Fail-Safe Security Architecture to Prevent Privacy Leaks from E-commerce Servers

Hiroshi Fujinoki*, Christopher A. Chelmecki, and David M. Henry
Southern Illinois University Edwardsville
Edwardsvill, Illinois, USA

Abstract

We propose new security architecture, called Fail-Safe Security Architecture (FSSA), which eliminates, or at least minimizes, the chance of privacy leaks for e-commerce customers, protecting their privacy even for the worst cases: the security administrators of the e-commerce servers convert to attackers or the merchants servers are hijacked by external attackers, giving the attackers full access to anything in the servers. FSSA is based on a security design that allows each party to access only the information necessary to perform their business and it makes sure no party, except the customer and the law enforcement authority, has access to the complete information of customers privacy. We analyzed the types of the security threats FSSA covers. The results of our analyses indicated that, FSSA protects customer privacy against the internal attackers (converted administrators and full hijacks), as well as the known security threats by external attackers of eavesdropping, replay, masquerading, man-in-middle, and traffic analyses, except denial of service attacks. Our performance studies suggested that the cost factor of running FSSA is 1.8 (1.8 times more computational power) to achieve the same response time and transaction throughput compared to the existing architecture, where there is no protection against the customer private information leaks.

Keywords: e-commerce security, security against insiders, prevention of privacy leaks, confirmation of delivered products in e-commerce, network application security

1 Introduction

Privacy leaks from e-commerce servers are a serious problem today. Privacy information stolen by attackers ranges from e-mail account password, credit card information, personal identification, such as name, address, telephone numbers, and occupation information, as well as account information for on-line shopping sites. Pavia reported that acquiring legal credit cards using the stolen personal identity occupies 62 percent of all credit card frauds today, instead of stealing the information of the existing credit cards [15].

E-commerce customers' fear of their privacy leaks and their distrust of e-commerce merchants' security due to frequent leaks of their privacy have escalated to the point that has impacted to acceptance of e-commerce applications. Glover conducted an empirical study to find that the risk of information misuse is currently one of the three major factors that discourage customers from using online e-commerce applications [8].

While privacy leaks can cause financial damages to e-commerce merchants and credit card companies, the US Federal Law (Section 901 of title IX of the Act of May 29, 1968 (Pub. L. No. 90-321)) protects customers from most of the financial losses due to credit card frauds. However, the damages from privacy leaks are sometime outside of the Federal Law's coverage. For example, illegally accessed

*Correspondence author: Department of Computer Science, Southern Illinois University Edwardsville, Edwardsville, Illinois 62026-1656, USA, Tel: +1-618-650-3727, Email: `hfujino@siue.edu`, Web: `www.siue.edu/~hfujino` privacy was often used for sales promotion in the past. Privacy leaks from adult web sites have been recently used by attackers to extort money from their users by threatening their social statuses.

From a technical viewpoint, most of the privacy leaks stem from the way online e-commerce applications are designed. For example, many of the online shopping web sites are currently implemented based on the client server model, in which servers hold customers' privacy information. Most of the online shopping web sites require their customers to provide their personal information to the servers owned by merchants. Once customers upload their personal information to servers, there is nothing the customers can do to protect their privacy, but trusting the merchants.

What make the situation worse are the crimes committed by malicious insiders. A trend of increasing crimes by insiders is reported [5], which seriously ruins trust by customers for using online e-commerce applications. Malicious insiders are not the unique problem to online e-commerce, but also to any face-to-face commerce. However, the impact of the crimes by insiders will be more significant for online e-commerce applications since customers do not have a chance to see the merchants face-to-face, making it quite difficult to measure the level of trust to their business.

Handling customers' information in electronic commerce has another dimension. Disputes regarding transactions, such as claims about delivery of a damaged product, may require involvements of all the parties. All of the merchant, the shipping carrier, and the credit card company, have to share the complete information to resolve the disputes, which contains all the details about the customers, as well as the products and services purchased. Sharing customers' information this way by all the three parties increases the risk of privacy leaks.

Also from the viewpoint of the legal issues, the ability to reconstruct the complete information for each transaction is mandatory, since the United States Money Laundering Act (U.S. Code 12, Section 1829) prohibits anonymous transactions valued over 100USD [1]. The complete information about each such online transaction must be reconstructed and disclosed to a legislative authority up on a court's disclosure order.

To solve the trade-off problem described above, we propose secure network architecture, called Fail-Safe Security Architecture (FSSA) that protects customers' privacy information not only from external attackers, but also from internal attackers (insiders) at the same time it satisfies the legal requirements. As a framework for the new security architecture, we assumed e-commerce systems that consist of the four parties of customers, e-commerce merchants, credit card companies, and shipping carriers. E-commerce merchants set up and maintain their online web sites to take orders from customers. Customers visit online shopping web sites set up by merchants, where customers make orders to products offered by the merchants. Credit card companies take payment requests from customers for the products ordered by the customers and make the payments to the merchants. Shipping carriers accept packages from merchants to transport them to customers.

The term, "internal attackers", means malicious insiders in the rest of this paper. Internal attackers are those who have access to customers' information stored in a server while external attackers are anyone who does not. Note that the attackers working in e-commerce merchants, credit card companies, and shipping carriers are still external attackers as long as they are not given official access to customers' information stored in their server.

The rest of this paper is organized as follow. Section 2 describes the existing related work. In Section 3, the proposed security architecture is described by applying the architecture to online transactions typically performed in online shopping web sites today. The proposed security architecture is analyzed for what security risks it covers and how. Section 4 describes performance evaluations of the proposed architecture for response time and throughput of e-commerce transactions issued by customers. Section 5 summarizes the conclusions, followed by the references.

2 Existing related work

Many of the existing information protection methods, such as encrypting a whole disk [9] and separation of stored data from code execution [14] will be effective to external attackers, but not to internal attackers if the information is not protected in the application level since the information must be decrypted or available to the applications when the information is processed by the applications. At that time, customers' privacy is accessible also to the internal attackers. The problem is even more serious for e-commerce applications running in a cloud environment.

To secure personal information from internal attackers, encrypting personal information using customers' biometric information without storing decryption keys in the server side has been proposed by Uludag [18] and Zhang [20]. Although these solutions will be effective in hiding customers' information from anyone except the customers, none of the merchants, shipping carriers, or credit companies has access to the customers' information, which fails to satisfy the legal requirements.

Mazieres proposed a distributed file system, SUNDR, which prohibits any unauthorized users from performing read access to the files stored in their storage devices [12]. SUNDR chops the contents in each file and spread the chops to multiple segments, called "data blocks" in such a way that the mapping of the segments in a file is hidden by "virtual i-node". This solution guarantees that only the authorized users can access their files by reconstructing the files' contents using the virtual i-node.

Mattsson proposed column-level database encryption to protect users' information from both external and internal attackers [11]. The column-level database encryption is performed by encrypting security sensitive data, calculating the HMAC hash of the data, and attaching the hash to another column in the same row of the table. The owner of the data uses the hash to find target data by matching the hash value. The decryption key for the secure data is not stored in the database, but in the applications that use the database. The encryption keys stored in the applications in the server are accessible to the insiders.

Although the above two solutions are effective for protecting customers' information from external attackers, the solutions will not be effective to internal attackers. It is because internal attackers can access even the information held by applications. They can intercept the information before it is passed from the applications to the system level [19]. There should be a security mechanism that spans the application layer and the OS layer, instead of two isolated security mechanisms, one in the application layer and the other in the OS layer. Otherwise, the communication between the two mechanisms can be the target of the internal attackers.

Secure Electronic Transaction (SET) protocol has been proposed to secure electronic payments using credit cards for online transactions such as purchasing goods through merchants' web sites [4]. Since its introduction, several extensions to SET have been proposed, some of which were proposed for realizing customers' anonymity in SET. For example, Rennhard proposed new network application architecture for online shopping using SET, which hides customers' network address from the merchants and credit card companies to protect customers' privacy [16]. Rennhard's solution consists of an overlay network called "Pseudonymity Network (PN)", which uses intermediate proxies and a security protocol (for key exchanges and encryptions) called "Pseudonymous Secure Electronic Transaction (PSET) protocol" for the messages exchanged by the involved parties to hide customers' network addresses and identities.

Although SET and its enhancements will be effective for hiding customers' credit card information from the merchants, none of them prevents customers' privacy from leaking at the merchant's servers, if merchants' servers store customers' privacy, such as their full names and shipping addresses. This is a problem, since most of the online customers do not pick up the products at the merchants' facility, while various services in online shopping, such as payments for the goods and shipping the goods to customers, should be integrated to automate businesses for better transactional turnarounds and for reducing labor cost [2] [13].

