the Services Provider. The Services Provider checks if the order information is supported or not and sends this transaction information to the Payment Gateway. The Payment Gateway forwards the data to the Issuer via secure channel. The Issuer verifies the transaction and sends the confirmation message to the Acquirer and Services Provider. Then, the Acquirer stores amount of added-value services to Service Provider's account. Finally, the User can obtain the services from the Services Provider. The overview of this phase is presented in Figure 5. The scenario is described as follows:

Step 1: The Vehicular enters PW, selects $OI(OI = service, price, time)$ and makes a signature Sig_{V_1}

$$Sig_{V_1} = S_U(ID_U, OI, T_V) \quad (14)$$

Then the User computes C_{V_1} and C_{V_2}

$$C_{V_1} = E_{SK_{U\_I}}(ID_U, PW, value_C) \quad (15)$$

$$C_{V_2} = E_{SK_{V\_SP}}(OI). \quad (16)$$

Finally, the Vehicular sends the message $(ID_V, C_{V_1}, C_{V_2}, Sig_{V_1}, T_V)$ to the Services Provider.

Step 2: After receiving the message, the Services Provider decrypts C_{V_2}

$$(OI) = D_{SK_{V\_SP}}(C_{V_2}) \quad (17)$$

and checks if *OI* is supported by the server. The Services Provider generates *TN* and computes

$$d = h(TN \oplus OI). \quad (18)$$

Then the Service Provider computes C_{SP}

$$C_{SP} = E_{SK_{SP\_I}}(C_{V_1}, d, OI, Cert_A, TN, Sig_{V_1}) \quad (19)$$

Finally, the SP sends the message (ID_{SP}, C_{SP}, T_V) to the Payment Gateway.

Step 3: Upon receiving the message, the Payment Gateway forwards the transaction message (ID_{SP}, C_{SP}, T_V) to the Issuer via secure channel.

Step 4: Once receiving the message, the Issuer decrypts C_{SP_1}

$$(C_{V_1}, d, OI, Cert_A, TN, Sig_{V_1}) = D_{SK_{SP\_I}}(C_{SP}) \quad (20)$$

Next, the Issuer checks d. If d has the same parameter in the database, i.e. the User has double-spending, this transaction will be rejected. Otherwise, it stores d in the database.

