Journal of Computing Science and Engineering
Vol. 5, No. 1, March 2011. pp. 51-70

DOI: 10.5626/JCSE.2011.5.1.051
pISSN: 1976-4677 eISSN: 2093-8020

# A Secure Credit Card Transaction Method Based on Kerberos

Jung Eun Kim
Microsoft Corporation, Redmond, WA, USA, kicom95@live.com

Yoohwan Kim
University of Nevada, Las Vegas, Las Vegas, NV, USA, Yoohwan.Kim@unlv.edu

Received 28 October 2010; Revised 25 February 2011; Accepted 3 March 2011

This paper introduces a new credit card payment scheme called No Number Credit Card that can significantly reduce the possibility of credit card fraud. The proposed payment system is loosely based on Kerberos, a cryptographic framework that has stood the test of time. In No Number Credit Card, instead of card numbers, only payment tokens are exchanged between the customers and merchants. The tokens are generated based on the payment amount, payment type, client information, and merchant information. However, it does not contain the credit card number, so the merchant or a database hacker cannot acquire and illegally use any credit card numbers. The No Number Credit Card system is ideal for online e-commerce transactions and can be used with any credit card that users possess. It can be used with minor modifications to the current card payment system. We provide the principles of its operation through scenario analysis, a sample implementation, and a security analysis

Keywords: Network Security, Network Protocol, Computer Security, Cryptography, Authentication

## 1. INTRODUCTION

Electronic payment systems (EPSs) are an essential part of modern business. Credit cards or debit cards have been widely used for on-site or remote transactions, greatly reducing the need for inconvenient cash transactions. Furthermore, EPS has become a critical piece for the operation of e-commerce systems where cash transactions are impractical.

However, there have been numerous incidents of credit card fraud over the Internet due to the weaknesses of EPS. For example, on August 16, 2009, a computer criminal named Albert Gonzalez was accused of stealing 170 million credit and ATM card numbers and reselling them [Stone 2009]. In another incident reported by the Chronology of Data Breaches, a credit card fraud incident database, on April 22, 2009, a former employee at the New York state tax department was accused of gathering secret data including credit card numbers and using them illegally [Privacy Rights Clearinghouse 2011]. The losses from credit card fraud are also large and the number of cases is increasing. According to the FBI's Internet Crime Complaint Center's 2008 Annual

Report [Bresson and Butterworth 2009], the total loss from online fraud amounted to 265 million dollars in 2008, a 33.1% increase compared with 2007.

The situation is getting worse as more people shop on-line. Shoppers need to submit their credit card numbers to e-commerce sites, and in many cases the credit card numbers are stored in a merchant's database either on purpose or by negligence. This increases the chance of credit card fraud because the merchants' database may be cracked by a hacker or used illegally by the insiders of the merchant site. Thus, protecting the credit card numbers from the database or while they are in transit is very important to reduce loss. To this end, a number of solutions or protocols such as SSL [Freier et al 1996], SET [VISA International AND Mater Card International 1997], and PayPal [PayPal 2011] have been proposed. While some of the solutions are used currently, some are considered impractical. One of the challenging issues in developing a scheme is to satisfy both groups of users, namely the cardholders and the merchants. Cardholders do not want to give their card numbers directly to the merchants, while merchants want to get the card number to conveniently charge payments.

In this paper, we present a new secure payment system called No Number Credit Card (NNCC) that does not reveal the credit card number to the merchant while minimizing inconvenience. This system is based on the Kerberos framework [Neuman and Ts'o 1994] and only exchanges tokens between buyers and merchants. Tokens are generated based on the payment amount, the client information and merchant information, but the token does not contain the actual credit card number. So the merchant cannot acquire and illegally use the credit card number. A token itself is cryptographically secure and valid only for the designated merchant, so it is robust against eavesdropping. The NNCC scheme can be best used for on-line e-commerce transactions. NNCC is more practical than other Kerberos-based systems as it can work under the current card payment system with only minor changes.

## 2. RELATED WORK

Generally an EPS falls into one of the two categories: token-based systems (electronic cash systems, or electronic currency systems), and account-based systems (Credit-debit) [Abrazhevich 2001]. However, the credit card system is considered a separate category in some cases, so we further divide the EPS into three categories. Figure 1 shows the classification of the electronic payment systems to be covered in this section.

### 2.1 Electronic Cash System

In electronic cash systems, customers buy digital tokens and surrender them to the merchant when they buy an item [Abrazhevich 2001]. Electronic cash systems are further divided into two systems: smart card-based systems which use smart cards to store E-Cash, and Web Cash where a user's E-Cash is stored in a user's online account. The smart card-based system is not suitable for an Internet Payment System due to the need for physical contact to make a payment. Web Cash systems do not suffer from this problem, and there are several systems proposed, e.g., the Millicent Protocol [Glassman et al. 1995], PayWord and MicroMint [Rivest 1996], NetCash [Medvinsky and Neuman 1993], and eCash (or DigiCash) invented by David Chaum

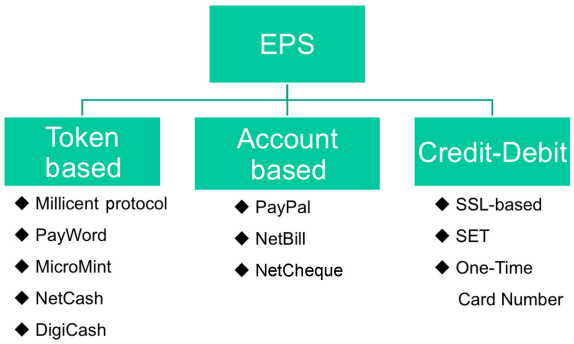


Figure 1. Electronic payment system (EPS) classification.

[Chaum 1983].

Millicent Protocol is designed to process the small amount of money that can be a fraction of cents for inexpensive Internet content. The most important parts of the Millicent Protocol are Broker and Scrip. Broker provides account management and billing, and Scrip is digital cash that is valid for a specific merchant [Glassman et al. 1995]. PayWord is credit-based. Customers need a digital certificate signed by a broker. A digital certificate consists of a customer's name, broker name, public key that is used for signature verification and some other things. PayWord shows its strong efficiency in multiple transactions to the same merchant [Rivest 1996]. In the MicroMint system, a coin is an essential factor. A Broker sells coins to a user, and customers give coins to merchants as payment. Merchants return tokens to the broker to get money. The validity of a token is easily checked, and it is almost impossible to forge [Rivest 1996]. NetCash provides anonymous transactions and real time payment processing under multiple server environments [Medvinsky and Neuman 1993]. DigiCash (or eCash) is based on the RSA blinding signature in which the content of a payment token is unknown to its signer. It provides public verification to its participants. Usually, this scheme is used where the owner and signer of a token or message are different [Chaum 1983].

The advantage of token-based systems is that anonymous transactions are possible in some systems. For example, in DigiCash, it is impossible to know to whom a specific token was issued because the content of a token is blind until it is signed by a bank [Chaum 1983]. Also, transaction processing is efficient because the exchange of tokens is performed locally without connecting to a remote transaction server [Abrazhevich 2001]. However, ECS still needs to maintain a large database of past transactions to prevent double spending from a single token. Furthermore, it is required for both customers and merchants to purchase and install hardware and software to deal with electronic tokens, which is a burden to the participants and prevents it from being widely used [Abrazhevich 2004].