Tygar introduced the concept of "certified delivery" [17]. The term "certified delivery" means that a customer will receive the product if and only if the money for the product is transferred from the customer to the merchant and that the delivered goods must be correct. The primary problem Tygar identified in designing the certified delivery is in the trade-off between disclosures of customer's information to all the involved parties and achieving anonymity of customers. The more information about each transaction is disclosed to each party, the easier it will be for them to handle any dispute for each transaction, but it obviously sacrifices anonymity in each transaction.

Zhang proposed security enhancements to SET [20]. Zhang solution uses Active-X to protect customers' private keys stored in the customer's host computer. When a user's private key is referenced by a local process, active-X prompts the user to enter the user's secret code to approve the use of the private key. A potential threat to this approach is that, if the active-X process is modified especially by internal attackers, the active-X process can be used to collect the user's secret code for the attackers.

Lekkas proposed a new approach for securing electronic payments, called ”E-coin” [10]. E-coin was proposed as an off-line electronic payment method without contacting a bank. E-coin is a new approach in that it performs payments by securely transferring virtual currency (i.e., E-coins) issued by an individual, instead of transferring payment information for actual currency as SET does. One of its features is ”untraceability”, meaning that it is not possible to identify who (as a real human) issued the currency and for what product it was used for. Lekkas solution is based on the concept of ”on-line virtual currency” proposed by Chaum [3]. An electronic payment system proposed by Eslami [6] is based on the same concept.

3 Designs and implementation of FSSA

The proposed FSSA security architecture protects customers' privacy information from both external and internal attackers by making sure that none of the four parties, except the customers, has access to the complete information for each e-commerce transaction, while FSSA allows the three parties of a merchant, a credit company, and a shipping carrier, to reconstruct the complete information for each transaction when necessary. By ”complete information for each e-commerce transaction”, we mean a set of data that collectively describes all the details about an e-commerce transaction, such as the real name of a customer, the shipping address, phone number, the product(s) ordered, the dollar amount charged to the transaction, the date and time of the order, and the credit card used.

The capability to reconstruct the complete information for each transaction is an essential requirement for not only satisfying the legal requirements, but also for handling any claim about transactions. For example, if a customer has a question about the payment amount charged by the credit card company for a particular order, the merchant and the credit card company need to collaborate on the investigation. If the products are lost during their shipping, the customer may request the merchant to provide its shipping information to the carrier for an insurance claim. On a disclosure order from a court, the three parties need to collaborate to reconstruct the complete information about the transactions. To handle such situations, there must be a mechanism that logically integrates users' information about a particular transaction while no single party accesses to the complete information about each transaction.

3.1 FSSA Organization

A high-level view of the FSSA organization is shown in Figure 1. Each of the four parties has a host computer. The four host computers are logically connected using the star topology with the merchant's server at the hub. To prevent masquerading, eavesdropping, replay, repudiation, and man-in-middle attacks during their transmissions, the host computers are connected by a secure connection, such as SSL. Denial of service attacks are not a scope of FSSA.

To facilitate the secure connections, each of the merchant, the shipping carrier, and the credit card company is assumed to have their digital certificate that contains their digitally signed public key. Merchants obtain the digital certificates of the shipping carriers and the credit card companies and make the certificates available to their customers through their online shopping sites. FSSA does not require customers to have their digital certificate. It is not reasonable to mandate customers to have their digital certificate, since requiring digital certificate, as many existing secure transaction protocols do, costs customers and such solutions will not be widely adopted by customers because of the cost.

In FSSA, none of the three parties of the merchant, the credit card company, and the shipping carrier has the complete information about each transaction. The premise of this design is that each party can access only the information needed to perform their business. For example, only the information merchants should know is about the ordered products, but not about the customers, such as their real names, mailing address and phone numbers, as well as their credit card number. Likewise, the only information shipping carriers should know is the information needed for delivering the packages, but not about the products ordered by customers.

If none of the three parties (merchants, shipping carriers, and credit card companies) holds the complete information about a transaction, there must be a mechanism that bundles the separate pieces of information together, when the complete information is needed by all three. Since each merchant may have multiple transactions that involve the same shipping carrier and the credit card company for the same customer, there must be the information that uniquely identifies every single transaction. This introduces the following two requirements: (1) transactions made by a merchant must be uniquely identified by unique transaction numbers assigned by the merchant and (2) each merchant must be uniquely identified by the three parties of customers, shipping carriers and credit card companies.

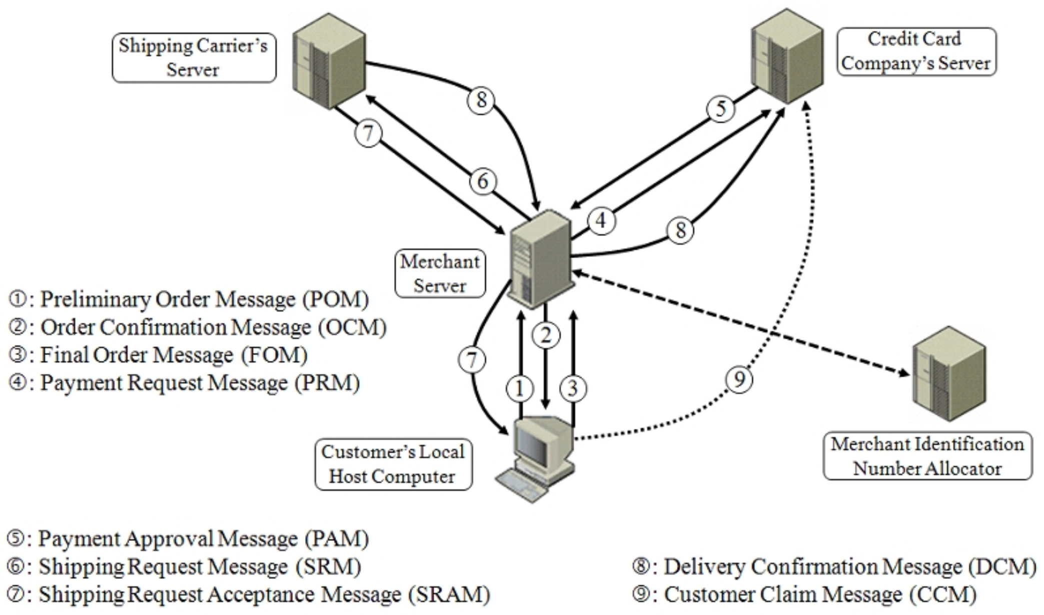

Shipping Carrier's Server
Credit Card Company's Server
Merchant Server
Customer's Local Host Computer
Merchant Identification Number Allocator
①: Preliminary Order Message (POM)
②: Order Confirmation Message (OCM)
③: Final Order Message (FOM)
④: Payment Request Message (PRM)
⑤: Payment Approval Message (PAM)
⑥: Shipping Request Message (SRM)
⑦: Shipping Request Acceptance Message (SRAM)
⑧: Delivery Confirmation Message (DCM)
⑨: Customer Claim Message (CCM)

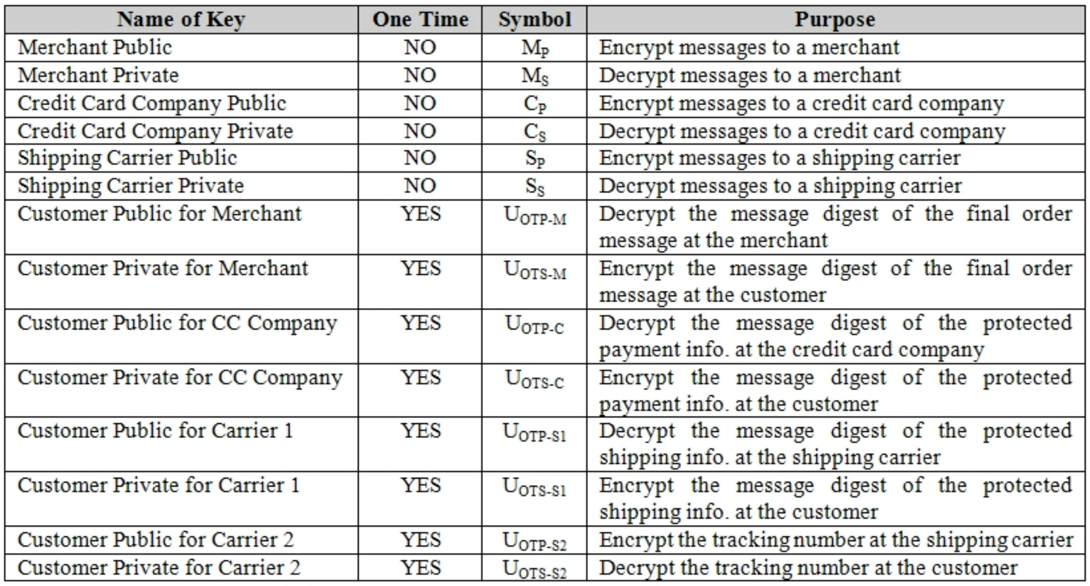
Figure 1: High-level view for FSSA organization

To meet the second requirement, we propose the merchant identification number allocator (MINA) that assigns a unique identification number to each merchant. MINA should be owned and managed by an organization, possibly by a consortium of e-commerce merchants. By combining a unique transaction number assigned by a merchant and a unique merchant identification number assigned by MINA, every single transaction should be uniquely identified. A combination of a unique transaction number and a merchant number works as glue that puts pieces of information together to reconstruct the complete information for a transaction when customers or law enforcement authorities need the information. Each merchant needs to contact MINA only once when the new merchant starts her business.