Then it decrypts C_{V_1}

$$(ID_U, PW, value_C) = D_{SK_{U\_I}}(C_{V_1}) \quad (21)$$

and verifies the signature Sig_{V_1}

$$V_U(Sig_{V_1}) \stackrel{?}{=} (ID_U, OI, T_V). \quad (22)$$

If Eq. (22) holds, it checks if the $value_C$ is true or not

$$\text{checks } value_C \stackrel{?}{\geq} price. \quad (23)$$

Then it computes

$$P' = h(ID_U \oplus s) \quad (24)$$

$$h(P' \oplus PW \oplus ID_I) \stackrel{?}{=} v. \quad (25)$$

If Eq. (25) holds, the User is a legal user. Afterwards, it computes

$$value_C = value_C - price \quad (26)$$

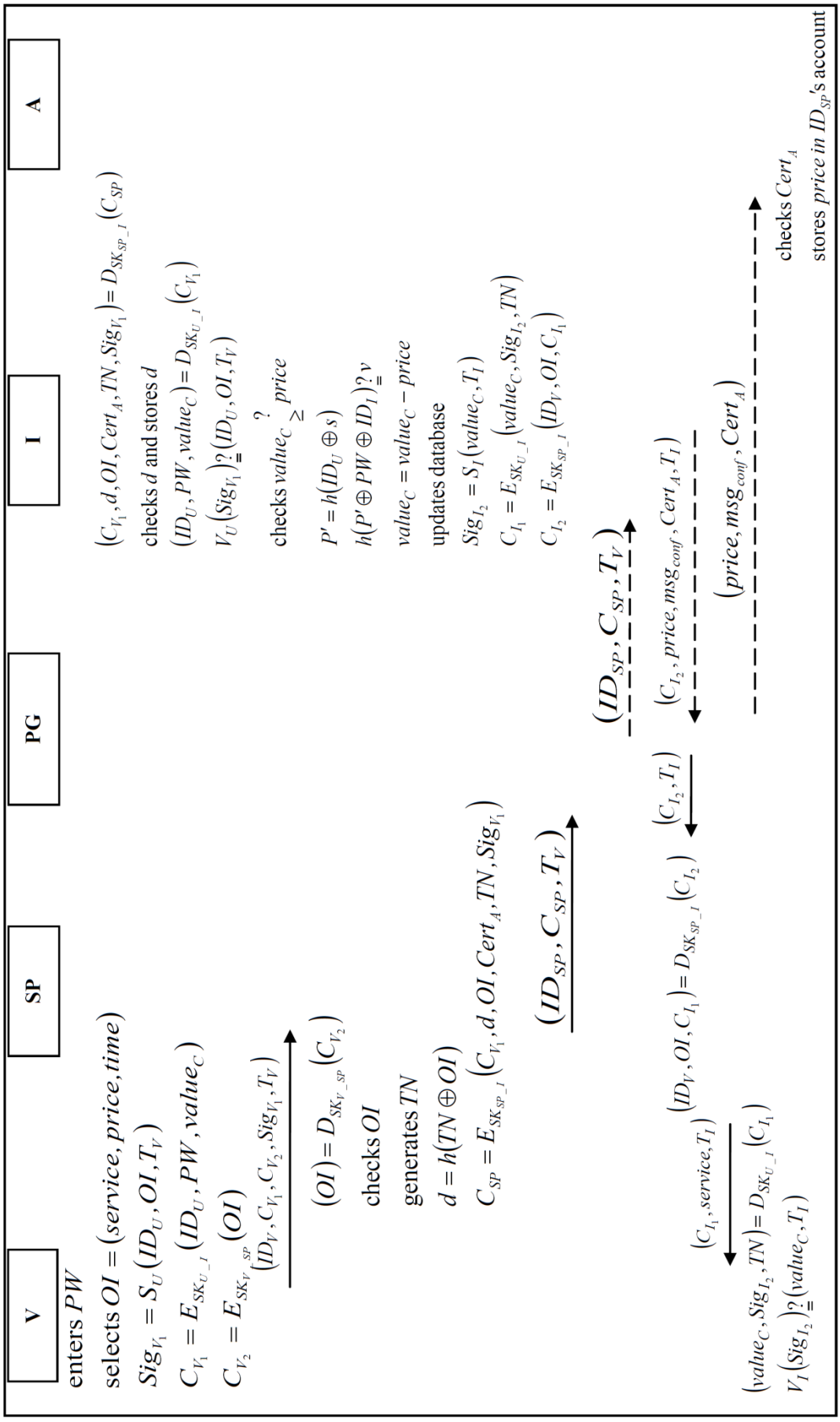

Figure 5. Overview of the payment phase

and updates to the database. The Issuer makes a signature Sig_{I_2}

$$Sig_{I_2} = S_I(value_C, T_I) \quad (27)$$

and computes C_{I_1} and C_{I_2}

$$C_{I_1} = E_{SK_{V\_I}}(value_C, Sig_{I_2}, TN) \quad (28)$$

$$C_{I_2} = E_{SK_{SP\_I}}(ID_V, OI, C_{I_1}). \quad (29)$$

Finally, the Issuer sends the message $(C_{I_2}, price, msg_{conf}, Cert_A, T_I)$ to the Payment Gateway.

Step 5: After receiving the message, the Payment Gateway forwards the message (C_{I_2}, T_I) to the Services Provider and sends $(price, msg_{conf}, Cert_A)$ to the Acquirer.

Step 6: When receiving the confirmation message, the Acquirer checks if the certificate $Cert_A$ is stored in the database. If it is valid, the Acquirer will store $price$ to Services Provider's account.

