SPP Secure Payment Protocol: Protocol Analysis, Implementation and Extensions

by

Gerry Kovan

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Mathematics

in

Computer Science

Waterloo, Ontario, Canada, 2005

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Internet commerce continues to grow rapidly. Over 60% of US households use the internet to shop online. A secure payment protocol is required to support this rapid growth. A new payment protocol was recently invented at IBM. We refer to the protocol as SPP or Secure Payment Protocol.

This thesis presents a protocol analysis of SPP. It is essential that a thorough security analysis be done on any new payment protocol so that we can better understand its security properties. We first develop a method for analyzing payment protocols. This method includes a list of desirable security features and a list of proofs that should be satisfied. We then present the results of the analysis. These results validate that the protocol does contain many security features and properties. They also help understand the security properties and identify areas where the protocol can be further secured. This led us to extend the design of the protocol to enhance its security.

This thesis also presents a prototype implementation of SPP. Three software components were implemented. They are the Electronic Wallet component, the merchant software component and the Trusted Third Party component. The architecture and technologies that are required for implementation are discussed. The prototype is then used in performance measurement experiments. Results on system performance as a function of key size are presented.

Finally, this thesis presents an extension of SPP to support a two buyer scenario. In this scenario one buyer makes an order while another buyer makes the payment. This scenario enables additional commerce services.

Acknowledgements

I would like to thank my thesis supervisor, Dr. Johnny Wong for his insightful suggestions and recommendations and also for being one the co-inventors of the SPP protocol. I would also like to thank Professor Richard Holt and Professor Grant Weddell for their time to be the readers of my thesis. Thanks to Lev Mirlas, who provided guidance during the early stages of the thesis. He was also a co-inventor of the SPP protocol.

This thesis would not have been possible without the support of my family. I would first like to thank my wife Natalie and my baby boy Daniel for their encouragement and support and most of all for their inspiration. I would also like to thank my parents for their love and support and for providing me the educational opportunities and resources necessary to pursue my academic goals. I would like to thank my sister Viola for her help and encouragement and for being a great role model for me throughout my life.

Finally, I would like to thank IBM for supporting my academic studies. IBM supported me both financially and with the time to pursue a Masters degree. IBM is truly a great company to work for.

Table of Contents

List of Tables

List of Figures

1 Introduction

Internet commerce has been booming over the last few years and it continues to do so. To enable the rapid growth of internet commerce there is a need for a secure payment protocol. A secure payment protocol is required to prevent the many types of fraudulent transactions that occur today. Fraudulent transactions can occur by buyers who purchase goods and services online with a stolen credit card or by merchant impersonators who pose as legitimate merchants in order to obtain valid credit card numbers from unsuspecting shoppers. When a problem does occur in a payment transaction between the participants, there is a need for a dispute resolution process. A payment protocol must therefore provide for a way to resolve disputes. There currently exist several payment protocols that are prevalent in the industry today that offer various levels of security. Two popular protocols today are SET [SET] and Cybercash [Cybercash]. A new protocol has recently been invented at IBM. This new protocol will be referred to as SPP [SPP] (Secure Payment Protocol) in this document.

In a SPP payment transaction, a TTP (Trusted Third Party) acts as a mediator between the buyer and the merchant in order to facilitate payment transactions. The protocol can be extended to multiple TTP's. In this scenario the buyer would be represented by one TTP and the merchant would be represented by another TTP.

For any new payment protocol it is critical that a thorough security analysis be performed. A security analysis should attempt to find any and all ways that the security of the protocol can be compromised and make recommendations to further enhance security of the payment protocol.

This thesis has four main goals. They are:

1. design a method for performing a security analysis of a payment protocol and use this method to analyze the single TTP scenario of the SPP protocol,
2. design and implement a prototype simulation of the single TTP scenario of the SPP protocol,
3. evaluate the performance of SPP using the prototype implemented, and
4. design an extension to SPP that will support the multiple buyer scenario.

Two approaches were taken to analyze a payment protocol. The first approach involved creating a list of desired features of a secure payment protocol and analyzing how the SPP protocol supported each of these features. Existing protocols were investigated and the desirable features of these protocols were used to create the list. These features include:

1. Dispute Resolution,
2. Confidentiality,
3. Anonymity,
4. Non-Repudiation,
5. Message Integrity, and
6. Availability and Reliability.

The second approach is what we call a proof analysis. It involves creating a list of proof requirements for each participant involved in a payment transaction in order for them to feel secure and analyzing the payment protocol to determine if these

requirements are indeed satisfied. As an example, the merchant requires proof that the customer has agreed to the terms and conditions of the order and authorized the transaction. The reason the merchant requires an undeniable proof of this is that if a dispute arises in the future, it can easily be resolved. This has the net affect of avoiding potential monetary losses as well as unnecessary problems.

A prototype SPP payment protocol was implemented as part of this thesis. The implementation uses a synchronous HTTP request/response architectural style to communicate messages between the participants in a SPP payment transaction. To implement the prototype the following software systems had to be implemented:

1. SPP Electronic wallet application used by the buyer or shopper,
2. Simulation of a merchant commerce server running SPP merchant software, and
3. TPP SPP server.

The prototype was built using advanced technologies such as the IBM Java Cryptography Extensions API, which provides the PKCS cryptography algorithms used for signing messages and verifying signed messages, IBM XML 4J parser which parses and generates the XML SPP messages, Java Plug-in technology to implement the SPP electronic wallet application, eXtensible Schema Language or XSL to define the XML SPP messages, DB2 to be the data store, Entity Enterprise Java Beans (EJBs) to act as the persistent code layer. Entity EJBs are components defined in the J2EE specification that access data in a database in a transactional context. The application tools used to implement the prototype are, Visual Age for Java development environment for coding, Visual Age WebSphere Test Environment to execute and test the execution of the

payment system and Rational Rose Enterprise Edition to model the key java classes in the payment system using class diagrams.

The prototype implemented was then used to do a performance analysis of the protocol. The performance analysis focused on the impact of the use of digital signatures on the performance of the protocol. We measured the performance for key sizes of 512 bits, 1024 bits, 2048 bits and with digital signatures disabled.

We have also extended the SPP protocol to support the 2 buyer scenario. In a 2 buyer scenario, one person can order the goods or services and have another person actually pay for them. This scenario is useful for providing merchant services such as gift certificates.

The thesis is organized as follows. Chapter 2 presents an overview of online payment authorization processing. Chapter 3 presents an overview of two of the industry leading protocols, SET and Cybercash. For each of the protocols the main features are discussed and a typical transaction scenario is described. Chapter 4 describes the SPP protocol. Both the single TTP scenario and the multiple TTP scenarios are described. Chapter 5 contains the protocol analysis where security features of the protocol are discussed and analyzed. A proof analysis is performed and optional security enhancements are suggested. Chapter 6 presents the design of the SPP prototype that was implemented as part of this thesis. This includes a description of the software components that had to be built. Chapter 7 contains the performance analysis while chapter 8 discusses the extensions to the protocol to support the 2 buyer scenario. Conclusions and future work are discussed in Chapter 9.

2 Overview of Electronic Payment Processing

Payment processing in the online world is similar to traditional methods of processing credit card payments. In the online world, the store and the transaction are both virtual meaning that the card is "not present" in the transaction. The merchant can not physically see the card and verify the customer's name or signature on the card. In such a transaction, merchants are held liable for fraudulent transactions by the credit card companies. Merchants must take additional steps against online fraud such as verifying that the credit card information is being submitted by the actual card owner and protecting their online store and network infrastructure against hackers.

Payment processing can be divided into two major phases or steps: authorization and settlement. Authorization verifies that the card is active and that the customer has the necessary funds available to complete the transaction. Settlement involves transferring money from the customer's account to the merchant's account. In this thesis we will focus on the authorization phase.

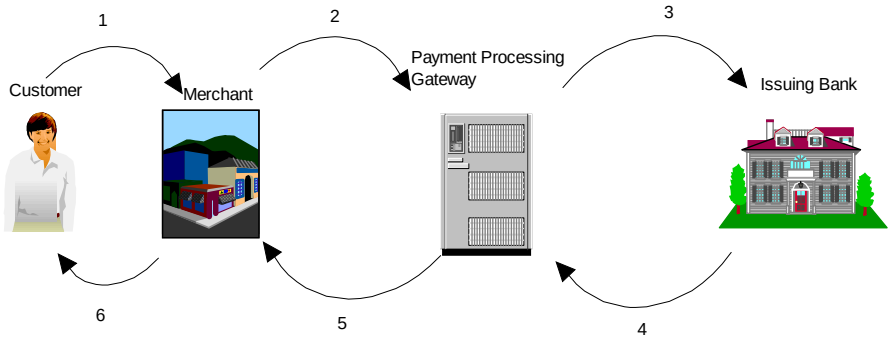

Figure 2-1 Typical Online Payment Authorization Process

A typical online payment authorization process is as follows. In step 1 the customer decides to make a purchase on a merchant's web site and proceeds to check-out and provide his/her credit card information. In step 2, the merchant's web site receives the customer order and payment information and sends the payment transaction information to the Payment Processing Gateway. In step 3, the Payment Processing Gateway sends the payment information to the Issuing Bank of the customer's credit card. In step 4, the Issuing Bank validates the payment information and sends the result, either an authorization number or a decline of payment to the Payment Processing Gateway who in turn passes the result to the merchant in step 5. The complexity of the validation algorithm depends on the payment protocol used. A simple form of validation would ensure that the billing address supplied by the customer along with the credit card information matches what the Issuing Bank has in its records. A more complex validation would use cryptography techniques such as digital certificates, digital signatures and digests to verify that the customer is the legitimate owner of the credit card. Finally in step 6 the merchant either accepts or rejects the transaction and ships the goods if necessary [VER1].

