Enhancing the Security of Credit Card Transaction based on Visual DSC

Kuo-Jui Wei, Jung-San Lee* and Shin-Jen Chen
Information Engineering and Computer Science, Feng Chia University
Taichung, Taiwan, ROC
[e-mail: leejs@fcu.edu.tw]
*Corresponding author: Jung-San Lee

Received December 9, 2014; revised February 4, 2015; accepted March 1, 2015; published March 31, 2015

Abstract

People have transferred their business model from traditional commerce to e-commerce in recent decades. Both shopping and payment can be completed through the Internet and bring convenience to consumers and business opportunities to industry. These trade techniques are mostly set up based on the Secure Sockets Layer (SSL). SSL provides the security for transaction information and is easy to set up, which makes it is widely accepted by individuals. Although attackers cannot obtain the real content even when the transferred information is intercepted, still there is risk for online trade. For example, it is impossible to prevent credit card information from being stolen by virtual merchant. Therefore, we propose a new mechanism to solve such security problem. We make use of the disposable dynamic security code (DSC) to replace traditional card security code. So even attackers get DSC for that round of transaction, they cannot use it for the next time. Besides, we apply visual secret sharing techniques to transfer the DSC, so that interceptors cannot retrieve the real DSC even for one round of trade. This way, we can improve security of credit card transaction and reliability of online business. The experiments results validate the applicability and efficiency of the proposed mechanism.

Keywords: Dynamic security code, SSL, visual secret sharing, e-commerce

http://dx.doi.org/10.3837/tiis.2015.03.022 **ISSN : 1976-7277**

1. Introduction

E-commerce techniques grow dramatically fast nowadays, and people have changed their business model from the traditional marketing mode to the e-commerce mode. Thus, consumers are able to make purchase whenever and wherever as long as they can access the Internet without having to stand in line to save time. Besides, visual environment techniques have made shopping online more realistic. With the lead of yahoo, eBay, and Alibaba, e-commerce has become a good business model, which benefits more than ten billions with consumers more than ten millions . Among these businesses, most people use credit cards to make purchase transactions [1-5]. Thus, protecting privacy of individuals and their credit cards information is of high priority.

So far, there have been lots of secure e-commerce protocols to protect individual information through the Internet. Secure Sockets Layer (SSL) and Secure Electronic Transaction (SET) techniques obtain the most favor [6-8]. These two techniques employ secure cryptography method to protect information, including symmetric-key cryptography, public-key cryptography, and hash function. SET is the most secure online business mechanisms among current protocols. To satisfy this protocol, however, basic public-key cryptosystem needs to be set up first, which is very difficult in real. This is the main reason why it has not been widely used. Comparing with SET, SSL is easy to set up and thus most current e-commerce mechanisms have employed this protocol for credit card transactions.

SSL helps to identify individuals through asymmetric-key cryptography mechanism and encrypt information with symmetric-key system. It sets up the secure socket within insecure environment to protect the transferred information through the handshake. The handshake process is shown as **Fig. 1**.

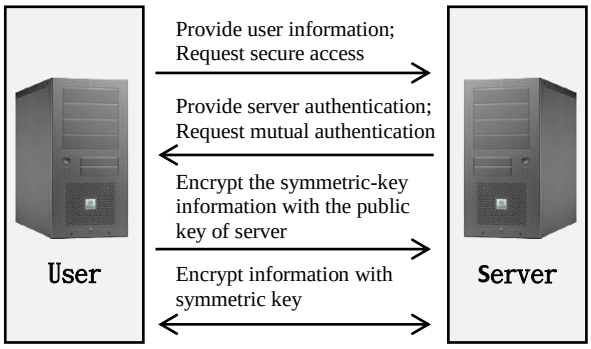

Fig. 1. Handshake work-flow

In this protocol, only the authenticated user and server can obtain symmetric key. So even the information is intercepted during transferring, no secret information can be disclosed without knowing the symmetric key. However, there are still insecure factors for applying SSL to online payment. The main reason is that the seller needs to know the credit card number and card security code (CSC). Actually, researchers have developed different kinds of one time password (OTP) technology to settle this problem. These mechanisms could be divided into two types. The first one is based on the short message system (SMS) [9-10], and

the other is device-oriented [11-12]. In the SMS based OTP mechanism, the user needs to send a request to bank to ask for an OTP, and then the bank will send a random password to user through a SMS. As to the device-oriented mechanism, the user obtains an OTP from a specific device allotted by the bank. Though these two types can solve the misusing problem, they have brought heavy cost to the device or SMS transmission. Even, the SMS based one may suffer from the malware problem [13-14]. Therefore, we propose a new online payment mechanism to eliminate these problems.

