

Spring 2010

## A Secure on-line credit card transaction method based on Kerberos Authentication protocol

Jung Eun Kim  
University of Nevada Las Vegas

Follow this and additional works at: <https://digitalscholarship.unlv.edu/thesesdissertations>



Part of the [Computer Sciences Commons](#), [Digital Communications and Networking Commons](#), and the [E-Commerce Commons](#)

---

### Repository Citation

Kim, Jung Eun, "A Secure on-line credit card transaction method based on Kerberos Authentication protocol" (2010). *UNLV Theses, Dissertations, Professional Papers, and Capstones*. 7.  
<https://digitalscholarship.unlv.edu/thesesdissertations/7>

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Theses, Dissertations, Professional Papers, and Capstones by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact [digitalscholarship@unlv.edu](mailto:digitalscholarship@unlv.edu).

A SECURE ON-LINE CREDIT CARD TRANSACTION METHOD  
BASED ON KERBEROS AUTHENTICATION PROTOCOL

by

Jung Eun Kim

Bachelor of Engineering, Computer Engineering  
Kyungil University, South Korea  
1999

A thesis submitted in partial fulfillment  
of the requirements for the

**Master of Science Degree in Computer Science**  
**School of Computer Science**  
**Howard R. Hughes College of Engineering**

**Graduate College**  
**University of Nevada, Las Vegas**  
**May 2010**



THE GRADUATE COLLEGE

We recommend the thesis prepared under our supervision by

**Jung Eun Kim**

entitled

**A Secure On-line Credit Card Transaction Method Based on Kerberos Authentication Protocol**

be accepted in partial fulfillment of the requirements for the degree of

**Master of Science in Computer Science**

Yoohwan Kim, Committee Chair

Ajoy K. Datta, Committee Member

Laximi Gewali, Committee Member

Ju-Yeon Jo, Graduate Faculty Representative

Ronald Smith, Ph. D., Vice President for Research and Graduate Studies  
and Dean of the Graduate College

**May 2010**

## ABSTRACT

### **A Secure On-line Credit Card Transaction Method Based On Kerberos Authentication Protocol**

By

Jung Eun Kim

Dr. Yoohwan Kim, Examination Committee Chair  
Assistant Professor of Computer Science  
University of Nevada, Las Vegas

Nowadays, electronic payment system is 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. However, there have been a huge number of incidents of credit card frauds over the Internet due to the security weakness of electronic payment system. A number of solutions have been proposed in the past to prevent this problem, but most of them were inconvenient and did not satisfy the needs of cardholders and merchants at the same time.

In this thesis, we present a new secure card payment system called NNCC (No Number Credit Card) that significantly reduces the possibility of credit card frauds. This scheme is primarily designed for on-line shopping. NNCC is based on the Kerberos cryptographic framework that has been proven to be secure after being used in real world for decades. In this proposed system, instead of card numbers, only the payment tokens are exchanged between the buyers and merchants. The token is generated based on the payment amount, the client information, and merchant information. However it does not contain the credit card number, so the merchant cannot acquire and illegally use the credit card number. A token is cryptographically secure and valid only for the designated merchant, so it is robust against eavesdropping.

This thesis describes the underlying cryptographic schemes, the operating principles, and the system design. It explains the concept of Kerberos and the background in Cryptography. Then it discusses the new proposed system and the associated payment processes. We have implemented a proof-of-concept prototype comprised of ecommerce web sites, client modules, payment server, and database. We show the architecture and protocol of the system, and discuss the performance.

## TABLE OF CONTENTS

ABSTRACT .....	iii
LIST OF TABLES .....	vii
LIST OF TABLES .....	viii
ACKNOWLEDGEMENTS .....	ix
CHAPTER 1 INTRODUCTION .....	1
CHAPTER 2 RELATED WORK .....	4
2.1 Electronic Cash System (ECS) .....	4
2.2 Account Based Systems .....	6
2.3 Credit (or debit) Card Payment Model .....	7
2.3.1 Card Payment Processing Network .....	7
2.3.2 Protocols for Secure Transaction of Credit Cards .....	8
CHAPTER 3 TECHNICAL BACKGROUNDS .....	11
3.1 Cryptography and Mcrypt .....	11
3.2 Kerberos .....	12
3.2.1 Participants of Kerberos .....	13
3.2.2 Keys in Kerberos .....	14
3.2.3 Flow of Kerberos .....	15
3.3 SSL (Secure Socket Layer) and Open SSL .....	17
CHAPTER 4 SECURE ON-LINE CREDIT CARD TRANSACTION METHOD .....	18
4.1 Advantage of the System .....	21
4.2 Scenario .....	22
4.2.1 Payment .....	22
4.2.2 Refund .....	26
4.2.3 Installment (Periodic Payment) .....	28
CHAPTER 5 IMPLEMENTATION .....	30
5.1 System Organization .....	30
5.2 Data Transfer modules .....	31
5.2.1 Randomization of Cryptography Algorithm .....	31
5.2.2 Data Transfer with SSL .....	31
5.3 Token in real system .....	32
5.3.1 Session Token .....	32
5.3.2 (Payment) Token .....	33
5.4 Keyboard Interface .....	33
5.5 Database Structure .....	34
5.5.1 User Table .....	34
5.5.2 Card Information Table .....	35

5.5.3 Transaction Table.....	36
5.6 Demonstration.....	38
5.6.1 Session Token Generation .....	38
5.6.2 Payment Token Generation.....	38
5.6.3 Payment Process .....	39
CHAPTER 6 PERFORMANCE EVALUATION.....	41
6.1 Security Analysis .....	41
6.1.1 Attack by Hacker .....	41
6.1.2 Fraud by Customer.....	42
6.1.3 Fraud by Merchant.....	42
6.2 Communication Cost Comparison.....	42
6.2.1 Client.....	43
6.2.2 Merchant .....	43
6.2.3 Payment Gateway (PGS) .....	44
6.3 Client (User) Input Cost Comparison .....	44
6.4 Processing Cost.....	46
CHAPTER 7 CONCLUSION AND FUTURE WORK.....	48
7.1 Conclusion .....	48
7.2 Future Work .....	48
REFERENCES .....	49
VITA.....	52

## LIST OF TABLES

Table 1 Session Ticket .....	32
Table 2 (Payment) Ticket.....	33
Table 3 User Table.....	34
Table 4 Card Information Table .....	35
Table 5 Transaction Table .....	36
Table 6 Example of Transaction Table.....	37
Table 7 Client’s Communication Cost in NNCC .....	43
Table 8 Client’s Communication Cost in Current System .....	43
Table 9 Merchant’s Communication Cost in NNCC.....	43
Table 10 Merchant’s Communication Cost in Current System .....	44
Table 11 PGS’s Communication Cost in NNCC.....	44
Table 12 Payment Gateway’s Communication Cost in Current System .....	44



## LIST OF FIGURES

Figure 1	Current Payment Process under Credit Card Payment Model .....	7
Figure 2	Kerberos System.....	13
Figure 3	Keys in Kerberos System .....	14
Figure 4-A	Data Flow from Client to AS in Kerberos.....	15
Figure 4-B	Data Flow from AS to Client in Kerberos.....	15
Figure 4-C	Data Flow from Client to TGS in Kerberos .....	16
Figure 4-D	Data Flow from TGS to Client in Kerberos .....	16
Figure 4-E	Data Flow from Client to Server in Kerberos.....	16
Figure 4-F	Data Flow from Server to Client in Kerberos .....	17
Figure 5-A	System flow of NNCC .....	18
Figure 5-B	System Flow of NNCC with Actual Data .....	14
Figure 6	Keys in NNCC.....	20
Figure 7	Sequence Diagram of NNCC for Payment.....	20
Figure 8-A	Data Flow from Client to PAS in NNCC.....	22
Figure 8-B	Data Flow from PAS to Client in NNCC .....	23
Figure 8-C	Data Flow from Client to PGS in NNCC .....	23
Figure 8-D	Data Flow from PGS to Client in NNCC.....	24
Figure 8-E	Data Flow from Client to Merchant in NNCC .....	24
Figure 8-F	Data Flow from Merchant to PGS in NNCC .....	24
Figure 8-G	Data Flow from PGS to Merchant in NNCC .....	25
Figure 8-H	Data Flow from Merchant to Client in NNCC.....	25
Figure 9	Sequence Diagram of NNCC for Refund.....	26
Figure 10-A	Data Flow from Merchant to PGS in NNCC for Refund.....	27
Figure 10-B	Data Flow from PGS to Merchant in NNCC for Refund .....	27
Figure 10-C	Data Flow from Merchant to Client in NNCC for Refund.....	27
Figure 10-D	Data Flow from Client to PGS in NNCC for Refund .....	27
Figure 10-E	Data Flow from PGS to Client in NNCC for Refund.....	28
Figure 11-A	Token for Installment .....	29
Figure 11-B	Token for Periodic Payment.....	29
Figure 12	System Organization of NNCC .....	30
Figure 13	Data Communication Layer of NNCC .....	31
Figure 14	Number Input Interface against Key Logger.....	34
Figure 15	User Table in MySQL.....	35
Figure 16	Card Information Table in MySQL .....	36
Figure 17	Transaction Table in MySQL .....	37
Figure 18	Session Ticket Generation.....	38
Figure 19	Payment Ticket Generation .....	38
Figure 20-A	E-Commerce site using NNCC .....	39
Figure 20-B	Billing Information in the E-Commerce Site Using NNCC.....	39
Figure 21	Key Inputs under Credit Card Payment Model .....	45
Figure 22	Key Inputs under NNCC .....	45
Figure 23-A	Average Processing Time per Request in PAS .....	46
Figure 23-B	Average Processing Time per Request in PGS .....	47

## ACKNOWLEDGEMENT

I am heartily thankful to my supervisor, Dr. Yoohwan Kim, whose encouragement, guidance, and support from the initial to the final step enabled me to develop an understanding of the thesis work, and taught me how to become a good researcher by kindly correcting me.

I am thankful to Dr. Ajoy K Datta for his encouragement, guidance and support during my Master's Program. I would also thank Dr Laxmi P. Gewali and Dr Ju-Yeon Jo for serving committee member.

I also would like to thank the entire faculty members for their teachings and helps during my course work. I want also express my gratitude to friends who took the classes together with me at UNLV. We encouraged each other to do our best in academic work.

Last, I am grateful to my wife Heesun and my family. They always encourage and help me in many ways.

I am honored to be a graduate student at UNLV. I never forget this two years' wonderful life at school of computer science at UNLV.

## CHAPTER 1

### INTRODUCTION

Electronic payment system (EPS) is 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, the proliferation of EPS has brought forth an undesirable effect. The convenience of credit cards gives a purchasing power to whoever has the card number with some extra information associated with it. When a 3rd party person obtains the card number, he has the same purchasing power as the legitimate owner and can falsely use the card without the knowledge of the legitimate owner. This can happen either inadvertently or on purpose. Merchant may store the credit card numbers insecurely and get them stolen. Or fake web sites can be set up to grab the credit card information from unsuspecting victims. Once the card number and the associated information are given to a merchant, the number cannot be withdrawn. The present EPS does not provide a mechanism to hide the credit card numbers during transactions.

Not surprisingly, there have been numerous incidents of credit card frauds over the Internet due to the weakness 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 [1]. According to the data reported by Chronology of Data Breaches, a credit card fraud incident database, on April 22, 2009, a former employee at New York state tax department was accused of gathering secret data including credit card numbers and using them illegally [2]. The loss from the credit card

fraud is also large and the number of the cases is increasing. According to FBI's Internet Crime Complaint Center's 2008 Annual Report [3], the total loss from online fraud amounted to 265 million dollars in 2008, a 33.1% increase compared with 2007. Obviously the situation gets worse as people shop more on-line. People give out the credit card numbers to more e-commerce sites, which means higher chances of their credit card number stolen. Protecting credit card numbers is thus very important to reduce the loss. To this end, a number of solutions or protocols such as SSL [4], SET [5], and PayPal [6] have been proposed. While some of the solutions are used currently, some others 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 directly give their card numbers to the merchants, while merchants want to get the card number to charge the payment conveniently.

