Friday , October 29, 2004

A Secure On-Line Software Transaction Scheme

Shiang-Feng Tzeng[1] Min-Shiang Hwang[2] Hsing-Bai Chen[3]

Department of Computer Science and Information Engineering[1]
National Central University
No. 300, Jung-da Rd., Jung-li City, Taoyuan, Taiwan 320, R.O.C.
sftzeng@csie.ncu.edu.tw

Department of Management Information System[2]
National Chung Hsing University
250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C
mshwang@nchu.edu.tw

Department of Information Management[3]
Chaoyang University of Technology
168, Gifeng E.Rd., Wufeng, Taichung County, Taiwan 413, ROC.
s9014604@mail.cyut.edu.tw

Abstract

Our main purpose in this paper is to present an effective and secure on-line software transaction protocol. On-line software transaction should meet some requirements such as security and non-repudiation. Our research will be mainly basing on public cryptosystem, digital signature and non-repudiation protocol to establish an on-line software transaction protocol.

Keywords: On-line software transaction, Public cryptosystem, Digital signature, Non-repudiation protocol

1. Introduction

Thanks to the advancement of information technology, which causes e-commerce to become flourishing, more and more commercial goods through the convenience of Internet can be provided with a comprehensive communication before a transaction is done [1]. These transactional goods through the Internet can be physical or nonphysical. Physical goods have to go through the traditional approach (i.e. delivery) of which it leads to higher cost due to the inventory pressure, packaging cost, distributing and what not before they reach the customers. Nonphysical goods are different, in which they can be sent through the Internet (ex. Software). Therefore, they are easier to be sent through the Internet than physical goods. Establishing a secure protocol for a transaction in which nonphysical goods are sent through the Internet is made may effectively build up the customers' confidence when they shop on-line, so higher profit can be obtained. Software is one such example of nonphysical goods. If software can be purchased through on-line transaction, costs (such as traditional packaging, distributing, and middleman) will be reduced effectively and profits will be earned substantially.

To make sure the software data transfer through the Internet to reach the certain place, we have to first discuss the characteristics of Internet:

The characteristics of Internet have the following:

(1) Publicity: Over the Internet, there is no need to know the user's identity, his location or application programs he uses. Each user can send his data or receive any information through the Internet.
(2) Anonymousness: We may occasionally obtain the user's IP address while sending or receiving data but we are unable to get other information about the user, unless some special approaches are taken.
(3) Diversity: Anyone can in any form develop or do anything because Internet is public, so that the information contained in the Internet is quite diverse.
(4) Non-geographical limitation: Internet users come from all over the world without being affected by geographical differences and are able to communicate with one another from anywhere at any time.
(5) Digitalized environment: Any information transfer over the Internet is digital file, so copying or obtaining it is an easy task after all.

From what the characteristics of Internet reveal, the Internet is a perfect match for these digital files transfer because of its convenience and efficiency as well as much lower cost of information transfer.

There are nevertheless some factors such as security and efficiency that are to be taken into account for on-line transactions. To this point, our research is to explore how a secure and effective on-line software transaction protocol is established in the public Internet. We propose an effective protocol based on Public cryptosystem [3, 5, 8] and Digital signature [2, 8, 9]. The protocol meets the security and non-repudiation requirements [4, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 18]. The Public cryptosystem achieves the security and integrity of data transfer, while the Digital signature completes the non-repudiation.

Elsevier 1 of 8

Our paper is organized as follows: Section 1 is our Introduction. Section 2 is the Related Works Discussion. Section 3, Research Architecture, where an explanation on the architecture of our proposed research is presented. Section 4, On-line Software Transaction Protocol, where an explanation of our proposed method is presented. Lastly, Section 5 is our Security Analysis and Discussion,

2. Related Works Discussion

2.1. Software Copyright Protection

Our software copyright protection is mainly based on the proposed method by Cynthia et al. [2] in that our purpose is to restrain customers' intention from distributing their purchased software. The method proposed by Cynthia et al. [2] is to protect the large quantity of data stored in compact discs, in which there are two types of files with the one being the encrypted software and the other the program which generates the decryption key. When users want to use the software, they have to first run the program to generate the decryption key then decrypt the software with the key.

As for the program generating the decryption key, users will need some parameters before obtaining the key. The parameters to be entered may contain some personal information such as credit card information, identification card number or those of which obtained by phone from the software suppliers. Because more personal information is used when generating the key, there is less likelihood of illegal software distribution and infringement in order for software copyright protection.