Our proposed mechanism focuses on the design and modification of the CSC. The original CSC is a set of static number, which is only known to the bank and owner of the card. CSC is generated by certain algorithm with the information of credit card number, expire date, and other available credit data. The purpose of CSC is to help authenticate the card holder, which is printed on the credit card and easy to be obtained by attackers.

In our proposed method, we aim to use Dynamic Security Code (DSC) and employ the visual secret sharing (VSS) technique to print the random base transparence on credit card instead of the traditional CSC. Thus, when users need to use security code to make transactions, they will make request to the bank through their registered mobile device. Once the bank receives the request, it randomly picks a DSC to get the sharing image together with the information of base transparence on the current user's card and send this image to the corresponding user. When users receive this sharing image, they can easily stack it with their own base transparence and obtain the DSC for the current transaction round.

Herein, the security of online purchase is guaranteed and even if the seller gets the current security code, it is still invalid to use the code during next purchase. Besides, along with the growing popularity of mobile devices, such authentication is feasible and helps to provide the multi-authentication schema, which is the trend of online payments. The rest of this paper is organized as below. Section 2 provides a brief introduction of the VSS technique. Detailed description of the whole payment system is provided in Section3, followed by the experiments and results in Section 4. Section 5 gives out the analysis and proof for a list of potential attacks, and we conclude our work in Section 6.

2. VSS Technique

Visual Secret Sharing (VSS) [15-25] is a kind of technique that hides secret first and partitions it to multi-parties for sharing purpose. This technique can be dated back to 1979, Shamir [15] and Blakley [16] provide two different (t, n) secret sharing mechanisms, respectively. This technique first divides a secret into n parts and partitions the n parts to n individuals. When k $(k \geq t)$ individuals stack their sharing together, the original secret will be disclosed. Compared with traditional secret sharing, VSS makes use of visual information instead of complex algorithm to improve applicability and efficiency.

The key for VSS is to embed a secret into two or more meaningless, noise-like images, from which human beings cannot detect the existence of secrets, so as to guarantee the security of the images during transferring. During the deciphering process, VSS needs not to rely on dense computation as traditional cryptography, but to stack the designed images together to obtain the secret within them, whose framework is shown as in **Fig. 2**.

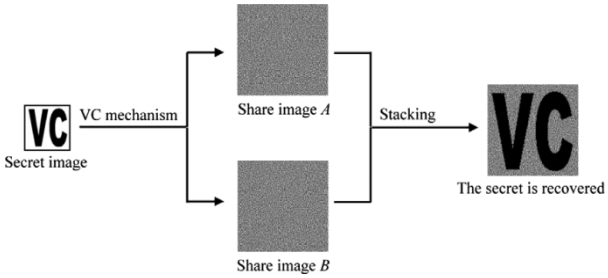

Fig. 2. Concept of VSS

There are two methods to implement VSS, polynomial based visual cryptography (VC) and random grids (RG). VC is proposed by Naor and Shamir [17]. In their (t, n)-VSS method, n sharing images will be generated, each of whose pixels will be enlarged into $m \times m$. Therefore the generated image will also be $m \times m$ times of the original one. In 1987, Kafri and Keren [18] proposed the RG-VSS method using random grid concept. In their method, each image is a two dimension matrix made up with transparent (white) and opaque (black) pixels, where there is no relationship among pixels. The RG_VSS algorithm is described as below.

Input：A binary image S $(M \times M)$

Output：Base transparence R_1 and sharing image R_2

Step1. Randomly generate a matrix R_1，

where $R_1(i, j) \in \{0,1\}$

Step2. For each $(i, j) \in \{(i, j) \mid 1 \le i \le M, 1 \le j \le M\}$，

compute $R_2(i, j) = \begin{cases} R_1(i, j) & \text{, if } S(i, j) = 0 \\ \overline{R_1(i, j)} & \text{, otherwise} \end{cases}$

Step3. Output two sharing images R_1 and R_2

When the two images R1 and R2 are stacked together, the secret will be disclosed, which can be represented as $R_1 \| R_2 = R_1(i, j) \| R_2(i, j)$.

In our proposed method, in order to improve the security of credit card, we apply the RG-VSS method provided by Kafri and Keren to generate base transparence. Whenever consumers need to use the online credit card payment, they will request the sharing image for current payment round from the bank and stack the obtained sharing image with their own base transparence to retrieve the DSC for that round of purchase. In the experiment, we put a random anti-counterfeited line into the original DSC image. When the user stacks out the DSC image, he/she can decide to ignore this round depends on the result of this line.