Step 7: When receiving the confirmation message, the Acquirer checks if the certificate $Cert_A$ is stored in the database. If it is valid, the Acquirer will store $price$ to Services Provider's account.

Step 8: Upon receiving the message, the Services Provider decrypts C_{I_2}

$$(ID_V, OI, C_{I_1}) = D_{SK_{SP\_I}}(C_{I_2}) \quad (30)$$

and sends the message $(C_{I_1}, service, T_I)$ to the Vehicular.

Step 9: The Vehicular decrypts C_{I_1}

$$(value_C, Sig_{I_2}, TN) = D_{SK_{V\_I}}(C_{I_1}) \quad (31)$$

and verifies the signature Sig_{I_2}

$$V_I(Sig_{I_2}) \stackrel{?}{=} (value_C, T_I). \quad (32)$$

If Eq. (32) holds, it completes this transaction; otherwise, the User can request the Issuer to check and resend the message.

2.8. The recovering phase

In this phase, when the user lost his or her stored-value card, he or she can send the recovering request to the Issuer. First, the Issuer verifies the User's legality, and then the Issuer can recover a new stored-value card to the User. The overview of this phase is presented in Figure 6. The scenario is described as follows:

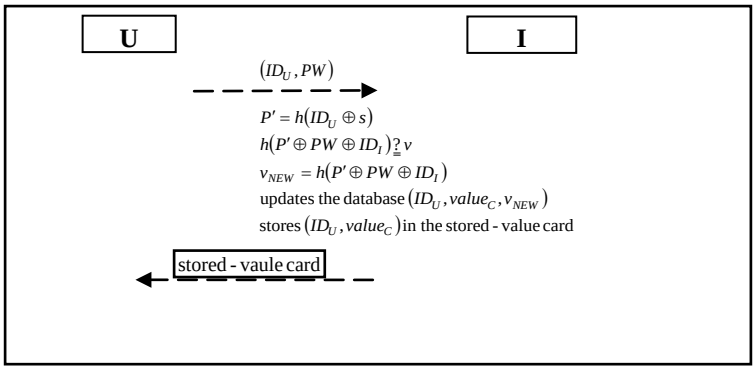

Figure 6. Overview of the recovering phase

Step1 : The User sends (ID_U, PW) to the Issuer.

Step 2: After receiving the message, the Issuer computes

$$P' = h(ID_U \oplus s) \quad (33)$$

$$h(P' \oplus PW \oplus ID_I) \stackrel{?}{=} v. \quad 34)$$

If Eq. (34) holds, the User is a legal user. Then the Issuer computes

$$v_{NEW} = h(P' \oplus PW \oplus ID_I), \quad (35)$$

updates the database $ID_U, value_C$ and v_{NEW}, and stores $(ID_U, value_C)$ in the stored-valued card. Finally, the Issuer sends a new stored-value card to the User.

3. Security analysis

In this section, we will analyze the security issues and determine whether our scheme conforms to the above mentioned security requirements.

3.1. Resists replay attack

In our scheme, we use a timestamp to prevent a replay attack in added-value and payment phase. If an attacker wants to replay the used message, it will fail because the sending message includes the timestamps T_x. After receiving the replay message, the receiver checks the validity of the signature. If it does not hold, the receiver will terminate this transaction. Therefore, our scheme can prevent the replay attack.

Scenario 1: If an *Attacker* intercepts the transaction message and sends it to Issuer in the added-value phase, he or she will fail.

Proof: When receiving the replay message, the Issuer will check if the signature Sig_{U_1} is valid or not via Eq. (8). If Eq. (8) does not hold, the Issuer will reject this transaction.

Scenario 2: If an *Attacker* intercepts the transaction message and sends it in the payment phase, he or she will fail.

Proof: When an *Attacker* replays the massage (ID_{SP}, C_{SP}, T_V) to the Issuer, the Issuer will decrypt C_{SP} and C_{VI} to get (ID_U, OI, T_V) via Eqs. (20), (21). Afterwards, the Issuer can check if the signature Sig_{V_1} is valid or not via Eq. (22). If Eq. (22) does not hold, the Issuer will reject this transaction.