In this thesis, we present a new secure payment system that does not reveal the credit card number to the merchant while minimizing the inconvenience. This system is based on the Kerberos framework [26] and only tokens are exchanged between the buyers and merchants. The tokens are generated based on the payment amount, the client information and merchant information, but it does not contain the 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. This scheme can be best used during on-line shopping. We describe the underlying cryptographic scheme and the system construction.

This thesis is organized as follows. Chapter 2 surveys the current payment systems. We categorize them into three groups and discuss the pros and cons of each payment

system. Chapter 3 explains the concept of Kerberos and the background in Cryptography. Chapter 4 discusses our proposed system and the payment processes in our system. In chapter 5 we implement a prototype system and discuss its performance. In chapter 6, we evaluate proposed system. Then we conclude the Thesis in chapter 7 with the list of future works.

## CHAPTER 2

### RELATED WORK

Generally an EPS falls into one of two categories: token based systems (Electronic cash system, or electronic currency systems), and account-based systems (Credit-debit) [7][8][9]. However the credit card system can be considered a separate category in some cases. In this Thesis, we also divide the EPS into three categories because the credit card system is most popular among the payment system.

#### 2.1 Electronic Cash System

In electronic cash systems, customers buy digital tokens and surrender them to the merchant when they buy an item [7]. 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 user's E-Cash is stored in users' online account. The smart card-based system is not suitable for Internet Payment System due to the need for a physical contact to make a payment. Web Cash systems do not suffer from this problem and there are several systems proposed, e.g., Millicent Protocol [10], PayWord [11] and MicroMint [11], NetCash [12], eCash (or DigiCash) invented by David Chaum [13], and so forth.

Millicent Protocol is designed to process the small amount of money which can be a fraction of cents for the inexpensive internet contents. The most important parts of Millicent Protocol are Broker and Scrip. Broker provides account management and billing, and Scrip is digital cash which is valid for the specific merchant [10].

PayWord is credit-based. Customers need digital certificate signed by a broker. Digital certificate consists of customer's name, Broker name, public key which is used

for signature verification and so forth. PayWord shows its strong efficiency in the multiple transactions to the same merchant [11].

In MicroMint system, a coin is essential factor. A Broker sells coins to user, and customers give coins to merchants as payment. Merchants return tokens to the broker to get money. The validity of token is easily checked, but it is almost impossible to forge it [11].

Net Cash provides anonymous transaction and real time payment processing under multiple server environments [12].

DigiCash (or eCash) is based on the RSA blinding signature in which the content of 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 token or message are different [13].

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 token is blind before it is signed by bank [13]. Also, transaction processing is efficient because the exchange of tokens is performed locally without connecting a remote transaction server [7]. 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 burden to them and made it not widely used [8].

## 2.2 Account Based Systems

In the account-based system, the exchange of money between users' accounts is performed by a payment service provider [7][8]. The examples of Account-Based systems include PayPal [6], Netbill [14], NetCheque [15], and so forth

PayPal is a very popular service for web-based transactions. PayPal users can send or receive money by using their email address. It is widely used in C2C (Customer-to-customer) transactions but can be also used in B2C (Business-to-customer) transactions [6] [16]. PayPal does not reveal the detailed account information of the transaction partner to the other users; its transactions assure some privacy. However, its authentication scheme is primitive and several ways are known to hack PayPal's ID easily using the Internet. Also, the management company of PayPal system seems to be a risk factor because it could go bankrupt with users' money. Also PayPal requires Credit Card Number to deposit money to the PayPal account, thus it has the same problems as credit card payment models have.

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 [14].

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 electronic signature, and then send it to the merchant [15]. NetCheque is based on the Kerberos concept [24] 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, NetCheck 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 [7][8]. This model is most widely used in the Internet due to its simplicity and convenience for 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 credit card system operates currently.

#### 2.3.1 Card Payment Processing Network

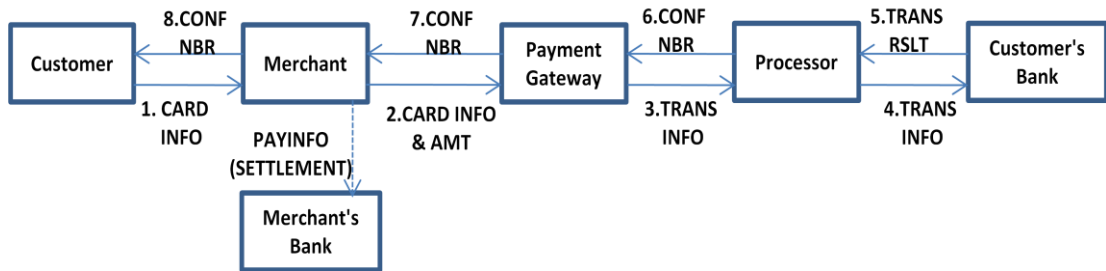


Figure 1 Current Payment process under Credit Card Payment Model

As shown Figure 1, several elements are involved with the Card Payment Networks. Payment Gateway provides connectivity between Merchant (i.e. Payment site) and

Processor (i.e. Financial Networks). The Processor is a large data center which processes credit card transactions. This is how payment processing works [17].

- 1) A customer submits a card number to the merchant.
- 2) Merchant sends payment information such as card number and amount of money to Payment Gateway.
- 3) Payment Gateway passes the information from merchant to processor.
- 4) Processor sends the information to customer's bank.
- 5) Customer's bank sends transaction result (approved or rejected) to the processor.
- 6) Processor passes the transaction result to the payment gateway.
- 7) Payment gateway sends the transaction information to the merchant.
- 8) Merchant saves transaction information for the settlement and sends confirmation number to the customer.

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

### 2.3.2 Protocols for Secure Transaction of Credit Cards

As shown Figure 1, a credit card number should be sent over the Internet. 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 encrypted using SSL (Secure Socket Layer). SSL (Secure Socket Layer) is a very popular web content

encryption technology, not a payment protocol, and virtually all credit card transactions are encrypted using SSL nowadays. SSL employs asymmetric key encryption for the communication between customers and merchants, and allows the user to authenticate the identity of the merchant using digital certificate [4]. In many cases, the credit card numbers are stored at merchant's database either on purpose or by negligence. So, there is a possibility that the database is cracked by hacker or used illegally by insiders on the merchant side. Most of card numbers fraud comes from this. Besides, recently SSL based on HTTP (-HTTPS) could be hacked by several tools such as sslstrip which can trick Web Browsers into thinking they are on a secure site [18][19].

SET (Secure Electronic Transaction) is security protocol for the 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) [5]. Unlike SSL, SET prevents merchants from using customer's credit card number illegally because cardholder shares order information with merchant and shares payment information only with bank (dual signature). However, SET failed to be implemented in the industry due to its complexity which gave burden to the customers.

The concept of one-time Credit card number (deposable number, single-use number) [20] is recently used in the market place such as American Express, Discover, MBNA, and Visa's Gift Cards [21]. One-time credit card number is only for single use. After using single-use number, illegal use of the card number is impossible. From this way, customers do not need to worry about credit card number theft. However, whenever they buy things, customers should have online connection with card issuer to have new card

number which is burden to both customers and card issuers. To solve this problem, Yingjiu Li proposed one-time payment scheme which generate card numbers with hash function [22]. However, in this scheme, we cannot use current credit card because it require smart card reader.

## CHAPTER 3

### TECHNICAL BACKGROUNDS

#### 3.1 Cryptography and Mcrypt

Cryptography is hiding information [23]. When Alice wants to share information only with Bob, she encrypts the original data (plaintext) with her key and sends it (encrypted data – cipher text) to Bob. Bob can read the cipher text by decrypting it with his key. Anyone else Bob and Alice cannot share the information without specific key. This is how Cryptography works. We can divide cryptography algorithm into two groups – symmetry key and asymmetry key. In a symmetry key system, we encrypt and decrypt data with same key. AES, DES, RC4 and so forth belongs to symmetry key system. Unlike symmetry system in an asymmetry key (called also public-key) system, we use different key for encryption and decryption of data. RSA is an example of asymmetry key.

Hash function is used in the cryptography for the integrity of data. It gets block of data as its argument and returns a bit string called hash value or message digest. This value is used to check of the data modification detect by adding it to the end of the data at the sender's side and by comparing hash value from sender and new hash value generate by a receiver. SHA1 and MD5 is an example of cryptographic has function.

There are lots of secured Cryptography algorithms, and some programming libraries offer these algorithms. Mcrypt is an open source implementation of cryptography algorithm library which includes symmetric-key, hash, and public-key algorithms [24].

### 3.2 Kerberos

Kerberos is designed at MIT to protect network services provided by Project Athena. Currently, we use Kerberos Version 5 defined in RFC 4210.

It is authentication protocol to prove client's identity attempting to log on server and encrypts their communications through secret-key cryptography for an unsecure network [25][26]. It never sends password over the network, so the password is protected against \*eavesdropping and \*replay attacks. It also solves Key Distribution problems between client and server with Ticket.

- Eavesdropping attacks (Man-in-the-middle attack) - attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other.
- Replay attack - a valid data transmission is maliciously or fraudulently repeated or delayed.

It requires additional servers -Authentication Server, and Ticket-Granting Server. When client is authenticated, it gets TGT (Ticket-Granting Ticket) from Authentication Server. With TGT, client can get Ticket for specific server from a Ticket-Granting Server. With Ticket, client can login application server.

The advantage of Kerberos system is that a user does not need to register in each server, and each server does not need to have each user's username and password in their storage.

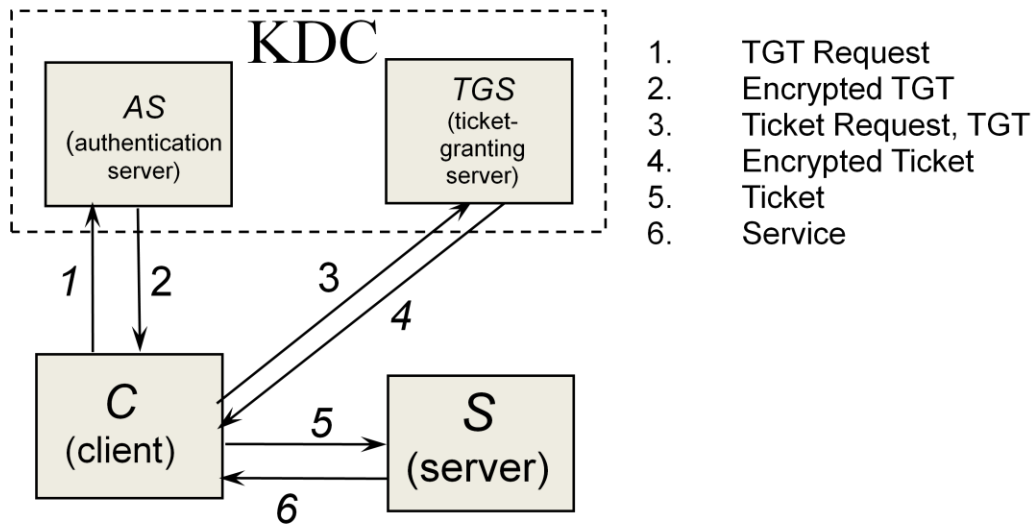


Figure 2 Kerberos System

### 3.2.1 Participants of Kerberos

- Client (user)
- S: Application Server
- Authentication server (AS) – check whether a user is registered in system and generates Ticket Granting Ticket (TGT). It does not care about how many application servers are in the system.
- Ticket Granting Server (TGS) – verify TGT and issue Ticket. It does not care how many users are registered in the system.
- Ticket: Kerberos data structure that can be safely sent across the networks. When the ticket is valid, the sender of it can prove its identity.
- Ticket has two properties like this.
  - All tickets in Kerberos are encrypted with the key of the final recipient.
  - Client has no knowledge of the tickets' content.