Our research employs on its concept, user must input his/her own sensitive information and the parameter supplier authored before using the purchased software every time, yet we send the parameters over the Internet provided that the parameters to be sent are under strict protection. The key-generated program Cynthia et al. proposed is a starter enabling the inhibited software to work every time, which is like an login mechanism in this paper. In this way, user has to input his personal information and the parameter supplier authored to generate an enabling key in the program and to start the software up before using the software every time. There are two steps involved in a transaction: First is that software supplier confirms the order and customer information after an order is placed then send out the inhibited software as well as authored parameters to the third entrusted party. Second is that, after the bank confirms the payment to supplier, the customer can take the paid receipt to the third entrusted party and obtain the authored parameters to enable the software to work with his/her own credit card number.

2.2. Basic Requirements of On-line Software Download and Transaction

According to the characteristics of Internet in [19], we propose some basic requirements of on-line software download and transaction so as to establish a standard for our on-line software transaction protocol.

(1) Data Transfer Security
Generally speaking, on-line shopping has always been placed a great deal of emphasis. When customers shop on-line, they provide their credit card number or some very personal information that is to be kept unknown to others. It is mandatory for the supplier to be highly regarding this information and keeping it as secure as possible in order to avoid unnecessary damages or losses, so that the customers are more confident to shop on-line.

(2) Software Product Integrity
The quality assurance of software products is usually stricter than that of common goods. The software specification from supplier should be in accord with that of its real product. If not, customers may experience difficulty such as making mistakes and even not being able to execute the program during the installation or usage. A defective product like this is not allowed to happen in software purchases.

(3) Customer and Software Supplier Non-repudiation
Non-repudiation – one cannot refuse to have anything to do with his own act or acknowledge an act he has nothing to do with. Usually a third party steps in and helps the clarification. For a customer, he maliciously claims he has not received the software when he actually has. For a software supplier, he claims he has sent out the software when he actually has not. Situation like this always need an evidence to support and avoid causing any damage or loss.

(4) Reduce User Intention and Ability for Software Copyright Infringement
Software copyright infringement has long been a problem of software supplier, which is also the largest part of supplier's loss. In today's world of highly advanced Internet technology, there are a far more number of illegally obtained software users than legally authorized users. From this point, we propose a method that reduces the ability of legally authorized users from providing access to the software for other unauthorized users in order to cope with the piracy issues.

(5) Legal Rights of Authorized Users against Software Suppliers
Authorized user is entitled to request a reasonable response from the supplier for an updated version or careless damage, and the supplier should be able to tell the user' authorization and the type of software to help he out.

3. Research Architecture

Our paper is to establish a secure on-line software transaction protocol. We combine data integrity, anonymousness and non-forgery for a secure electronic transaction protocol, plus non-repudiation we establish a secure on-line credit card transaction. With its security as basis in addition to which software copyright