3 Overview of Existing Protocols

An overview of some of the prevalent payment protocols will be discussed in this section. The two main protocols discussed are SET and Cybercash as they share many of the same properties as the SPP protocol.

3.1 SET

SET is a protocol whose specification was published by VISA and MasterCard on February 1, 2001. The protocol's security is based on RSA cryptography, which includes 56 bit DES and RSA public/private key pairs with 1024 bit modulus.

3.1.1 Features of SET Protocol

The features of the SET protocol will be discussed in this section.

3.1.1.1 Confidentiality of Payment and Order Information

Two types of information are typically exchanged in a payment transaction: Payment and Order information. Payment information includes data such as payment method and payment method details. An example payment method is credit card and its payment details are card type (e.g. Visa, MasterCard), card number and expiry date. Order information consists of the details of the order such as product names and numbers, descriptions, prices, quantities and terms and conditions regarding the order.
The merchant cannot access the shopper's payment information. The shopper encrypts such information with the Payment Gateway's (acquirer bank's) public key. As a result, only the payment gateway can view the payment information [SET1] [SET2].

The bank cannot access the shopper's order information as the bank never receives any information regarding the contents of the order. The bank only receives the payment information so that it can authorize the payment [SET1] [SET2].

3.1.1.2 Integrity of Transmitted Messages

The SET protocol is responsible for ensuring the integrity of the messages that are sent between the participants in a transaction. Ensuring integrity means to ensure that the messages are not tampered with during transmission. When a merchant submits a message to the bank the following occurs [SET2]:

- Merchant creates a DES symmetric key.
- Merchant signs the message with its private key then encrypts the resulting message with the DES symmetric key. The resulting message will be referred to as (A).
- Merchant then encrypts (A) plus the DES symmetric key with the Banks public key-exchange key. The resulting message will be referred to as (B).
- Merchant transmits (B) to the bank

Only the bank can decrypt and view the message (B) and furthermore, the bank can verify that the contents of (B) were created by the merchant.

3.1.1.3 Cardholder Authentication

In SET, the cardholder is issued a digital certificate by his/her bank. This digital certificate is the means by which the cardholder is authenticated. It contains a digital signature signed by the bank which contains a digital hash of the card type, card number and expiry date. The bank uses this information to verify that the payment information entered by the shopper is valid and that the shopper is using his/her own credit card. This feature prevents the fraud situation where a shopper attempts to use a stolen credit card to pay for an order.

3.1.1.4 Merchant Authentication

The merchant is issued a certificate by the acquirer bank. This certificate enables other parties to authenticate the merchant. It contains information pertaining to the credit card brands (e.g. Visa, MasterCard) that the merchant can accept. This feature reduces the chances of merchant fraud where an entity pretends to be a legitimate merchant to obtain valid credit card information and subsequently use it in fraudulent transactions.

3.1.1.5 Interoperability Across Network and Software Providers

SET is an open specification/standard. Any software vendor can implement and sell their version of the product. Example vendors include IBM, Oasis, etc.

3.1.1.6 Protocol is not Dependant on Transport Security

SET ensures that all transmitted messages are protected. Therefore any transport protocols can be used to implement SET, e.g., HTTP, FTP and RPC.

3.1.2 SET Transaction Scenario

A typical SET payment transaction scenario [SET1] starts once the shopper notifies the merchant of his/her willingness to pay for an order. The transaction is described as follows:

1. The shopper selects a credit card (VISA, MasterCard) for payment from those that can be used with their SET electronic wallet software.
2. The shopper’s software initiates the payment process by sending a request to the merchant's software for the merchant’s and acquiring bank’s public keys.
3. The merchant replies to the shopper with the requested information.
4. The shopper's software verifies the merchant's and acquirer bank certificate. It then generates two packets of information, order information and payment information. The order information is encrypted with the merchant’s public key and the payment information is encrypted with the bank’s public key, so only the bank can see it. The shopper’s software transmits the encrypted order information and payment information to the merchant.
5. The merchant verifies the message for tampering and then proceeds with requesting an authorization from the bank. The merchant sends the bank a message containing a transaction id, the payment information provided by the shopper and the merchant’s certificate.
6. The bank decrypts the message and checks to ensure that the transaction identifier in the authorization matches that in the buyer's payment information packet and that the merchant has not tampered with the data in the buyer's payment information packet.
7. The acquirer bank then sends a request for payment authorization to the shopper's credit card issuer through customary bankcard networks.
8. The issuing bank responds with either an approval or denial.

9. The acquirer bank generates an authorization response message to be returned to the merchant. This message contains the issuing bank's response and an optional capture token to be used by the merchant when requesting capture of the payment.
10. The acquirer bank encrypts and sends the authorization response message to the merchant's software.
11. The merchant's software decrypts the authorization notice. If the transaction is approved, the merchant's software creates a purchase response message and sends it to the buyer's software. This message informs the buyer that payment was accepted and that the product or service that he/she has purchased will be delivered.

3.2 Cybercash

Cybercash is a company that provides a gateway service between the internet and traditional credit card authorization networks [Cybercash]. The Cybercash protocol is also based on RSA cryptography, which includes 56 bit DES and RSA public/private key pairs with 1024 bit modulus.

3.2.1 Features of Cybercash Protocol

The features of the Cybercash protocol will be discussed in this section.

3.2.1.1 Confidentiality of payment and order information

The merchant cannot access the shopper’s payment information as it is encrypted with the shopper’s private key [CYBP]. Only the Cybercash server has the public key to decrypt the payment information.

The bank cannot access the shoppers order information as the bank never receives any information regarding the contents of the order [CYBP]. The Cybercash server has access to both order and payment information.

3.2.1.2 Integrity of Transmitted Messages

Cybercash uses DES symmetric key encryption along with RSA digital signatures to ensure that messages are not tampered with during transmission [CYBP]. The process to accomplishing this is very similar to that used in SET.

3.2.1.3 Cardholder Authentication

The cardholder is issued a digital certificate by Cybercash upon registration. The Cybercash server maintains a record of public keys for all registered customers. Therefore, Cybercash can authenticate the cardholder, however, the merchant cannot [CYBP].

3.2.1.4 Merchant Authentication

The merchant is issued a digital certificate by Cybercash upon registration. The Cybercash server maintains a record of the public key of all registered merchants. Therefore, Cybercash can authenticate the merchant, however, the customer cannot [CYBP].

3.2.2 Cybercash Transaction Scenario

A typical Cybercash payment scenario starts once the shopper notifies the merchant of a willingness to pay for an order using the Cybercash payment method. The transaction is described as follows:

1. The merchant’s Cybercash software sends an invoice to the shopper's CyberCash electronic wallet software.
2. The shopper selects a credit card from the ones bound to their wallet. The shopper's Cybercash wallet then digitally signs and encrypts the invoice and credit card information with the key assigned to that Wallet-ID. The encrypted packet is sent to the Merchant's Cybercash payment server.
3. The merchant digitally signs and encrypts the payment packet with its Cybercash key. The packet is sent to the Cybercash server.
4. CyberCash decrypts the message and checks to make sure that the merchant has not tampered with the original invoice agreed upon by the shopper. The credit card information is encrypted and sent over dedicated lines to the merchant's acquiring bank.
5. The merchant's acquiring bank processes the merchant's request as it would for any other credit card transaction. It forwards the request through the card association’s network to the card issuing bank.
6. The card-issuing bank sends an approval or denial code back to the acquiring bank. The acquiring bank then sends this code to Cybercash.
7. Cybercash sends the merchant an encrypted message indicating success or failure of the credit card payment transaction.

8. The merchant’s software sends a message back to the shopper's Cybersash electronic wallet indicating success or failure of the payment transaction.

3.3 Other Protocols

3.3.1 SSL

Merchant sites use SSL [SSL] to enable their clients to send payment information such as credit card numbers, in a confidential manner. SSL provides many desirable properties for a payment system such as merchant authentication, client authentication (this is optional and in most cases not implemented), message integrity and confidentiality. Since SSL is supported by major client applications –such as browsers from Microsoft, Netscape, and Mozilla, SSL security services are readily available to users worldwide, without the need to install any proprietary software to activate secure browser sessions with server applications for on-line shopping.

SSL runs on top of a reliable transport protocol, such as TCP/IP, and below any application protocol, such as HTTP, encrypting/decrypting and calculating/verifying a secure hash code of the application protocol byte stream to ensure its privacy and integrity. In the handshake phase of the SSL 3.0 protocol, the server sends its public key certificate to the client to prove its right to use the domain name and the company name and is authenticated to the client using a challenge-response mechanism with a public key algorithm (most commonly RSA). The server's public key is also used to encrypt the client generated session keys and Message Authentication Code (MAC) computation which are subsequently sent to the server. MAC is a secure one-way hash-code computed

on the data being sent between sender and recipient to detect possible modifications on the data while in transit. SSL uses dedicated one-way hash functions - SHA or MD5, for MAC generation and verification. The recipient of the message recalculates the hash code on the received message and compares it with the received hash code. If the two hash codes are identical, the message is assumed to be "authentic". This assumption is based on the fact that no one else except the sender knows the shared secret key and so no one else can produce the correct hash code. Both the message and its MAC-code are encrypted with a symmetric algorithm to preserve the confidentiality of transactions.

Once the merchant has successfully received the payment information it can process the transaction. In many cases a manual process is used. For the case of payment by credit card, the merchant would call the credit card company to obtain an authorization code for the amount of purchase. Subsequently the merchant can send a batch of credit card authorizations to the credit card company in order to capture the payment.

4 Description of the SPP Protocol

The features of the secure payment protocol (SPP) investigated in this thesis will be discussed in this section.