None of the merchant, the credit card company or the shipping carrier can disclose the information they have about a transaction to anyone else, if they are not requested by the customer or the law enforcement authority. Thus, the complete information of a transaction can be reconstructed only on the customer's or a court's request, and if all the three parties collaborate.

3.2 Implementation

FSSA uses asymmetric cryptographic keys, some of which are "one-time", meaning that a new key will be created for each transaction and that it will be discarded when a transaction is completed. The cryptographic keys used in FSSA are listed in Figure 2. The first letter in the key symbol indicates the owner of the key (i.e., M: merchant, C: credit card company, S: shipping carrier, and U: customer).



Name of Key	One Time	Symbol	Purpose
Merchant Public	NO	M_P	Encrypt messages to a merchant
Merchant Private	NO	M_S	Decrypt messages to a merchant
Credit Card Company Public	NO	C_P	Encrypt messages to a credit card company
Credit Card Company Private	NO	C_S	Decrypt messages to a credit card company
Shipping Carrier Public	NO	S_P	Encrypt messages to a shipping carrier
Shipping Carrier Private	NO	S_S	Decrypt messages to a shipping carrier
Customer Public for Merchant	YES	$U_{OTP\text{-}M}$	Decrypt the message digest of the final order message at the merchant
Customer Private for Merchant	YES	$U_{OTS\text{-}M}$	Encrypt the message digest of the final order message at the customer
Customer Public for CC Company	YES	$U_{OTP\text{-}C}$	Decrypt the message digest of the protected payment info. at the credit card company
Customer Private for CC Company	YES	$U_{OTS\text{-}C}$	Encrypt the message digest of the protected payment info. at the customer
Customer Public for Carrier 1	YES	$U_{OTP\text{-}S1}$	Decrypt the message digest of the protected shipping info. at the shipping carrier
Customer Private for Carrier 1	YES	$U_{OTS\text{-}S1}$	Encrypt the message digest of the protected shipping info. at the customer
Customer Public for Carrier 2	YES	$U_{OTP\text{-}S2}$	Encrypt the tracking number at the shipping carrier
Customer Private for Carrier 2	YES	$U_{OTS\text{-}S2}$	Decrypt the tracking number at the customer

Figure 2: Construction of the protected SI (PSI) message

The procedure of FSSA starts when a customer visits the web site of a merchant. When a customer visits the web site of a merchant, a secure connection (e.g., SSL) is established in the application level between the customer's web browser and the web server of the merchant. Then, the customer's web browser downloads the digital certificate (referred by "certificate" hereafter) of the merchant, confirms the identity of the merchant, and extracts the merchant's public key (M_P). The customer specifies the products, the number of the products, and the name of the shipping carrier and the credit card company the customer would like to use. These pieces of the information (but no other information is included) are packed in a message, called "the preliminary order information (POM)" and it is transmitted to the merchant's web server (message ① in Figure 1) through the secure connection.

When the merchant's web server receives a POM, it calculates the total payment amount. The merchant web server issues a unique transaction number in the day. The merchant's web server then issues an order token for this transaction. The order token is the information that uniquely identifies each transaction by all the four parties of the customer, the merchant, the shipping carrier, and the credit card company. Each order token consists of the merchant identification number, the date of the transaction in year, month, and day, followed by the transaction number in the day, but no other information is included.

Once the order token is constructed, the merchant's server creates "order information (OI)" that contains the details of the product(s) ordered by the customer, such as the product ID's, the number of the product(s) ordered, the name of the shipping carrier, the name of the credit card company, and the total amount for the payment (but no other information). The merchant's web server then bundles the OI, the order token, the certificates of the shipping carrier and the credit card company in a message, called "order confirmation message (OCM)". The OCM is transmitted from the merchant's web server to the customer's web browser using the secure connection between them (message ②).

When the customer's web browser receives the OCM from the merchant, the contents of the OI are displayed by the web browser to the customer, asking for confirmation. If the customer approves the contents in the OI, the web browser extracts the public keys of the shipping carrier (S_P) and the credit card company (C_P) from their certificates in the OCM. The customer's web browser generates the four pairs of asymmetric one-time keys for this transaction.

After the four pairs of the one-time asymmetric keys are created, the customer's web browser constructs the final order message (FOM). The FOM consists of the OI, the protected shipping information (PSI), the protected payment information (PPI), and the one-time public key for the merchant (U_{OTP-M}). The PSI contains the information the shipping carrier needs to deliver the products to the customer in such a way that the merchant can not either see or tamper with. Similarly, the PPI contains the credit card information the credit card company needs to process the payment request from the customer in such a way that the merchant can not either see or tamper with the information.

The customer's web browser constructs the PSI using the following procedure. The web browser first constructs the shipping information (SI). SI contains the mailing address of the customer, types of shipping (e.g., express, insured, etc.) and the other information needed for shipping the ordered products. Then, the customer's browser concatenates the SI, the order token from the merchant, and the two one-time asymmetric public keys for the carrier (U_{OTP-S1} and U_{OTP-S2}). The browser calculates the message digest of the whole concatenated message, and encrypts the message digest using the first one-time asymmetric private key (U_{OTS-S1}). The concatenated information is then encrypted using the public key of the carrier (S_P). Finally, the PSI is created by attaching the encrypted message digest to the concatenated information encrypted by S_P (Figure 3).

The PPI for the credit card company is constructed in a similar way. The customer's browser first constructs the payment information (PI). PI contains the information needed to obtain the payment approval from the credit card company. It contains the card holder name, the credit card number, expiration date, the requested amount and the card's security code. The browser concatenates the PI, the order token from the merchant, and the one-time asymmetric public key for the credit card company (U_{OTP-C}). The browser calculates the message digest of the whole concatenated information and encrypts the message digest using the one-time asymmetric private key (U_{OTS-C}). The concatenated information is encrypted using the public key of the credit card company (C_P). Finally, the PPI is created by attaching the encrypted message digest to the concatenated information encrypted by C_P (Figure 4).

After the PSI and PPI are constructed, the customer's browser creates the FOM using the following procedure. The browser concatenates the OI, the order token, the one-time asymmetric public key for the merchant (U_{OTP-M}), the PSI, and the PPI. The whole concatenated information is encrypted by the merchant's public key (M_P). Then, the message digest of the whole concatenated information is calculated by the customer's browser. The message digest is then encrypted by the one-time asymmetric private key for the merchant (U_{OTS-M}) and it is attached to the whole concatenated information encrypted by M_P to make it FOM (Figure 5). The FOM is transmitted from the customer's browser to the merchant web server (message ③).

After the FOM is delivered to the merchant's server, the procedure of FSSA consists of five phases.

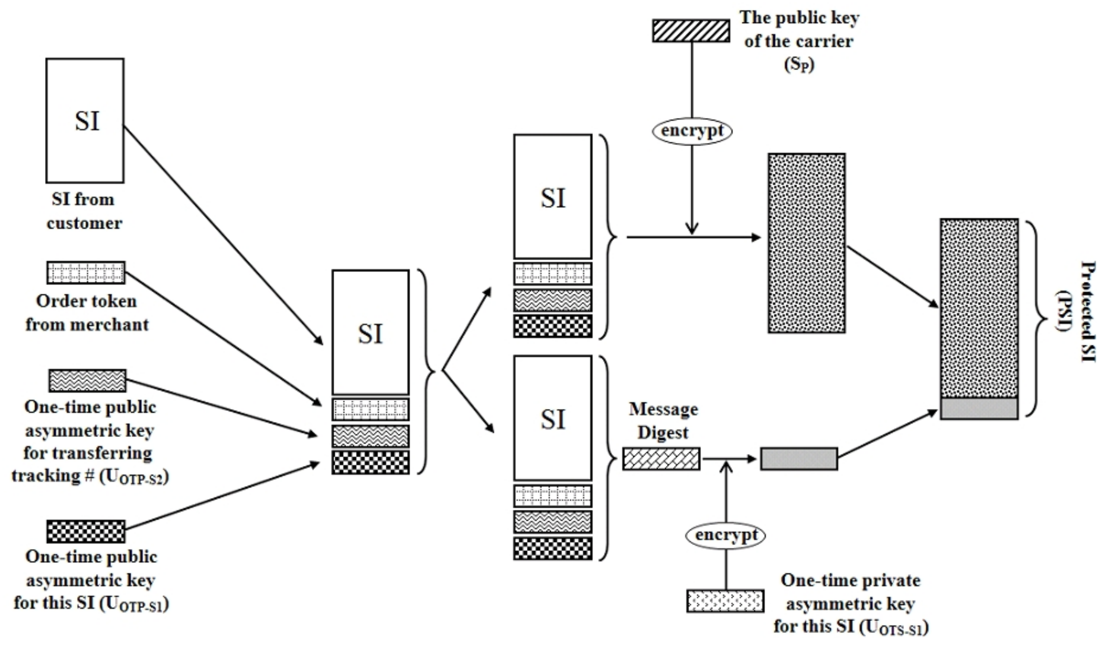

SI
SI from customer
Order token from merchant
One-time public asymmetric key for transferring tracking # (U_{OTP-S2})
One-time public asymmetric key for this SI (U_{OTP-S1})
SI
SI
SI
The public key of the carrier (S_P)
encrypt
Message Digest
encrypt
One-time private asymmetric key for this SI (U_{OTS-S1})
Protected SI (PSI)