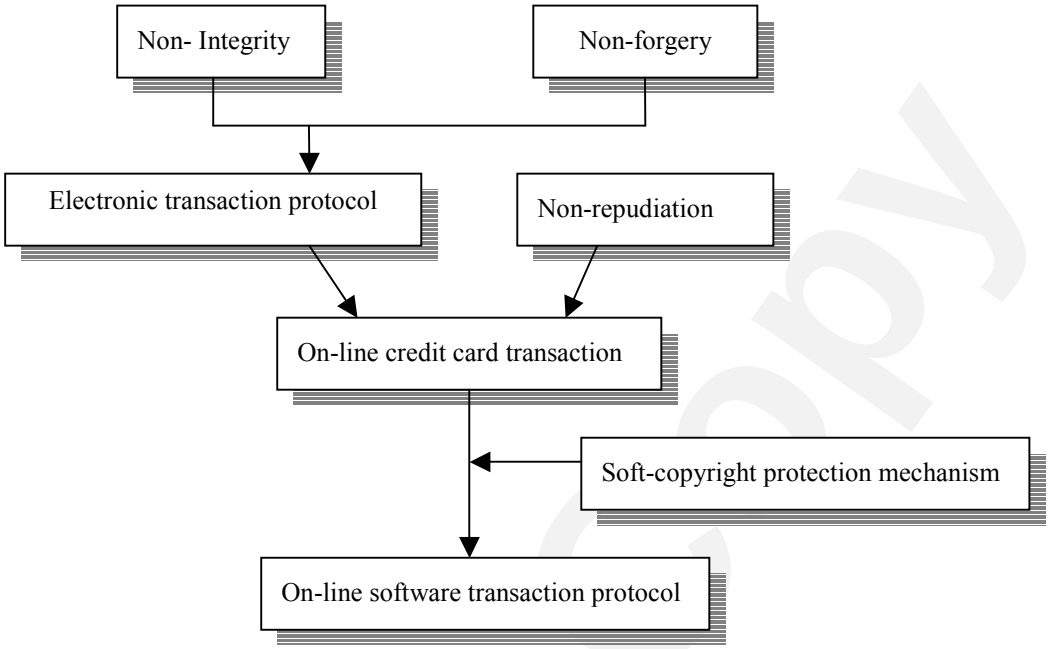

Figure 1. Our research architecture

protection mechanism combined together, a secure on-line software transaction protocol is established. Our research architecture is shown as Figure 1.

4. Our proposed scheme

For an on-line transaction to go through over the public Internet, it has to fulfill the following: Customers receive the exact software they expect to purchase. Merchants receive the right amount of payment. Electronic banks can prevent from double payments to the merchants and verify if they complete their jobs as required when they submit payment requests. Therefore we propose an efficient and secure on-line software transaction protocol, based on the proposed software copyrights protection by Cynthia, which can prevent the software from random distributing. We will be specifying the participating parties and executing steps of the transaction protocol in the following.

4.1. Participating Parties of On-line Software Transaction Protocol

In our proposed on-line software transaction protocol, these participating parties are customers, merchants, electronic banks and trustworthy third parties in which their tasks and characteristics are detailed respectively in the following:

1. Customers: They place orders over the Internet from merchants and provide identity and credit cards information for verification purposes.
2. Merchants: They provide valuable software for download. They will first verify the customer's identity then provide appropriate services as required depending on the order of products.
3. Electronic Banks: They have customer's identity and credit cards information that they pay for the customer as part of the service. Merchant will need to provide proof of customer's consent to pay before they request payment from the electronic bank.
4. Trustworthy Third Parties: Entrusted unit. Government may be the third party whose role function is to act as an agent who manages the important data then allocates tangible public key as well as private key and maintains duplicates of the important data.

Of transaction protocol, the physical relationship among customers, merchants, electronic banks and trustworthy third parties is shown in Figure 2.

4.2. Digital Certificate Patterns

Elsevier

Friday , October 29, 2004

It's not an easy task to stop people who maliciously give false information or deny having participated in the transaction that already went through, causing another party to be responsible for the loss and even for deceiving. Therefore we will need a proof to confirm the validity of the self and others identities, however the digital certificate mechanism under PKI (Public-Key Infrastructure) can solve the above repudiation and deceiving problems.

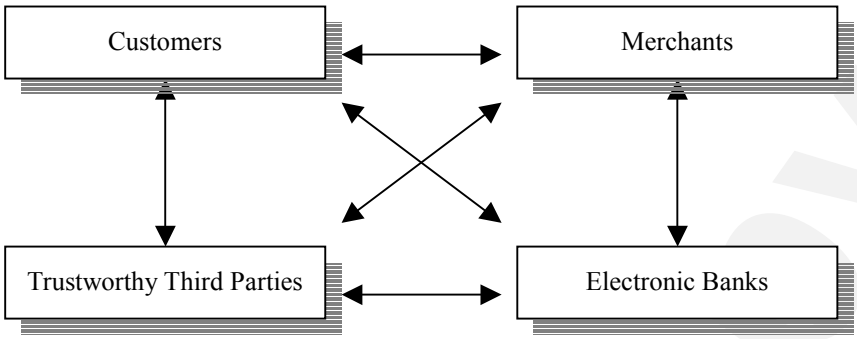

Figure 2. The physical relationship

Digital certificates are distributed by a valid and trustworthy certificate authority (CA for short). Applicants only need to apply for a certificate while sending their personal information out, the CA will then assign a digital certificate that contains the serial number, applicant's identification and public key. The CA will sign with its own private key to this digital certificate that is validated and fair on behalf of the applicant himself.

According to today's commonly used X.509 standards, the content of digital certificate is described in the following simple words as shown in Table 1.