4.1 Single TTP Scenario

This section describes the SPP protocol for the single trusted third party (TTP) scenario. In this scenario, the TTP is used by both the buyer and merchant in a payment transaction. SPP requires the availability of a public key authority. Any certificate

authority can be used, e.g., Pretty Good Privacy [PGP] and commercial providers such as Verisign [Ver].

The following notation is used in our description:

$CERT_j$	j's certificate (j = b for buyer, j = m for merchant, j = t for TTP)
H(x)	a cryptographic digest of x
$S_j(y)$	Signature on y using private key of j (j = b for buyer, j = m for merchant, j = t for TTP)
*abc	abc is an optional field

Table 4-1 Notation Used in Description of SPP Protocol

The basic steps of the SPP protocol are shown in Figure 3-1. For each step, any message transfer (if required) is secured using cryptographic technology, such as SSL.

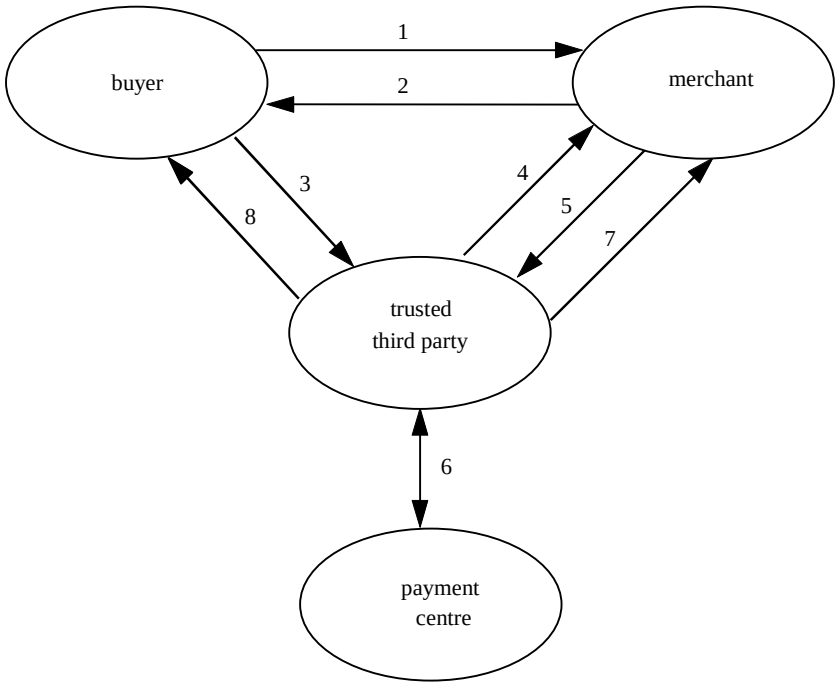

Figure 4-1 Payment Protocol with One TTP

The following table gives a short description of each of the steps in the SPP protocol.

Step	Step Description
1	*buyer sends merchant an Order message*
2	*merchant sends buyer an Order Payment Request message*
3	*buyer sends TTP an Order Payment message*
4	*TTP sends merchant an Order Confirmation Request message*
5	*merchant sends TTP an Order Confirmation message*
6	*Payment Confirmation Request/Payment Confirmation*
7	*TTP sends merchant a Merchant Transaction Receipt message*
8	*TTP sends buyer a Buyer Transaction Receipt message*

Table 4-2 Short Description of Steps in the SPP Protocol (single TTP Scenario)

Each step is described in the following sections.

4.1.1 Step 1: Buyer Sends Merchant an Order Message

The buyer sends an *Order* message to the merchant. The message contains the following information:

- items to be purchased

- shipping information
- *previously quoted price
- *timestamp

The timestamp is an optional field included to prevent a replay attack.

4.1.2 Step 2: Merchant Sends Buyer an Order Payment Request Message

The merchant, upon receiving the *Order* message, returns an *Order Payment Request* message to the buyer. The *Order Payment Request* message contains the following information:

- transaction ID
- amount
- *Order*
- validity period
- $CERT_m$
- *purchase agreement
- S_m(transaction ID, amount, *Order*, validity period, $CERT_m$, *purchase agreement)

The transaction ID is generated by the merchant and it is used by the merchant and the TTP to keep track of all the transactions. The Order information is the same as that provided by the buyer. The validity period specifies the time during which the payment must be confirmed. The merchant certificate can be used by the buyer to verify the

signature of the merchant. The purchase agreement is an optional field which contains information such as refund policy, product quality, warranty and other policies that are agreed upon. A digital signature signed by the merchant is included as part of the message.

4.1.3 Step 3: Buyer Sends TTP an Order Payment Message

The buyer verifies the merchant's signature on the *Order Payment Request* message and proceeds by sending the TTP an *Order Payment* message. The *Order Payment* message contains the following information:

- payment information
- amount
- merchant
- transaction ID
- $CERT_b$
- *timestamp
- S_b(payment information, amount, merchant, transaction ID, $CERT_b$, *timestamp)

The payment information field contains information such as credit card number, credit card holder and expiration date. The protocol is not limited to just processing credit card based payment transactions. The payment information field can contain details about other payment instruments such as debit cards, electronic checks, electronic tokens etc. The transaction ID is the same as that provided by the merchant. The buyer's

certificate can be used by the TTP to verify the buyer's signature. An optional timestamp may be included to prevent a replay attack. A digital signature is also included.

4.1.4 Step 4: TTP Sends Merchant an Order Confirmation Request Message

The TTP verifies the buyer's signature on the *Payment Request* message and proceeds by requesting a confirmation from the merchant by sending an *Order Confirmation Request* message to the merchant. The *Order Confirmation Request* message contains the following information:

- transaction ID
- amount
- status
- S_t(transaction ID, amount, status)

4.1.5 Step 5: Merchant Sends TTP a Order Confirmation Message

The merchant verifies the *Order Confirmation Request* message by verifying the transaction ID, amount and the TTP's signature. The merchant then proceeds by sending an *Order Confirmation* message to the TTP. The *Order Confirmation* message contains the following information:

- transaction ID
- amount
- status

- S_m(transaction ID, amount, status)
- *(H(transaction ID, amount, *Order*, validity period, *purchase agreement), Sm(H(transaction ID, amount, *Order*, validity period, *purchase agreement)))

As an option, a cryptographic digest of the transaction details as contained in the *Order Payment Request* message may be included. This digest will be useful for dispute resolution purposes.

4.1.6 Step 6: Payment Confirmation Request / Payment Confirmation

Upon receiving the *Order Confirmation* message, the TTP requests the authorized amount from the payment centre. The payment centre returns an approval to the TTP. Any payment method can be used at this step. The requirement for payment approval is tied to the TTP's policy. It is possible that in some cases (e.g. for preferred customers) the TTP would not wait for credit approval, but would process the payment right away. In this case, the TTP, rather than the payment centre, would be taking on the responsibility for the payment. Furthermore, different TTP's may have different policies on handling unknown or delayed credit approval requests. For example, if the approval request times out, the TTP may either refuse to process the payment or may take the risk of processing it. Similarly, even if the payment centre rejects the request, the TTP still can process it, taking on the payment responsibility as described above.

4.1.7 Step 7: TTP Sends Merchant a Transaction Receipt

The TTP sends a signed *Transaction Receipt* message to the merchant. The message contains the following information:

- payment ID
- transaction ID
- amount
- S_t(payment ID, transaction ID, amount)

4.1.8 Step 8: TTP Sends Buyer a Transaction Receipt

The TTP sends a signed *Transaction Receipt* message to the buyer. The message contains the following information:

- payment ID
- transaction ID
- amount,
- S_t(payment ID, transaction ID, amount)

4.1.9 Additional Details about the Protocol

After the TTP sends the buyer the *Buyer Transaction Receipt* message, the TTP captures the payment and transfers the funds to the merchant. This step happens offline and involves actual payment settlement. For example, if the payment is by credit card, the TTP would be charging the card. The specific payment capture process is beyond the scope of this thesis. For example, the payment capture could be done weekly as a batch

job, or on a per-order basis. Also, the issue of how the TTP transfers funds to the merchant is beyond the scope of this thesis.

The steps described above are sequential in nature. The payment transaction is not complete until the last step (TTP sends buyer a *Buyer Transaction Receipt* message) is performed. A timer is used at each step to protect against unusual situations where one of the parties (buyer, merchant, or TTP) is not proceeding with the next step within a predetermined time interval. For Steps 1 to 6, if the timer expires, the transaction is assumed to be aborted. Any subsequent messages regarding this transaction will be ignored. Up to Step 6, any party can abort the transaction by simply not continuing with the next step.

At Step 7, if the merchant does not receive a receipt within a time-out period, the merchant attempts to obtain the receipt by sending a request message to the TTP. If a receipt is not received after a pre-specified number of attempts, the transaction is assumed to be aborted. In this case, the buyer will not receive the order, but he/she can contact the TTP to request a refund.

At Step 8, if the buyer does not receive a receipt within a time-out period, the buyer may request a receipt from the TTP at a later time. This would not affect the transaction because the order will be shipped by the merchant as long as the merchant has received the receipt.

In case of dispute, the buyer has a signed payment request from the merchant and a signed receipt from the TTP. The merchant has a signed receipt from the TTP. The TTP has a signed payment from the buyer and a signed confirmation from the merchant. The above information is sufficient for dispute resolution purposes.

4.2 Extension to Two TTP's

For the case of a single TTP, a conflict of interest situation may arise when a dispute arises between the buyer and the merchant. This is because the TTP would be representing the interests of both parties. For this reason, candidates for trusted third parties are typically organizations such as major banks or major credit card companies that have no obvious vested interest in supporting either party in a transaction.