Figure 3: Construction of the protected SI (PSI) message

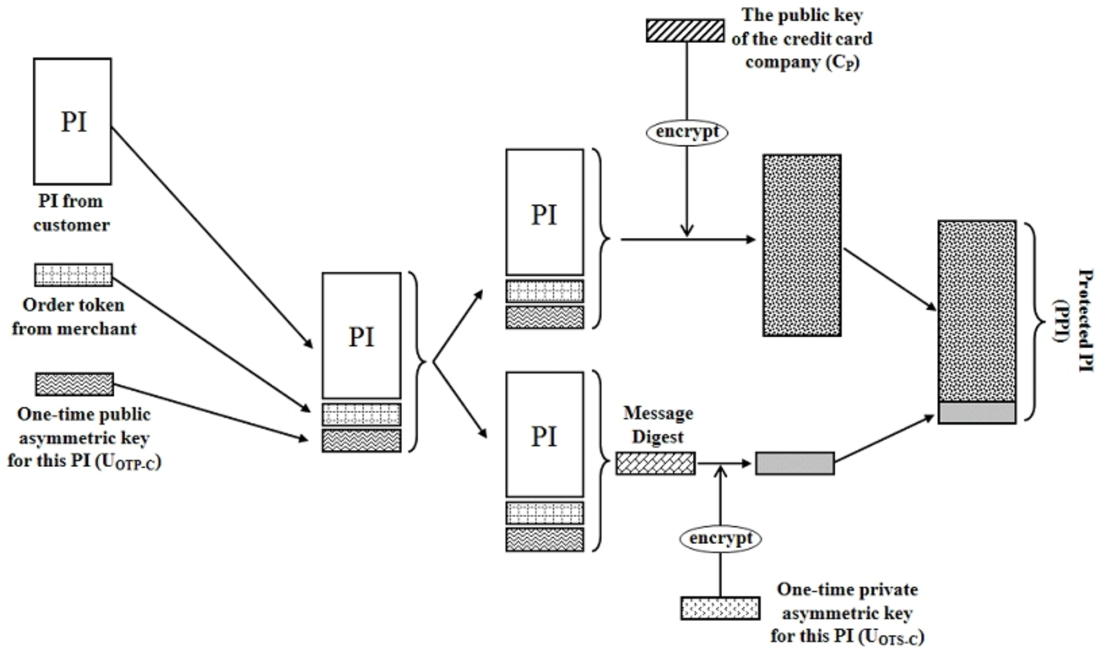

PI
PI from customer
Order token from merchant
One-time public asymmetric key for this PI (U_{OTP-C})
PI
PI
PI
The public key of the credit card company (C_P)
encrypt
Message Digest
encrypt
One-time private asymmetric key for this PI (U_{OTS-C})
Protected PI (PPI)

Figure 4: Construction of the protected PI (PPI) message

In the first phase, the merchant's web server performs the tasks shown in Figure 6. When the credit card company's server receives the PRM, it executes the tasks shown in Figure 7. If the merchant receives the PAM from the credit card company, the merchant performs the task shown in Figure 8. If the merchant receives a denial of payment, the whole procedure is terminated and it is notified to the customer. When the shipping carrier's server receives the SRM, it performs the tasks in Phase 4 (Figure 9). When the

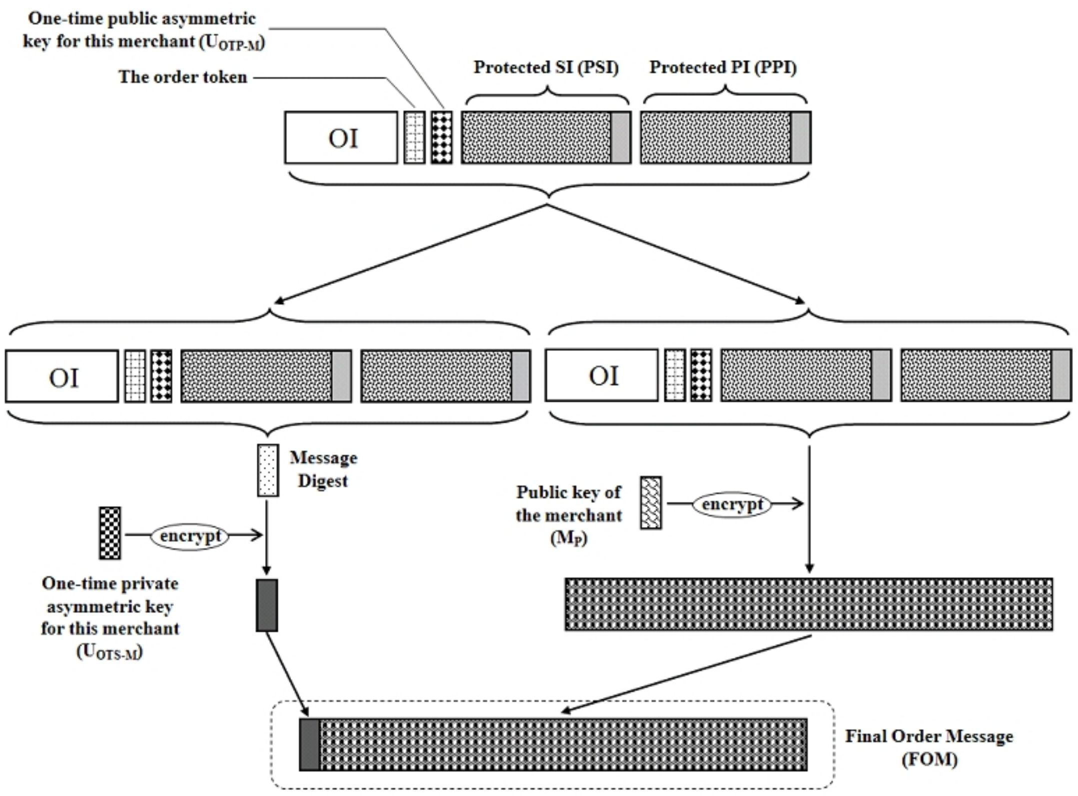
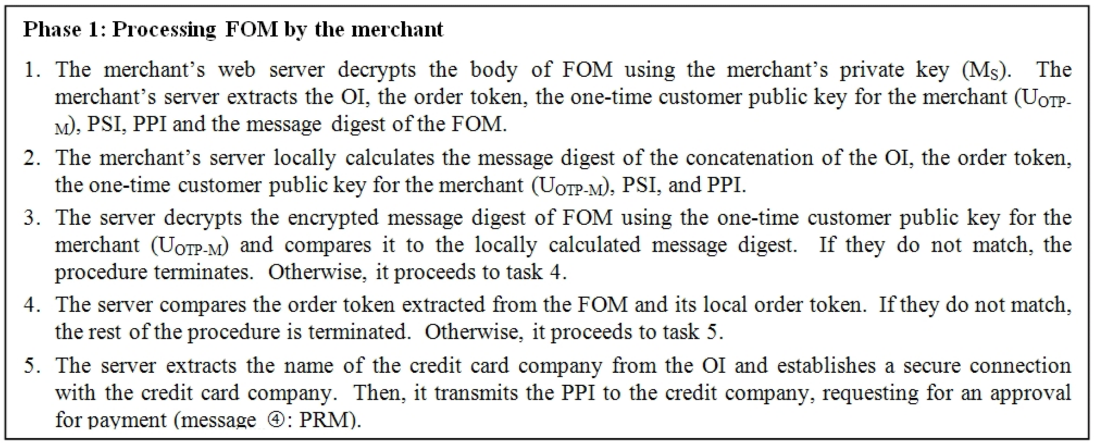