Table 1. Content of Digital Certificate by X.509 Standards

Certificate Version	Records the version number of this certificate
Certificate Serial Number	The serial number for the authorized applicant assigned from the CA
Identification Code of Algorithm	Records the algorithm for signature authentication on the certificate
Authorized Unit	Related information of the authorized applicant
Public Key	Public key of the authorized applicant
Registration Time	The time of authorized applicant registration
Valid Time Period	The valid time period of this certificate
Distributing Unit Identity	Related information of the certificate authority that distributes certificates
Distributing Unit Signature	The certificate authority signs with its own private key to the above data for validation

4.3. Preparation Process for the Protocol

In our transaction protocol, the participating customers, merchants, electronic banks and trustworthy third parties must all apply for a digital certificate from the CA. In addition, the customers and merchants both will create an account with the electronic bank so the bank can be a medium of money transfer between them.

4.4. Complete On-line Software Transaction Progress

We will divide the entire process of transaction protocol into four steps: Purchase, Transfer Payment Request, Sending Rights of Software Uses and Actualizing Rights of Software Uses. The complete progress of on-line software transaction protocol is shown as Figure 3.

In each of these steps, the four participating units, namely customers, merchants, electronic banks and trustworthy third parties are respectively explained in the following:

4.4.1. Purchase

The description of symbols used in our transaction protocol is shown in Table 2.

Step 1. Merchant sends the product information signed with his own private key and digital certificate to the

Elsevier

Friday , October 29, 2004

customer.

Step 2. Customer sends the encrypted order payment information and digital certificate to the merchant.

Step 2.1. Customer uses the public key provided by the merchant to verify the signature validity on product information.