In some cases however, buyers and merchants may want to be represented by a trusted party that is more active in supporting their individual concerns, and perhaps is targeted specifically at providing services for their needs. This can be realized by having two TTPs, one for the buyer (referred to as TTP-B), and the other for the merchant (referred to as TTP-M). In the case of dispute, TTP-B and TTP-M will be involved in dispute resolution, protecting the interest of the buyer and merchant respectively. In addition, the protocol with two TTPs has the potential of allowing more types of organizations to assume the role of a trusted third party.

The payment protocol can be extended to two TTPs. The basic steps are illustrated in Figure 3-2.

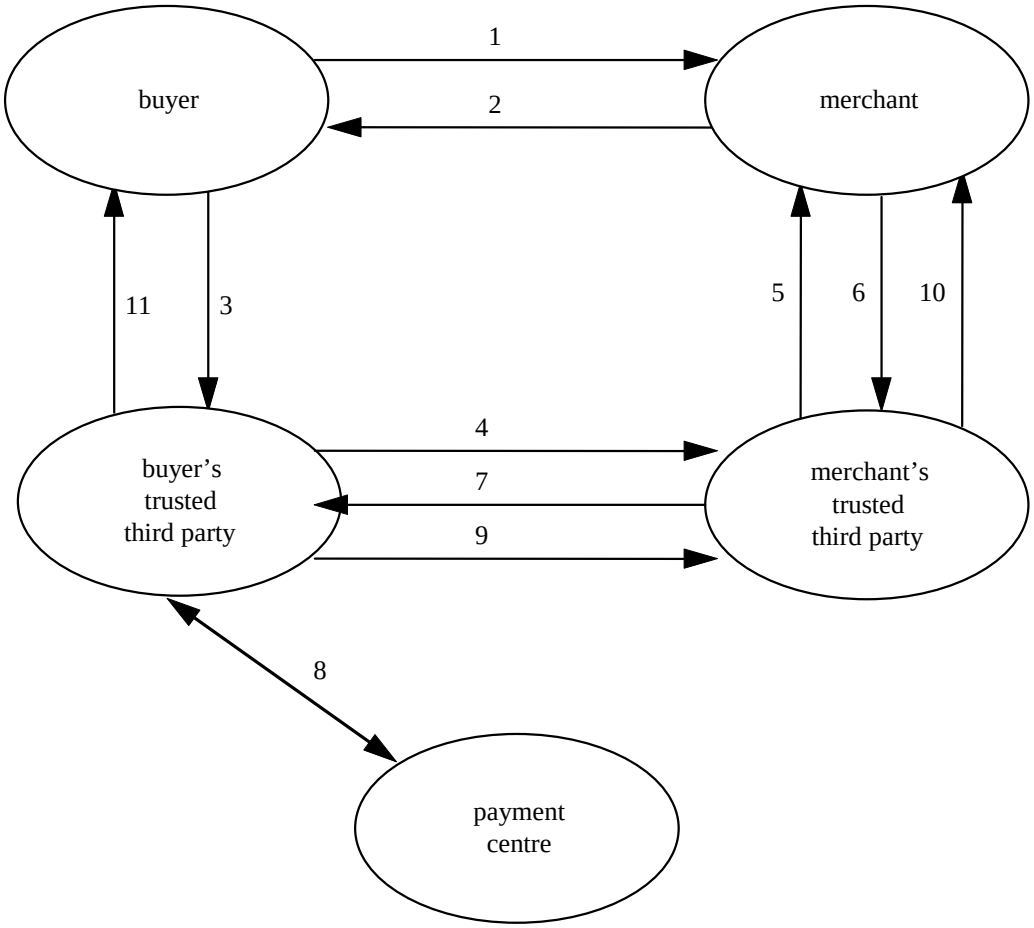

Figure 4-2 SPP Protocol with Two TTP's

Step 1 consists of the buyer sending order information to the merchant via an *Order* message. In step 2, the merchant requests payment from the buyer by sending the buyer a *Payment Request* message. This message would contain the address of the merchant's TTP (TTP-M). In step 3, the buyer sends the payment information to the buyer's TTP (TTP-B). This message would contain the address of TTP-M. Step 4 consists of TTP-B sending a message to TTP-M regarding a confirmation from the merchant. In step 5, the TTP-M requests a confirmation of the transaction and the amount of payment from the

merchant. The merchant returns to the TTP-M a confirmation via the *Confirmation Request* message in step 6. The TTP-M forwards the confirmation to the TTP-B in step 7. In step 8, the TTP-B gets a payment authorization from the payment centre. In step 9, the TTP-B sends a receipt to the TTP-M confirming payment authorization. In step 10 and 11 the TTP-M forwards the receipt to the merchant and the TTP-B sends a receipt to the buyer respectively. The message contents, together with the action taken by the buyer, merchant, TTP-B and TTP-M at the various steps are straightforward extensions of those for the case of one TTP. Therefore they will not be presented here.

5 Protocol Analysis

5.1 Desirable Security Features of a Payment System

In this section we discuss the desirable security features of a payment system.

5.1.1 Dispute Resolution

It is very common during a transaction that a dispute occurs between two or more parties. Examples of typical disputes are:

1. Seller does not ship the merchandise because seller claims the buyer has not paid for it.
2. Buyer claims the seller shipped the wrong merchandise.
3. Purchase agreement has been violated by either the seller or the buyer.
4. Amount of transaction is in dispute.

A payment system must provide a way to handle these types of disputes.

5.1.2 Confidentiality

Confidentiality means the restriction of knowledge about the various pieces of information related to a transaction [AsoS]. Such information includes the identity of payer/payee, purchase content, amount and so on. Typically, the confidentiality requirement dictates that this information be restricted only to the parties involved.

However the requirement can be extended to further limit the knowledge to certain subsets of the parties only.

5.1.3 Anonymity

Some buyers prefer to keep their payment activities private. They do not want parties not involved in their payment transactions to have access to order and payment information. Often they prefer that the merchants and in some cases the banks to be incapable of observing and tracking their payments [AsoS]. There are two levels of anonymity: un-traceability [AsoS] and un-linkability [AsoS]. Un-traceability means that an adversary cannot determine a player's identity in a run of the payment protocol. Un-linkability means that in addition, participation by the same player in two different payments cannot be linked.

5.1.4 Non-Repudiation

Non-repudiation means:

- A service that provides proof of the integrity and origin of data, both in an unforgeable relationship, which can be verified by any third party at any time; or,
- An authentication that with high assurance can be asserted to be genuine, and that can not subsequently be refuted [MacN].

The implication of a payment system that supports non-repudiation is that an action performed by any party in a transaction should not be able to be denied. For example, if a merchant agreed to sell a set a goods or services for a specified price under the

specified terms and conditions, he/she should not be able to refute this agreement in the future. The use of digital signatures enables this functionality. Also a buyer that willingly agrees to the conditions of a contract should not be able to refute or deny that the agreement was ever made.

5.1.5 Message Integrity

The protocol must guarantee that the message content is not altered during transmission between a sender and receiver. The message may contain information that is sensitive and private. If a message is altered in any way during transport then the transaction will not be processed correctly. Therefore the protocol must ensure that the information received matches the information sent.

5.1.6 Availability and Reliability

Payment transactions must be atomic. They occur entirely or not at all, and they should never hang in an unknown or inconsistent state. No payer would accept a loss of money due to a network or system crash. Availability and reliability presume that the underlying network services and all software and hardware components are sufficiently dependable [AsoS].

5.2 Analysis of SPP against Desirable Security Features

In this section we analyze the SPP protocol to see how the desirable security features of a payment system are provided.

5.2.1 Dispute Resolution

The SPP protocol messages sent between the different parties in a payment transaction are signed using digital signatures based on the PKCS #7 [PKCS7] standard. This provides a confidential audit trail for dispute resolution. To demonstrate this, we will consider the scenarios mentioned above one at a time and see how the protocol deals with the problems.

1. Seller does not ship the merchandise because seller claims buyer has not paid for it.

For the first scenario, the TTP has proof that the payment has been authorized by the Payment Centre. Upon receiving the SPP *Order Confirmation* message the TTP requests from the Payment Centre a payment authorization validation. The Payment Centre would respond to the TTP with a message containing the results of the request, either successful or unsuccessful. If successful an authorization code is also provided. The TTP stores the messages to and from the Payment Centre in its secure database for audit trail purposes. Once the payment authorization code has been received by the TTP, the TTP has proof that the payment has been assured to the seller. Once, the payment has been assured to the seller it is up to the seller to capture the payment and there is nothing to stop the seller from doing so.

2. Buyer claims the seller shipped the wrong merchandise.

For this second scenario, assuming the transaction has completed successfully, the buyer has a signed SPP *PaymentRequest* message from the merchant that contains the order details. The buyer can present this information to the merchant along with the packing slip to prove to the merchant that the wrong merchandise was shipped.

3. Purchase agreement has been violated by either the seller or the buyer.

The merchant sends to the buyer a signed *OrderPaymentRequest* message that contains the purchase agreement information. The buyer would review the contents of this message and if the terms and conditions of the order are acceptable, then he/she would submit a signed *OrderPayment* message to the TTP. If the transaction has completed successfully then the above steps were guaranteed to have taken place which prove that both the buyer and the merchant have agreed to the terms of the order including the purchase agreement. More specifically, the buyer has a copy of the *OrderPaymentRequest* message. This message is signed by the seller and it can be used to provide undeniable proof that the merchant has agreed to the terms of the order. The TTP has a copy of the *OrderPayment* message that is signed by the buyer. The merchant can present this as undeniable proof that the buyer has agreed to the terms of the order.

Note that there is no message that explicitly contains the purchase agreement signed by the buyer. The buyer signs the *OrderPayment* message which has the transaction id field in it and this id can be used to reference the *OrderPaymentRequest* message which does contain the purchase agreement information. The transaction id reference is sufficient to link together all of the SPP messages for a given transaction.