One-time public asymmetric key for this merchant (U_{OTP-M})
The order token
Protected SI (PSI)
Protected PI (PPI)
OI
OI
OI
Message Digest
Public key of the merchant (M_P)
encrypt
encrypt
One-time private asymmetric key for this merchant (U_{OTS-M})
Final Order Message (FOM)

Figure 5: Construction of the final order message (FOM)



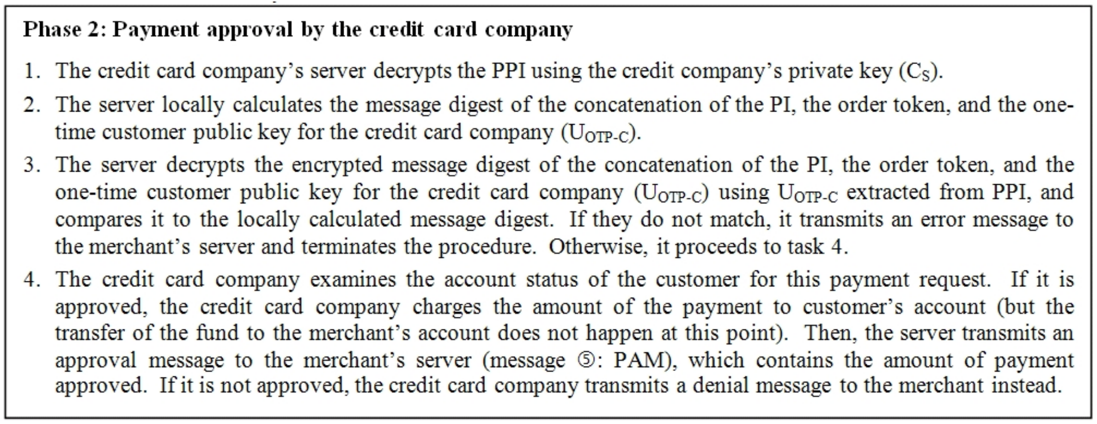
Phase 1: Processing FOM by the merchant

1. The merchant's web server decrypts the body of FOM using the merchant's private key (M_S). The merchant's server extracts the OI, the order token, the one-time customer public key for the merchant (U_{OTP-M}), PSI, PPI and the message digest of the FOM.
2. The merchant's server locally calculates the message digest of the concatenation of the OI, the order token, the one-time customer public key for the merchant (U_{OTP-M}), PSI, and PPI.
3. The server decrypts the encrypted message digest of FOM using the one-time customer public key for the merchant (U_{OTP-M}) and compares it to the locally calculated message digest. If they do not match, the procedure terminates. Otherwise, it proceeds to task 4.
4. The server compares the order token extracted from the FOM and its local order token. If they do not match, the rest of the procedure is terminated. Otherwise, it proceeds to task 5.
5. The server extracts the name of the credit card company from the OI and establishes a secure connection with the credit card company. Then, it transmits the PPI to the credit company, requesting for an approval for payment (message ④: PRM).

Figure 6: Tasks to process FOM at the merchant's server

shipping carrier delivered the package to the customer, the shipping carrier performs the tasks in Phase 5 (Figure 10).

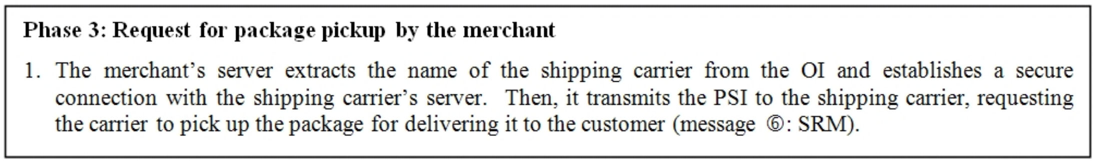
When the merchant receives the DCM from the shipping carrier, it forwards the DCM to the credit card company, requesting for the payment. The credit card company holds the payment for short time (e.g., three days) after the arrival of the DCM from the merchant. This time period allows the customer to file a claim (message ⑨: CCM), if there is any problem about the delivered product, which will suspend the transfer of the fund to the merchant's account. If the customer does not submit any claim during the



Phase 2: Payment approval by the credit card company

1. The credit card company's server decrypts the PPI using the credit company's private key (C_S).
2. The server locally calculates the message digest of the concatenation of the PI, the order token, and the one-time customer public key for the credit card company (U_{OTP-C}).
3. The server decrypts the encrypted message digest of the concatenation of the PI, the order token, and the one-time customer public key for the credit card company (U_{OTP-C}) using U_{OTP-C} extracted from PPI, and compares it to the locally calculated message digest. If they do not match, it transmits an error message to the merchant's server and terminates the procedure. Otherwise, it proceeds to task 4.
4. The credit card company examines the account status of the customer for this payment request. If it is approved, the credit card company charges the amount of the payment to customer's account (but the transfer of the fund to the merchant's account does not happen at this point). Then, the server transmits an approval message to the merchant's server (message ⑤: PAM), which contains the amount of payment approved. If it is not approved, the credit card company transmits a denial message to the merchant instead.

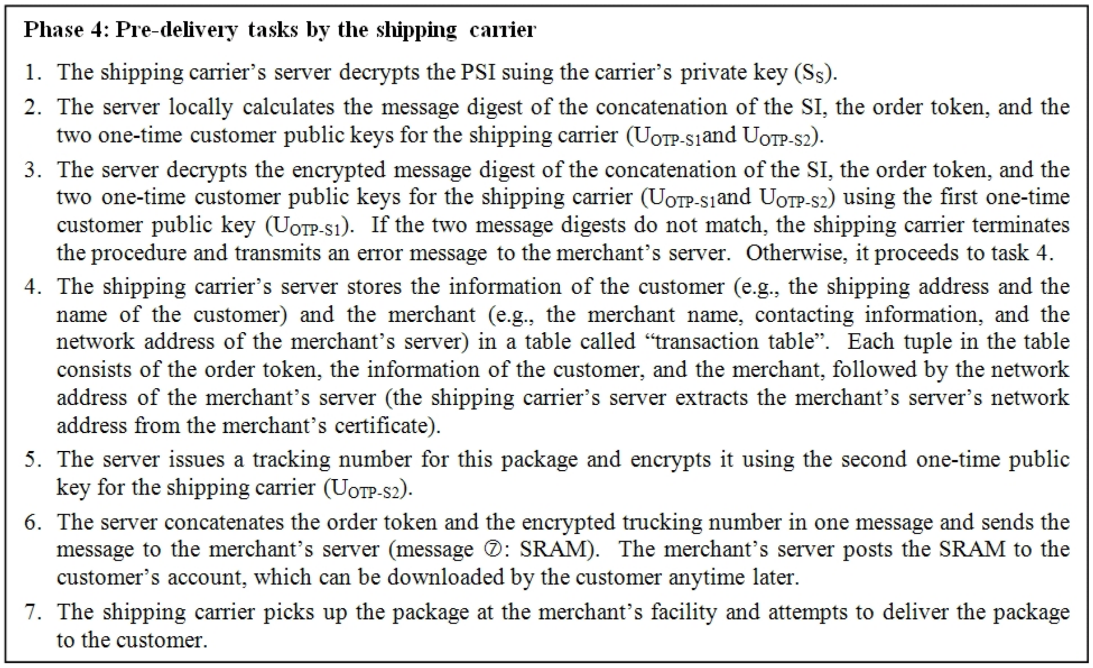
Figure 7: Tasks to process payment request the credit card company's server



Phase 3: Request for package pickup by the merchant

1. The merchant's server extracts the name of the shipping carrier from the OI and establishes a secure connection with the shipping carrier's server. Then, it transmits the PSI to the shipping carrier, requesting the carrier to pick up the package for delivering it to the customer (message ⑥: SRM).