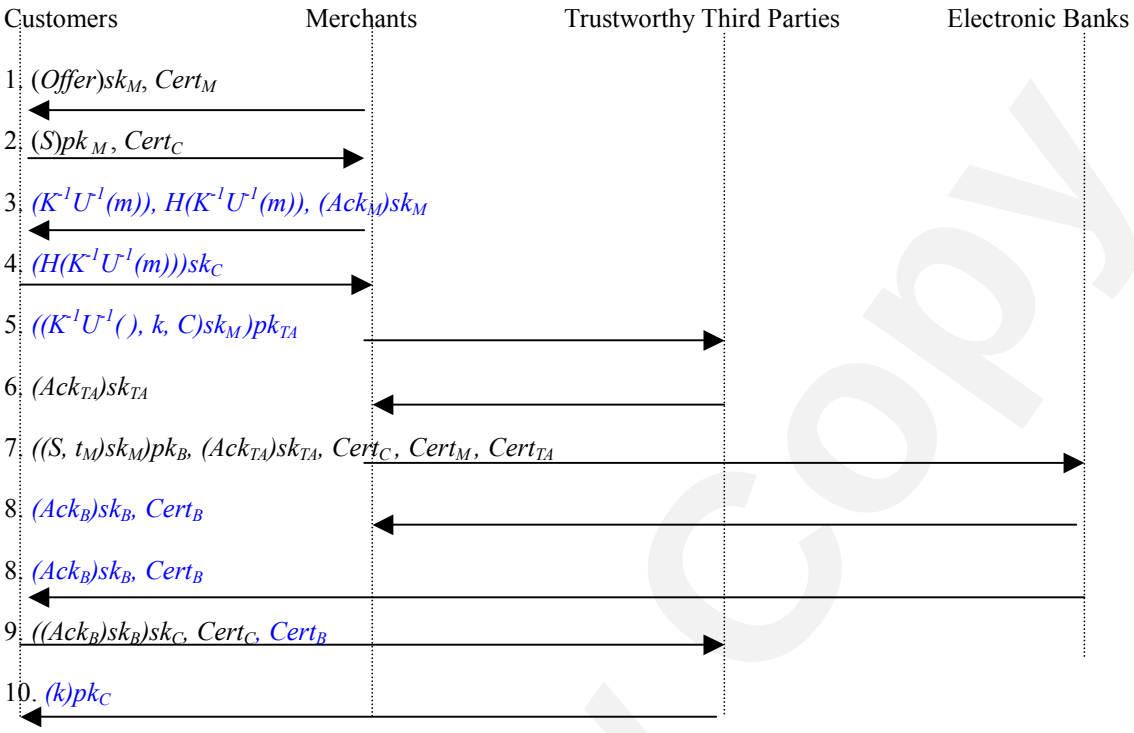

Figure 3. The complete progress of on-line software transaction protocol

Table 2. Description of Symbols in On-line Software Transaction Protocol

Customer Digital Certificate	$Cert_C$	Merchant Digital Certificate	$Cert_M$
Trustworthy Third Party Certificate	$Cert_{TA}$	Electronic Bank Digital Certificate	$Cert_B$
Customer ID	C	Merchant ID	M
Trustworthy Third Party ID	TA	Electronic Bank ID	B
Product Information	$Offer$	Credit Card Number	U
Order Information	O	Amount of Order Payment	$\$$
Time-stamping of Customer Purchase	t_C	Time-stamping of Merchant Payment Request	t_M
Acknowledgement	Ack	Communicative Key	k
Inhibited Software	$K^{-1}U^{-1}(m)$	Hash Function	$H()$
Key Generator	$K^{-1}U^{-1}()$	Public-Key Encryption	$()pk$
Private-Key Signature	$()sk$	Order Payment Information	S

Step 2.2. The order payment information is made up of the customer ID, merchant ID, credit card number, order for which products are to be purchased, total amount of order, the time-stamping of customer purchase, etc. The order payment information is $S=(U, C, M, O, \$, t_C)sk_C$.

Step 2.3. The customer encrypts the order payment information with the merchant's public key then sends the encrypted information as well as self-digital certificate together to the merchant.

Step 3. Merchant sends the encrypted software and the results from hash function to the customer.

Step 3.1. Merchant decrypts the message with his own private key.

Step 3.2. Verify the signature validity on order payment information with the customer's public key.

Step 3.3. From the customer's credit card number and self-generated communicative key, create a specific software key-by-key generator.

Step 3.4. Merchant combines the specific software with the corresponding key generator to produce inhibited software. The inhibited software is disabled to use except that user inputs his own credit card number and the corresponding communicative key when using the purchased software every time.

Elsevier

Step 3.5. Proceed on the inhibited software with hash function then send the inhibited software, hashed results and merchant signed order payment acknowledgement to the customer.

Step 4. Customer signs the hash-generated results and sends back to the merchant.
Step 4.1. Customer verifies the validity of the merchant signed acknowledgement.
Step 4.2. Proceed on the encrypted software with hash function.
Step 4.3. Match the hashed results with which the merchant sends.
Step 4.4. If matching is not successful, the customer will start the download over from the merchant until the results are matched.
Step 4.5. If matching is successful, the hash-generated results will be signed and sent to the merchant.

Step 5. Merchant sends the rights of software uses and his digital certificate to a trustworthy third party. The trustworthy third party manages the important data for enabling the purchased software, such as key generator and communicative key.
Step 5.1. Merchant verifies the signature validity on hashed results sent by the customer; upon the customer's identity is confirmed, the merchant recognizes that the customer has received the encrypted software for certain.
Step 5.2. Sign the key generator, communicative key, customer ID, and self-private key to generate the rights of software uses.
Step 5.3. Encrypt the rights of software uses with the trustworthy third party's public key.
Step 5.4. Send the encrypted rights of software uses and self-digital certificate to the trustworthy third party.

Step 6. The trustworthy third party sends the signature acknowledgement to the merchant.
Step 6.1. The trustworthy third party decrypts the information sent from the merchant with his own private key.
Step 6.2. From the merchant's public key, it's possible to verify the signature validity on the rights of software uses and obtain information such as the key generator, communicative key, customer ID and what not.
Step 6.3. If the integrity of the previous information is doubted, request the merchant to send again.
Step 6.4. If the integrity of information is confirmable, send the signed acknowledgement receipt to the merchant.

4.4.2. Transfer Payment Request
Step 7. Merchant sends the payment request and the signed acknowledgement by the trustworthy third party to electronic bank.
Step 7.1. Merchant verifies the signature validity on the acknowledgement by the trustworthy third party to determine whether the third party does actually receive the information such as key generator, communicative key, customer ID…etc.
Step 7.2. Sign with the private key to the order payment information including the signed customer information, merchant ID, credit card number, order for which the products are to be purchased, total amount of order and time-stamping of customer purchase plus the time-stamping of merchant payment request.
Step 7.3. Encrypt the order payment information and time stamping of merchant payment request with the electronic bank's public key; a payment request proof will be produced. Send this proof as well as the signed acknowledgement by the trustworthy third party to the bank for payment application.