4. Amount of transaction is in dispute.

The resolution of this dispute would be similar to the third scenario. Additionally, for this scenario, the amount of the transaction is explicitly signed by both the merchant and the buyer. The merchant signs the amount information in the *OrderPaymentRequest* message and the buyer signs it in the *OrderPayment* message.

Furthermore, a SPP payment transaction includes a TTP (trusted third party) who acts as an impartial player in the transaction. The TTP can be called upon to help settle disputes.

5.2.2 Confidentiality

There are essentially two types of information encapsulated in the SPP protocol messages. They are 1) order information and 2) payment information. Order information includes information such as the product list, the prices of the products, the purchase agreement and shipping information. Payment information consists of the payment method and details of payment. Credit card is an example of a payment method and the card number, expiry date, and amount are payment details. To analyze the confidentiality feature of a payment protocol we need to analyze who has access to these two types of information.

We first check whether anyone other than the participants of a transaction (buyer, merchant, TTP) can access any of the information in the transaction. There are two places where the confidentiality of SPP messages may be compromised. They are, during transmission and while stored in a database (e.g. TTP stores SPP messages in a database for audit trail purposes).

The protocol states that during transmission, the SSL protocol [SSL] is used to encrypt the message. The message is secure during transmission as SSL employs 128 bit DES synchronous cryptography for encryption. However, once the message reaches its destination, it is no longer in an encrypted state. The problem arises when a message gets sent to an incorrect destination. The receiver of the message would be able to fully view the details of the message. This security hole is discussed in more detail in section 5.4.1 entitled "Potential for Compromised SPP Messages".

The second place where the confidentiality of a SPP message could be compromised is while it is stored in the database. The TTP is required to store the SPP messages in order to maintain an audit trail for dispute resolution purposes. The protocol does not discuss the details of how the message should be stored in a database. If the SPP messages were stored in raw form, then any TTP database administrator (DBA) with access to the database would be able to view the details of the messages. Therefore, a dishonest TTP DBA would be able to steal sensitive data such as credit card information. Also, an unauthorized individual who happens to break into the DBA's account will gain access to the database. This security hole is discussed in more detail in section 5.4.2 entitled "Un-trusted TTP Database Administrator".

Now let's examine which parties involved in a transaction can access the various pieces of information, specifically order and payment information. Note that only the merchant and the buyer can access the order information. The TTP only has a digest of the order information; therefore the TTP is not able to view the order details. Note also that only the buyer and the TTP can access the payment information. The buyer sends the payment information to the TTP via the *Order Payment* message; therefore the merchant is not involved in the collection of the payment information.

The identities of the parties involved in a SPP transaction are uniquely identified. The use of digital certificates ensures that this is the case. The digital certificates of each party are exchanged during a run of a SPP payment transaction. A digital certificate that is issued by a respectable certificate authority uniquely identifies the party.

The following table summarizes the confidentiality features of the SPP protocol:

Description of Confidentiality Requirement	Is Requirement Satisfied by SPP Protocol?
Information is restricted to only the parties involved in transaction.	Partial. (See table below for further details.)
Information is restricted to only parties involved in transaction and information is further restricted to a subset of the parties.	Yes
Order information is available to only the buyer and the merchant	Yes
Payment information is available to only the buyer and the TTP	Yes
Identity of payee/payer is known.	Yes

Table 5-1 Summary of SPP Confidentiality Analysis

The following table lists scenarios and states whether or not the confidentiality is preserved.

SPP Message Confidentiality Scenarios	Is Confidentiality Preserved?
During transmission and message delivered at proper destination	Yes
During transmission and message delivered at improper destination	No
SPP message stored in TTP database and TTP DBA is honest	Yes
SPP message stored in TTP database and TTP DBA is dishonest	No
SPP message stored in TPP database and TTP DBA user account is compromised	No

Table 5-2 Summary of Confidentiality Scenarios

5.2.3 Anonymity

The first anonymity requirement is un-traceability. It states that an adversary should not be able to determine a party's identity in a run of the payment protocol. In the SPP protocol, the identities of the parties are known since digital certificates are used for authentication. The identities of the parties can be determined by an adversary if the adversary can gain access to the SPP messages. This is because digital certificates are embedded within these messages. There exist two places where access can be gained to the SPP messages. They are during transmission and from the database, assuming the database is not secured.

The SPP messages are protected during transmission by the SSL protocol. We can therefore assume that these messages will be protected during transmission. However, there is still a possibility for an SPP message transmitted using SSL to be compromised by an attack called a Man-in-the-Middle attack. This is discussed further in the section 5.4.1 entitled "Potential for Compromised SPP Messages".

A secure database is required to prevent a dishonest DBA from tampering with the data. The un-trusted DBA security hole is discussed further in the section 5.4.2 entitled "Un-trusted TTP Database Administrator".

The second anonymity requirement is un-linkability. It states that participation by the same party in two different payment transactions cannot be linked. In the SPP protocol, the merchant and the TTP can link multiple buyer transactions together. The merchant can do this only if multiple transactions of the buyer involve that merchant. An adversary cannot link two buyer transactions unless the adversary has access to the

messages of the SPP protocol. The un-linkability requirement will be met if the security enhancements discussed in section 5.4 entitled "Discussion of Security Enhancements to Protocol" are implemented. Without that enhancement, the SPP protocol cannot claim that the requirement of un-linkability is satisfied completely.

The following table summarizes the anonymity features of the SPP protocol:

Description of Anonymity Requirement	**SPP Protocol Support**
Un-traceability – untraceable by parties within transaction	No
Un-traceability – untraceable by an adversary not involved in transaction	Partial
Un-linkability - un-linkable by parties within transaction	No
Un-linkability – un-linkable by an adversary not involved in transaction	Partial

Table 5-3 Summary of SPP Anonymity Analysis

5.2.4 Non-Repudiation

Non-repudiation is enabled in the SPP protocol through the use of digital signatures. Every message except for the initial *Order* message sent by the buyer to the merchant is signed using a digital signature. A digital signature provides proof of the originator of the message and the integrity of the message content. Therefore, the originator of the message cannot claim that the message was altered in any way or that the originator is someone else. For example, a buyer cannot claim that he/she did not agree to the details of the transaction. This is because the buyer has sent to the TTP a signed *OrderPayment* message which contains a transaction id, which can be used to link all the SPP messages of a transaction together. Therefore, it would be impossible

for the buyer to dispute the details of a transaction. The same principle holds true for the merchant thereby protecting the buyer.

This requirement can only be broken if the validity of the digital signatures is compromised. This is possible in the SPP protocol. In Section 5.4.4 entitled "Validity of Digital Signatures", we discuss how digital signatures can be compromised and ways to secure them in order to enhance the security of the SPP protocol.

5.2.5 Message Integrity

Message integrity is supported in the SPP protocol, by means of digital signatures. For example the SPP *Order Payment* message which is sent by the buyer to the TTP, consists of the following:

- payment information
- amount
- merchant
- transaction ID
- $CERT_b$
- *timestamp
- S_b(payment information, amount, merchant, transaction ID, $CERT_b$, *timestamp)

The buyer calculates a message digest or hash value of the message (*Order Payment*). The buyer signs the message digest with the buyer's private key and sends the signed digest along with the original *Order Payment* message to the TTP. The TTP uses the

sender's public key to decrypt the signed message digest that was received. It also performs a separate calculation of the message digest from the received *Order Payment* message. If there is a match between the two message digests, then the TTP has verified that the message has not been altered. Since the buyer's public key was used for the decryption, the TTP has also verified that the originator of the message is the buyer. The above feature can only be broken if the validity of the digital signatures is compromised. As mentioned previously, this issue will be discussed in Section 5.5.4.

5.2.6 Availability and Reliability

The underlying technology must be reliable in order for the SPP protocol to be reliable. The underlying technology includes internet technologies such as TCP/IP, HTTP, database such as DB2 [IBM DB2] and RSA PKCS [RSA][PKCS7] cryptography. It is inevitable that any one of the above technologies will fail at some point in time. Therefore contingency must be built into the system. For example what will happen if a network failure occurs, or the merchant or TTP database becomes corrupted or a security hole is discovered in the cryptography algorithms. For this thesis we will assume that all the underlying technology can be made reliable through techniques such as redundancy so if one system component fails, a backup will automatically kick in.

If a security hole is found in the cryptography algorithms, then the whole protocol breaks down. However, the RSA [RSA] cryptography algorithms have so far stood the test of time, therefore it is fair to assume at this time that they are indeed reliable and secure for all practical purposes.

To analyze the requirement that each transaction must be atomic, we need to determine the point at which the transaction is completed, and examine the scenarios that

can cause a transaction to be aborted. A SPP transaction is considered complete once the TTP sends a *Transaction Receipt* message to the merchant and the merchant receives this message successfully. At this point, the merchant has confirmation from the TTP that the payment has been authorized and is waiting to be captured. The merchant would now typically ship the goods to the buyer. Some scenarios that can prevent a transaction from completing are listed in the table 4-4.

SPP Transaction Abort Scenarios	**Behavior of the SPP Protocol**
Network stops functioning in the middle of a SPP transaction. (Example: the network fails after the TTP sent the *OrderConfirmationRequest* message to the merchant.)	The SPP protocol has a built in time out mechanism. Each event triggers the starting of a timer. Sending a message is an example of an event. If a response is not received before the timer expires then the transaction is aborted. In the example, the TTP will wait for a pre-determined amount of time for an *OrderConfirmation* response message from the merchant. If no response is received then the SPP transaction is aborted.
Merchant decides to abort transaction. (Example, merchant does not send a *OrderConfirmation* response message to the TTP.)	The merchant can abort the transaction by simply choosing to not reply to a SPP message it has received. In the example, the merchant decides not to send an *OrderConfirmation* response to the TTP thereby causing the TTP to mark the transaction as aborted.