Figure 8: Task to request package pickup

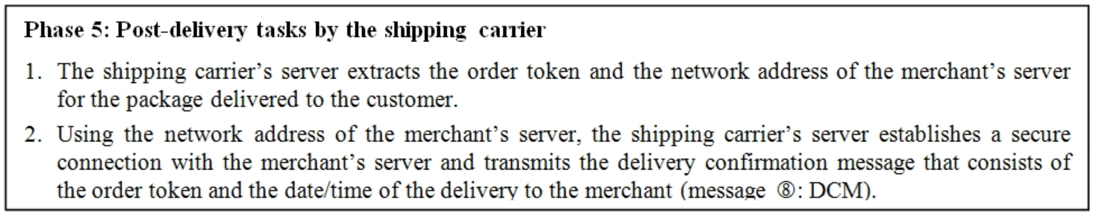


Phase 4: Pre-delivery tasks by the shipping carrier

1. The shipping carrier's server decrypts the PSI suing the carrier's private key (S_S).
2. The server locally calculates the message digest of the concatenation of the SI, the order token, and the two one-time customer public keys for the shipping carrier ($U_{OTP-S1}$and U_{OTP-S2}).
3. The server decrypts the encrypted message digest of the concatenation of the SI, the order token, and the two one-time customer public keys for the shipping carrier ($U_{OTP-S1}$and U_{OTP-S2}) using the first one-time customer public key (U_{OTP-S1}). If the two message digests do not match, the shipping carrier terminates the procedure and transmits an error message to the merchant's server. Otherwise, it proceeds to task 4.
4. The shipping carrier's server stores the information of the customer (e.g., the shipping address and the name of the customer) and the merchant (e.g., the merchant name, contacting information, and the network address of the merchant's server) in a table called "transaction table". Each tuple in the table consists of the order token, the information of the customer, and the merchant, followed by the network address of the merchant's server (the shipping carrier's server extracts the merchant's server's network address from the merchant's certificate).
5. The server issues a tracking number for this package and encrypts it using the second one-time public key for the shipping carrier (U_{OTP-S2}).
6. The server concatenates the order token and the encrypted trucking number in one message and sends the message to the merchant's server (message ⑦: SRAM). The merchant's server posts the SRAM to the customer's account, which can be downloaded by the customer anytime later.
7. The shipping carrier picks up the package at the merchant's facility and attempts to deliver the package to the customer.

Figure 9: Tasks performed by the shipping carrier before the delivery of the products

time period, the fund will be transferred to the merchant's account.


Phase 5: Post-delivery tasks by the shipping carrier

1. The shipping carrier's server extracts the order token and the network address of the merchant's server for the package delivered to the customer.
2. Using the network address of the merchant's server, the shipping carrier's server establishes a secure connection with the merchant's server and transmits the delivery confirmation message that consists of the order token and the date/time of the delivery to the merchant (message ⑧: DCM).

Figure 10: Tasks performed by the shipping carrier after the delivery of the products

3.3 Analyses on the security requirements

The possible threats to e-commerce applications are classified as those from internal and external attackers. It is assumed that internal attackers can access anything in a server, including the information held by application processes running in the server. The security threats external attackers possibly carry out are eavesdropping, replay, man-in-middle, masquerading, traffic analysis, bug exploits, and non-repudiations of transactions. Protections from internal attackers:

(a) At merchants: The attackers have access to any information stored in the server as long as it is not encrypted by a key the attackers do not have access. Thus, the only information accessible to the attackers is about the product(s) ordered by customers, the shipping carrier and the credit card company of the transactions, the information that recognizes each customer (but not customer's real identity) and the network address of the customers. The attackers do not have access to the name, mailing address, or the credit card number, which prohibits the attackers from reconstructing the complete information about transactions. The attackers can modify the contents of the PPI and PSI by overwriting their contents, but such modifications can not be performed in a way the attackers control the other two parties as the attackers wish.

(b) At shipping carriers: The shipping carriers' servers hold the name, shipping address of the customers, and the merchant's ID (from the order tokens), but they do not hold the information about the products ordered by the customers or their credit card information. However, using the merchant's ID, the attackers may identify the name of the merchants. If the merchant's name provides any clue about the products they deal with, it is possible for the attackers to infer the products ordered by customers.

(c) At credit card companies: The credit card company's servers hold the information that identifies each customer, such as the customer's full name, mailing address, telephone number, credit card number, and the amount of the payment for each transaction, but not the information about the products. The only possible risk is that, similar to the attackers at shipping carriers, the order tokens may allow the attackers to identify the merchants, which lets the attackers infer what products were purchased by the customers. Protections from external attackers:

Eavesdropping: Protection against eavesdropping on the message contents during their transmissions is provided by encryption possibly in multiple layers. For example, SSL, IPSec, STS, WPA, and encryptions performed in the application processes prevents the message contents from being released to external attackers, assuming the following two conditions:(1)the encryption keys are securely stored, and(2)the keys for a secure connection are exchanged properly (man-in-middle attacks are theoretically possible for Diffie-Hellman key-exchange algorithm, which is the basis for IPSec). Regarding the first condition (1), since most of the important keys at a customer host computer are "one-time", eavesdropping will not happen unless an attacker can access the one-time keys while a transaction is in progress. The second problem (2) is dealt with in the section of "man-in-middle attacks" later.

Replay: The external attackers may have physical access to network equipment, allowing them "wiretap" to the messages on the fly. Although this allows the attackers to perform replays, they will not cause any effect to a transaction since each of the eight messages in FSSA is unique. The uniqueness of each message is guaranteed by the unique order token issued by a merchant. If two identical messages arrive at any party, whichever arrives later will be automatically dropped without any effect to the transaction.

Masquerading: Masquerading merchants, shipping carriers, or credit card companies is not possible, since the correctness of their public keys is guaranteed by their certificates. If an attacker tries to masquerade a customer using a fake shipping address, such masquerading can be detected by comparing the fake address and the customer's correct address registered to the customer's credit card account.

Man-in-middle: It is known that man-in-middle attacks can be performed by intercepting Diffie-Hellman key exchanges when a new secure connection is established. Since the contents of message ③, ④, ⑥, and ⑦ are protected by customer's one-time keys, and since attackers can not masquerade customers, the contents of the message can not be accessed to attackers or the attackers can not modify a portion of the messages in any controlled way (the message digest will detect such modifications). Attackers can not meaningfully synthesize the messages either, since they do not know the customer name, address, and the credit card number. Messages ⑤ and ⑧ are also protected from the main-in-middle attacks since they are encrypted by a public key endorsed by a certificate and are digitally signed. The only messages attackers can access by man-in-middle arracks are message ① (POM) and ② (OCM). POM contains only the information about the products ordered by a customer, the name of the shipping carrier and the credit card company, but nothing else. OCM contains the order token and OI, but neither of them contains the customer's name, shipping address or the credit card number. Reconstructing the complete information about a transaction requires the attackers to obtain the information from the three parties, such as the full name of the customer, which the attackers can not obtain. Thus, man-in-middle attacks, even to the known weakness in Diffie-Hellman key exchange algorithm, will not cause any threat to FSSA.

Traffic analysis: Traffic analysis can take place between any two parties.

(a) Between a customer and a merchant: It was assumed that external attackers have access only to the packet headers, but not to their payload field when transport-layer secure connections are used. Since the merchant's network address may identify the merchant (and possibly what business the merchant does), it is important to make sure that customer's network address does not identify the customer. If the above conditions are met, the traffic analyses will not cause a risk of privacy leak.

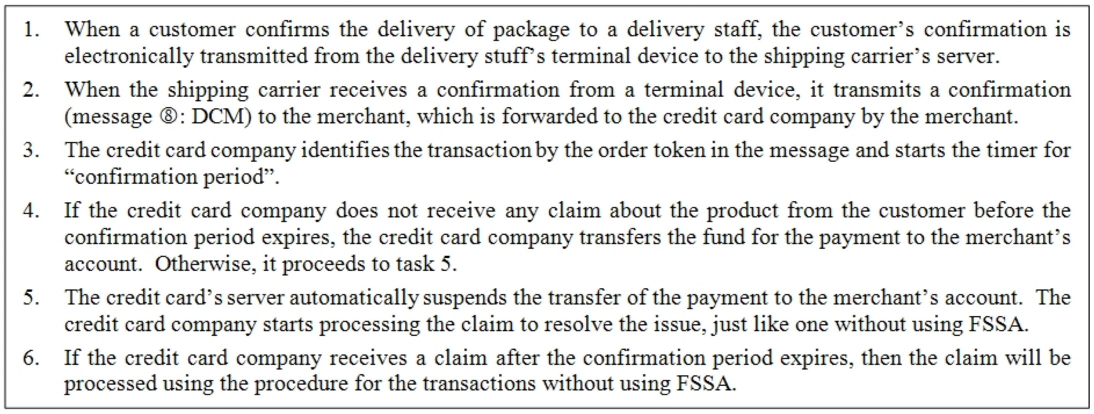
(b) Between a merchant and a carrier: From the network addresses of a merchant and a shipping carrier, they can be identified easily from their network addresses. However, determining the merchant and the shipping carrier will not reveal who is the customer. Thus, anonymity of the customer is preserved.

(c) Between a merchant and a credit card company: In the same way as (b) above, anonymity of the customer will be preserved as long as the transport-layer connections are properly performed.

Bug exploits: We defined ”bug exploits” as any security breach at a host computer due to software bugs, which, in the worst case, can give the full control of a host computer to attackers, just like the ones an administrator of a host computer would have. Therefore, the security risks by bug exploits are essentially the same as the ones from the internal attackers. As described before, since none of the three servers holds the complete information about each transaction, an external attacker who manages to gain the full access to a server still can not obtain the complete information about a transaction by the same reasons for the internal attackers.