3. System Structure

In the credit card system, we employ visual cryptography to make DSC and replace the traditional CSC with it to improve reliability of the online payment mechanism. There are four phases for the whole proposed mechanism, registration phase, login phase, request phase, and transaction phase. In this system, the bank is assumed to keep the information of user rigorously, and no one can steal sensitive information from the bank. Notations used in the proposed mechanism is shown in **Table 1**.

Table 1. Notations table

Notations	Definition
ID	Last four digit number of the card holder identity
BD	Birthday of card holder
Phone	Phone number of card holder
PI	Personal information, including *ID*, *BD* and *Phone*
CN	Credit card number
VT	Expire date of the credit card
CardInf	Credit card information, including *CN* and *VT*
IMEI	International mobile station equipment identity
PFValue	Authentication information to represent successful/fail authentication
DSC	Dynamic security code
n	Random number
$image_{base}$	Credit card base transparence
$image_{share}$	Sharing image
$image_{DSC}$	Image with *DSC* $image_{DSC} = image_{base} \| image_{share}$
TransInf	Transaction information

3.1 Registration Phase

A user should provide the registration application to bank through a secure channel. The flowchart is shown as **Fig. 3**.

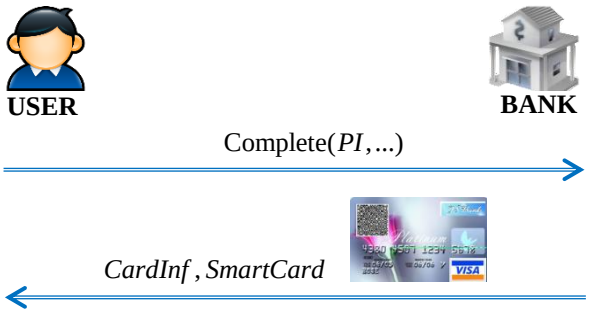

Fig. 3. Flowchart for the registration phase

Step 1. First, the user should fill out the registration form for credit card and send the form to bank. The form contains *ID*, *BD* and *Phone*.

Step 2. Once the bank receives the application form, it will make sure that the applier satisfies all the conditions. If the applier cannot meet all the requirements, the bank would reject the application. Otherwise, bank would generate a random base transparence $image_{base}$, and print it together with the *CardInf* on the back of credit card. The card is sent to applier then, and the $image_{base}$ is kept in the database.

3.2 Login Phase

When the user first requests DSC, he/she should download the application software (APP) for his/her mobile device and complete the login phase to pass the identity authentication. Also,

he/she would use $IMEI$ for the current mobile device to bind the credit card for one-time login. The flowchart of the login phase is shown in **Fig. 4**.

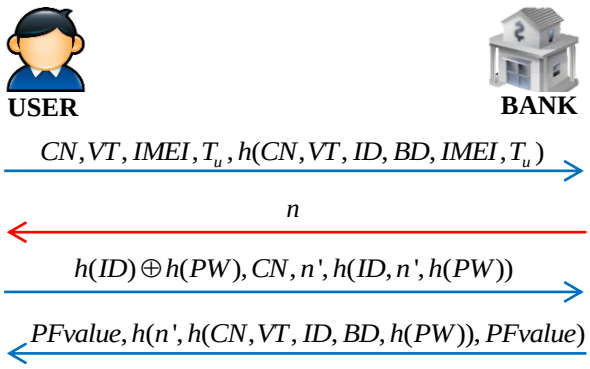

Fig. 4. Flowchart for the login phase

Step 1. The user downloads the APP and installs it in the mobile device. Then the user enters the information of CN , VT , ID and BD to the APP. After login, the APP would automatically retrieve $IMEI$ of the mobile device and calculate a hash value with above information as $h(CN, VT, ID, BD, IMEI, T_u)$. Then CN , VT , $IMEI$ and T_u would be sent to the bank with the hash value.

Step 2. Once the bank receives the logging information, it will check whether VT and T_u are valid. It then applies CN to compute $h(CN, VT, ID, BD, IMEI, T_u)$ and compares the result with the one received. If they are not the same, the login request would be rejected. Otherwise, the bank will generate a random number n and inform the user with text message.

Step 3. When obtaining n, the user will enter it into the mobile device and set a sequence of password for future protection. Here we denote the random number and password as n' and PW . After the user finishes the entering, the mobile device can calculate $h(ID) \oplus h(PW)$ and $h(ID, n', h(PW))$. Then, the hash values are sent to the bank along with CN and n' .