### 2.2 Account Based Systems

In an account-based system, the exchange of money between user accounts is performed by a payment service provider [Abrazhevich 2001]. Examples of account-based systems

include PayPal, NetBill [Cox et al. 1995], and NetCheque [Neuman and Medvinsky 1995].

PayPal is a very popular service for web-based transactions. PayPal users can send or receive money using their email addresses. It is widely used in customer-to-customer (C2C) transactions but can be also used in business-to-customer (B2C) transactions [Sorkin 2001; González 2004]. PayPal does not reveal detailed account information of the transaction partner to other users. Thus its transactions assure some privacy. However, its authentication scheme is primitive and PayPal's ID can be easily hacked using the Internet in several ways. Also, the management company of PayPal is considered a risk factor because it could go bankrupt with user money [González 2004]. Also, PayPal requires a credit card number to deposit money to the PayPal account, thus it inherits the same problems as credit card payment models.

Net bill is designed for micro-payments, i.e., a small amount such as a fraction of a dollar, especially for information content delivered over the Internet. However, Net bill is currently a theoretical system and hasn't been deployed yet [Cox et al. 1995].

NetCheque is a distributed accounting service. Users of NetCheque have accounts on account servers. When a customer buys an item, they write an electronic document (a check) with an electronic signature, and then send it to the merchant [Neuman and Medvinsky 1995]. NetCheque is based on the Kerberos concept [Neuman and Ts'o 1994] and Electronic Check. It is the first attempt to apply the Kerberos concept to a payment system, which allows the system to maintain high security and reliability. However, NetCheque is not considered practical and hasn't been implemented in the industry.

### 2.3 Credit (or debit) Card Payment Model

The greatest difference between an account-based system and a credit card payment system is that customers do not need to make an account to use a credit card system, and credit card information is the only thing needed for authentication [Abrazhevich 2001]. This model is most widely used on the Internet due to the simplicity and convenience of customers. However, due to this simple authentication scheme, it results in numerous problems such as credit card fraud and counterfeiting. The purpose of our proposed system is fully utilizing the benefits of the credit card system, while removing this vulnerability by not using the credit card number directly. Therefore, it is necessary to understand how the credit card system currently operates.

2.3.1 *Card payment processing network.* As shown in Figure 2, several elements are involved with Card Payment Networks. A Payment Gateway provides the connectivity between Merchant (i.e., Payment site) and a Processor (i.e., Financial Networks). The Processor is a large data center which processes the credit card transactions. The following steps describe how the payment processing system works [VeriSign n.d.].

1) A customer submits a credit card number to the merchant.
2) Merchant sends the payment information such as the card number and the amount of money to a Payment Gateway.
3) The Payment Gateway passes the information from the merchant to the Processor.

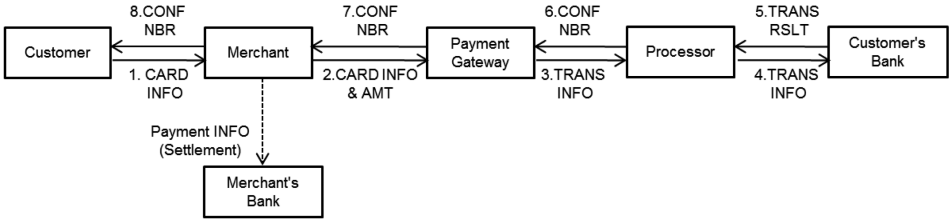


Figure 2. Current payment process under credit card payment model.

4) The Processor sends the information to the customer's bank.
5) The customer's bank sends the transaction result (approved or rejected) to the Processor.
6) The Processor passes the transaction result to the Payment Gateway.
7) The Payment Gateway sends the transaction information to the merchant.
8) The Merchant saves the transaction information for the settlement and sends the confirmation number to the customer.

In Step 8, the money is not transferred from the customer's bank to the merchant's bank at the moment of card processing. Instead, a settlement is delivered to the merchant's bank for later processing. Settlement is a merchant's electronic bookkeeping for the transaction of payment information.

2.3.2 *Protocols for secure transaction of credit cards.* During an e-commerce transaction, a credit card number is sent over the Internet at Step 1. To make it secure between each element, especially between the Customer (i.e., the card holder) and the Merchant, a number of methods have been suggested. To prevent eavesdropping during the transmission process, the transaction information is generally encrypted using secure socket layer (SSL). SSL (or https) is not a payment protocol, but a very popular web content encryption technology and virtually all credit card transactions are encrypted using SSL nowadays. It employs an asymmetric key algorithm for encryption, and allows the user to authenticate the identity of the merchant using a digital certificate [Freier et al. 1996]. However, it was recently shown that SSL can fail to protect the data against some attack tools such as sslstrip [Marlinspike 2009; Kerner 2009].

Secure electronic transaction (SET) is a security protocol for card payment over the Internet proposed by Visa, MasterCard, and other companies. It consists of five protocols (cardholder registration, merchant registration, purchase request, payment authorization, and payment capture) [Visa International 1997]. Unlike SSL, SET prevents merchants from using a customer's credit card number illegally because the cardholder shares order information with the merchant and shares the payment information only with banks (i.e., dual signature). However, SET failed to be implemented in the industry because its complexity became a burden to customers.

The concept of a one-time credit card number (a.k.a., disposable card number or single-use number) [Shamir 2002] is used by some credit card issuers such as American Express, Discover, MBNA, or Visa's Gift Cards [Kenny 2010]. A one-time credit card number is generated for each transaction for a single use. After using the single-use

number, additional and potentially illegal uses of the card number are rejected. In this way, customers do not need to worry about credit card number theft. However, whenever customers need a credit card transaction, they should have an online connection with the card issuer to have a new card number, which can be a burden to both customers and card issuers. To resolve this problem, Li and Zhang [2005] proposed a one-time payment scheme that generates card numbers with hash functions. However, users cannot use the one-time card number if the card issuer does not offer the one-time card number service.

Tokenization is a recent trend to replace the vulnerable card numbers with a random number. A number of card processors are offering these services such as TrueToken [Shift Corporation n.d.], TransactionVault [Merchantlink n.d.], or BuyerWall [Electronic Payment Exchange 2009]. These services replace the original card number with a randomized token without any user intervention. While they are useful and convenient for off-line card-present transactions, they fall short for on-line transactions because the users still have to submit the card numbers over the Internet. Although they provide a dedicated website independent from merchants for accepting card numbers, whenever users submit the card number over the Internet, there is still some possibility of a data breach.

The Chip and Pin system [Wikipedia n.d.] has been effective in preventing fraud in card-present transactions by using the PIN and smart card technology. But it cannot be used for on-line transactions.