### 3.2.2 Keys in Kerberos

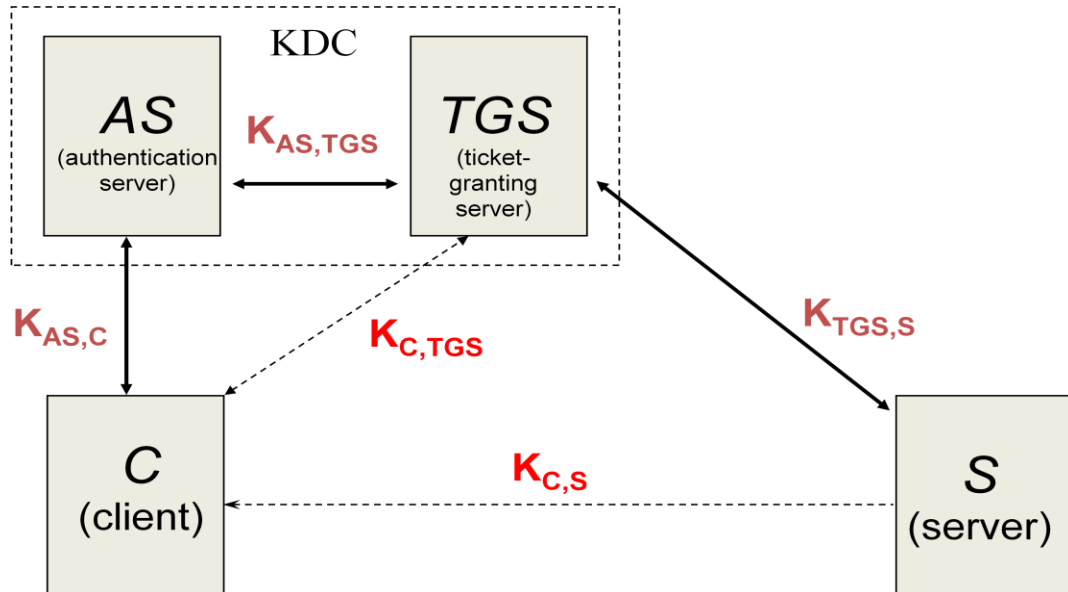


Figure 3 Keys in Kerberos

As shown above, different keys are shared between elements.

Notation

- $K_{AB}$  : Secret Key between A and B
- {Clear Text}  $K_{AB}$  : clear text encrypted with key  $K_{AB}$
- $K_{AS,TGS}$ ,  $K_{AS,C}$ ,  $K_{TGS,S}$  is pre-defined, and  $K_{C,TGS}$ ,  $K_{C,S}$  is dynamically generated
- NC and N'C are a nonce generated by Client
- Nonce: A randomly chosen value, different from previous choices, inserted in a message to protect against replays.
- $L_k$ : ticket lifetime (valid period).
- $T_k$  : Time Stamp.



### 3.2.3 Flow of Kerberos

0) User enters ID and Password (Password is secret key between Client and AS).

1) Client sends TGT (Ticket Granting Ticket) request message to the AS.

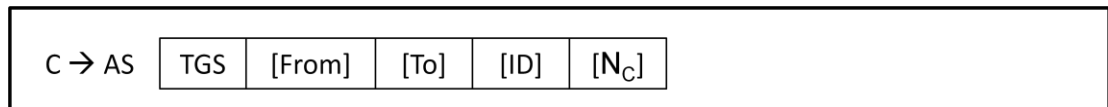


Figure 4-A Data Flow from Client to AS in Kerberos

‘From’ and ‘to’: time interval used for Key validation.

2) When AS receives message, AS checks if the user is in the database. If the user exists, AS generates session key ( $K_{C,TGS}$ ) and TGT, and sends them into message composed with 2 parts to the client.

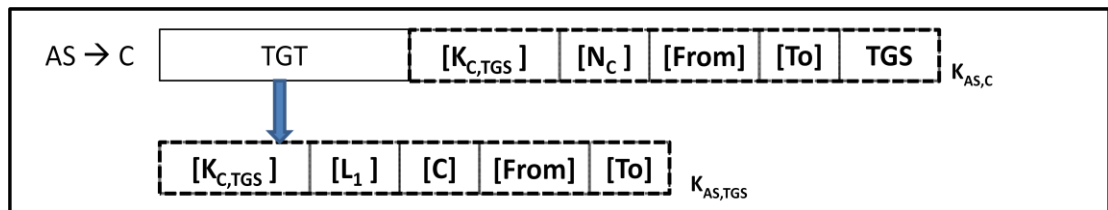


Figure 4-B Data Flow from AS to Client in Kerberos

As shown step 1, 2 Password is never sent over the network. It is used as a Key for encryption and decryption.

3) Clients receive message from AS, and encrypt the second message with Password ( $K_{AS,C}$ ) to get a session Key ( $K_{C,TGS}$ ) and other information. After then, client sends message including TGT after encrypting it with  $K_{C,TGS}$  to the TGS.

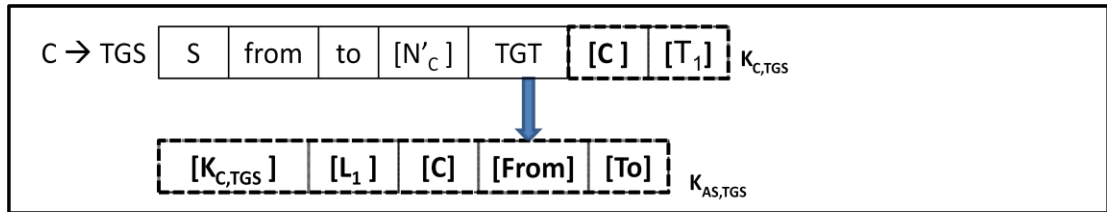


Figure 4-C Data Flow from Client to TGS in Kerberos

4) TGS receives message from the client and with TGT from which it gets  $K_{C,TGS}$ . And then TGS decrypt the rest of message. If TGS is valid, it generates Ticket and Client/server session key ( $K_{C,TGS}$ ) and send them to the client.

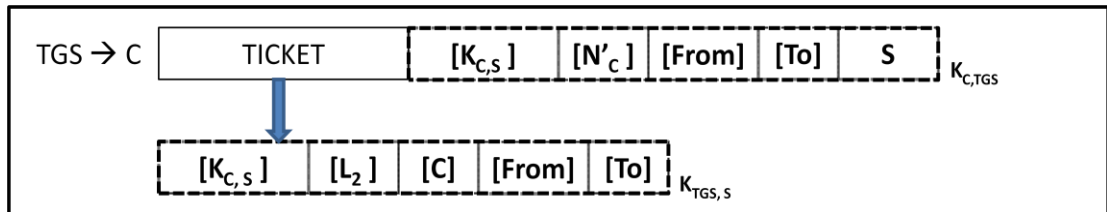


Figure 4-D Data Flow from TGS to Client in Kerberos

5) After getting message from TGS, Client sends Ticket to Server to get access permission to server.

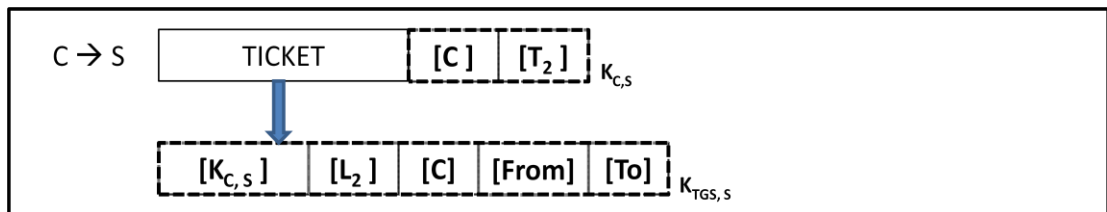


Figure 4-E Data Flow from Client to Server in Kerberos

6) Server sends Confirmation Message encrypted with Client/server session key ( $K_{C,s}$ ) if Ticket is valid.



Figure 4-F Data Flow from Server to Client in Kerberos

### 3.3 SSL (Secure Socket Layer) and Open SSL

The SSL is used for secure communication over a network which works on Transport Layer between two applications. It is based on public key cryptography to accomplish its tasks, and provides server authentication and encrypted data communication channel. It uses digital certificate for certification of participants, especially server. SSL guarantee higher security providing Confidentiality, Identity authentication, and Message integrity verification [4]. Open SSL is an open source implementation of SSL [27]. Open SSL is multi-platform library which supports C and C++.

## CHAPTER 4

### SECURE ON-LINE CREDIT CARD TRANSACTION METHOD

As we discuss chapter 3, Kerberos system is very secure and reliable [26]. Some payment protocols such as NetCheque [15] are suggested based on this protocol. However, they are theoretical and never implemented industry domain. Our solution is also based on Kerberos, but we modify the protocol to make it practical payment system. So our new system works under current card payment network and card holders and merchants can easily migrate from current payment process to our system with minor change.

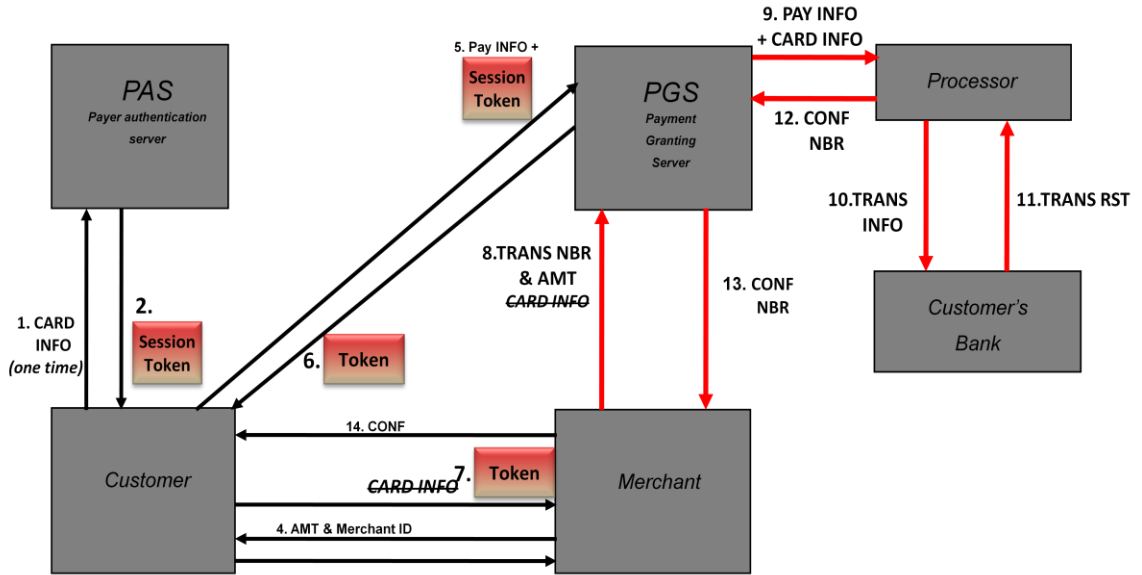


Figure 5 - A System flow of NNCC

This new system looks similar to Kerberos because of its structure. You can regard customer as client, PAS as AS, PGS as TGS, and merchant as server in Kerberos. However, it has several different things. First, the operation sequence is different. Second,

it does not have key between merchant and customer while Kerberos has key between client and server. Third, it has new communication relationship between merchant and PGS. Fourth, it has reverse operation (refund operation). Besides, its token contents are totally different from the content of Kerberos Ticket.