Step 8. Electronic bank sends the signed receipt to both customer and merchant.
Step 8.1. Electronic bank verifies the signature validity on the signed acknowledgement by the trustworthy third party to determine whether the trustworthy third party does actually receive the rights of software uses sent from the merchant.
Step 8.2. Decrypt the payment request information from the merchant with self-public key to obtain the merchant signed order payment information and time stamping of merchant payment request.
Step 8.3. Then verify the signature validity with merchant's public key to obtain the order payment information and time stamping of merchant payment request.
Step 8.4. Verify the signature validity on order payment information with customer's public key to obtain information such as the customer ID, merchant ID, credit card information, order in which products are to be purchased, total amount of order and time-stamping of customer purchase.
Step 8.5. After all verifications are completed, proceed with the payment transfer according to the available information.
Step 8.6. Once the payment transfer is done, send the signed receipt by the bank's private key to the customer and merchant.

4.4.3. Sending Rights of Software Uses

Review Copy

Step 9. Customer sends the digital certificate and signed receipt to the trustworthy third party.
Step 9.1. Customer verifies the signature validity on the receipt with the electronic bank's public key.
Step 9.2. Sign with the customer's own private key to the receipt and send along with the customer's digital certificate to a trustworthy third party.

Step 10. Trustworthy third party checks the payment information and sends the encrypted communicative key to the customer.
Step 10.1. Trustworthy third party verifies the signature validity on the receipt with customer's public key.
Step 10.2. Encrypt the corresponding communicative key of customer ID then send to the customer.

4.4.4. Actualizing Rights of Software Uses
Step 11. Customer applies the software key to use his purchased software.
Step 11.1. Customer decrypts the message with his own private key to obtain communicative key.
Step 11.2. To enable the inhibited software to work, Customer keys the communicative key and his own credit card number in the key generator combined with the purchased software every time to use.

5. Discussion and Analysis

5.1 Proof of Non-repudiation

In our proposed on-line software transaction protocol, as well as in data transferring, there are many evidences of non-repudiation already included. The participants are advised to keep these evidences properly in case of any conflicts that they can show the proof to solve.

When there is actually a conflict, the participants can show proof to solve the conflict. The following are some possible situations we assume might happen in an on-line software transaction protocol and how the participants demonstrate their evidences to solve the conflicts:

(1) Customer denies having sent the purchasing message (Step 2). The merchant can provide the non-repudiation evidence $S=(U, C, M, O, \$, t_C)sk_C$ to prove the customer did actually send a message to purchase the software. This prevents customers from maliciously denying having sent the purchasing message (Step 2).
(2) Merchant denies having received the purchasing message and credit card number from the customer (Step 3). The customer can provide the non-repudiation evidence $(Ack_M)sk_B$ to prove the merchant did receive the purchasing message and his credit card number. This prevents merchants from maliciously denying having received the purchasing message (Step 2).
(3) Customer denies having received the inhibited software download message (Step 4), the merchant can provide the non-repudiation evidence $(H(K^{-1}U^{-1}(m)))sk_C$ to prove he customer did receive the inhibited software. This prevents customers from maliciously denying having received the inhibited software (Step 3).
(4) In order to prevent the merchant from not sending the communicative key for inhibited software to the trustworthy third party, the merchant provides the non-repudiation evidence $(Ack_{TA})sk_{TA}$ to prove the trustworthy third party did actually receive the communicative key (Step 5) and the merchant did actually send the communicative key to the third party.
(5) Merchant denies having withdrawn money from the electronic bank (Step 7). The customer can provide the non-repudiation evidence $(Ack_B)sk_B$ or the bank can provide $(S, t_M)sk_M$ to show the merchant did actually go to withdraw the money from the bank. This prevents merchants from maliciously denying having received the money from the bank (Step 7).

5.2 Data Integrity

In our on-line software transaction protocol, we have also added the analysis of data integrity and correctness in data transferring among the participants.