SPP Transaction Abort Scenarios	Behavior of the SPP Protocol
Buyer decides to abort transaction. (Example, buyer receives the *OrderPaymentRequest* message from the merchant and decides not to send an *OrderPayment* message to the TTP.)	The buyer also has the ability to abort the transaction. This is accomplished by simply not sending an *OrderPayment* message to the TTP.

Table 5-4 SPP Transaction Abort Scenario Analysis

5.3 SPP Proof Analysis

In this section, we present a proof analysis of the SPP protocol. The main purpose of this analysis is to enumerate the various pieces of proof that each party requires in a SPP transaction in order to feel secure and confident about the transaction. It is also a good way to find security holes in the protocol. A summary of the proof analysis is shown in the Table below.

Actor	Description of Requirement	Supported By SPP	Discussion
TTP	Proof Buyer has authorized transaction	Yes	Buyer sends a signed OrderPayment message to TTP
	Proof Merchant has authorized transaction	Yes	Merchant sends a signed *OrderConfirmation* message to TTP

Actor	Description of Requirement	Supported By SPP	Discussion
Merchant	Proof Buyer has authorized transaction	Yes	The merchant does not have direct proof of this. Only the TTP does. The merchant would have to contact the TTP in this case to get the direct proof, which consists of a signed *OrderPayment* message that the buyer sends to the TTP.
	Proof TTP has authorized transaction	Yes	TTP generates a *TransactionReceipt* message and sends it to the merchant
	Proof Buyer is legitimate owner of credit card	Limited Support	The TTP is responsible for getting the credit card authorized; therefore the risk for a fraudulent transaction would have to be negotiated between the TTP and the merchant. The only information the TTP has to validate the credit card is the buyers digital certificate. From the certificate one can extract buyer information such as the name. Assuming the certificate is valid, the TTP can verify that the name on the credit card is the same as the name associated with the buyer's digital certificate. This information would be sent to the payment center along with the payment information as the payment center is ultimately the one who does the validation.
Buyer	Impossibility of unauthorized payment	Limited Support	If the buyer's certificate is compromised (i.e. gets into someone else's hands) then someone other then the buyer can make unauthorized payments that appear to be authorized. See section 5.3.4 for discussions on how to handle situations when a digital certificate has been compromised.
	Proof of Merchant accreditation	Yes	The Merchant has to register with a TTP. Therefore the TTP would perform a screening process on the merchant before processing transactions for the merchant.

Actor	Description of Requirement	Supported By SPP	Discussion
Buyer (cont)	Proof TTP has authorized the transaction	Yes	The TTP sends the customer a signed Receipt of the transaction.
	Proof Merchant has authorized transaction	Yes	The merchant sends the customer a signed *OrderPaymentRequest* message also the merchant sends the TTP a signed *OrderConfirmation* message.
Payment Center	Proof Buyer is legitimate owner of the credit card.	Limited Support	Same discussion as for the merchant.

Table 5-5 SPP Proof Analysis

5.4 Discussion of Security Enhancements to Protocol

In this section we present security enhancements that can be made to the protocol. There are many levels of security. Let's use a single family home as an analogy. The home would most likely have at least a single lock on every door of the house. In many cases this would be an acceptable level of security. However, if you want to add additional security to the home there are many enhancements that could be made such as putting an additional chain lock on every door, installing metal bars on the windows, installing motion detectors in the home, installing a closed circuit video and so on. In this section we propose methods to provide additional levels of security to the protocol. Our approach is to list the possible security exposures to the protocol and present a solution to each of the exposures.

5.4.1 Potential for Compromised SPP Messages

A SPP message can be compromised if the message can get into the possession of an unwanted entity and the details of the message can be accessed by that entity. To analyze this problem, we examine the scenario where the buyer sends the TTP an *OrderPayment* message. This message contains sensitive payment information belonging to the buyer. The protocol states that the message is sent by the buyer to the TTP using SSL. There will be no problems if we can guarantee that the TTP is the one receiving the message. The message is encrypted during transport so that it cannot be compromised.

Suppose someone other then the TTP has received or intercepted the message. The message is now compromised because once the message is received, it is no longer in an encrypted state. The question to ask is how can someone other then the TTP receive or intercept the message when SSL is used as the secure transport protocol? There is an attack called a “man-in-the-middle” [SSLIntro] attack where such a situation can happen. The man-in-the-middle is a rogue program that intercepts all communication (via SSL) between the client and the server. This rogue program intercepts the legitimate keys that are passed back and forth during the SSL handshake, substitutes its own, and makes it appear to the client that it is the server, and to the server that it is the client [SSLIntro].

The encrypted information exchanged at the beginning of the SSL handshake is actually encrypted with the rogue program's key, rather than the client's or server's keys. The rogue program ends up establishing one set of session keys for use with the server, and a different set of session keys for use with the client. This allows the rogue program

not only to read all the data that flows between the client and the server, but also to change the data without being detected. To prevent this problem, the client should perform an additional check that is not covered in the SSL protocol. Specifically, the client should check that the domain name in the server's certificate is the same as the domain name of the server with which the client is attempting to communicate [SSLIntro]. If this is not the case, then all communication should be halted immediately.

Another solution to the above problem involves applying the cryptographic concept of enveloping the data. This can be accomplished by using the PKCS7 EnvelopedData class. This class encrypts the signed message with the public key of the recipient so that only the recipient can decrypt the message using its private key. For example, the buyer would encrypt the signed *OrderPayment* message using the public key of the TTP, and then send the message to the TTP. Only the TTP can access the details of the message because only the TTP's private key can decrypt the message. Note that if all SPP messages are enveloped while being transported, then the SSL protocol is no longer needed. Any unsecured transport mechanism can be used for transmitting SPP messages.

5.4.2 Un-trusted TTP Database Administrator

The TTP stores the SPP messages that it sends or receives into its database for a period of time for audit trail purposes, e.g., *OrderPayment*, *OrderConfirmation*, *OrderConfirmationResponse* and *OrderPaymentRecipt*. These messages contain sensitive data, especially the *OrderPayment* message which contains payment information of the buyer. If these messages are stored into the database without being

encrypted then the data in the messages are not protected from an untrustworthy TTP database administrator (DBA). There is nothing stopping the TTP DBA from viewing the contents of the SPP messages and stealing credit card numbers for fraudulent use.

To solve this problem, the TTP should encrypt each SPP message with its public key before storing the message into the database. To view the contents of the SPP messages in the database, the messages would first have to be decrypted using the private key of the TTP. Of course the TTP DBA would not have access to this private key and therefore the data would be protected.

5.4.3 Unauthorized Access to TTP Database Server

If an unauthorized person gains access to the TTP Database server then they would be able to view the details of all the SPP messages that have been sent and received by the TTP. This would be a major security problem for the TTP.

How could an unauthorized person gain access to the TTP Database Server? The most common way is for someone to obtain a valid username and password for the system. There are many ways that this can be done. Since our goal is to protect the data in the database, we can use the same solution as for the "Un-trusted TTP Database Administrator" situation. This means that the TTP should encrypt the SPP messages before storing them in the database, using the public key belonging to the TTP. As a result, these messages can only be decrypted with the private key of the TTP.

5.4.4 Validity of Digital Signatures

If the validity of the digital signatures become compromised then the message integrity, non-repudiation and hence the dispute resolution properties of the protocol will

be lost. This subsection discusses ways in which digital signatures can become invalid and suggests methods that can be applied to the protocol to further secure it.

The timestamps on the SPP messages acts as a critical piece of information to maintain the validity of digital signatures [ZhouO]. It should be generated by an online timestamp authority such as DigiStamp [DigiStamp]. As an example, for the SPP *Order Payment* message, a signed timestamp can be added as follows.

1. B -> TS: S_b(payment information, amount, merchant, transaction ID, $CERT_b$)
2. TS -> B: T_s, S_{TS}(S_b(payment information, amount, merchant, transaction ID, $CERT_b$), T_s)

The buyer (B) sends the timestamp authority (TS) a signed *Order Payment* message. The TS adds a timestamp to the buyer's signed message.
All signed SPP messages should contain a timestamp from a timestamp authority. This is because if a private key or the signing key of one of the participants of a SPP transaction becomes compromised, the digital certificate of that participant needs to be revoked. Once the digital certificate has been revoked all messages signed with that signing key should no longer be valid. When a signed message is being verified, the verifier should check the Certificate Revocation List to see if there exists a record of the revoked certificate. If the time of revocation is before the timestamp on the signed message then the signature will be deemed invalid [ZhouO].

5.4.5 Validity of Digital Certificates

The security of the SPP protocol depends on digital certificates for authentication and on signature services for the participants in a transaction. A digital certificate can become invalid because:

- The private key corresponding to the public key in the certificate may be lost or compromised [LeviR],
- The Certificate Authority's (CA's) signature key may be compromised [LeviR], or
- The certification contract may be terminated or the certificate holder's status and abilities described in the certificate may change or may be cancelled [LeviR].

The SPP protocol must specify how to check and verify the validity of digital certificates as it is inevitable that these certificates become invalid. The concept of certificate revocation was introduced into PKI (Public Key Infrastructure) based systems for this reason. The best-known revocation mechanism is the Certificate Revocation List (CRL), which keeps a signed list of the serial numbers of revoked certificates. Usually, the CA is the signer of the CRL for the certificates that it issued [LeviR]. The verifier of a digital certificate would obtain a CRL from the CA and check if the certificate has been revoked.

6 Design of SPP Prototype

The SPP Protocol prototype that was implemented as part of this thesis supports the single TTP scenario.