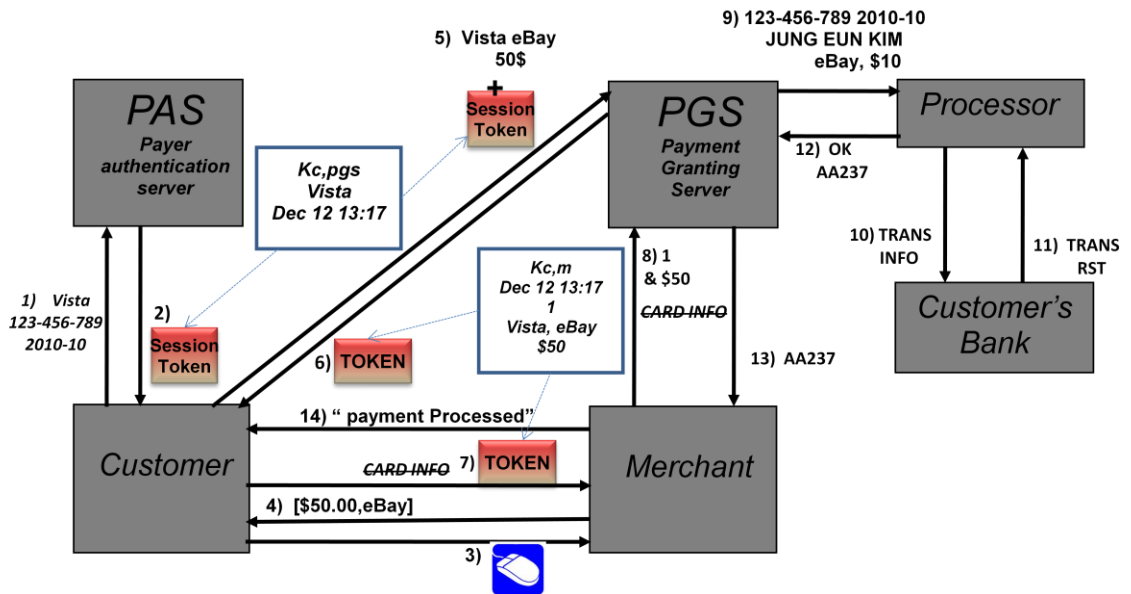


Figure 5 - B System Flow of NNCC with Actual Data

As shown Figure 1, in current card payment flow, customer is connected to merchant and merchant is linked with Payment Gateway. In our solution illustrated by Figure 4-A and 4-B , PGS server (TGS server in Kerberos) which replaces Payment Gateway is connected to both customer and merchant together unlike current payment system.

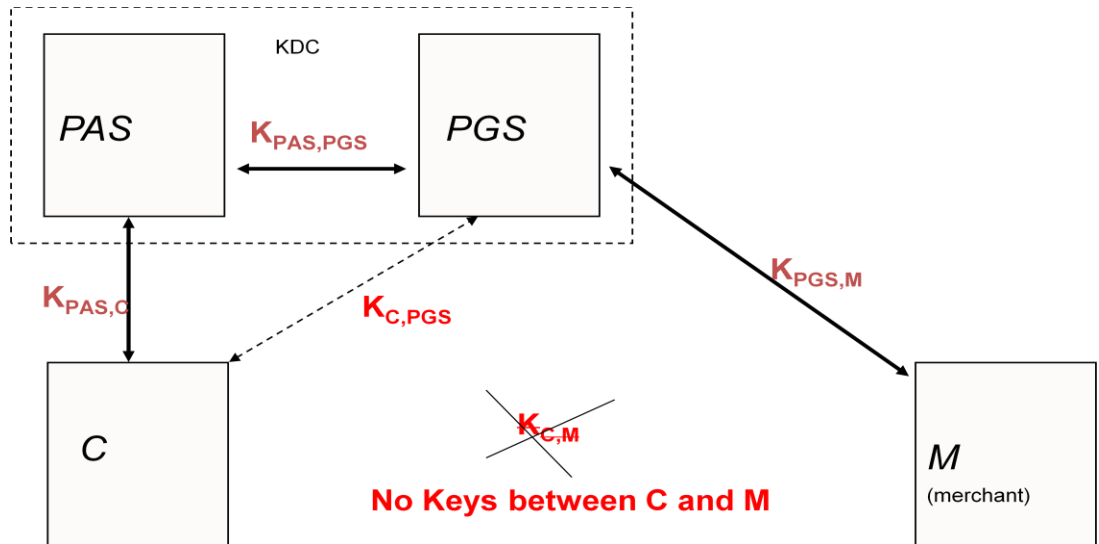


Figure 6 Keys in NNCC

Two tokens are used – Session Tokens and Payment Tokens.

Session Token means the authorization of customer like TGT in Kerberos.

Payment Token contains information of client and merchant, and the amount of money paid.

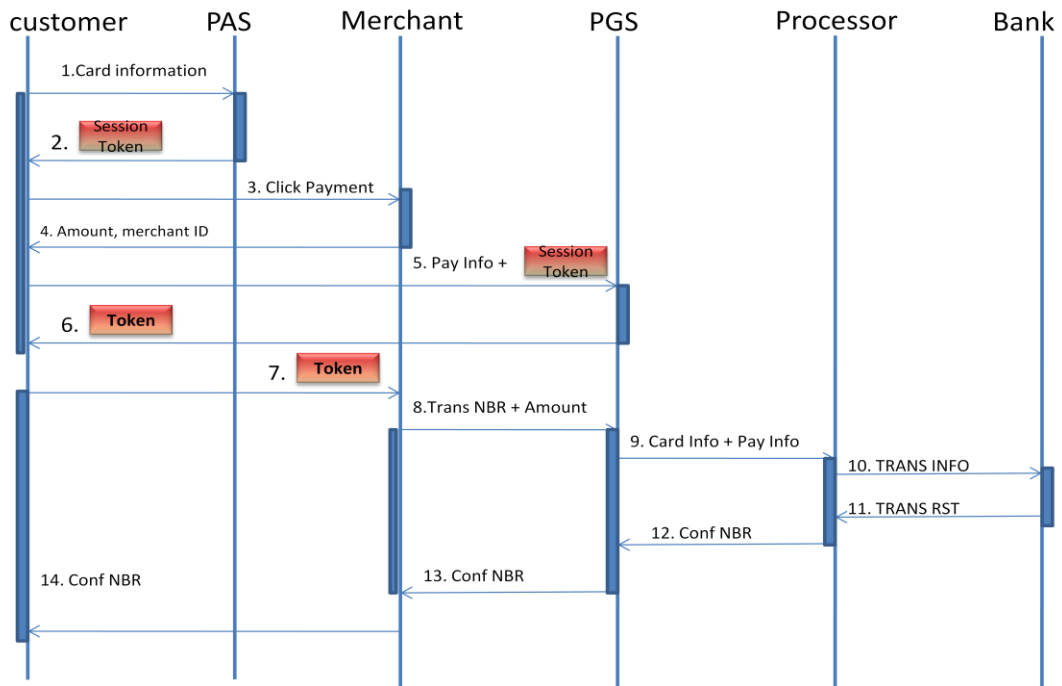


Figure 7 Sequence Diagram of NNCC for Payment

The rough process is like this. First of all, customer sends card information to the PAS server, and gets a session token. After that, Customer sends payment information (the amount of money and merchant name) with session token to the PGS instead of merchant, and gets Payment Token from PGS. And then, customer sends Payment Token to the merchant. Last, to make actual payment, merchant sends transaction number included in Payment Token to PGS server. Payment token is also used for merchant's settlement processing afterwards. After PGS receives transaction number, it proceeds actual payment processing.

#### 4.1 Advantage of the System

The advantages of Kerberos system becomes those of this payment system. User does not need to send card information to the each merchant and merchant does not need to have user card information. This prevents merchant or its employee from stealing customers' card information because merchant cannot save card information in its database. Under this flow, Credit card number encrypted with user password is sent to PAS. So even if card number is known to other, he or she cannot get session Token from Payer Authentication Server without knowing user's password which is pre-registered through the trust path. In other words, he or she cannot get payment Token. Also, mutual authentication is implemented among payment systems, merchant and customer because each participant needs to know its shared key to communicate with each other. So, merchant can verify whether the customer is an actual owner or not. This is important because the merchant is entirely responsible for the loss from the card number theft in some countries. Furthermore, Merchant cannot modify payment information under this system.

We are also sure that the migration from current existing system to our new system is easy and simple. Customer only needs to download tiny software program, and merchant just need to send transaction number in the Payment Token from the customer, instead of card information and the amount to the Payment Gateway (Payment Granting Server). This is important because several secure payment system has been proposed, but they are not implemented in the industry because of its complexity or burden to customer and merchant.

## 4.2 Scenario

### 4.2.1 Payment

Detailed payment process is like this.

- 1) User enters ID, card Information and Password (Password become Key between Client and PAS).
- 2) Client sends data (ID, card Information) after encrypting them with user password.

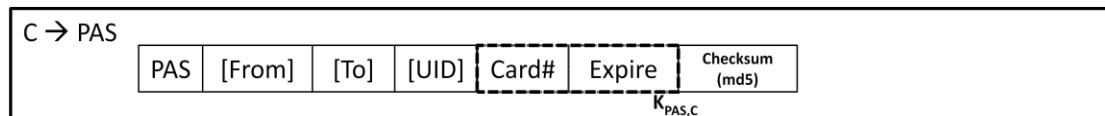


Figure 8-A Data Flow from Client to PAS in NNCC

‘from’ and ‘to’ indicates a specified time interval.

- 3) PAS process user request as like this
  - a. Check whether user is registered or not. If registered, decrypt message.
  - b. Card information is registered to the database. The information is automatically deleted on the time when session token is expired in client side.



c. Generate session token and send it to the client.

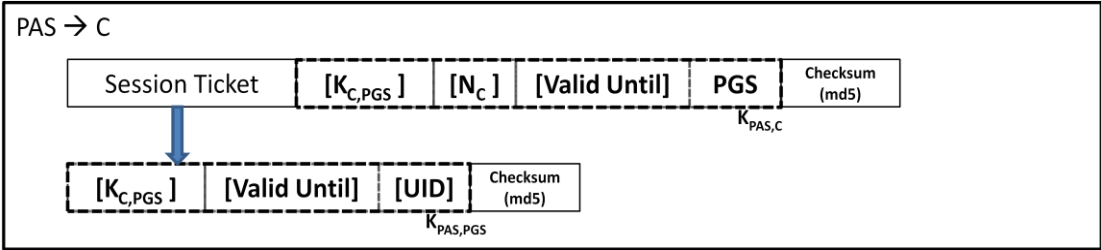


Figure 8-B Data Flow from PAS to Client in NNCC

- 4) Customer decides to purchase things.
- 5) Merchant sends Amount and Merchant ID to Customer.
- 6) Client Program sends Payment Information such as Merchant ID and amount of money encrypted with key and session token to PGS server.

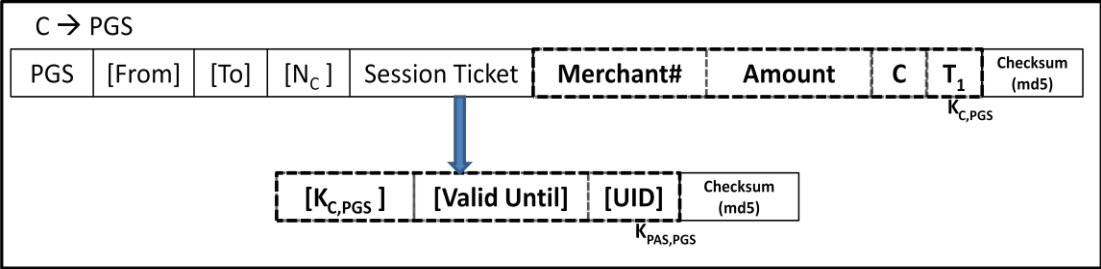


Figure 8-C Data Flow from Client to PGS in NNCC

- 7) PGS work like this.
  - a. Check if Session token is valid.
  - b. Generate transaction information (amount, customer, merchant etc) on the database.

c. Generate Key shared between merchant and client, and Payment Token for merchant.

d. Send Key ( $K_{C,M}$ ) and Payment Token to the Client.