3.2. Resists Double spending

In our scheme, we can detect double spending if the User sends the same request more than once through an online authentication. In the payment phase, Services Provider generates a unique transaction number *TN* in each transaction and computes a parameter. Then, the Issuer can use it to detect double spending.

Scenario 3: If an *Attacker* intercepts the transaction message and sends it to cheat the Issuer, he or she will fail.

Proof: In each transaction, the Service Provider generates a unique transaction number *TN* and computes a parameter *d* via Eq. (18). After receiving the message, the Issuer will check if *d* exists in the database. If it is true, the Issuer will reject this transaction.

3.3. Unforgeability

In our scheme, the Issuer is a trusted party, and only it can issue the stored-value card. Each e-cash must be generated and signed by the Issuer. After receiving the stored-value card, the User verifies if the signature was signed by the Issuer. If it holds, this e-cash is valid. Therefore, our scheme offers unforgeability.

Scenario 4: If an *Attacker* wants to forge the e-cash, he will fail.

Proof: In the added-value phase, each e-cash is generated by the AVM. After receiving the confirmable message, the AVM makes a signature Sig_{I_1} via Eq. (12). On the other hand, if *Attacker* wants to forge the e-cash, he or she must forge the Issuer's private key. In fact, it is impossible.

Scenario 5: If an *Attacker* wants to forge a stored-value card and use it, he or she will fail.

Proof: In the payment phase, if an *Attacker* forges the e-cash then there should be a payment. When receiving the transaction message, the Issuer checks the database through Eq. (23). If it is false, the Issuer will reject this transaction.

3.4. Anonymity

In our scheme, when the User takes the added-value services, the User remains anonymous for each transaction. In the payment phase, when the User sends the transaction message to the Services Provider, the Services Provider only knows the Vehicle's Identity ID_V. It does not know the User's real identity, only the Issuer can decrypt C_{V_1} to obtain the User's Identity ID_U. So, our scheme offers anonymity.

Scenario 6: If an *Attacker* tries to distinguish between transaction message and real identity of the User, he or she will fail.

Proof: In the payment phase, the User uses the Vehicle's Identity ID_V to perform a payment. The Vehicle's Identity ID_V is like Electronic license. The Service Provider can use it to correspond the Vehicle. Only the Issuer knows the User's real identity. It can decrypt C_{V_1} to obtain the User's Identity ID_U via Eq. (21). Hence, if the *Attacker* wants to know ID_U., he must decrypt C_{V_1} to obtain the real identity. In fact, it is impossible.

3.5. Recovery issue

When the User loses their stored-value card, or the stored-value card cannot be used, the User can initiate the recovering phase through the Issuer. In the recovering phase, the Issuer has to support the recovering service for the User. The User then sends ID_U and *PW* to the Issuer via a secure channel. Our scheme therefore satisfies the recovery issue.

Scenario 7: If an *Attacker* wants to recover the User's stored-value card, he or she will fail.

Proof: In the recovering phase, if *Attacker* wants to recover the User's stored-value card, he or she has to know the User's password and generate a P' via Eq. (33). When the *Attacker* forges a password PW', the Issuer can identify the *Attacker's* identity through Eq. (34). If Eq. (34) does not hold, the *Attacker* is an illegal user. The Issuer will reject this request.

3.6. Non-repudiation

In our proposed scheme, we use the signature to achieve the non-repudiation such that it can constitute evidence of the same transaction. When issuing the stored-value card to the User, the Issuer must provide the User with a signature. The User can then verify whether the e-cash is valid or not. Therefore, our scheme offers non-repudiation. The verifiable proofs of non-repudiation are shown in Table 1.