Non-repudiation: Repudiations of e-commerce transactions by customers (customers deny that they had ordered some products at a merchant's online site even though they had done so) are not possible because none of replay, man-in-middle, and masquerading can happen. Repudiations of transactions by a merchant, a shipping carrier, or a credit card company is impossible, since (1) each message issued by a party is digitally signed by their private key, (2) each public key has been endorsed by a certificate from a legitimate certificate authority, and (3) each message in FSSA is uniquely identified by the order token.

Confirmation of the delivered products: Most of the shipping services available today provide ”delivery confirmation”. However, they do not provide ”confirmation of delivered products”. When delivery staff ask customers' signature on the delivery of a package, customers usually do not have a chance to confirm the products in the packages nor the condition of the products. It is usually after customers sign for delivery confirmation that they can see the products or the conditions of the products in the packages. FSSA logically provides ”confirmation of delivered products” by holding the transfer of the customers' payment to the merchants for a specific time period, called ”confirmation period”. Dishonest merchants can fake ”confirmations of delivered products” to credit card companies. However, such misbehaviors are easily detected and prevented by termination of the contract with the credit card companies. Figure 11 describes the procedure for ”confirmation of delivered products”.



1. When a customer confirms the delivery of package to a delivery staff, the customer's confirmation is electronically transmitted from the delivery stuff's terminal device to the shipping carrier's server.
2. When the shipping carrier receives a confirmation from a terminal device, it transmits a confirmation (message ⑧: DCM) to the merchant, which is forwarded to the credit card company by the merchant.
3. The credit card company identifies the transaction by the order token in the message and starts the timer for "confirmation period".
4. If the credit card company does not receive any claim about the product from the customer before the confirmation period expires, the credit card company transfers the fund for the payment to the merchant's account. Otherwise, it proceeds to task 5.
5. The credit card's server automatically suspends the transfer of the payment to the merchant's account. The credit card company starts processing the claim to resolve the issue, just like one without using FSSA.
6. If the credit card company receives a claim after the confirmation period expires, then the claim will be processed using the procedure for the transactions without using FSSA.

Figure 11: the FSSA procedure that realizes confirmation of delivered products

4 Performance evaluation

The feasibility of FSSA was evaluated from the viewpoint of its runtime overhead. The overhead of FSSA was quantified and compared to that of an existing system that did not perform any specific security feature except encrypting the messages exchanged between two host computers. The two performance metrics of our interest were the response time and the throughput of online shopping transactions issued by customers. The response time and the throughput of the transactions were studied using experiments implemented on a test bed.

4.1 Experiment test bed

The test bed was developed in an isolated local area network where four host computers were connected. Each of the merchant's, credit card company's and shipping carrier's server processes was executed in a host computer, while multiple customers were simulated at a host computer (called ”customer simulator”). The merchant server was executed by a PC with an Intel Core 2 Quad CPU running at 2.4 GHz. Each of the shipping carrier, the credit card company, and the customer simulator was executed by a PC with an Intel Pentium 4 CPU running at 3.2 GHz. Cable length between two hosts was less than 10 feet to minimize the impact from the signal propagation delay.

4.2 Experiment designs

The response time and the throughput of online shopping transactions were measured when the workload was increased. The amount of the workload was controlled by adjusting the number of the transactions issued from customers to the merchant server. The response time was defined to be the elapsed time (T_Δ)

since the arrival time (T_S) of FOM (message ③ in Figure 1) at the merchant's server to the completion time (T_E) of processing SRAM (message ⑦) at the merchant's server (T_Δ = T_E - T_S).

In generating customers' requests, Poisson distribution was used to control the interval time between two consecutive requests. The tested intervals were 0.1, 1, 10, 50, 100, 150, 175, 200, 225, 235, 250, 300, and 500ms. Each experiment simulated 1,000 transactions (n = 1,000) for one of the 14 workload levels and the same experiment was repeated for 20 times (m = 20). The average response time was defined as:

$$\frac{\sum_{j=0}^{m}\sum_{i=0}^{n}(T_\Delta, i, j)}{m \times n}$$

where a tuple (T_Δ, i, j) indicates the response time for the i-th transaction in the j-th run of the experiment. We reset all the hosts at the end of each run of the experiment. For the throughput, we ran an experiment that generated 100,000 transactions to observe how many transactions were completed at the merchant server (completion of processing SRAM message at the merchant's server) per second. The same experiment was repeated 20 times and the average transaction throughput of the 20 runs was calculated.

4.3 Observations of the outcomes from the experiments

Transaction Response Time: Figure 12 shows the observed average response time for the 14 different levels of the workload offered to the merchant's server. We observed "takeoff" at 8 transactions per second (125ms interval - at (a) in Figure 12) and 4.19 transactions per second (225ms interval - at (b)) for the existing system (without using FSSA) and the proposed system (with FSSA) respectively. Its ratio was 125:225 = 1.0:1.8. At the highest workload we tested in the experiments (transaction interval of 0.1ms or 10,000 transactions per second), the response time for the existing system and the proposed system was 15.8 and 29.4ms respectively. The ratio of the response time of using FSSA to that of without using FSSA was 15.8:29.4 (approximately 1.00:1.86).

Transaction Throughput: Figure 13 shows the observed transaction throughput for the 14 different levels of workload offered to the merchant's server. The peak throughput was 7.8 transactions for the existing system (without using FSSA) and 4.2 transactions for the proposed system (using FSSA). It was demonstrated that the transaction throughput and the number of transactions issued by customers approximately matched for both of the existing and the proposed systems up to the peak throughout.

5 Conclusion

New security architecture, called Fail-Safe Security Architecture (FSSA), is proposed to prevent privacy leaks from e-commerce servers. FSSA is designed to protect customers' privacy even if the e-commerce merchants' servers are hijacked by attackers or even if the administrators of the servers involve in the leaks. As a result, FSSA protects customer's privacy information from leaking not only to external attackers, but to internal attackers, including the administrators of the e-commerce servers.

The technical challenge we faced was in meeting the legal requirements at the same time the proposed solution must protect the customers' privacy in the worst cases: the involvements of the insiders and hijacks of the server host computers by external attackers. In many countries, their law requires the complete information about each e-commerce transaction to be presented on a court order. In addition to the legal requirements, the three parties of e-commerce merchants, shipping carriers, and credit card companies may have to collaborate in processing claims submitted by customers. The proposed security

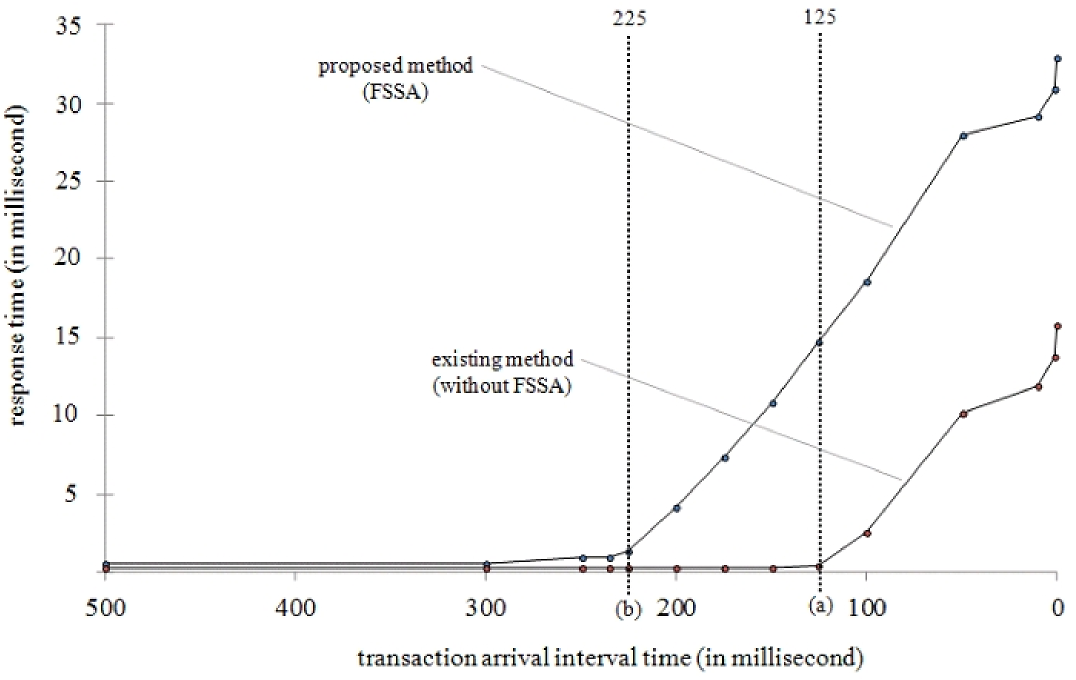


Figure 12: Transaction response time for a system using FSSA and a system without using FSSA