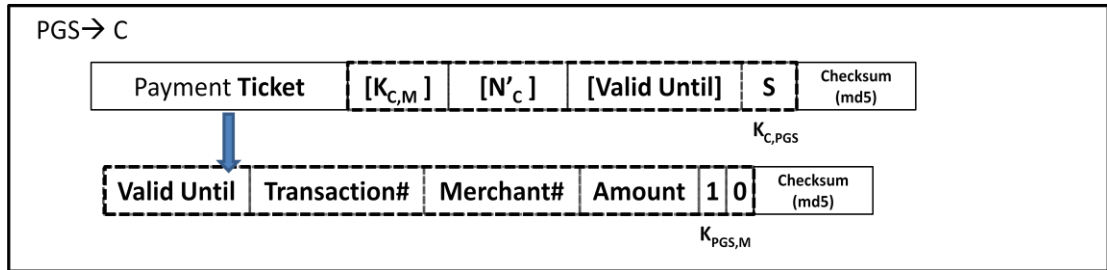


Figure 8-D Data Flow from PGS to Client in NNCC

8) Client sends Payment Token to the merchant.

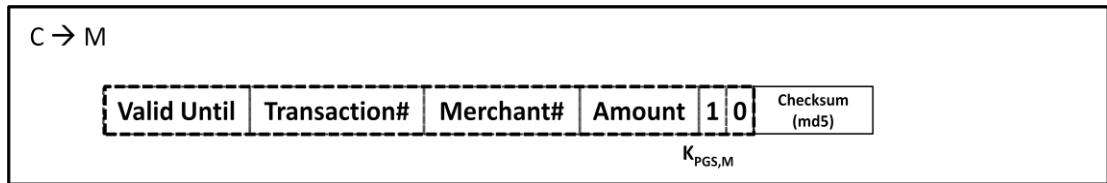


Figure 8-E Data Flow from Client to Merchant in NNCC

9) Merchant extracts Transaction number from the Payment Token and sends with amount to PGS server.

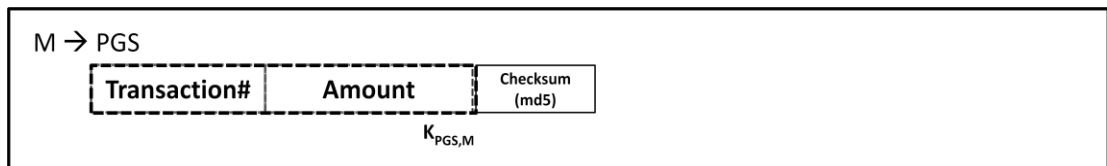


Figure 8-F Data Flow from Merchant to PGS in NNCC

- 10) After PGS server getting Transaction number from the merchant, actual payment process starts.
  - a. From the Transaction number, PGS server retrieves Payment Information (card #, amount, merchant ID) from its database.
  - b. PGS sends the Information to the Processor.
- 11) Processor request services to Client's bank.
- 12) Customer's Bank response to the Processor (Accept or Reject).
- 13) PGS receives confirmation number from the Processor.
- 14) PGS sends confirmation number to the merchant.



Figure 8-G Data Flow from PGS to Merchant in NNCC

- 15) The merchant sends this number to the client.



Figure 8-H Data Flow from Merchant to Client in NNCC

Stage 0, 1, 2 is the process of getting session token. If a client has a valid session token, it does not need to get a new session token as long as the session token is valid. However, it has to request payment token every payment process.

#### 4.2.2 Refund

Refund is a just reverse process of payment. We assume client has a valid session token. Customer can get session token without credit card number, because refund process does not require card information but transaction Reference Number.

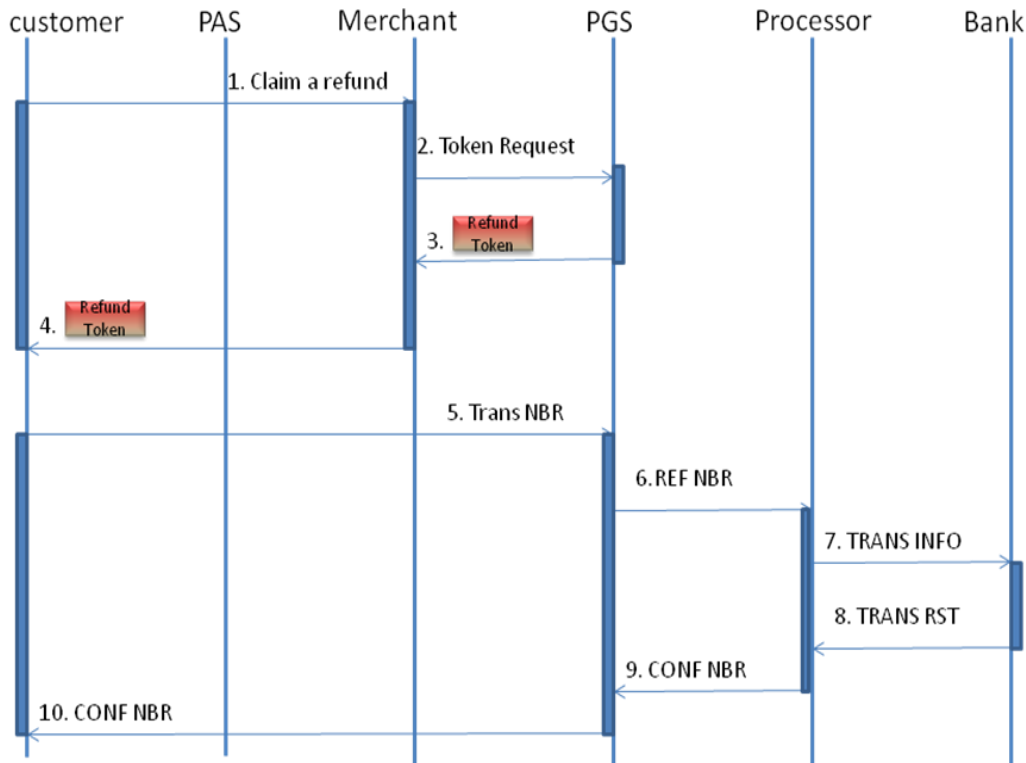


Figure 9 Sequence Diagram of NNCC for REFUND

- 0) Customer claims a refund.
- 1) Merchant requests Refund Token to PGS.

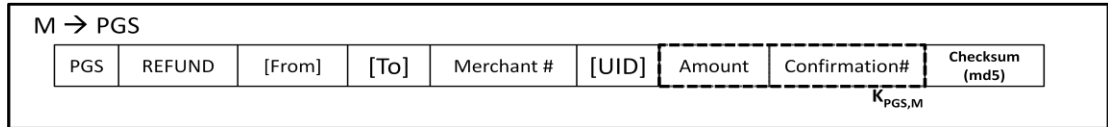


Figure 10-A Data flow from Merchant to PGS in NNCC for Refund

2) After getting request, PGS sends REFUND token and data to the Merchant.

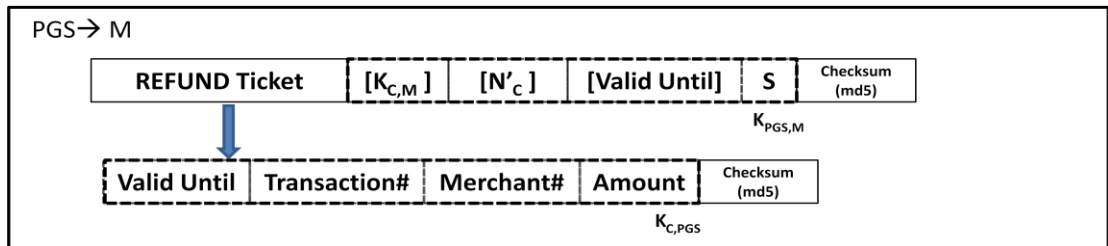


Figure 10-B Data flow from PGS to Merchant in NNCC for Refund

3) Merchant sends “REFUND TOKEN” to the Client.

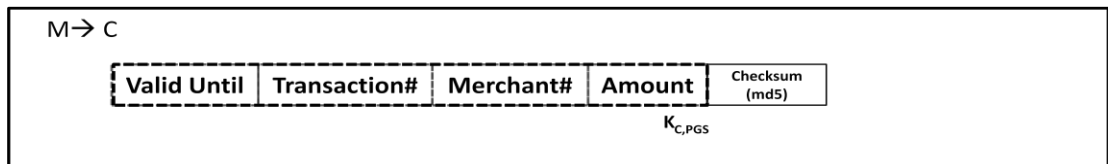


Figure 10-C Data flow from Merchant to Client in NNCC for Refund

4) Client sends Transaction number and amount to the PGS after getting “REFUND TOKEN”.

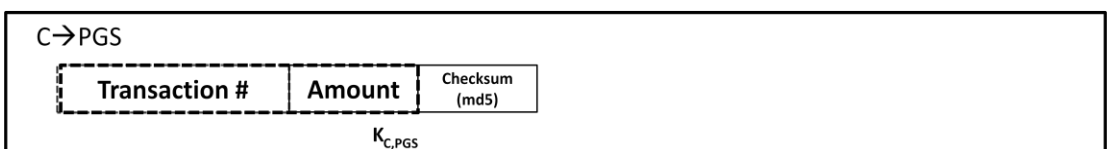


Figure 10-D Data flow from Client to PGS in NNCC for Refund

5) After PGS server getting Transaction number from the client, actual REFUND process starts.

a. From the Transaction number, PGS server retrieves Payment Information (Transaction Reference Number, amount, merchant ID, UID) from its database.

b. PGS sends the information to the Process

After several steps, PGS receives confirmation number from the Processor.

6) PGS sends confirmation number to the client.



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

#### 4.2.3 Installment (Periodic Payment)

In the current payment system, Installment through the Credit Card Company requires additional Interest. Besides, Debit card does not allow it. In our system, we can provide this function without Interest to customer s regardless of their card type (debit or credit). The idea is simple. PGS has additional information on its database for this (refer to database structure 5.5.3). PGS will do most of work. At the first payment, PGS use customer's credit card number. From second payment, PGS will use Transaction Reference Number of previous transaction which does not require customer's credit card number. Actual process will be same with the general payment Process to both customer and merchant. Customer just needs to request TOKEN for Installment instead of general payment. We can think two cases for Installment – real Installment or periodic payment.

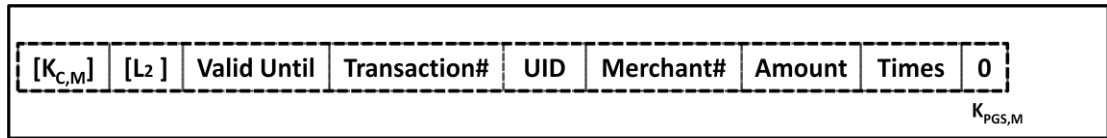


Figure 11-A TOKEN for Installment

Figure 11-A shows Token for Installment. PGS will divide the Amount with n (the value of ‘times’) and request payment with the quotient to the processor for n times.