## 3. OUR PROPOSED SYSTEM – NNCC

### 3.1 Background in Kerberos

Kerberos is an authentication protocol to verify a client's identity to allow logging on to a server and to encrypting their communications through secret-key cryptography over an insecure network [Neuman and Ts'o 1994]. It resolves the key distribution problems between client and server with Tickets. Clients can log on to a server in two steps. First, they get authenticated at the Authentication server (AS), and get a Ticket-Granting Ticket (TGT). The client presents the TGT to the Ticket-Granting Server

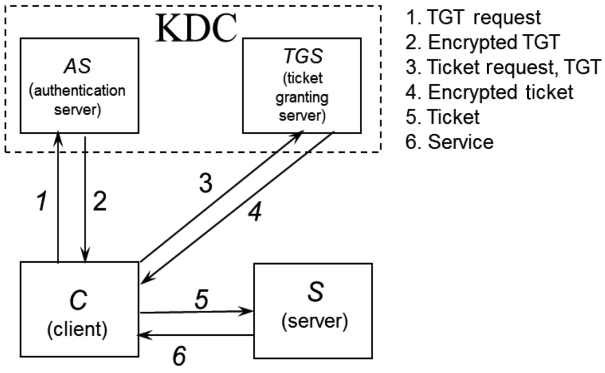


Figure 3. Kerberos system.

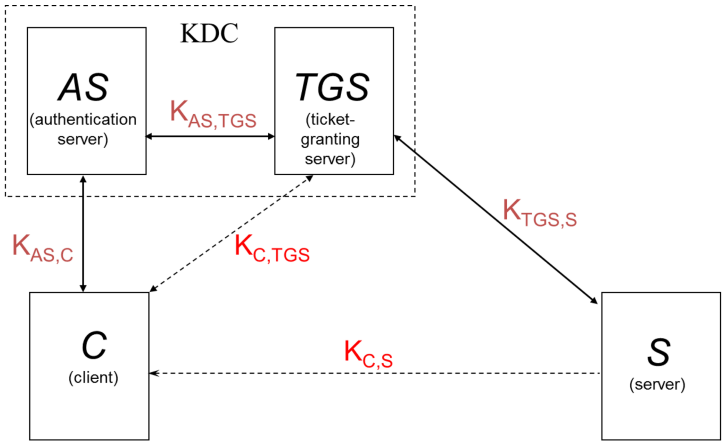


Figure 4. Keys in Kerberos.

(TGS) and gets a real Ticket for a specific server. With the Ticket, clients can log on to an application server. This concept is illustrated in Figure 3.

Using the Tickets, Kerberos never sends a password over the network, so the password is protected against eavesdropping or replay attacks. A ticket contains the authentication information of the ticket holder, and is encrypted with the key of the final recipient. Therefore a client holding a ticket has no knowledge of the ticket's content nor can they modify it. As it is encrypted, it can be safely sent across the networks. The types of keys are described in Figure 4. The shared keys, $K_{AS,TGS}$, $K_{AS,C}$, $K_{TGS,S}$ are predefined during the registration process, and $K_{C,TGS}$, $K_{C,S}$ are dynamically generated during authentication.

Our proposed No Number Credit Card (NNCC) system is loosely based on Kerberos. Since Kerberos is known to be secure and reliable [Neuman and Ts'o 1994], it has been also used in other payment protocols such as NetCheque [Neuman and Medvinsky 1995].

### 3.2 NNCC Architecture

The NNCC system architecture is shown in Figure 5. Figure 6 shows the same diagram with sample messages between the components.

At a glance, the NNCC system looks similar to Kerberos. The AS in Kerberos is replaced by Payer Authentication Server (PAS), Client by Customer, Ticket-Granting Server (TGS) by Payment Granting Server (PGS), and the application server by Merchant. However, it has a number of key differences. First, the operation sequence is quite different because NNCC has to accommodate the back-end process of credit card processing. Second, it has a new communication relationship between merchant and PGS. Third, it has a reverse operation (refund operation). And finally, the token contents are significantly different from the content of the Kerberos tickets.

Note that NNCC uses the term *Tokens* instead of *Tickets* to emphasize the aspect of financial values in the messages and also to avoid the confusion with Kerberos tickets. Two tokens are used in NNCC, i.e., Session Token and Payment Token. A Session Token is similar to a Kerberos TGT, and a Payment Token is similar to a

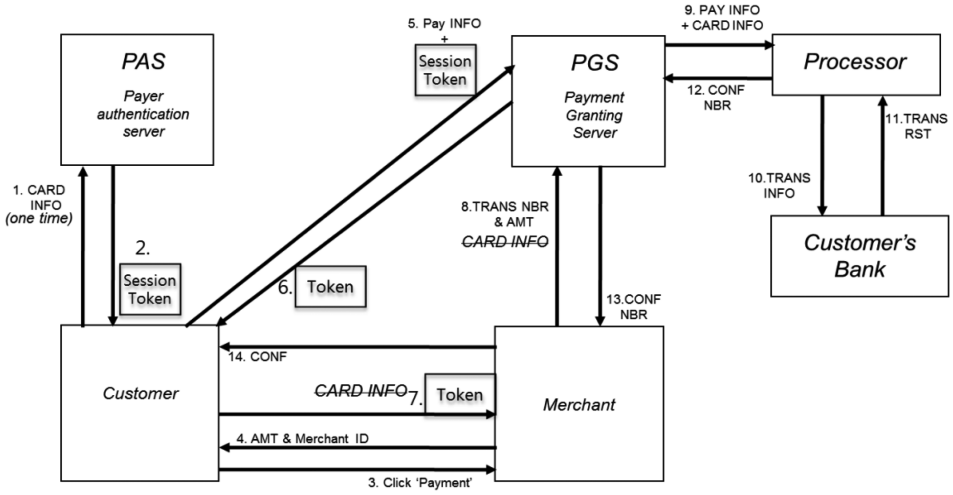


Figure 5. System architecture of No Number Credit Card (NNCC).

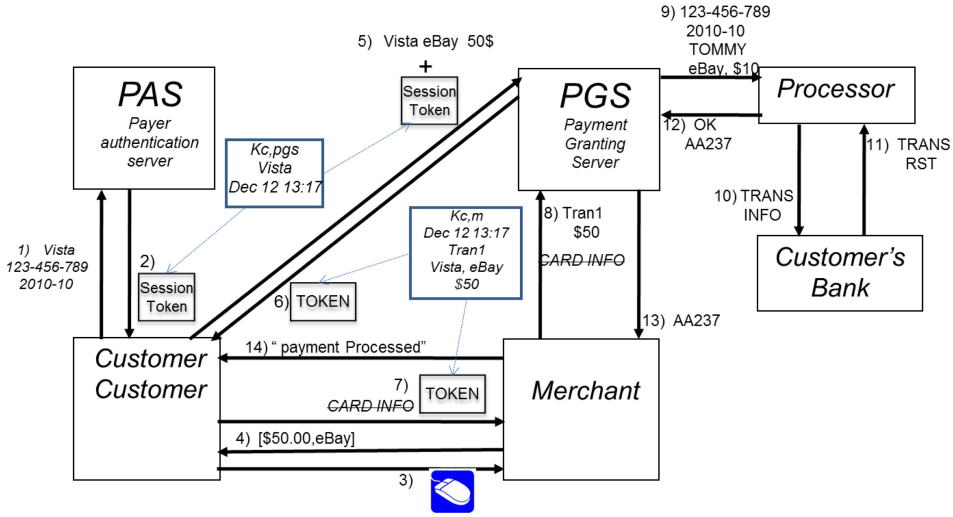


Figure 6. No Number Credit Card (NNCC) system with sample data.