Table 1. The verifiable proofs of non-repudiation

Evidence	Evidence issuer	Evidence holder	Evidence verification
$(ID_U, C_{U_1}, value_{add}, Sig_{U_1}, T_U)$ $Sig_{U_1} = S_U(ID_U, value_C, value_{add}, T_U)$	User	Issuer	$V_U(Sig_{U_1}) \stackrel{?}{=} (ID_U, value_C, value_{add}, T_U)$
$(Sig_{I_1}, value_C, T_I)$ $Sig_{I_1} = S_I(value_C, T_I)$	Issuer	User	$V_I(Sig_{I_1}) \stackrel{?}{=} (value_C, T_I)$
$(ID_V, C_{V_1}, C_{V_2}, Sig_{V_1}, T_V)$ $C_{V_1} = E_{SK_{U\_I}}(ID_U, PW, value_C)$ $C_{V_2} = E_{SK_{V\_SP}}(OI)$	Vehicular	Issuer	$V_U(Sig_{V_1}) \stackrel{?}{=} (ID_U, OI, T_V)$
$(C_{I_1}, service, T_I)$ $C_{I_1} = E_{SK_{V\_I}}(value_C, Sig_{I_2}, TN)$ $Sig_{I_2} = (value_C, T_I)$	Issuer	Vehicular	$V_I(Sig_{I_2}) \stackrel{?}{=} (value_C, T_I)$

4. Discussion

In this section, we summarize the security comparisons and computation cost of related payment protocol. In 2012, Isaac et al. [9] proposed a lightweight secure mobile payment protocol for vehicular ad-hoc networks. They designed and implemented a secure payment protocol that allowed the merchant to send a message to the Acquirer through a Client for authentication. We also use the similar architecture to design the payment protocol. Thus, we analyze our scheme and Isaac's scheme to prove our scheme to be more efficient and securer than their scheme.

4.1. Security comparison

As Table 2 shows, Isaac et al. do not provide unforgeability because they do not focus on withdraw or added-value the e-cash. Our scheme uses the stored-value card to add value into the card. Their scheme also does not support recovery issue and unforgeability. But, our scheme supports recovery issue when the user has lost his or her stored-value card. Even Isaac et al. claim that their scheme satisfies the non-repudiation requirement, but the authors only use a hash function rather than digital signature. They do not meet the non-repudiation requirement. In our scheme, we use the digital signature to prove the non-repudiation. However, our scheme stresses these weaknesses, so our scheme is more secure than that by Isaac et al.

4.2. Computation cost

In this section, we compare the computation cost presented by us and Isaac et al.'s scheme in Table 3. Isaac et al.'s scheme uses a lot of symmetric encryption / decryption operations to protect the communication messages. The computation cost encloses seven hash operations and twelve symmetric encryption / decryption operations in the payment phase. Even though the computation cost is higher than Isaac et al.'s, our scheme uses the signature and the asymmetric encryption mechanism to achieve the non-repudiation requirement. Our scheme is more secure than Isaac et al.'s scheme.

Table 2. Security comparison of the proposed scheme and Isaac's scheme

	Ours	Isaac et al. [9]
Replay attack	Yes	Yes
Double spending	Yes	Yes
Unforgeability	Yes	NA
Anonymity	Yes	Yes
Recovery issue	Yes	NA
Non-repudiation	Yes	No

Table 3. Computation cost of the proposed scheme and Isaac et al.'s scheme.

Phase	Ours	Isaac et al. [9]
Added-value / Withdrawing	$2\ T_h + 2\ T_{sym} + 4\ T_{sig}$	NA
Payment	$3\ T_h + 10\ T_{sym} + 6\ T_{sig}$	$7\ T_h + 12\ T_{sym}$
Recovering	$3\ T_h$	NA

T_h: the time for executing a one-way hash function
T_{sym}: the time for executing a symmetric encryption / decryption operation
T_{sig}: the time for executing / verifying a signature

5. Conclusion

In this paper, we propose a stored-value card to provide an added-value service of payment protocol in VANET. The user can exploit the stored-value card to use added-value service from the service provider. Our scheme proposes a recovering mechanism when the

user has lost the stored-value card, the user can perform the recovering process through the Issuer. We use symmetric cryptography and digital signature to offer the security of payment scheme in VANET. Moreover, our scheme can defend against replay attack and satisfy some of security requirements as follows:

1. Double spending
2. Unforgeability
3. Anonymity
4. Recovery issue
5. Non-repudiation

In our scheme, we provide various added-value services for user. The user can use stored-value card to enjoy the service anywhere. Thus, we expect that our scheme can provide a convenient trading architecture for the added-value service in VANET.