Step 4. Upon the bank gets above information, it will compare if n' equals to n and employ $h(ID) \oplus h(PW)$ to retrieve $h(PW)$ to calculate $h(ID, n, h(PW))$. If the authentication information is correct, the bank would link CN to $h(PW)$ and the $image_{base}$ stored in its database for future request usage. Otherwise, the bank would reject the login. No matter if the login phase is successful, the bank would compute a hash value $h(n', h(CN, VT, ID, BD, h(PW)), PFvalue)$ and send it to the user together with the authentication information $PFValue$ to notify the status of login.

3.3 Request Phase

Whenever the user wants to perform online payment, he/she can request DSC from the bank with the mobile device as long as the mobile device finished the login session in Step 2. The flowchart for request phase is shown in **Fig. 5**.

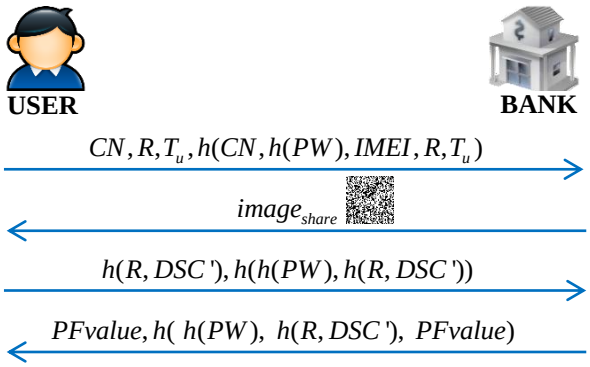

Fig. 5. Flowchart for the request phase

Step 1. The user enters the password and forwards a request. The mobile device generates a random number R and timestamp T_u. Afterwards, the device computes the hash value $h(CN, h(PW), IMEI, R, T_u)$ with CN and other information. Finally, it sends CN, R, T_u and the hash value to bank.

Step 2. Once the bank receives the information and validate T_u. It searches the corresponding $IMEI$, $h(PW)$, and $image_{base}$ from the database based on CN, and calculates the hash value to compare with the one received. If the hash values hold the same, the bank will generate $image_{DSC}$ and employ $image_{base}$ to generate $image_{share}$ according to VSS. Then it sends the result back to user.

Step 3. After the user gets $image_{share}$, he/she can stack it with $image_{base}$ which is printed on the credit card to obtain $image_{DSC}$. From $image_{DSC}$, the user can retrieve DSC and enter it to the mobile device. Then the mobile device computes the two hash values $h(R, DSC')$ and $h(h(PW), h(R, DSC'))$ and transfers them to the bank for authentication purpose.

Step 4. When the bank receives two hash values, it starts to validate them. As long as one of the value does not match, the bank will reject this DSC and record the number of failure times. If the user fails continuously for three times, this card will be locked. Otherwise, the DSC will be enabled. No matter what if the request gets approved, the bank will compute $h(h(PW), h(R, DSC'), PFvalue)$ and send it to the user together with the authentication information $PFValue$ to notify the request status.

3.4 Transaction Phase

After the bank authorized the DSC, the user can complete his/her online payment through SSL website with credit cards. The flowchart of the purchase is shown as **Fig. 6**.

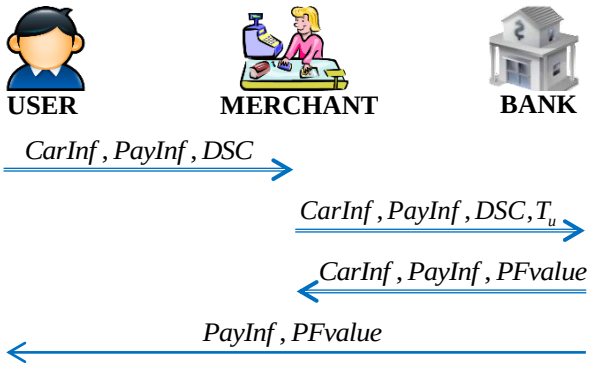

Fig. 6. The flowchart of transaction phase

Step 1. After shopping online, the user proved the *CardInf* , *PayInf* and retrieved *DSC* to the seller for payment purpose.

Step 2. Once the seller gets and authenticates the transaction information, it will generate a timestamp T_u and transfer all the information to bank through a secure channel for authorization.

Step 3. Upon the bank obtains the information, it will authenticate T_u and *DSC* . If one of them cannot meet the condition, the authorization is failed. Otherwise, the authorization passes and the bank will transfer *CardInf* , *TransInf* and *PFvalue* to notify the seller about the authorization results.

Step 4. No matter if the authorization is success, the bank will send the user current transaction information and result to protect the legal right of user.