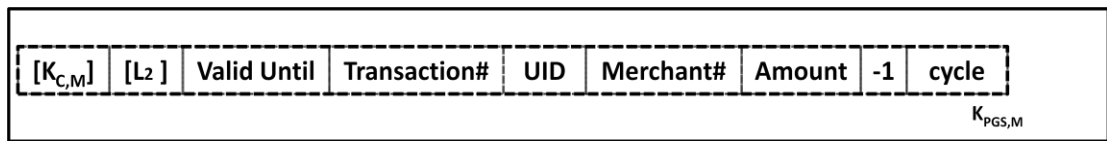


Figure 11-B TOKEN for Periodic Payment

Figure 11-B shows Token for Periodic Payment. PGS will request payment with the amount to the processor every ‘cycle’.

## CHAPTER 5

### IMPLEMENTATION

The proposed protocol is implemented using C. We use Crypt library for the encryption and open SSL library for data communication. To show all the payment process during the payment process, we install open source shopping mall package written in PHP and modify it to link our payment system. PAS and PGS module works under both UNIX and Windows system. PAS and PGS could be physically separated depending on database type. In this chapter, we describe how we implement our system in detail.

#### 5.1 System Organization

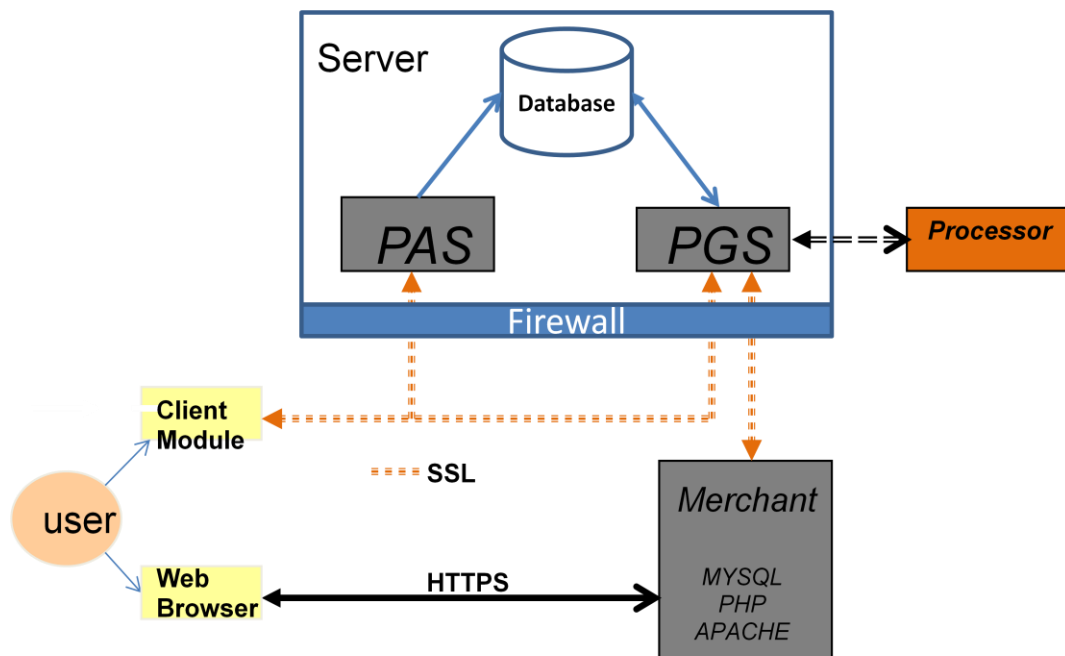


Figure 12 Sequence Diagram of NNCC for REFUND



The connection between customer and Merchant is based on HTTPS or HTTP. Merchant keeps connection to the PGS continuously for the efficiency.

### 5.2 Data Transfer Modules

To achieve highest secure channel among the element, we implement new data communication module. In new module, we use two functions - Send and Recvline. Figure 9, shows protocol stack of this library.

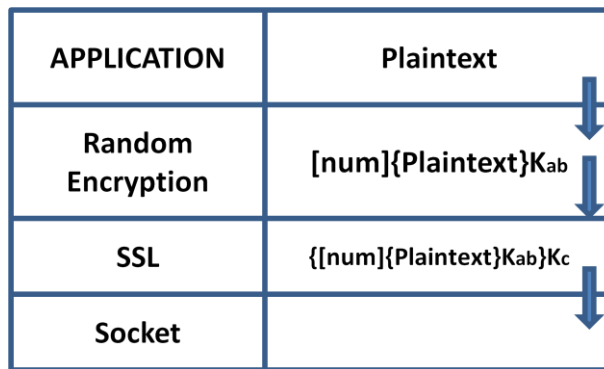


Figure 13 Data Communication Layer of NNCC

#### 5.2.1 Randomization of Cryptography Algorithm

Encryption method (or algorithm) is also random. MCrypt library we are using in our program provides a lot of encryption algorithms. If we use one of them continuously, there could be possibility of password cracking by Guessing, dictionary and brute force attacks. To decrease the possibility of this attack, we can choose an algorithm at random when each element sends data to other. Before we send data, we put the 2bit flag (called num) to indicate which algorithm is used.

#### 5.2.2 Data Transfer with SSL

To maintain most secure channel among elements in the system, we can use SSL protocol in our communication. It means that we encrypt data with key, the encrypted

data is transferred under [27] open-SSL library instead of socket library. Figure 13 shows the implementation of 5.2.1 and 5.2.

### 5.3 Token in Real System

Normally, when we send well-organized data, we use structure (or class) in the programming. We read and write data by structure. However, in our system we represent data in other way. Each data ends with newline, and fields in each data group are distinguished with special character. Before sending data, we encrypt the data, change the encrypted data into hexadecimal value, and we add hash value of the data to check data modification during the transmission. We also generate token in this way.

#### 5.3.1 Session Token

Table 1 Session Token

$K_{C,PGS}$	Valid Until	UID
Key between client and PGS	The time Token expires	User ID
Abcdef	OCT 10 2009 13:00	Vista
Session Token data	abcdef  OCT 10 2009 13:00  Vista	
Token after encryption	8c3c6438dd4de73f9cb536252894d293318e17e57958a41ef0fff903240314c4	

As shown in Table 1, session token in NNCC is similar to session ticket in Kerberos except for that fact token in NNCC has hash value at the end of it. With hash value, we can check the integrity of token.

### 5.3.2 (Payment) Token

Table 2 (Payment) Token

$K_{C,M}$	$L_2$	Valid Until	Trans#	UID	MID	AMT	TIMES	CYCLE
Key	Time stamp	The time token expires	Transaction Number	USER ID	Merchant ID	Amount (price)	Number of Installment	Period of Payment
Abcdef	2 am	Oct 10...	100	vista	eBay	\$50	1	0
Token data	Abcdef  2 am  oct 10 2009 13:30  100  vista  eBay  \$50  1  0							
Token	ac080b691f5f32fc4e9a4913e52b799d6d4b82d91af6db87fb44c8ef07a4777bac57222							
after encryption	710b2cbc31e0df4e0b2ae917e1c28881481860244149c2b1b							

Payment token in NNCC is totally different from ticket in the Kerberos. Payment token has fields related to payment while ticket has data for the access of the system. Also, final recipient of payment token is a merchant or customer while that of ticket in Kerberos is only server. Because of this, NNCC could be applied to B2B, C2C as well as C2B.

### 5.4 Keyboard Interface

Attackers can steal information by installing Key logger software in a user's computer. Key logger software catches keyboard input by interrupting key board message, and sends it to an attacker. In our system, we do not allow keyboard input in registering card number from a keyboard directly and provides button interface for it instead. However, each button always has same value; it could be cracked by another malware. So, we change value of each button whenever a user clicks a button.

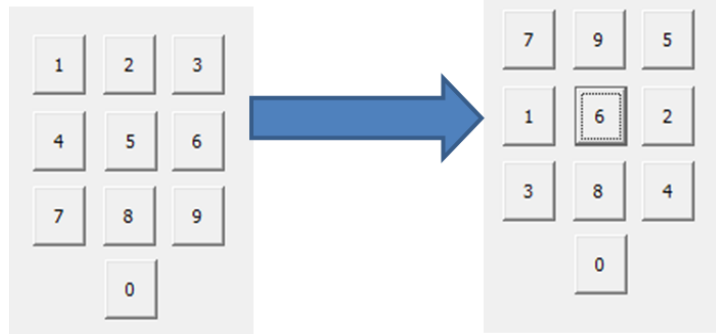


Figure 14 Number Input Interface against Key Logger

## 5.5 Database Structure

These are database schema we are proposing in NNCC.

### 5.5.1 User Table

Table 3 User Table

UID	NAME	Password
User login ID	User Name	$K_{PAS,C}$
Vista	JUNG EUN KIM	123

When PAS receives message from a client, it needs  $K_{PAS,C}$  to decrypt the message.

From the user table, PAS gets  $K_{PAS,C}$  of the client. (\* 4.2.1 – 1)

In real system,  $K_{PAS,C}$  should be encrypted

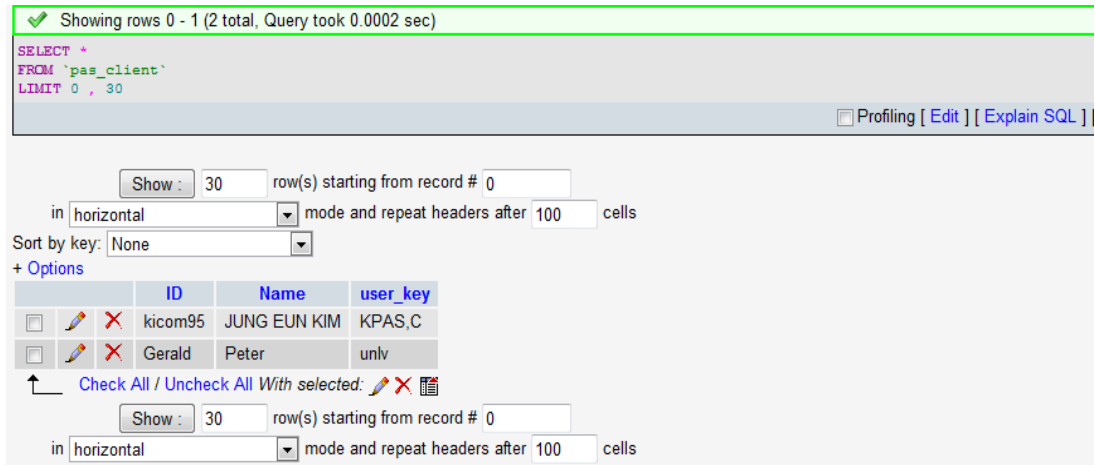


Figure 15 User Table in mySQL

### 5.5.2 Card Information Table

Table 4 Card Information Table

UID	CNUM	EXP	Availability
User login ID	Card Number	Card expiration date	Valid Until
Vista	1234	June 10 2009	17 sep 19:30

Each row is automatically deleted on the specific time (the value of availability field).

When PAS decrypts card information, it inserts the information in this table and generates Session Token. The value of “availability” field in a each row is the same with that in the session token. (i.e when the session token expires, the related row in this table is deleted at the same time) (\*4.2.1 – 2)

When PGS sends Transaction Information to the processor, it uses this table to retrieve card information. (\*4.2.1 – 9)

In the real system, we need to encrypt card information table.