(1) Whether or not the customer receives the right encrypted software:
Customer can use Step 3 data to verify if the received software has been modified in any way or just not the right one. The customer can proceed on the inhibited software $H(K^{-1}U^{-1}(m)$ with hash function H(), then match with $H(H(K^{-1}U^{-1}(m)))$. Successful matching means the received software is the right one whereas the non-successful matching indicates a problem with the received software. At this point, request the merchant to re-transfer the inhibited software for download (Step 3).
(2) Whether or not the trustworthy third party receives the right decryption key:
Trustworthy third party can use Step 5 data to verify if the received communicative key has been modified in any way or just not the right one. The third party can decrypt the received information $((K^{-1}U^{-1}(), k, C)sk_M)pk_{TA}$ as well as signature algorithm from the third party's private key sk_{TA} and merchant's public key pk_M. The third party computes the signature digest of key generator, communicative key and customer ID and then verifies whether it is equal to received one. A successful matching indicates the received decryption key

Elsevier

Friday , October 29, 2004

is the right one whereas a non-successful matching implies a problem with the received important data for enabling the purchased software. At this point, request the merchant to re-send the Step 5 message.

6. Conclusion

With an increasing number of transactions gone through over the Internet everyday, its security mechanism has since been a customers' main concern in deciding whether or not they want to make on-line transaction. Therefore, we employ the PKI security mechanism and propose a secure and feasible transaction protocol that meets customers' demands when which they want to purchase on-line, download software and make payment requests. What's more on this mechanism is with which the customers' credit card information is also required to initiate the software. As a result, the customers are less than likely to illegally distribute the software that they purchase because they don't want to reveal their credit card information to anyone, and the software copyright is protected at the same time.

Although in our proposed on-line software transaction protocol, we provide a secure and feasible yet copyright protected on-line transaction mechanism. Because today's society has been greatly concerned with privacy, and we would like to extend our research further with our current accomplishment as basis, we are developing a newly secure and feasible yet copyright protected on-line transaction method to enhance the customers' anonymousness in our future research, provided that the customer information is ensured its safety and his credit card information can be verified before the transaction.

Acknowledgement

This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC90-2213-E-324-004. The authors would also like to thank Chih-Wei Lin from Chaoyang University for his help.

Reference

1. Alexandris, N., Burmester, M., Chrissikopoulos V., & Desmedt, Y. (2000). Secure linking of customers, merchants and banks in electronic commerce. Future Generation Computer Systems, 16, 393-401.
2. Cynthia, D., Lotspiech, L.J. & Naor, M. (1996). Proceedings of the 8th Symposium on the Theory of Computation (pp. 489-498). Digital signets: self-enforcing protection of digital information.
3. Diffie, W. & Hellman, M.E. (1976). New directions in cryptography. IEEE Trans. Information Theory, 6, 664-654.
4. Herda, S. (1995). Non-repudiation constituting evidence and proof in digital cooperation. Computer Standards & Interfaces, 17, 69-79.
5. Jonge W.D. & Chaum, D. (1985). Proceeding of the CRYPTO'85 (pp. 18-27). Attacks on Some RSA Signatures.
6. Kim, K., Park, S., & Baek, J. (1999). Improving Fairness and Privacy of Zhou-Gollmann's Fait Non-repudiation Protocol. IEEE Parallel Processing, 140-145.
7. Kalla, M., Wong, J.S.K., Mikler, A.R., & Elbert, S. (1999). "Achieving non-repudiation of Web based transactions. The Journal of Systems and Software, 48, 165-175.
8. Rivest, R.L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), 120-126.
9. Rivest, R.L. (1992). The MD5 message digest algorithm, *RFC 1321.*
10. Schneider, (1998). Formal analysis of a non-repudiation protocol. IEEE Computer Security Foundations Workshop, 11, 54-6. 11, 54-65.
11. Zhou, J., & Gollmann, D. (1996). Observations on non-repudiation, *Lecture Notes in Computer Science*, 1163, 133-144.
12. Zhou, J., & Gollmann, D. (1997) Evidence and non-repudiation. Journal of Nerwork and Computer Applications, 20, 267.281.
13. Zhou, J. and Lam K.Y. (1999) Securing digital signatures for non-repudiation, *Computer Communications*, 22, 710-716.
14. Zhou, J. & Gollmann, D. (1997) An efficient non-repudiation protocol. IEEE Computer Security Foundations *Workshop*,. 10, 126-132.
15. Zhou, J. & Gollmann, D. (1996) A fair non-repudiation protocol. IEEE Symposium on Security and Privacy, 55-61.
16. Secure Electronic Transaction Specification, Book1.
17. Secure Electronic Transaction Specification, Book2.
18. Secure Electronic Transaction Specification, Book3.

Elsevier