Kerberos Ticket. A Payment Token contains the information of the client and the merchant, and the amount of money to be paid. A session token is long-lived, *e.g.*, hours, but the payment token is strictly for one-time use. If a client has a valid session token, it does not need to get a new session token. However, it has to request a new payment token for every payment process.

In the current card payment system shown in Figure 2, a customer is connected to a merchant and the merchant is subsequently connected to a Payment Gateway. In NNCC, PGS replaces the Payment Gateway, but unlike in the current payment system, it is connected to both customer and merchant. PGS acts as a clearinghouse for all

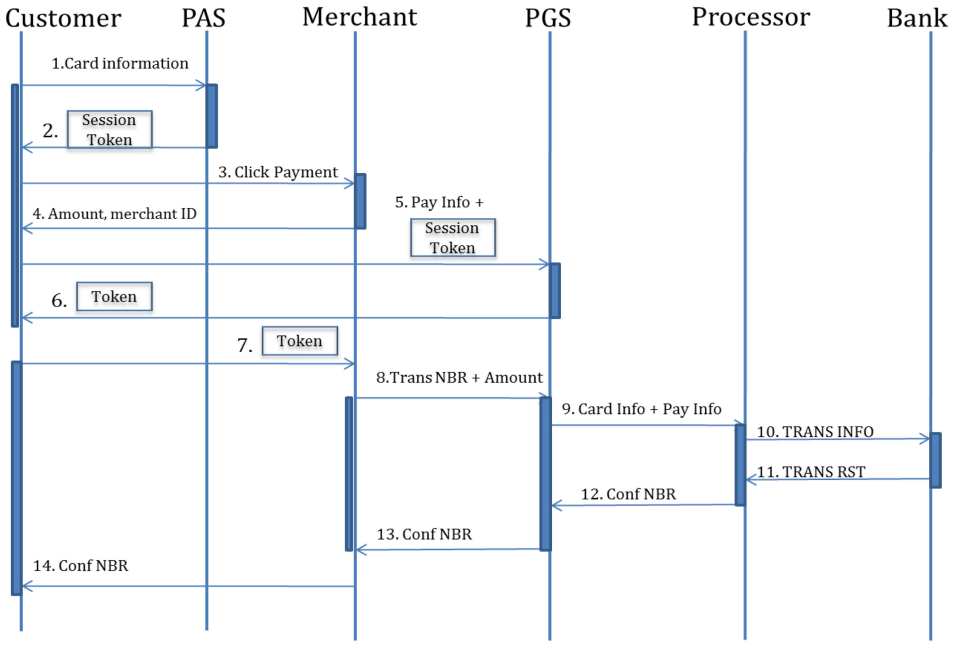


Figure 7. Sequence diagram of NNCC for payment.

credit card transactions. For each credit card transaction, it creates a token with certain monetary value and gives it to the client who requested it. However, it does not know the actual credit card number because it creates the token based on the session token created by PAS. Likewise the Merchant does not know any credit card number. Only the customer and PAS know the actual credit card number. Although it sounds complex, a similar concept is used in single-use credit cards where a user communicates with the card-issuing bank (far right line in Figure 7) to generate a new card number. In our system, a user communicates with the PGS instead of the card-issuing bank.

The operation scenario is described in Figure 7. First, a customer sends the credit card information to the PAS server, and gets a session token. Then the customer sends the payment information (the amount of money and merchant ID) along with the session token to the PGS, and gets a Payment Token from PGS. The customer then sends the Payment Token to the merchant. Up to this point, it is similar to Kerberos. However, to accept the actual payment, a merchant sends the transaction number included in the Payment Token to PGS. Payment token is also used for merchant's settlement processing afterwards. After PGS receives the transaction number, it proceeds with the actual payment processing in the same way as in the current system.

### 3.3 NNCC Operating Scenarios

3.3.1 *Payment.*

1) User enters ID, card Information, and Password (A shared key between Client

and PAS is derived from this password. Normal precautions are required to avoid password attacks) in a terminal (e.g., home computer).

2) The Client sends card registration data (ID, card information) after encrypting them with the shared key. (From, To, fields are for time interval of the valid period.) If the credit card is already registered, the card number doesn't need to be sent again. Figure 8-A shows the message.
3) PAS checks if the credit card is registered and if so, generate a session token and sends it to the client. The message is described in Figure 8-B.
4) The Customer visits an e-commerce website, and decides to buy an item.
5) The Merchant presents the amount and Merchant ID to the Customer on its web site.
6) A program running on the Client's computer sends the Payment Information (Merchant ID and amount of money) encrypted with the session key along with


Figure 8-A. Data flow from Client to PAS in NNCC.

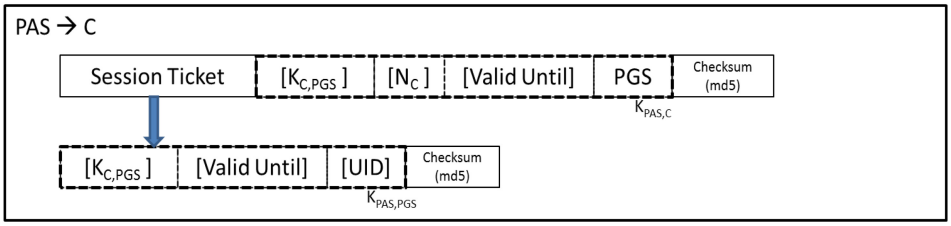


Figure 8-B. Data flow from PAS to Client in NNCC.

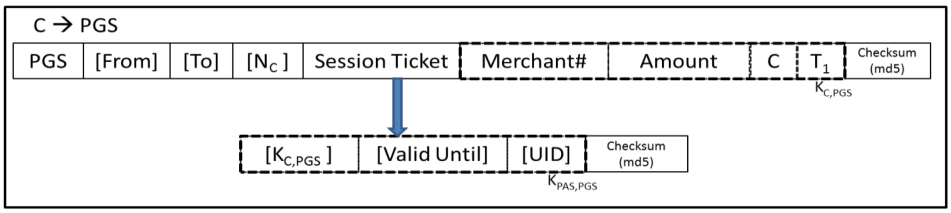


Figure 8-C. Data flow from Client to PGS in NNCC.