4. Experiments and Results Analysis

To prove the feasibility of the proposed mechanism, we deploy this system to Android mobile device and execute 1,500,000 rounds to get the statistical analysis of its efficiency. We use PC to simulate the bank server. CPU used for the server is Intel Core i7-2600 Quad-Core Processor 3.4GHz with 8GB RAM. The operating system is Window 7 with 64bit. The mobile device is HTC Sensation XE, with CPU Qualcomm MSM8260 Dual-core, 1.5 GHz and 768MB RAM. The operating system for mobile device is Android 4.0.3. Detailed implementation and experiments analysis will be described as below.

4.1 System Implementation

We assume that the user has finished the registration with the bank and been possessed of credit card with base transparence on the back as shown in **Fig. 7**. The dash lines on both the upper and lower bounds are localization axes used by the software to localize the coordinates.

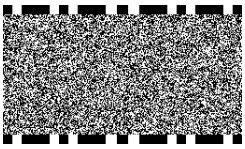

Fig. 7. User base transparence

The system implementation can be divided into two parts. The first is the login phase that a user has to download and setup the software at the beginning. The second part is that a user requests DSC from bank.

4.1.1 Login Phase Implementation

When starting the software, the APP will ask the user to enter basic information, including credit card number, expire date, identity number, and birthday as in **Fig. 8**. Then APP will perform the preliminary authentication to check if the information format is correct, for example, the credit card number and expire date. When one of them is incorrect, the APP will show the error for the entered information. Otherwise, APP will compute the values for login phase and send it to server for authentication.

Once the server receives the information, it will retrieve the user related information from its database and compare it with the one obtained. If they are the same, it will generate the authentication code and notify the user with text message. The APP will automatically guide to the authentication page as in **Fig. 9**. Otherwise, the user will get the error message as shown in **Fig. 10**.

After the user gets the text message, he/she will enter the authentication code and set his/her own password. Then the APP will enter step 3 and compute the hash value and transfer it to server for validation. If the authentication fails, the verified code error will be sent back as in **Fig. 12**. And, the user needs to re-login. Otherwise, APP will automatically jump to the request page for user as **Fig. 13**.

Fig. 8. First time login interface

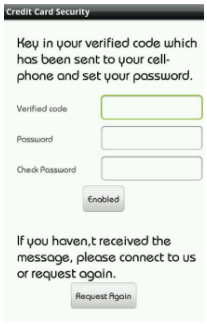

Fig. 9. Authentication code and password set interface

Fig. 10. User error interface

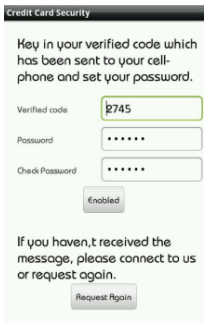

Fig. 11. Authentication code and password verification

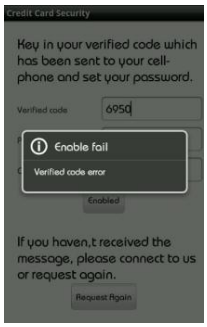

Fig. 12. Authentication failure interface

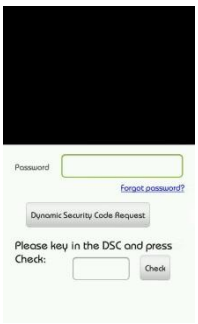

Fig. 13. DSC request interface

4.1.2 Request Phase Implementation

A user only needs to login and set up the password for the first time. After that, he/she can enter the request phase directly without bothering about the complex credit card information. He/she can just key in the password and click the "DSC Request" button, and then the mobile device will automatically compute values used by the first step of Request phase and send it to the server.

When the server receives request, it will authenticate the information accepted. If the authentication successes, server will generate four digits DSC and random picks the character style to make with anti-counterfeited line as shown in **Fig. 14**.

After the server generates $image_{DSC}$, it will employ $image_{base}$ of the user to generate $image_{share}$ based on RG-VSS technique and send the result to user as shown in **Fig. 15**.

Fig. 14. DSC example.

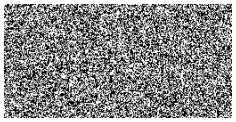

Fig. 15. Sharing image with DSC.

When the user gets $image_{share}$, APP will automatically start webcam and detect the localization axes as in **Fig. 16**. According to the localization axes and size, $image_{share}$ can be accurately stacked as shown in **Fig. 17**. If any attack occurs on the extracted sharing image, the anti-counterfeited line is used to help user recognize it. **Fig. 18**. illustrates an example.