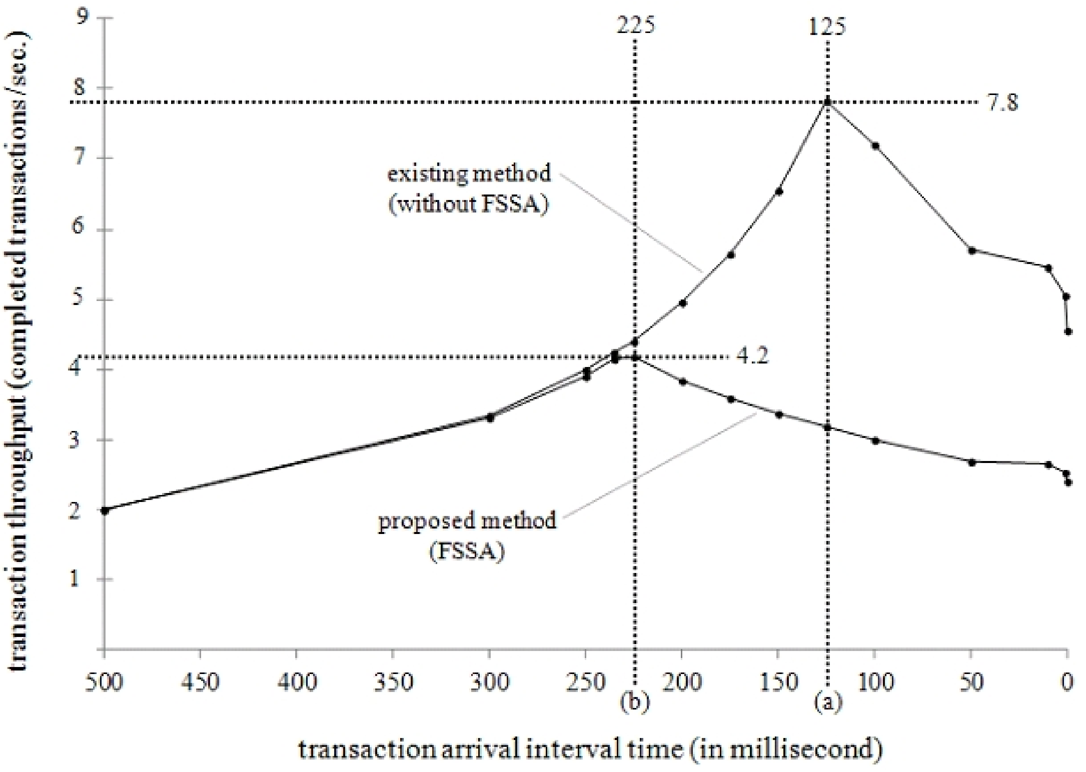


Figure 13: Transaction throughput of a system using FSSA and a system without using FSSA

architecture solves the problems, by making sure that each of the three parties holds only the information needed for their business.

To study the feasibility of FSSA from the viewpoint of its runtime overhead, we evaluated the overhead of FSSA from an existing system that does not perform any security means except secure connections. Our performance studies suggested that the cost factor of running FSSA is 1.8 (1.8 times more computational power) to achieve the same response time and throughput compared to the existing architecture.

From our analyses on the types of the security threats FSSA covers and its overhead factor, the proposed security architecture will contribute to the promotion of wide adoption of e-commerce transactions by eliminating the fear of privacy leaks from potential e-commerce customers. Because of the evidences, we are sure that the main objectives in the proposed security architecture are met and it contributes to safety in e-commerce.

The future work of FSSA includes the following issues. FSSA is not currently designed to handle shipping to foreign countries, which necessitates declaration of the package contents to customs. Another missing feature is the capability for the shipping carrier and the credit card company to confirm if the shipping address for product delivery match the one registered to credit card company without the merchants' seeing them. This is another important missing feature since most of e-commerce merchants rely on this method to prevent credit card frauds.

References

[1] T. 12-BANKS and C. .-F. D. I. C. BANKING. Retention of records by insured depository institutions, January 2007. pages: 1084-1087.

[2] B. Benatallah, H. R., and M. Nezhad. Service oriented architecture: Overview and directions. *Advances in Software Engineering*, 5316:116–130, 2008.

[3] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proc. of Advance in Cryptography (CRYPTO'88), Santa Barbara, California, USA, LNCS*, volume 403, pages 319–327. Springer-Verlag, 1990.

[4] I. Corporation. An overview of the ibm set and the ibm commerce point products. Technical report, IBM Corporation, 1998.

[5] A. Cummings, T. Lewellen, D. Mcintire, A. P. Moore, R. Trzeciak, and C. Officer. Insider threat study: Illicit cyber activity involving fraud. Technical report, Institute, Carnegie Mellon University, 2012.

[6] Z. Eslami and M. Talebi. A new untraceable off-line cache system. *Electronic Commerce Research and Applications*, 10(1):59–66, 2011.

[7] Z. Ge and Z. Chao. A biometric-based framework for digital rights protection. In *Proc. of the 7th International Conference on Signal Processing (ICSP'04), Troia, Turkey*, volume 3, pages 2314–2317, September 2004.

[8] S. Glover and I. Benbasat. A comprehensive model of perceived risk of e-commerce transactions. *International Journal of Electronic Commerce*, 15(2):47–48, 2011.

[9] L. Hars. Discryption: Internal hard-disk encryption for secure storage. *Computer*, 40(6):103–105, 2007.

[10] D. Lekkas and D. Spinellis. Implementing regular cash with blind fixed-value electronic coins. *Computer Standards and Interfaces*, 29(3):277–288, 2007.

[11] U. T. Mattsson. A practical implementation of transparent encryption and separation of duties in enterprise databases: Protection against external and internal attacks on databases. In *Proc. of the the 7th IEEE International Conference on E-Commerce Technology (CEC'05), Munich, Germany*, pages 559–565. IEEE, July 2005.

[12] D. Mazieres and D. Shasha. Don't trust your file server. In *Proc. of the 8th Workshop on Hot Topics in Operating Systems (HOTOS'01), Elmau, Germany*, pages 113–118. IEEE, May 2001.

[13] M. P. Papazoglou and W.-J. van den Heuvel. Service oriented architectures: Approaches, technologies and research issues. *International Journal on Very Large Data Bases*, 16:389–415, 2007.

[14] B. Parno, J. M. McCune, D. Wendlandt, D. G. Andersen, and A. Perrig. CLAMP: Practical prevention of large-scale data leaks. In *Proc. of the 30th IEEE Symposium on Security and Privacy (SP'09), Berkeley, California, USA*, pages 154–169. IEEE, May 2009.

[15] J. M. Pavia, E. J., Veres-Ferrer, and G. Foix-Escura. Credit card incidents and control systems. *International Journal of Information Management*, 32(6):501–503, 2012.

[16] M. Rennhard, S. Rafaeli, L. Mathy, B. Plattner, and D. Hutchison. Towards pseudonymous e-commerce. *Electronic Commerce Research*, 4:83–111, 2004.

[17] D. Tygar. Atomicity versus anonymity: Distributed transaction electronic commerce. In *Proc. of the 24th International Conference on Very Large Data Bases (VLDB'98), New York City, New York, USA*, pages 1–12, August 1998.

[18] U. Uludag, S. Member, S. Pankanti, A. K. Jain, S. Member, S. Prabhakar, Anil, and K. Jain. Biometric cryptosystems: Issues and challenges. *Proceedings of the IEEE*, 92(6):948–960, June 2004.

[19] M. E. Whitman. Enemy at the gate: Threats to information security. *Communications of the ACM*, 46(8):91–95, 2003.

[20] X. Zhang, Q. Huang, and P. Peng. Implementation of a suggested E-commerce model based on SET protocol. In *Proc. of the 8th ACIS International Conference on Software Engineering Research, Management and Applications (SERA'10), Montreal, Quebec, Canada*, pages 67–73, May 2010.

Author Biography

Hiroshi Fujinoki is currently an associate professor at Department of Computer Science at Southern Illinois University Edwardsville (SIUE). He received a Ph.D. degree from the Department of Computer Science and Engineering at the University of South Florida in August 2001. His research areas include routing in large-scale networks, traffic engineering, server performance optimization, and network/cyber security. Dr. Fujinoki has been working especially on security threats by the insiders for e-commerce network applications and private/community clouds.

Christopher A. Chelmecki[1] is a former graduate student at the Department of Computer Science of SIUE. Chris' primary interest is in network/cyber security, especially in developing new technologies, such as honeypots, to cope with recent security threats. Mr. Chelmecki has a couple of publications in the area, some of which have been cited by researchers in the world. Mr. Chelmecki graduated from SIUE in December 2011.

David M. Henry is a former graduate student at the Department of Computer Science of SIUE. David's primary interest is cryptography. David Henry graduated from SIUE in May 2012.

[1]No Photo is available.