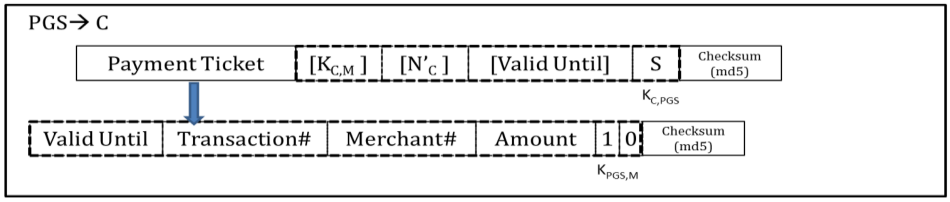


Figure 8-D. Data flow from PGS to Client in NNCC.

the session token to PGS. This message is described in Figure 8-C.

7) PGS checks if the session token is valid. If so, it
   i) generates the transaction information (amount, customer, merchant) on the database.
   ii) generates a shared key between merchant and client ($K_{C,M}$), and Payment Token.
   iii) sends the key and a Payment Token to the Client.
   Figure 8-D describes the message.
8) The Client sends the Payment Token to the merchant as described in Figure 8-E.
9) Merchant extracts the Transaction number from the Payment Token and sends it with the amount to PGS server. This message is described in Figure 8-F.
10) After the PGS server gets the Transaction number from the merchant, the actual payment process starts in a similar way to the current system.
   i) From the Transaction number, the PGS server retrieves the Payment Information (card #, amount, merchant ID) from its database.
   ii) PGS sends the Information to the Processor.
11) Processor request services to the Client's bank.
12) The Client's Bank responds to the Processor (Accept or Reject).
13) PGS receives a confirmation number from the Processor.
14) PGS sends a confirmation number encrypted by the shared key to the merchant.

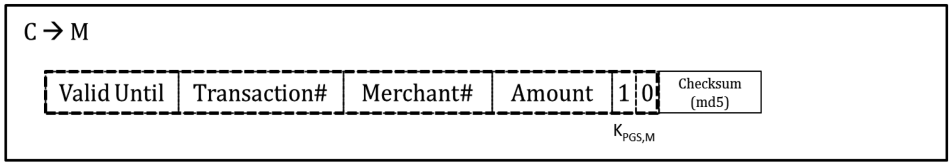


Figure 8-E. Data flow from Client to Merchant in NNCC.

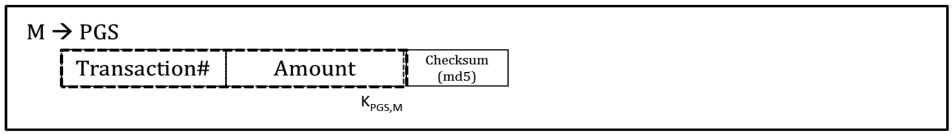


Figure 8-F. Data flow from Merchant to PGS in NNCC.


Figure 8-G. Data flow from PGS to Merchant in NNCC.

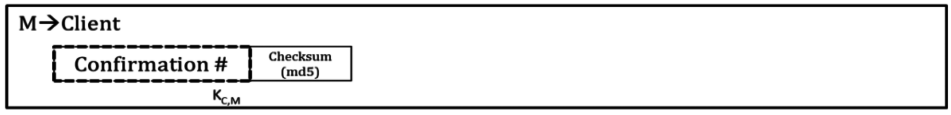


Figure 8-H. Data flow from Merchant to Client in NNCC.

Figure 8-G describes this message.

15) The merchant sends the confirmation number encrypted with the shared key to the client. This message is described in Figure 8-H. For added security, the communication between the client and the merchant web site may be further encrypted by SSL. (We actually employ SSL in our demonstration system described in the next section.) However, even if the site does not use SSL, the communication is still secure, at least for the Token and the confirmation number, thanks to this encryption.

3.3.2 *Refund scenario.* Refund is just a reverse process of payment. We assume that the customer has a valid session token. The customer can get a session token without a credit card number because the refund process does not require the card information, but needs only the transaction reference number. Figure 9 describes the refund procedure.

0) Customer claims a refund.
1) Merchant requests a Refund Token to PGS. (Figure 10-A)
2) After getting the request, PGS sends the refund token and data to the Merchant. (Figure 10-B)
3) Merchant sends the refund token to the Client. (Figure 10-C)

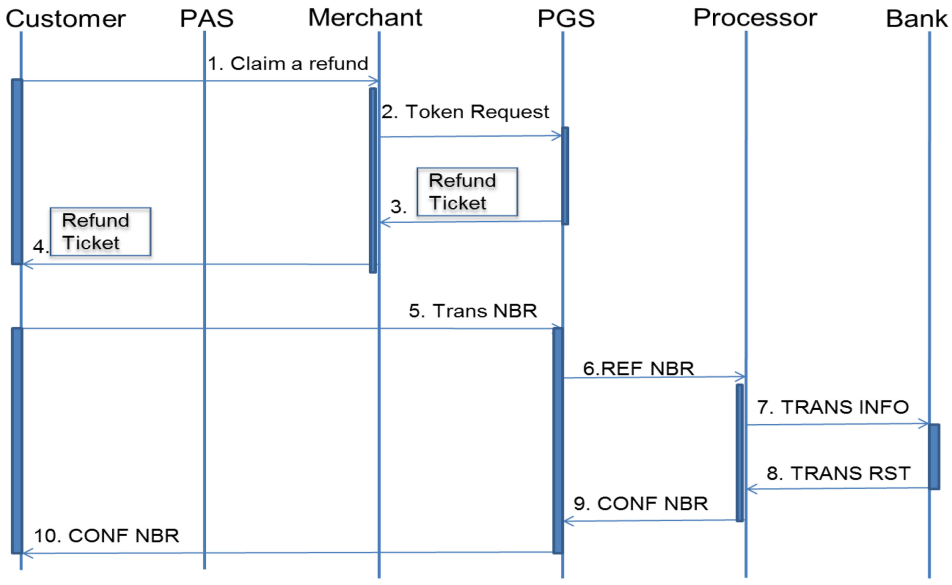


Figure 9. Sequence diagram of NNCC for REFUND.

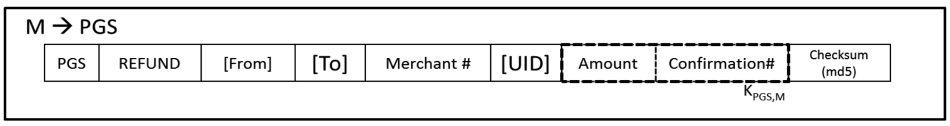


Figure 10-A. Data flow from Merchant to PGS in NNCC for Refund.

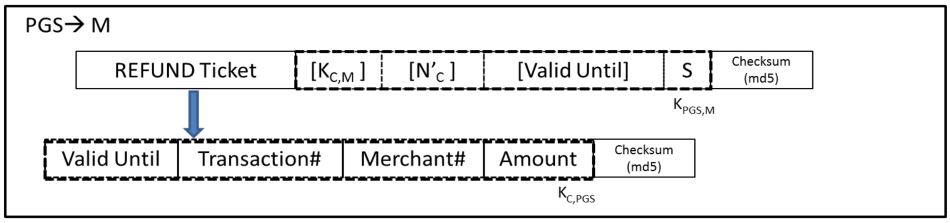


Figure 10-B. Data flow from PGS to Merchant in NNCC for Refund.