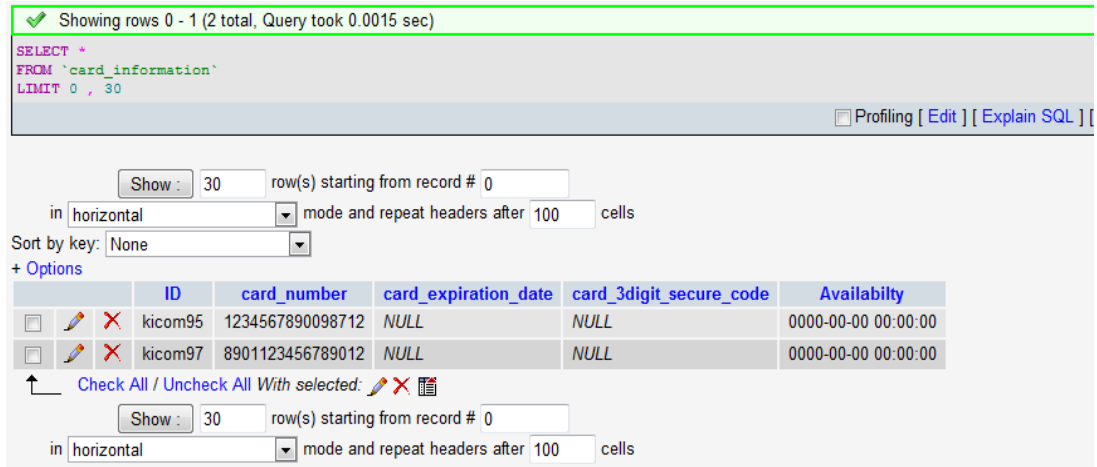


Figure 16 Card Information Table in MySQL

### 5.5.3 Transaction Table

TABLE 5 Transaction Table

TYPE	SEQ	UID	MID	CNUM	Amount	Availability	TxnRef	status
Transaction	Transaction	User	Merchant	Confirmation	Amount	Valid Until	Transaction	
Type	Number	ID	ID	Number			Reference	

Transaction type: Normal: N Refund: R Installment: I

Status: Waiting (for request) :W Processed: P N: newly triggered by Installment

TxnRef : Transaction Reference Number ( returned by Processor used Refund )

When PGS receives Token request message including amount, merchant ID from a client, it inserts information in this table and generates Token. (4.2.1 – 6)

When PGS receives Transaction number from a merchant (4.2.1 – 8), sends Transaction Information to the processor in order to make actual payment (4.2.1 – 9) after joining Card Information Table and Transaction Table.

This is an example of actual data in the Transaction Table.

TABLE 6 Transaction Table Example

TYPE	SEQ	UID	MID	CNUM	Amount	Availability	TXNREF	Status
N	1	Vista	eBay	NULL	\$50.00	17-01 19:30	NULL	W
N	2	Ms	eBay	4A2BC	\$30.00	NULL	1237367	P
R	3	Ta	Yahoo	A1313	\$30.00	NULL	4646788	P

Transaction (SEQ) 1: Payment Token is issued to the client (vista), but PGS does not receive transaction number from the merchant (yahoo). When PGS receives transaction number (1), it will send the processor all the information including customer's (vista) card information from the card information table

Ex) – [1234567890/June 10 2011/yahoo.com/\$50.00]

Transaction 2: Payment Token was issued and PGS got transaction number from eBay.

Transaction 3: Refund Token was issued to yahoo and PGS got transaction number from the customer.

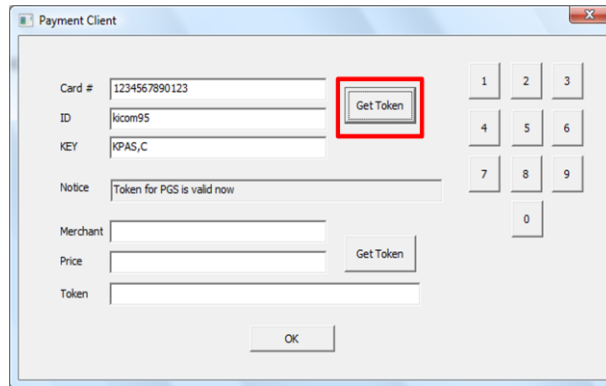
transaction_id	TYPE	USER	MERCHANT	CONFIRMATION_NUMBER	AMOUNT	Availability	TxnRef	status
1	N	kicom95	yahoo.	NULL	30	NULL	NULL	
2	N	kicom95	yahoo.com	NULL	30	2010-01-05 17:43:24	NULL	
3	N	kicom95	yahoo.com	NULL	30	2010-01-05 17:57:54	NULL	
4	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W
5	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W
6	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W
7	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W
8	N	kicom95	yahoo.com	NULL	30	2010-01-06 15:24:27	NULL	W
9	N	kicom95	yahoo.com	NULL	30	2010-01-06 15:37:34	NULL	W
10	N	kicom95	yahoo.com	NULL	30	2010-01-06 15:42:37	NULL	W
11	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W
12	N	kicom95	yahoo.com	NULL	30	0000-00-00 00:00:00	NULL	W

Figure 17 Transaction Table in MySQL

## 5.6 Demonstration

In this chapter, we will show how our payment system works

### 5.6.1 Session Token Generation



The screenshot shows a 'Payment Client' window with the following fields and controls:

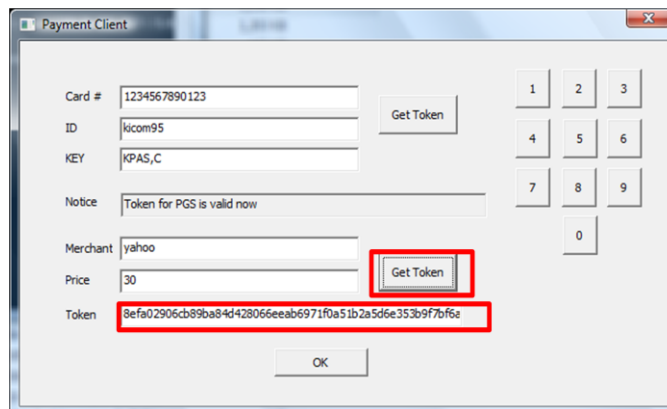
- Card #: 1234567890123
- ID: kicom95
- KEY: KPAS,C
- Notice: Token for PGS is valid now
- Merchant: (empty)
- Price: (empty)
- Token: (empty)

There are two 'Get Token' buttons. The first 'Get Token' button, located to the right of the ID and KEY fields, is highlighted with a red box. To the right of the input fields is a numeric keypad with buttons for digits 1-9 and 0. An 'OK' button is located at the bottom center of the window.

Figure 18 Session Token Generation

As you see in Figure 18, we need to input card information, ID, and Key to get a session token. We do not need to get the session token as long as we have valid one.

### 5.6.2 Payment Token Generation



The screenshot shows the 'Payment Client' window with the following fields and controls:

- Card #: 1234567890123
- ID: kicom95
- KEY: KPAS,C
- Notice: Token for PGS is valid now
- Merchant: yahoo
- Price: 30
- Token: 8efa02906cb89ba84d428066eeab6971f0a51b2a5d6e353b9f7b6a

There are two 'Get Token' buttons. The second 'Get Token' button, located to the right of the Merchant and Price fields, is highlighted with a red box. The Token field is also highlighted with a red box. The numeric keypad and 'OK' button are also visible.

Figure 19 Payment Token Generation



We can get Payment Token if we have valid session token. To acquire payment token, we need to give Merchant ID and amount of money. After getting Payment Token, it is automatically saved in the clipboard.

### 5.6.3 Payment Process

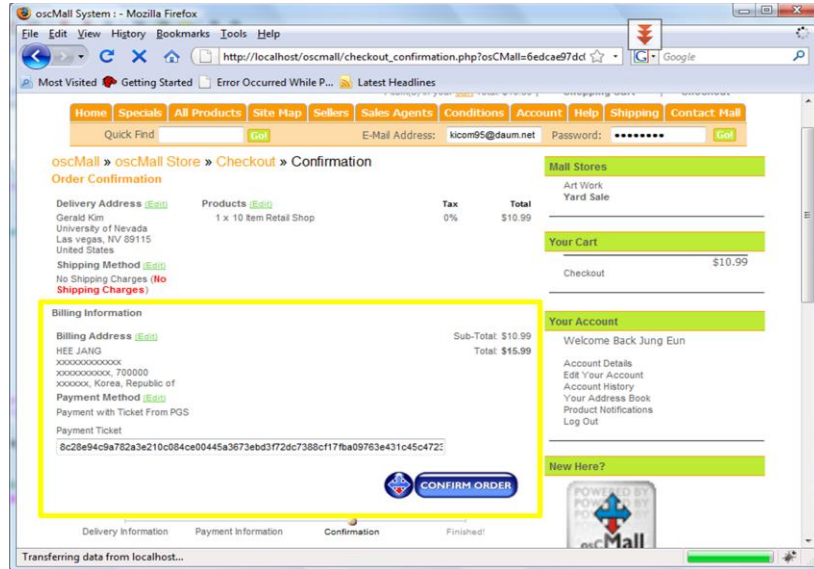


Figure 20–A E-Commerce Site Using NNCC



Figure 20–B Billing Information in the E-Commerce Site Using NNCC

Figure 20-A and 20-B show a payment page of E-commerce site using NNCC. Instead of giving card information, we pass payment token to the page. Passing payment token to the edit box is easily done by copying the token in the clipboard.

## CHAPTER 6

### PERFORMANCE EVALUATION

#### 6.1 Security Analysis

##### 6.1.1 Attack by Hacker

- Packet Sniffing (Password Guessing Attack)

All the data is transferred over the SSL in our system. SSL can be cracked with password guessing attack. However, if the Key of SSL Key is 128 bits, it is almost impossible to crack the SSL communication. Unfortunately, even if SSL is cracked within a very short time by chance, the attacker has another challenge to decrypt the message or token also decrypted with other algorithm. It means that Packet sniffing is impossible against our system while Kerberos Ticket and message are vulnerable against offline password guessing attack.

- Replay Attack

Replay attack in Kerberos between client and server which intercepts Ticket for server and reuse it later seems to be impossible because of time stamp in the ticket. However, if two systems' time clocks are different, it could be possible. Besides, according to Kasslin and Tikkanen's research, Replay attack exists against SMB and Kerberos 5 on a Windows domain [28].

In NNCC, even without time stamp, replay attack is impossible because Payment token is for the single use. Payment token has transaction number on it, so after merchant sends it to the PGS and PGS processes it, the token with the transaction number is invalid.

- Database stealing

Merchants' databases do not have any card information, so an attack to such merchant is useless. Attacker also will target our server. We delete card information in the database when user's session token expires. So, attacker cannot get much card information from the database. Also, each card's information is divided to several data unit; each unit is encrypted before being saved to the database. These will make very difficult for attacker bring customers' complete card information within short time.

#### 6.1.2 Fraud by Customer

A customer can cheat a merchant. The key  $K_{PGS,M}$  might be guessed by clients because the key is fixed. In this case, customer requests Payment Token with less money than merchant ask for, and then send it merchant after changing the amount in the token to the amount merchant asked. We can solve this problem easily. If a merchant sends transaction number with the amount in the token to the PGS, we can know if the payment Token is modified or not.

#### 6.1.3 Fraud by Merchant

It is impossible in our system because merchant does have customer's credit card number and does not decide the amount of money in the payment Token.

### 6.2 Communication Cost Comparison

In this section, we show the communication cost (size of data transferred) of our system compared to current SSL based payment system. We ignore the overhead from the SSL and other data. We just focus on the data needed to card transaction. We assume that the amount of data between Payment Gateway and Merchant is at least 400 bytes as the result of searching some payment Gateway libraries [29].

### 6.2.1 Client

Table 7 Client's Communication Cost in NNCC

	From Client	To the Client	Total
PAS	32 (card) + 32 (etc)	32 (Session Token)	96 bytes
PGS	32 (Session Token)	64 (Token)	96 bytes
Merchant	64 (Token)	8 (confirmation )	72 bytes
Total			264(*168 ) bytes

\*If a customer has a session, he or she does not need to connect PAS.