Otherwise, the user can enter the DSC content to the field according to the stacked images. After pushing "Check", APP will automatically send out the authentication information to server. The DSC will be enabled only if it is the same as the one bank issued. No matter what result we have, the server will send back the authentication acknowledgement. Furthermore, the user has to use the DSC within a pre-setting expired date. Otherwise, he/she needs to send the request again.

Fig. 16. Automatic detection after receiving sharing image.

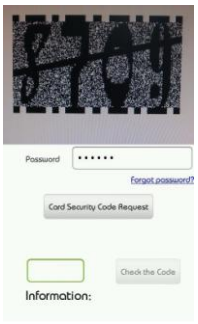

Fig. 17. Stacking with the base transparence.

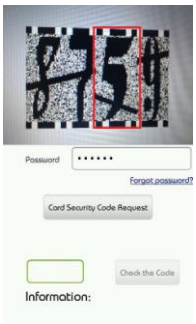

Fig. 18. Stacking the base transparence with a tampered share.

4.2 Experiments results

Considering the feasibility of mobile device, we executed 1,500,000 rounds and collected the statistical results for authentication and image generation time. **Table 2** and **Table 3** show the execution time for login and request phase, respectively. We can see that average time for login is 1.4 *ms*, while that of the request phase including generating sharing images takes about

8.2 *ms*. This shows that the proposed schema will not gain computation burden for mobile device.

T1: Time of computing the hash value for the first step of login phase by mobile device
T2: Time of hash value authentication for the first step of login phase on server side
T3: Time of computing the hash value for the third step of login phase by mobile device
T4: Time of hash value authentication for the third step of login phase on server side
T5: Time of computing the hash value for the fourth step of login phase by mobile device
T6: Time of computing the hash value for the first step of request phase by mobile device
T7: Time of hash value authentication for the first step of request phase on server side
T8: Time of generating sharing image with base transparence on server side
T9: Time of hash value authentication for the third step of request phase on server side
T10: Time of hash value authentication for the third step of request phase on server side
T11: Time of hash value authentication for the fourth step of request phase by mobile device

Table 2. Time for login phases

Time (*ms*)	Maximum	Minimum	Average	Standard deviation
T1	3.0518	0.7019	0.9515	0.1815
T2	0.0201	0.0012	0.0016	0.0012
T3	1.2512	0.1831	0.2409	0.0885
T4	0.0246	0.0015	0.0021	0.0009
T5	0.8240	0.1831	0.2191	0.0403
Total	5.1717	1.0708	1.4153	

Table 3. Time for request phases

Time (*ms*)	Maximum	Minimum	Average	Standard deviation
T6	4.3030	0.7019	0.9198	0.2325
T7	0.0120	0.0012	0.0016	0.0007
T8	8.0372	6.5013	6.9439	0.2754
T9	0.7324	0.1221	0.1687	0.0566
T10	0.0102	0.0006	0.0008	0.0005
T11	0.9155	0.1526	0.1917	0.0676
Total	14.0104	7.4796	8.2265	

5. Security Analysis

This proposed mechanism can not only achieve the requirements of mutual authentication, double key factors authentication, dynamic secure code characters, but also withstand malicious attacks, including the server spoofing attack, impersonation attack, replay attack, guessing attack, and image tampering attack. The security analysis is based on the properties of hash function [26] described as below.

Property 1. Pre-image resistance: given $Y = H(M)$, it is hard to find M' that $Y = H(M')$.

Property 2. The second pre-image resistance: given M and $H(M)$, it is hard to find $M \neq M'$, so that $H(M) = H(M')$.

Property 3. Collision resistance: it is hard to find a set of $M \neq M'$, so that $H(M) = H(M')$.

5.1 Mutual Authentication

During the login phase, a user must hold personal information, including identity number and birthday date to compute the hash value for the first step. By providing with these information, the bank is able to authenticate the user identity. Besides, a user can authenticate the identity of the bank during the fourth step. It is because that the user uses the hash value to verify the

personal information during the login phase. According to the properties of hash function, if the bank is able to calculate the hash value containing correct personal information, it indicates that the bank indeed holds true personal information. Based on the two statements above, the mutual authentication during login phase is proved.

During the request phase, a user needs to enter correct password as well as the sharing image to disclose the unified DSC and pass the authentication for step 3, which help to authenticate the user identity. In step 4, the bank has to provide correct hash value for the password, which is only known to the authorized user and bank. Therefore, the user can also authenticate the identity of back in this way. Based on the content stated above, we can guarantee the mutual authentication for the request phase.