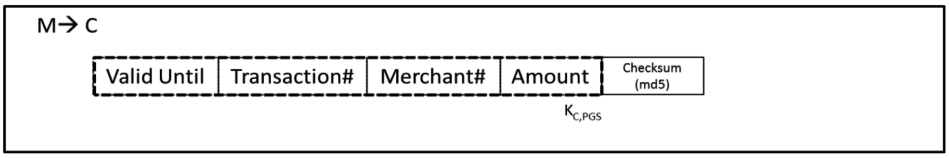


Figure 10-C. Data flow from Merchant to Client in NNCC for Refund.

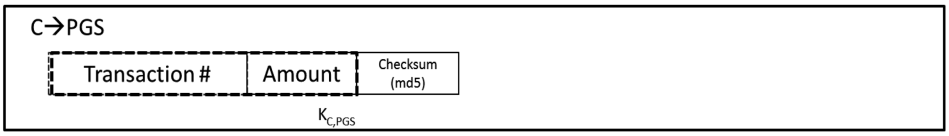


Figure 10-D. Data flow from Client to PGS in NNCC for Refund.


Figure 10-E. Data flow from PGS to Client in NNCC for Refund.

4) Client sends the transaction number and the amount to the PGS after getting the refund token. (Figure 10-D)
5) After the PGS server gets the transaction number from the client, refund process starts.
   i) From the Transaction number, PGS server retrieves Payment Information (Transaction reference number, amount, merchant ID, UID) from its database.
   ii) PGS sends the information to the Processor.
   After several steps, PGS receives a confirmation number from the Processor.
6) PGS sends the confirmation number to the client. (Figure 10-E)

### 3.4 Advantages of NNCC System

NNCC inherits most of the advantages of Kerberos. User does not need to send the credit card information to the each merchant and merchant does not need to have users' card information. This prevents merchant or its employees from stealing customers' card information.

The only place where users' credit card numbers are stored is the PAS. Credit card

numbers are encrypted with the shared key between the user and the PAS, and then sent to PAS. PAS offers anadditional layer of security through a session token. Even if the card number is stolen by an attacker, the attacker cannot get a session Token from PAS without the user's password which is pre-registered through the trusted path. Also, mutual authentication is implemented among the payment systems, merchant, and customer because each participant and its communicating partner need to know their shared key to communicate with each other. So, the merchant can verify whether or not the customer is the legitimate owner of the token. This is important because the merchant is entirely responsible for the loss from card number theft in some countries. Furthermore, the merchant cannot modify payment information under this system.

The migration from the current payment system to NNCC is reasonably straightforward. Customers only need to download a small software program, and merchants just need to send a transaction number in the Payment Token from the customer, instead of the card information and the amount to the Payment Gateway (i.e., Payment Granting Server). As explained earlier, tokenization is widely used already, and adding this capability to the payment processing system is not impractical. This aspect of simplicity is important in practice because several secure payment system previously proposed haven't been implemented in the industry due to their complexity or the burden to the customers and merchants.

## 4. IMPLEMENTATION OF NNCC

A demonstration system of the proposed concept was implemented using C. We use Crypt library [Mcrypt] for the encryption and open SSL library [OpenSSL Project] for the data transfer. To show all the payment processes during the payment transactions, we install an open source shopping mall package [OSCMall] written in PHP and modify it to link our payment system. PAS and PGS modules work under both UNIX and Windows systems. In this section, we describe briefly how we implemented the demonstration system and how it worked.

### 4.1 System Organization

Figure 11 shows the high-level design of the NNCC demonstration system. To enhance security, a few additional techniques were employed. All participants in NNCC transported the data over the SSL protocol for stronger encryption. The client program always checks the credentials of PAS and PGS.

This demonstration system itself had a few security vulnerabilities that should be addressed in a production system.

- Availability: Like the AS and TGS in the original Kerberos system, the PAS and PGS are vulnerable to a DoS attack. If the PAS or PGS becomes unavailable, the entire system becomes disabled. To avoid a single-point-of-failure and maintain high availability, PAS and PGS should be made redundant in the hardware layer. Further multiple distributed PAS and PGS may be used.
- Brute-force attack: As in all password-protected systems, a weak password can allow attackers to gain access to private information. Strong passwords along with account locking mechanisms should be used to avoid brute-force attacks.

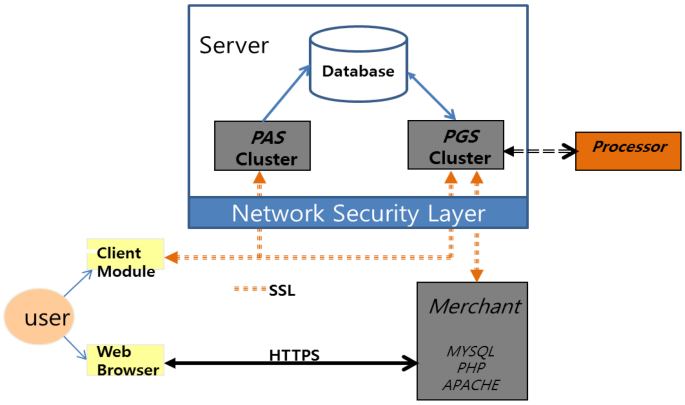


Figure 11. System organization of NNCC.

- Scalability: To cover diverse geographical regions and reasonable response time, PAS and PGS should be distributed. Similar schemes to Kerberos, such as Kerberos realm and realm keys, may be used.

### 4.2 Demonstration

In this section, we show how our demonstration system works.

4.2.1 *Session token generation.* As shown in Figure 12, a user enters the card information, ID, and Key to get a session token. She does not need to get a new session token as long as she has a valid one.

4.2.2 *Payment token generation.* A user can get a payment token as shown in Figure 13 if she has a valid session token. To acquire a payment token, she needs to give the Merchant ID and the amount of money. After getting a payment token, it is

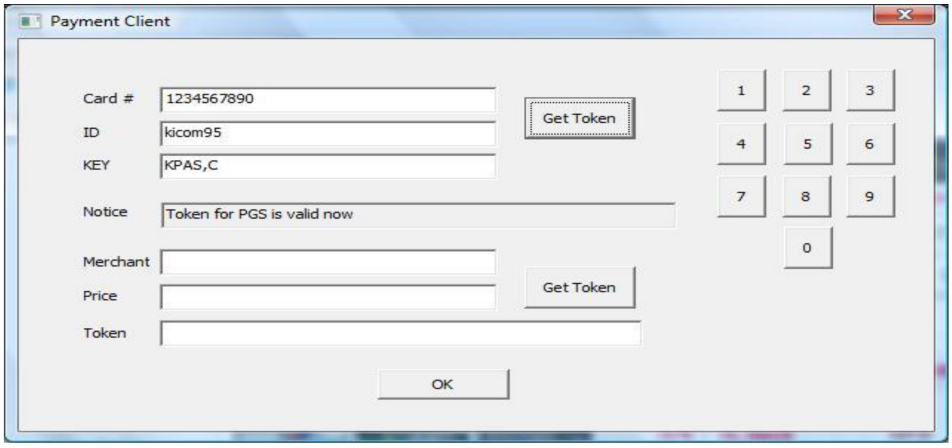


Figure 12. Session token generation.