Table 8 Client's Communication Cost in Current System

	From Client	To the Client	Total
Merchant	32 (card information)	8(Confirmation)	40 bytes

As you shown in Table 8, client on the SSL-based system (40bytes) is more efficient than ours (264 bytes). However, the size of data is lower than 1k bytes. In other world, this difference does not make any inconvenience to customers.

### 6.2.2. Merchant

Table 9 Communication of Merchant in NNCC

	From Merchant	To the Merchant	Total
Client	8 bytes	64 bytes	72 bytes
PGS	16 bytes	8 bytes	24 bytes
			96 bytes

Table 10 Merchant’s Communication Cost in Current System

	From Merchant	To the Merchant	Total
Client	8 bytes	32 bytes	40 bytes
Payment Gateway			400 bytes
			440 bytes

In merchant sides, our system (96 bytes) is much more efficient than current existing system (440 bytes).

### 6.2.3 Payment Gateway (PGS)

Table 11 Communication Cost of PGS in NNCC

	From PGS	To PGS	Total
Client	64 bytes	32 bytes	96 bytes
Merchant	8 bytes	16 bytes	24 bytes
			120 bytes

Table 12 Payment Gateway’s Communication Cost in Current System

	From P.G	To P.G	Total
Merchant			400 bytes

Even if our PGS have to process two requests for one transaction, the total amount of data for PGS is much less than payment gateway in the current system.

### 6.3 Client (User) Input Cost Comparison

In this section, we will show that the number of key input in our system is less than in Current Payment system in the payment process.

Indicate user input

**Current System**

*Merchant's Site*

Amount	\$ xxx.xx	
Name	①	
Card Number	②	16 digit
Expiration	③	4 digit
Security Code	④	4 digit

**Submit**

Key Input : At least 34 chars

Figure 21 Key Inputs under Credit Card Payment Model

As you know, we need to give card holders name, card number, expiration date, and 3 digit security codes in the current SSL-based payment system. Sometimes merchant also ask 5 digit zip codes. If we assume our name consists of 10 characters, we need to type at least 34 characters.

Indicate user input

**Our New System**

*Token Generator*

Merchant ID	①	At most 8 digit
Amount	②	At most 6 digit

**Get Token** ← *Generate Token & copy it to the Clipboard.*

*Merchant's Site*

Amount	\$ xxx.xx
Merchant ID	www.shop.com
Payment Token	Ctrl+v

**Submit** ↑ *Paste Token here From the Clipboard*

Key Input At most 14 chars

Figure 22 Key Inputs under NNCC

In our system, we only need merchant ID and Amount of money to make a payment in case we have valid session token. Besides, we can give payment token to the merchant

with just one key input because token is copied into clipboard. From this, we just need at most 14 characters input to make payment. If we do not have valid session token, we need to type 23 characters more which make total input 37 characters.

#### 6.4 Processing Cost

Server should allow concurrent users to request service for the customers' satisfaction. PAS and PGS are also designed to process multiple requests concurrently. Each request will hold server resource and compete with other requests, which delays response time to each client (customer). Processing time in PAS or PGS divided into three parts – communication time, time to perform cryptography algorithm, and the payment processing time.

In this section, we will show the average processing time of PAS and PGS for each request. We execute client, PAS and PGS at the same server during the measurement process. We found that when number of concurrent user increase, the processing time also increases. However, even if the processing time increases, it will not give customers inconvenience because the increased time is still less than 0.1 second. If we separate PAS, PGS and client, we will get better performance.

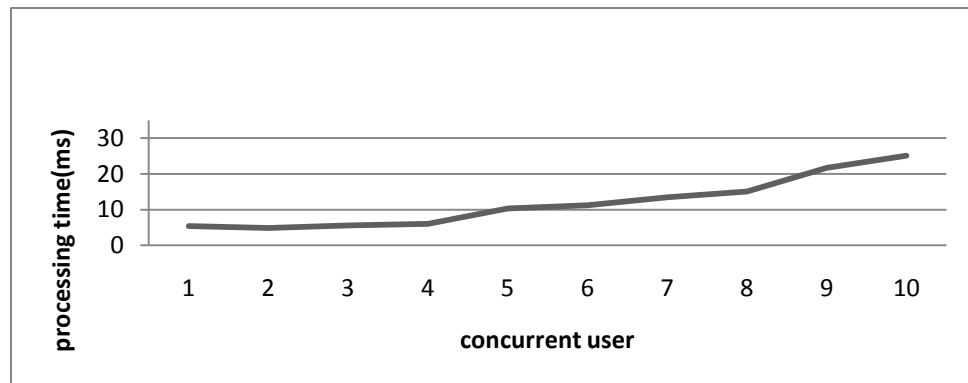


Figure 23-A Average Processing Time per Request in PAS



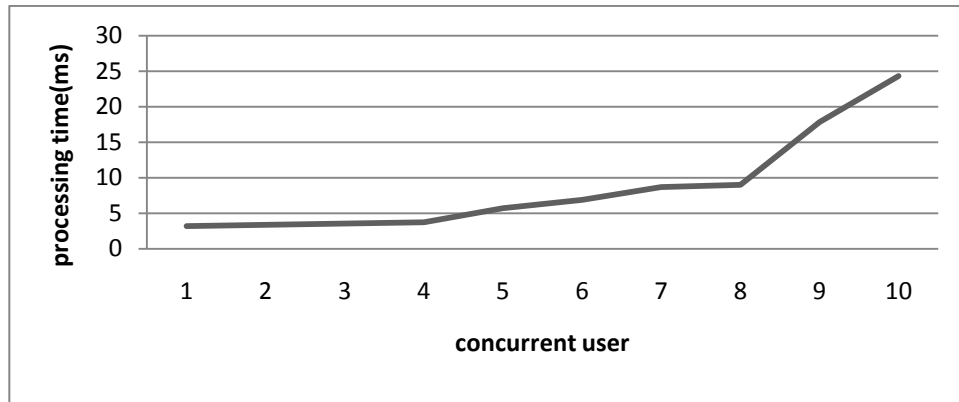


Figure 23-B Average Processing Time per Request in PGS

## CHAPTER 7

### CONCLUSION AND FUTURE WORK

#### 7.1 Conclusion

It is obvious that if we do not provide credit card numbers to the merchants during the electronic payment, we can tremendously decrease the number of credit card fraud. To acquire this goal, we implement new payment system (called NNCC) based on 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 modification of current payment workflow by sending token to the Payment Gateway instead of credit card information in the merchant side. From this, we can conclude that NNCC is secure and reliable credit card payment system which can be easily implemented in the current Electronic Payment System.

#### 7.2 Future Work

We show that the proposed system works practically in the current card payment environment. However, to be widely used, we need to implement NNCC in the distributed environment in order to process millions of transactions in a second and to provide more convenient user interface to the customers.

## REFERENCES

1. Hacking Suspect's Lawyer Criticizes Federal Prosecutors  
<http://bits.blogs.nytimes.com/2009/08/19/accused-hackers-lawyer-criticizes-federal-prosecutors/> (accessed: 2010 Feb )
2. A Chronology of Data Breaches  
<http://www.privacyrights.org/ar/ChronDataBreaches.htm#2009> (accessed: 2010 Feb)
3. IC3 2008 Annual Report on Internet Crime Released :  
<http://www.ic3.gov/media/2009/090331.aspx> (accessed: 2010 Feb )
4. SSL 3.0 Specification  
<http://www.freesoft.org/CIE/Topics/ssl-draft/3-SPEC.HTM> (accessed: 2010 Feb)
5. Visa International and Mastercard International, SET Secure Electronic Transaction Specification Book 3: Formal Protocol Definition, May 1997.
6. Andrés Guadamuz González, PayPal: the legal status of C2C payment systems  
Computer Law & Security Report Volume 20, Issue 4, July-August 2004, Pages 293-299
7. Abrazhevich-Dennis, Classification and Characteristics Of Electronic Payment systems, Lecture Notes in Computer Science 2115, 2001, pp. 81-90
8. Dennis Abrazhevich : Electronic payment systems: a user-centered perspective and interaction design (ISBN 90-386-1948-0)
9. Medvinsky, G. and Neuman, B.C. Netcash: A design for practical electronic currency on the internet. In Proceedings of first ACM Conference on Computer and Communication security (1993) 102-196.
10. Glassman, S., Manasse, M., Abadi, M., Gauthier, P., and Sobal-varro, P. The Millicent Protocol for inexpensive electronic commerce. In Proceedings of the 4th WWW Conference pp. 603-618, O'Reilly, 1995.
11. RL Rivest, A Shamir : PayWord and MicroMint: Two simple micropayment schemes : Lecture Notes in Computer Science, 1997 – Springer
12. Gennady Medvinsky , Clifford Neuman, NetCash: a design for practical electronic currency on the Internet, Proceedings of the 1st ACM conference on Computer and communications security, p.102-106, November 03-05, 1993, Fairfax, Virginia, United States
13. e-cash : David Chaum, Blind signatures for untraceable payments, Advances in Cryptology - Crypto '82, Springer-Verlag (1983), 199-203

14. Benjamin Cox, JD Tygar, and Marvin Sirbu. NetBill security and transaction protocol. In Proceedings of the First USENIX Workshop in Electronic Commerce, pages 77-88, July 1995
15. BC Neuman, G Medvinsky: Requirements for network payment: The netcheque perspective : IEEE Proceedings of Comcon '95: Technologies for the Information Superhighway, Digest of Papers, pp. 32-36
16. Sorkin, op cit; pp.11-12.
17. Online Payment Processing :  
[https://www.verisign.com/stellent/groups/public/documents/white\\_paper/001879.pdf](https://www.verisign.com/stellent/groups/public/documents/white_paper/001879.pdf)  
(accessed: 2010 Feb )
18. sslstrip  
<http://www.thoughtcrime.org/software/sslstrip/> (accessed: 2010 Feb )
19. Black Hat : Hacking SSL with sslstrip  
<http://blog.internetnews.com/skerner/2009/02/black-hat-hacking-ssl-with-ssl.html>  
(accessed: 2010 Feb )
20. A Shamir:: Secureclick: A web payment system with disposable credit card numbers  
Lecture notes in computer science, 2002 – Springer
21. 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](http://www.streetdirectory.com/travel_guide/149328/credit_cards/how_to_use_a_one_time_credit_card.html)
22. Y. Li and X. Zhang. Securing credit card transactions with one-time payment scheme.  
Electronic Commerce Research and Applications, 4(4):413–426, 2005
23. DR Stinson: Cryptography: theory and practice
24. mdecrypt : <http://mdecrypt.sourceforge.net> (accessed: 2010 Feb )
25. BC Neuman, T Ts'o - Kerberos: An authentication service for computer networks :  
IEEE Communications magazine, 1994
26. Lecture note – Internet Security Class at UNLV by Professor Yoohwan Kim
27. OPENSLL - The Open Source toolkit for SSL/TLS  
<http://www.openssl.org/> (accessed: 2010 Feb )
28. Kimmo Kasslin, Antti Tikkanen : Replay Attack on Kerberos V and SMB  
<http://users.tkk.fi/autikkan/kerberos/> (accessed: 2010 Mar )

29. Payment Gateway Example :

<http://www.netregistry.com.au/support/kb/questions.php?questionid=74> (accessed:  
2010 Feb )

VITA  
Graduate College  
University of Nevada, Las Vegas

Jung Eun Kim

Degree:

Bachelor of Engineering in Computer Engineering, 1999  
Kyungil University, South Korea

Thesis Title: A Secure on-line credit card transaction method based on Kerberos  
Authentication protocol

Thesis Examination Committee:

Chairperson, Dr. Yoohwan Kim, Ph.D.  
Committee Member, Dr. Ajoy K Datta, Ph.D.  
Committee Member, Dr. Laximi Gewali, Ph.D.  
Graduate Faculty Representative, Dr. Ju-Yeon Jo, Ph.D.