5.2 Double Key Factors Authentication

After the user successfully logins the system to request DSC from the bank, he/she needs to enter the correct password to achieve identity authentication. After obtaining the sharing image, he/she also needs to hold the legal base transparent to stack out the DSC image to get through the authentication. Herein, a user needs not only to know the password, but also to hold the credit card with base transparence on it. Thus, this mechanism can achieve the double key factors authentication.

5.3 Dynamic Secure Code

Different with the traditional credit card mechanism, we apply the DSC to replace CSC. The main advantage is that DSCs must be different for each session of payment. Therefore, even the current secure code is retrieved, it will not threat the future transaction. Besides, in our implementation, the DSC has certain feasible time range to keep its security.

5.4 Server Spoofing Attack

According to the third property of hash function, if there is any difference between two inputs, there will be huge difference for the outputs. In this mechanism, if an attacker pretends to be the server to cheat users, he/she must feed the correct user information into hash function during login and request phase. However, the personal information is only known by the user and authorized bank, so the attacker cannot get valid hash value to pass authentication phase. Therefore, this mechanism can withstand the server spoofing attack.

5.5 Impersonal Attack

In order to avoid users being impersonated, they are required to enter the identity number, birthday date, and other personal information during login phase and password in the request phase. Under the first assumption of hash function, attackers cannot retrieve anything useful from the information transferred between user and bank. Thus, we can be sure that impersonal attack cannot be achieved in this mechanism.

5.6 Replay Attack

With the proposed mechanism, attacker cannot retrieve information to perform the replay attack. As there is timestamp in the first authentication step for both login and request phases, which is also fed into the hash function, attacker cannot generate new timestamp to play resending request.

5.7 Guessing Attack

To avoid malicious attackers using guessing attack to hack the system, we have added the upper bound for error accepts. When the number of failures meets the limitation, the DSC function of user will be locked. If an attacker intends to perform guessing attack during the login phase, both personal information authentication and text message authentication code errors can lead to system locked. During the request phase, both password error and DSC invalidation can result the lock up of the account. Once the system account is locked up, users need to contact the bank to reactive it. So this mechanism can perfectly reject the guessing attack.

5.8 Image Tampering Attack

According to the anti-counterfeited line on the DSC image, it is easy for users to perceive whether the sharing image has been tampered or not. Once the user is aware of the fact that the anti-counterfeited line is not a straight one, the user will detect that the received sharing image has been modified; thus the proposed method can effectively resist the image tampering attack.

6. Conclusions

We have applied the DSC to replace traditional credit card CSC. This mechanism can effectively address problems come across for online payment with SSL. Besides, we make use of VSS to guarantee the DSC security during transferring and add the double key factors authentication to improve the security level and address online credit card payment risk. We also implemented and simulated the whole system with PC and mobile device to prove the feasibility of this method. With more powerful server in the real world, this system would improve its efficiency and security on a large scale.

References

[1] Y. Li, X. Zhang, "Securing credit card transactions with one-time payment scheme," *Electronic Commerce Research and Applications*, vol. 4, no. 4, pp. 413-426, 2005. Article (CrossRef Link)
[2] C.W. Chan, C.H. Lin, "A new credit card payment scheme using mobile phones based on visual cryptography," in *Proc. of the International Conference on Intelligence and Security Informatics*, vol. 5075, pp. 467-476, 2008. Article (CrossRef Link)
[3] S. Gupta, R. Johari, "A new framework for credit card transactions involving mutual authentication between cardholder and merchant," in *Proc. of the International Conference on Communication Systems and Network Technologies*, pp. 22-26, 2011. Article (CrossRef Link)
[4] F. Buccafurri, G. Lax, "Implementing disposable credit card numbers by mobile phones," *Electronic Commerce Research*, vol. 11, pp. 271-296, 2011. Article (CrossRef Link)
[5] I. Jain, R. Johari, R. L. Ujjwal, "CAVEAT: Credit Card Vulnerability Exhibition and Authentication Tool," in *Proc. of Security in Computing and Communications*. vol. 467, pp. 391-399, 2014. Article (CrossRef Link)
[6] A. Elgohary, T. S. Sobh, M. Zaki, "Design of an enhancement for SSL/TLS protocols," *Computers and Security*, vol. 25, no. 4, pp. 297-306, 2006. Article (CrossRef Link)
[7] R. Oppliger, R. Hauser, D. Basin, "SSL/TLS session-aware user authentication revisited," *Computers and Security*, vol. 27, no. 3-4, pp. 64-70, 2008. Article (CrossRef Link)
[8] M. A. Kaljahi, A. Payandeh, M. B. Ghaznavi-Ghoushchi, "TSSL: improving SSL/TLS protocol by trust model," *Security and Communication Networks*, 2014. Article (CrossRef Link)
[9] A. Varghese, D. Mathews, "Securing SMS-based approach for two factor authentication,"