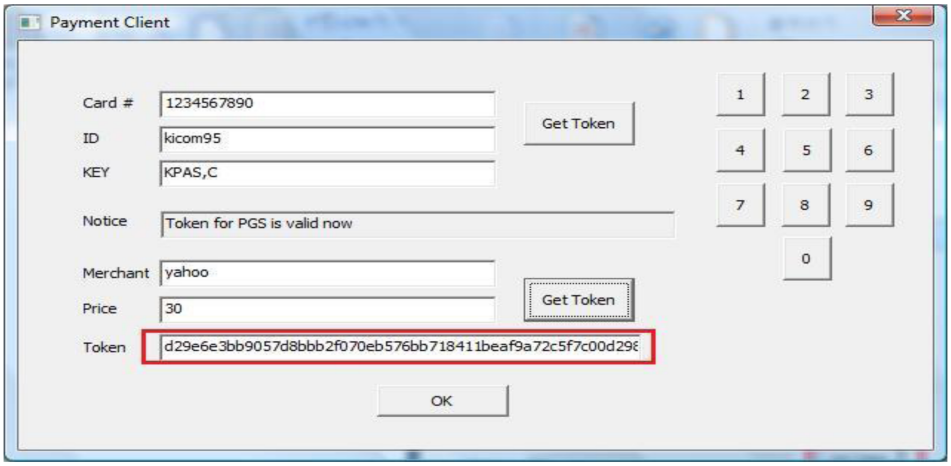


Figure 13. Payment token generation.

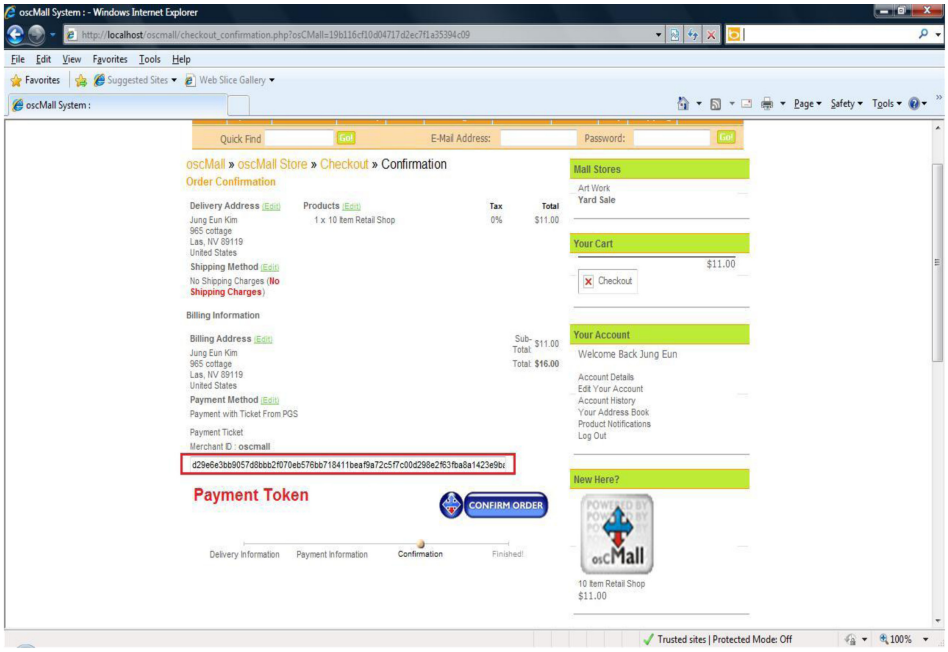


Figure 14. E-commerce site using NNCC.

automatically saved in the clipboard.

4.2.3 *Payment process.* Figure 14 shows a payment page of the mockup E-commerce site using NNCC. Instead of entering the card information, the user enters the payment token number on this page. Entering the payment token into the edit box is easily accomplished by pasting the token from the clipboard.

## 5. SECURITY ANALYSIS

The security of NNCC is inherited from the security of Kerberos. We describe just a few additional concepts beyond the basic Kerberos principles.

### 5.1 Attack by a 3rd Party

- Brute-force password-guessing attack: Kerberos servers are vulnerable to offline password guessing attacks. As a derivative of Kerberos, the PAS and PGS are vulnerable to the same attack. However, there are well-known techniques against password-guessing attacks such as account lockout schemes after a limited number of trials. Therefore, NNCC does not offer additional methods against this attack.
- Packet sniffing attack: Although the ticket is securely encrypted, we encrypt it again with SSL for defense-in-depth. Thanks to this double encryption, it is almost impossible to crack the communication in NNCC. In other words, packet sniffing is practically impossible against our system while the original Kerberos Tickets are vulnerable to offline password guessing attacks.
- Replay attack: A Replay attack in Kerberos is highly difficult because of the time stamp in the ticket. However, if two systems' clocks are not well aligned, a replay attack could be possible. Besides, according to Kasslin and Tikkanen's research, a Replay attack exists against SMB and Kerberos 5 on a Windows domain [Kasslin and Tikkanen 2003]. In NNCC, a payment token may be subject to a replay attack between the customer and the merchant. However, a replay attack of the payment token is ignored because the transaction number on the payment token is disposable once a token is accepted.
- Database stealing: Merchants' databases do not have any credit card information, so the damage from an attack on the merchant database is limited.

### 5.2 Fraud by Customer

A customer may launch a password-guessing attack to find out the key $\mathbf{K}_{\mathbf{PGS,M}}$ and try to change the amount in the token. In this case, a customer requests a payment token with less money than the merchant asked for, and then sends it to the merchant after changing the amount in the token to the amount the merchant asked. This attack is prevented in NNCC. When the merchant sends the transaction number with the amount in the token to the PGS, there will be a mismatch in the amount, and the merchant will know that the payment token is modified.

### 5.3 Fraud by Merchant

It is impossible to steal a credit card number using NNCC because the merchant does not have the customer's credit card number.

A merchant may try to change the amount in the token, but the payment amount information is registered in the PGS database by the user and the merchant merely verifies it when it receives the token. Changing the amount in the ticket at the merchant site has no effect on the PGS database.

## 6. CONCLUSIONS AND FUTURE WORK

It is conceivable that if we do not provide credit card numbers to the merchants

during the electronic payment process, we can greatly reduce the chance of credit card fraud. To achieve this goal, we implemented a new payment system (called NNCC) based on the Kerberos framework which has been proven to be secure. In the proposed system, payment tokens are passed into the merchant instead of card information. So the merchant or a database hacker cannot acquire and illegally use the credit card number. Besides, a token is cryptographically secure and valid only for the designated merchant, so it is robust against eavesdropping. Our approach can be applied to current card payment system with minor modifications on the current payment workflow by sending a token to the Payment Gateway instead of credit card information in the merchant side. From this, we can conclude that NNCC is a secure and reliable credit card payment system which can be easily implemented in the current Electronic Payment System. NNCC does introduce a higher complexity than the normal credit card processing flow, but it is still considered less complex than other comparable systems such as SET.