References

[1] **C. L. Chen, M. H. Liu.** A traceable E-cash transfer system against blackmail via subliminal channel. *Electronic Commerce Research and Applications*, 2009, Vol. 8, No. 6, 327-333.

[2] **C. L. Chen, Y. Y. Chen, J. K. Jan.** A secure authentication scheme for a public terminal before a transaction. *Lecture Notes in Computer Science*, 2007, Vol. 4658, 118-126.

[3] **Y. Chen, J. S. Chou, H. M. Sun, M. H. Cho.** A novel electronic cash system with trustee-based anonymity revocation from pairing. *Electronic Commerce Research and Applications*, 2011, Vol. 10, No. 6, 673-682.

[4] **E. K. Clemons, D. C. Croson, B. W. Weber.** Reengineering money: the mondex stored value card and beyond. *International Journal of Electronic Commerce*, 1996, Vol. 1, No. 2, 5-31.

[5] **X. Dai, J. Grundy.** NetPay: An off-line, decentralized micro-payment system for thin-client applications. *Electronic Commerce Research and Applications*, 2007, Vol. 6, No. 1, 91-101.

[6] **W. Diffie, M. E. Hellman.** New directions in cryptography. *IEEE Transactions on Information Theory*, 1976, Vol. 22, No. 6, 644-654.

[7] **Easycard Corporation.** Available at: http://www.easycard.com.tw/english/easycard/index.asp.

[8] **Z. Eslami, M. Talebi.** A new untraceable off-line electronic cash system. *Electronic Commerce Research and Applications*, 2011, Vol. 10, No. 1, 59–66.

[9] **J. T. Isaac, S. Zeadally, J. S. Cámara.** A lightweight secure mobile payment protocol for vehicular ad-hoc networks (VANETs). *Electronic Commerce Research*, 2012, Vol. 12, No. 1, 97-123.

[10] **J. T. Isaac, S. Zeadally, J. S. Cámara.** Implementation and performance evaluation of a payment protocol for vehicular ad hoc networks. *Electronic Commerce Research*, 2011, Vol. 10, No. 2, 209-233.

[11] **W. S. Juang.** An efficient and practical recoverable pre-paid offline e-cash scheme using bilinear pairings. *Journal of Systems and Software*, 2010, Vol. 83, No. 4, 638-645.

[12] **S. Kremer, O. Markowitch, J. Zhou.** An intensive survey of fair non-repudiation protocols. *Computer Communications*, 2002, Vol. 17, No. 1, 1606-1621.

[13] **W. Li, Q. Wen, Q. Su, Z. Jin.** An efficient and secure mobile payment protocol for restricted connectivity scenarios in vehicular ad hoc network. *Computer Communications*, 2012, Vol. 35, No. 2, 188-195.

[14] **Micropayment – Wikipedia**. Available at: http://en.wikipedia.org/wiki/Micropayment.

[15] **M. Raya, J. P. Hubaux.** The security of vehicular ad hoc networks. In: *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks(SASN '05)*, 2005, pp. 11-21.

[16] **E. Rescorla.** Diffie-Hellman Key Agreement Method, Internet Engineering Task Force (IETF) (RFC 2631) Working Group, 1999.

[17] **P. G. Schierz, O. Schilke, B. W. Wirtz.** Understanding consumer acceptance of mobile payment services: An empirical analysis. *Electronic Commerce Research and Applications*, 2012, Vol. 9, No. 3, 209-216.

[18] **Value-added service – Wikipedia.** Available at: http://en.wikipedia.org/wiki/Value-added_service.

Received May 2013.