International Journal of Research in Computer and Communication Technology, vol. 3, no. 3, 2014. Article (CrossRef Link)
[10] C. Mulliner, R. Borgaonkar, P. Stewin, J.P. Seifert, "SMS-based one-time passwords: attacks and defense," in Proc. of *Detection of Intrusions and Malware, and Vulnerability Assessment*. vol. 7967, pp. 150-159, 2013. Article (CrossRef Link)
[11] B. Vaidya, J. H. Park, S.S. Yeo, J. J. P. C. Rodrigues, "Robust one-time password authentication scheme using smart card for home network environment," *Computer Communications*, vol. 34, pp. 326-336, 2011. Article (CrossRef Link)
[12] W.B. Lee, T.H Chen, W.R. Sun, K. I. J. Ho, "An S/Key-like one-time password authentication scheme using smart cards for smart meter," in *Proc. of the Advanced Information Networking and Applications Workshops*, pp. 281-286, 2014. Article (CrossRef Link)
[13] C. Miller, "Mobile attacks and defense," *IEEE Security and Privacy*, vol. 9, pp. 68-70, 2011. Article (CrossRef Link)
[14] Y. Seungyong, K. Jeongnyeo, C. Hyunsook, "Detection of SMS mobile malware," in *Proceedings of the International Conference on Electronics, Information and Communication*, pp. 1-2, 2014. Article (CrossRef Link)
[15] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979. Article (CrossRef Link)
[16] G. R. Blakley, "Safeguarding cryptographic keys," in *Proc. of the National Computer Conference*, vol. 48, pp. 313-317, 1979. Article (CrossRef Link)
[17] M. Naor, A. Shamir, "Visual cryptography," in *Proc. of the Advances in Cryptology-EUROCRYPT 94*, LNCS. 950, pp. 1-12, 1994. Article (CrossRef Link)
[18] O. Kafri, E. Keren, "Encryption of pictures and shapes by random grids," *Optics Letters*, vol. 12, pp. 377-379, 1987. Article (CrossRef Link)
[19] M. Naor,B. Pinkas, "Visual authentication and identification," in *Proc. of the Advances in Cryptology — CRYPTO '97*, vol.1294, pp.322-336, 1997. Article (CrossRef Link)
[20] T.H. Chen, K.H. Tsao, "Visual secret sharing by random grids revisited," *Pattern Recognition*, vol. 42, no. 9, pp. 2203-2217, 2009. Article (CrossRef Link)
[21] J.Y. Chang, M.J. Li, Y.C. Wang, S.T. Juan, "Two-image encryption by random grids," *International Symposium on Communications and Information Technologies*, pp. 458-463, 2010. Article (CrossRef Link)
[22] T.H. Chen, K.H. Tsao, "Threshold visual secret sharing by random grids," *Journal of Systems and Software*, vol. 84, pp. 1197-1208, 2011. Article (CrossRef Link)
[23] Y.C. Hou, S.C. Wei, C.Y. Lin, "Random-grid-based visual cryptography schemes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, pp. 733-744, 2014. Article (CrossRef Link)
[24] X. Wu, W. Sun, "Random grid-based visual secret sharing with abilities of OR and XOR decryptions," *Journal of Visual Communication and Image Representation*, vol. 24, pp. 48-62, 2013. Article (CrossRef Link)
[25] K.S. Lin, C.H. Lin, T.H. Chen, "Distortionless visual multi-secret sharing based on random grid," *Information Sciences*, pp. 330-346, 2014. Article (CrossRef Link)
[26] A. J. Menezes, P. C. van Oorschot, S.A. Vanstone, *Handbook of applied cryptography*, pp. 321-376, 1996. Article (CrossRef Link)

Kuo-Jui Wei received the MS degree in information engineering and computer science in 2011. He is currently pursuing her Ph.D. degree in Information Engineering and Computer Science in Feng Chia University, Taichung, Taiwan. His current research interests include information security and mobile communications.

Jung-San Lee received the BS degree in computer science and information engineering in 2002 and his Ph.D in computer science and information engineering in 2008, both from National Chung Cheng University, Chiayi, Taiwan. Since 2012, he has worked as an associate professor in the Department of Information Engineering and Computer Science at Feng Chia University, Taichung, Taiwan. His current research interests include information security, image processing, and watermarking.

Shin-Jen Chen is currently pursuing his MS degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan. His current research interests include information security and visual secret sharing.