In this paper, we showed how the proposed system can fit in the current card payment environment. However, for wide adoption, we need to implement NNCC in a distributed environment in order to process millions of transactions in a second and to provide a more convenient user interface to the customers. Furthermore, to make NNCC usable in an off-line environment, a mobile phone application with an Internet connection should be developed.

## REFERENCES

ABRAZHEVICH, D. 2001.Classification and characteristics of electronic payment systems. In *Electronic Commerce and Web Technologies (Lecture Notes in Computer Science 2115)*, 81-90.

ABRAZHEVICH, D. 2004. *Electronic Payment Systems: A User-Centered Perspective and Interaction Design*. Technische Universiteit Eindhoven, Eindhoven.

BRESSON, P. AND BUTTERWORTH, C. 2009. IC3 2008 Annual report on internet crime released. http://www.ic3.gov/media/2009/090331.aspx. Acccessed on Feb 20, 2011.

CHAUM, D. 1983. Blind signatures for untraceable payments. In *Advances in Cryptology, Proceedings of Crypto 82*. 199-203.

COX, B., TYGAR, J. D., AND SIRBU, M. 1995. NetBill security and transaction protocol. In *Proceedings of the1st USENIX Workshop in Electronic Commerce*, 77-88.

ELECTRONIC PAYMENT EXCHANGE. 2009. Buyer Wall tokenization and encryption for payment processing and PCI compliance. http://www.epx.com/solutions/buyerwall.htm. Acccessed on Feb 22, 2011.

FREIER, A. O., KARLTON, P., AND KOCHER, P. C. 1996. The SSL protocol version 3.0. http://tools.ietf.org/html/draft-ietf-tls-ssl-version3-00. Acccessed on Feb 22, 2011.

GLASSMAN, S., MANASSE, M., ABADI, M., GAUTHIER, P., AND SOBALVARRO, P. 1995. The Millicent protocol for inexpensive electronic commerce. In *Proceedings of the 4th International World Wide Web Conference*, 603-618.

GONZÁLEZ, A. G. 2004. PayPal: the legal status of C2C payment systems. *Computer Law and Security Report 20*, 4, 293-299.

KASSLIN, K. AND TIKKANEN, A. 2003. Replay attack on Kerberos V and SMB. http://users.tkk.fi/autikkan/kerberos/docs/phase1/pdf/LATEST_replay_attack.pdf. Acccessed on Feb 22, 2011.

KENNY, P. 2010. How to use a one time credit card. http://www.streetdirectory.com/travel_guide/149328/credit_cards/how_to_use_a_one_time_credit_card.html. Acccessed on Feb 20, 2011.

KERNER, S. M. 2009. Black Hat: hacking SSL with sslstrip. http://blog.internetnews.com/skerner/

2009/02/black-hat-hacking-ssl-with-ssl.html. Acccessed on Feb 20, 2011.

LI, Y. AND ZHANG, X. 2005. Securing credit card transactions with one-time payment scheme. *Electronic Commerce Research and Applications 4*, 4, 413-426.

MARLINSPIKE, M. 2009. SSLStrip. http://www.thoughtcrime.org/software/sslstrip. Acccessed on Feb 20, 2011.

MCrypt. http://mcrypt.sourceforge.net. Acccessed on Feb 22, 2011.

MEDVINSKY, G. AND NEUMAN, B. C. 1993. NetCash: a design for practical electronic currency on the internet. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, 102-106.

MERCHANT LINK. Transaction vault--Remove the data, remove the risk. http://www.merchantlink.com/portal/community/merchant_link/v2.0/restaurant/transactionvault. Acccessed on Feb 22, 2011.

NEUMAN, B. C. AND MEDVINSKY, G. 1995. Requirements for network payment: the NetCheque perspective. In *IEEE Proceedings of Compcon '95 Technologies for the Information Superhighway, Digest of Papers*, 32-36.

NEUMAN, B. C. AND TS'O, T. 1994. Kerberos. an authentication service for computer networks. *IEEE Communications Magazine 32*, 9, 33-38.

OpenSSL project. http://www.openssl.org. Acccessed on Feb 22, 2011.

The OSCMall™ development site. http://www.oscdevshed.com/. Acccessed on Feb 22, 2011.

PayPal. http://www.paypal.com. acccessed on Feb 22, 2011.

PRIVACY RIGHTS CLEARINGHOUSE. Chronology of data breaches. http://www.privacyrights.org/ar/ChronDataBreaches.htm#2009. Acccessed on Feb 20, 2011.

RIVEST, R. L. AND SHAMIR, A. 1996. PayWord and MicroMint: two simple micropayment schemes. *CryptoBytes 2*, 1, 7-11.

SHAMIR, A. 2002. Secureclick: a web payment system with disposable credit card numbers. In *Financial Cryptography (Lecture Notes in Computer Science 2339)*, P. SYVERSON, Ed. Springer Berlin, Heidelberg, 232-242.

SHIFT4 CORPORATION. True Security, TrueTokenizationTM. http://www.shift4.com/tokenization.cfm. Acccessed on Feb 22, 2011.

SORKIN, D. E. 2001. Payment methods for consumer-to-consumer online transactions. *Akron Law Review 35*, 1, 1-30.

STONE, B. 2009. Hacking suspect's lawyer criticizes federal prosecutors (New York Times August 19, 2009). http://bits.blogs.nytimes.com/2009/08/19/accused-hackers-lawyer-criticizes-federal-prosecutors/. Acccessed on Feb 20, 2011.

VERSIGN INC. Online payment processing: what you and your customers need to know. https://www.verisign.com/stellent/groups/public/documents/white_paper/001879.pdf. acccessed on Feb 20, 2011.

VISA INTERNATIONAL AND MASTER CARD INTERNATIONAL. 1997. SET secure electronic transaction specification book 3: Formal protocol definition. http://www.cl.cam.ac.uk/research/security/resources/SET. Accessed on Feb 20, 2011.

WIKIPEDIA. Chip and PIN (smartcard payment system). http://en.wikipedia.org/wiki/Chip_and_PIN. acccessed on Feb 22, 2011.

**Jung Eun Kim** received his M.S degree in Computer Science from the University of Nevada Las Vegas, USA in 2010, and a B.S degree in Computer Engineering from Kyungil University, South Korea in 1999. He worked at Welcome Information System in Korea for 10 years as a Senior Software Engineer. Currently he is a Software Development Engineer in Test at Microsoft Corporation.

**Yoohwan Kim** is an Associate Professor of Computer Science at University of Nevada, Las Vegas. He received his MS and Ph.D. degrees in Computer Engineering from Case Western Reserve University in 1994 and 2004 respectively, and his Bachelor degree in Economics from Seoul National University, Korea in 1989. Between 1997 and 1999 he was a Member of Technical Staff at Lucent Technologies, Whippany, NJ, developing wireless telecommunication software. He then co-founded and managed a start-up company developing technologies for delivering video advertising over the Internet. His research interests include network security, Internet traffic analysis, and software architecture, where he has published over 50 papers.