6.1 Description of SPP Prototype Software Components

6.1.1 Software Components on Buyer Computer

The software components required on the buyer's computer are:

- Web browser such as Internet Explorer or Netscape.
- SPP Electronic Wallet Browser Plug-in Application.

A web browser such as Internet Explorer or Netscape is a standard client application that allows a buyer to interface with a merchant's web storefront in order to browse catalogs and create orders. The SPP Electronic Wallet Browser Plug-in application is based on the Java Plug-in 1.3 [JavaPlugin] specification. This application is used to sign the messages that the buyer sends to the merchant and the TTP. It also performs signature verification on the messages it receives. These are the functions that enable some of the security features of the protocol. The wallet application is launched automatically when the buyer's computer receives an *OrderPaymentRequest* message from the merchant.

The scenario can be described as follows:

- The buyer (or shopper) is shopping at a merchant's web storefront and decides to submit an order to the merchant by sending an *Order* message to the merchant.
- The merchant, upon receiving the *Order* message, verifies it and redirects the shopper to the OrderPaymentRequest.jsp page which launches the SPP Wallet application in the shopper's browser. The wallet receives the *Order Payment Request* message from the merchant, verifies the message and displays the contents of the message for the shopper to see. The contents displayed include the details of the order.
- The shopper verifies that the information displayed in the wallet is correct and presses the submit button on the wallet which generates an *Order Payment* message that is sent to the TTP.

In our implementation, it is assumed, for simplicity, that the URL of the TTP is known to both the merchant and the buyer.

6.1.2 Software Components on Merchant Commerce Server

The basic function of a merchant commerce server is to enable buyers to browse its catalog and place orders. The software components needed for the implementation of the prototype are:

- simulation of a commerce server
- SPP Merchant software

For the prototype a simulation of a merchant commerce server was implemented. The simulation deals with the tasks related to the submission of an order by a buyer and certain merchant related payment processing tasks. Therefore, the simulation does not

support browsing of catalogs, adding products to shopping carts or any task that would be done prior to submitting an order. The simulation is based on the IBM WebSphere Commerce Suite (WCS) V5.1 product [IBMWS] and consists of the following two commands:

- OrderProcess
- OrderConfirm

OrderProcess is used to submit an order. When the buyer submits an order, the client software calls the OrderProcess command and passes to it the SPP *Order* message.

The OrderConfirm command does not exist in the WCS product. It has been added specifically to support the SPP protocol. The purpose of this command is to confirm the details of the order for the TTP during a SPP payment transaction. The TTP would send the merchant an *Order Confirmation Request* message via a call to the OrderConfirm command. The OrderConfirm command processes the *Order Confirmation Request* message and if all is valid a response is sent back to the TTP consisting of the *Order Confirmation* message.

The SPP merchant software is a software component that plugs into the commerce server. This component performs the following services:

- Maintaining the integrity of the SPP protocol,
- Preparing and processing SPP merchant related messages,
- Signing the messages that are sent, and
- Verification of messages received.

6.1.3 Software Components on TTP SPP Commerce Server

The software components required on the TTP Commerce Server are:

- WebSphere Commerce Suite [IBMWS]
    - This software product provides a transaction based electronic commerce platform. In our prototype this product was not used but some of the necessary functions were simulated.
- SPP TTP Software
    - When installed on the TTP SPP Commerce Server, the software handles all the TTP related activities such as
        - Maintaining the integrity of the SPP protocol,
        - Preparing and processing SPP TTP related messages,
        - Signing the messages that are sent, and
        - Verification of messages received.

6.2 Architecture and Design of SPP Prototype Components

6.2.1 SPP Electronic Wallet Browser Plug-in Application

A high-level architecture of the SPP Electronic Wallet Browser Plug-in Application is shown in Figure 6-1. The various components are described below.

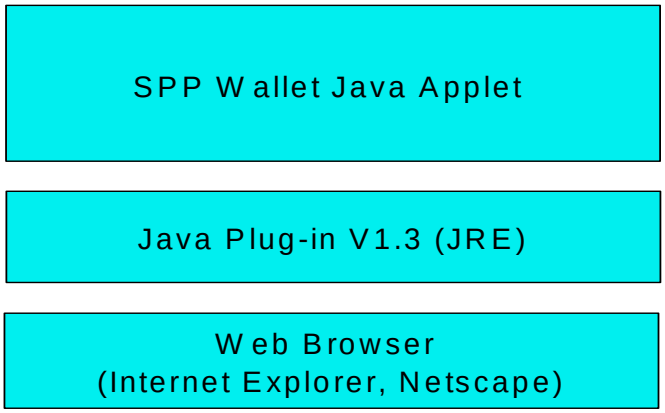

Figure 6-1 High level architecture of the SPP Electronic Wallet Browser Plug-in Application

1. SPP Wallet

The SPP wallet is implemented as a java applet.

2. Java Plug-in V1.3

Java™ Plug-in [JavaPlugin] is a product from Sun Microsystems that runs beans or applets (written in the Java programming language) in an HTML page using Sun's Java virtual machine (JVM). To launch Java Plug-in, the `OBJECT` tag and the `EMBED` tag in the HTML specification are used. The buyer computer must have the Java Plug-in installed in order to execute the SPP Wallet application.

3. Web Browser

A standard web browser such as Internet Explorer or Netscape Navigator.

The above 3 components work together in the following way. The buyer submits an *Order* message to the merchant. The merchant processes the *Order* message and redirects the buyer to the OrderPaymentRequest.jsp page. This jsp page contains the following code snippet:

```
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93" width="700"  height="500"
codebase="http://java.sun.com/products/plugin/1.2.2/jinstall-1_2_2-
win.cab#Version=1,2,2,0">
<PARAM name="java_code" value="spp.wallet.sppwallet.class">
<PARAM name="java_archive" value="/examples/jsp/store/sppwallet/sppwallet.jar">
<PARAM name="type" value="application/x-java-applet;version=1.2">
<PARAM name="SPPMSG" value="<%=xmlOrderPaymentRequestMsg%>">
<PARAM name="SIGNEDSPPMSG" value="<%=signedXmlOrderPaymentRequestMsg%>">
<COMMENT>
<EMBED type="application/x-java-applet;version=1.2"  width="400"  height="300"
pluginspage="http://java.sun.com/products/plugin/" java_code="sppwallet.class"
java_archive="/examples/jsp/store/sppwallet/sppwallet.jar" >
<NOEMBED>
</COMMENT>
Plugin tag OBJECT or EMBED not supported by browser.
</NOEMBED></EMBED>
</OBJECT>
```

The above code snippet is loaded into the Web browser which loads the Java applet specified by the 'java_code' tag and the 'java_archive' tag. The 'java_code' tag specifies the applet class and the 'java_archive' tag specifies the jar file where the applet code is located. These are enabled with the Java plug-in technology.

6.2.2 SPP Merchant Software

A high level design of the SPP Merchant Software is shown in Figure 6-2.

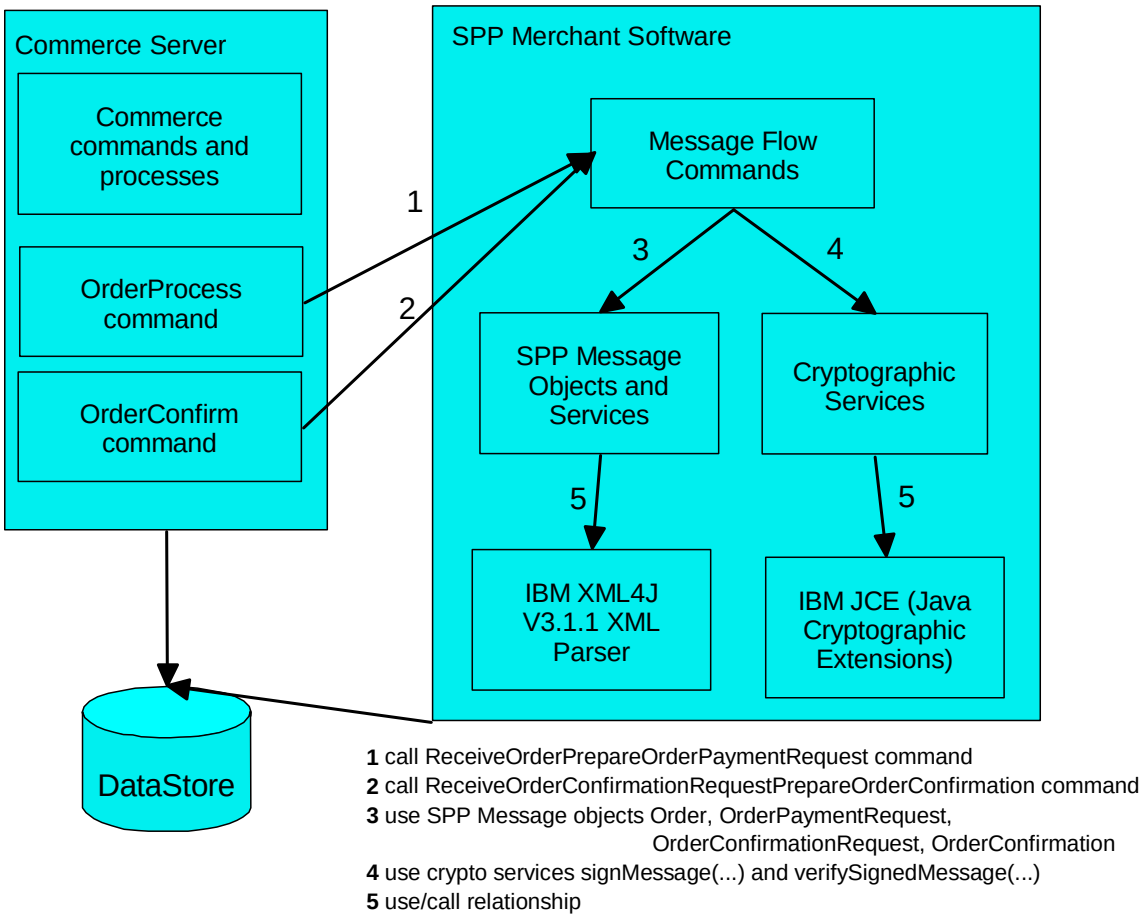

Figure 6-1 High Level Design of SPP Merchant Software

The details of each component are described below.

1. Message Flow Commands

A class diagram of the merchant message flow commands is shown in Figure 6-3. These commands extend the abstract base class *MessageFlowBaseCmd* and implement the *MessageFlowInterface.*

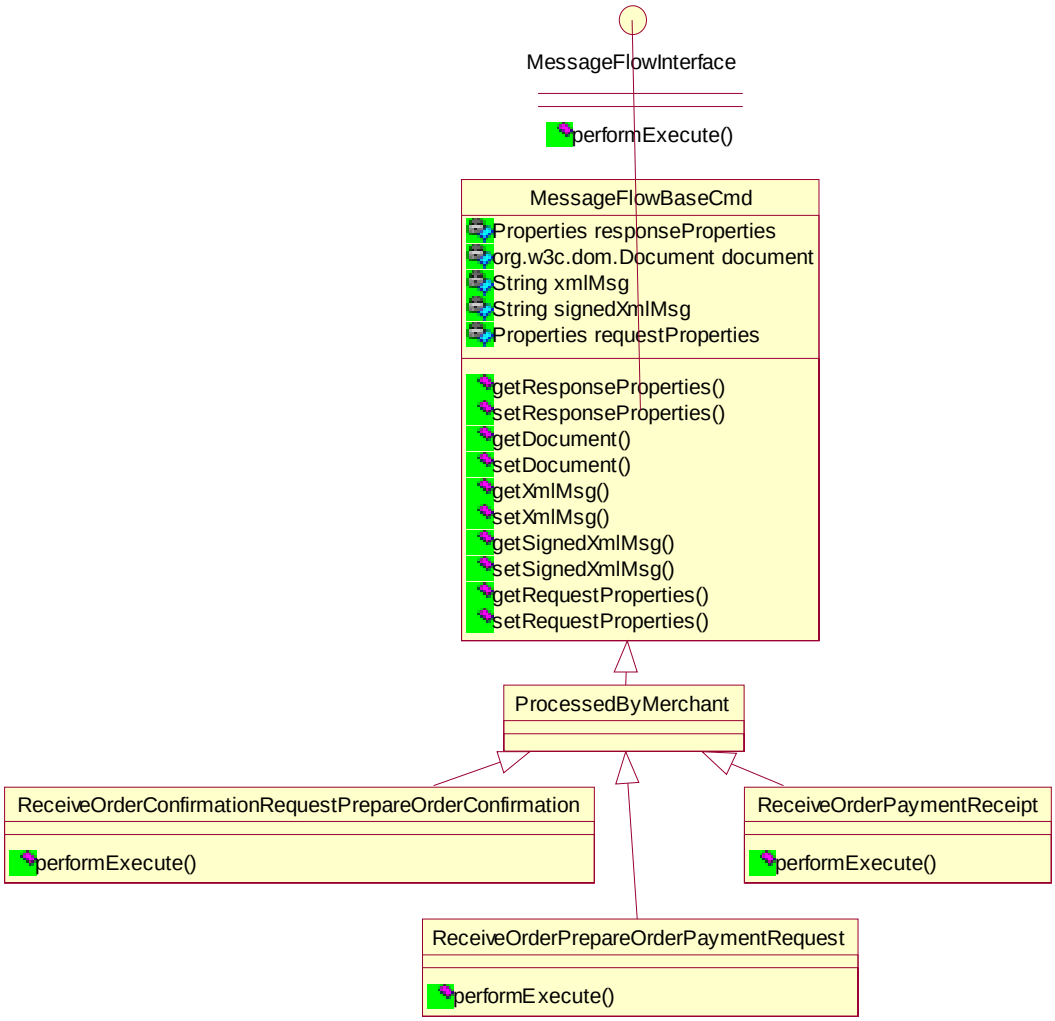

Figure 6-2 SPP Merchant Software Message Flow Commands

2. SPP Message Objects and Services

A class diagram of the SPP Message objects is shown in Figure 6-4. Each SPP message is represented by a class object. Each object provides helper routines such as *convertXMLToHashtable*, *convertHashtableToXML*, *formatMsgForDisplay* and *validateParameters*.

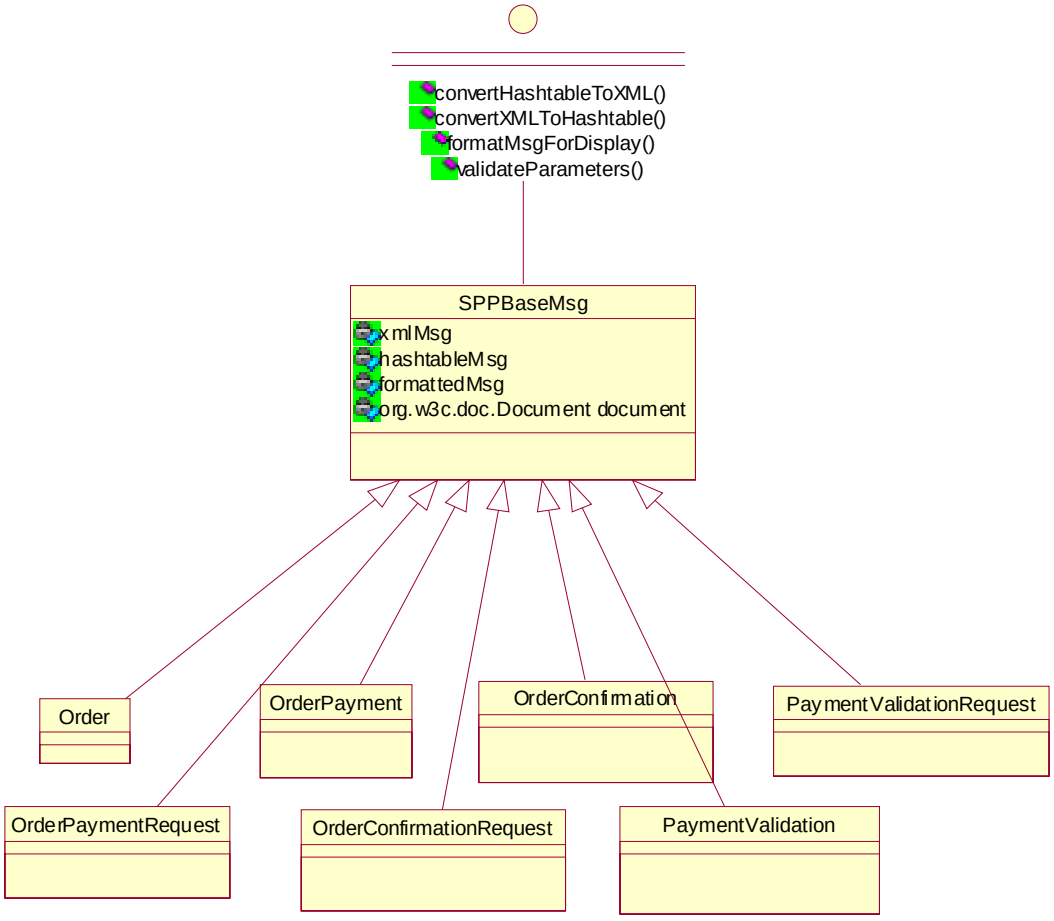

Figure 6-3 SPP Message Objects

3. Cryptographic Services

The cryptographic services class abstracts the IBM JCE API and provides routines to sign messages and verify message signatures. Figure 6-5 shows a class diagram of the *CryptographicServices* class that displays the services that it performs.

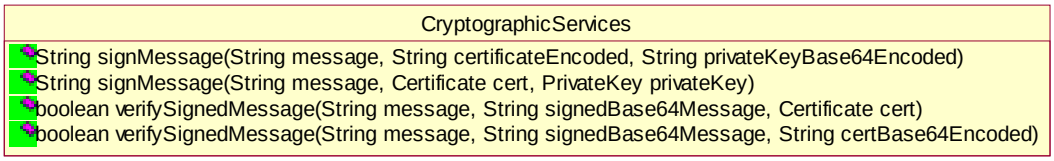

Figure 6-4 SPP Cryptographic Services

4. IBM XML4J Parser V3.1.1

This version of XML4J is based on the Apache Xerces version 1.2.0 [Xerces] codebase. The parser is used for parsing and generating SPP messages. It also supports xml schema [XmlSchema]. The use of xml schema provides a superior way to validate xml documents as opposed to traditional DTDs.

5. IBM JCE V1.3

IBM Java Cryptography Extension V1.3 provides the cryptographic services used in the prototype. The pkcs7 package is an implementation of PKCS #7 Version 1.5 [PKCS7]. PKCS #7 describes a general syntax for data that may have cryptography applied to it, such as digital signatures and digital envelopes. The IBM PKCS implementation supports all the content types defined in this standard, such as Data, SignedData, and EnvelopedData.

6. OrderProcess command

The shopper submits the SPP *Order* message to the merchant by calling the OrderProcess command and passing the *Order* message as a parameter.

7. OrderConfirm command

The TTP sends the *SPP OrderConfirmationRequest* message to the merchant's OrderConfirm command. The OrderConfirm command responds with an *OrderConfirmation* message to the TTP if the processing produced no errors.

6.2.3 SPP TTP Software

A high level design of the SPP TTP Software is shown in Figure 6-6.

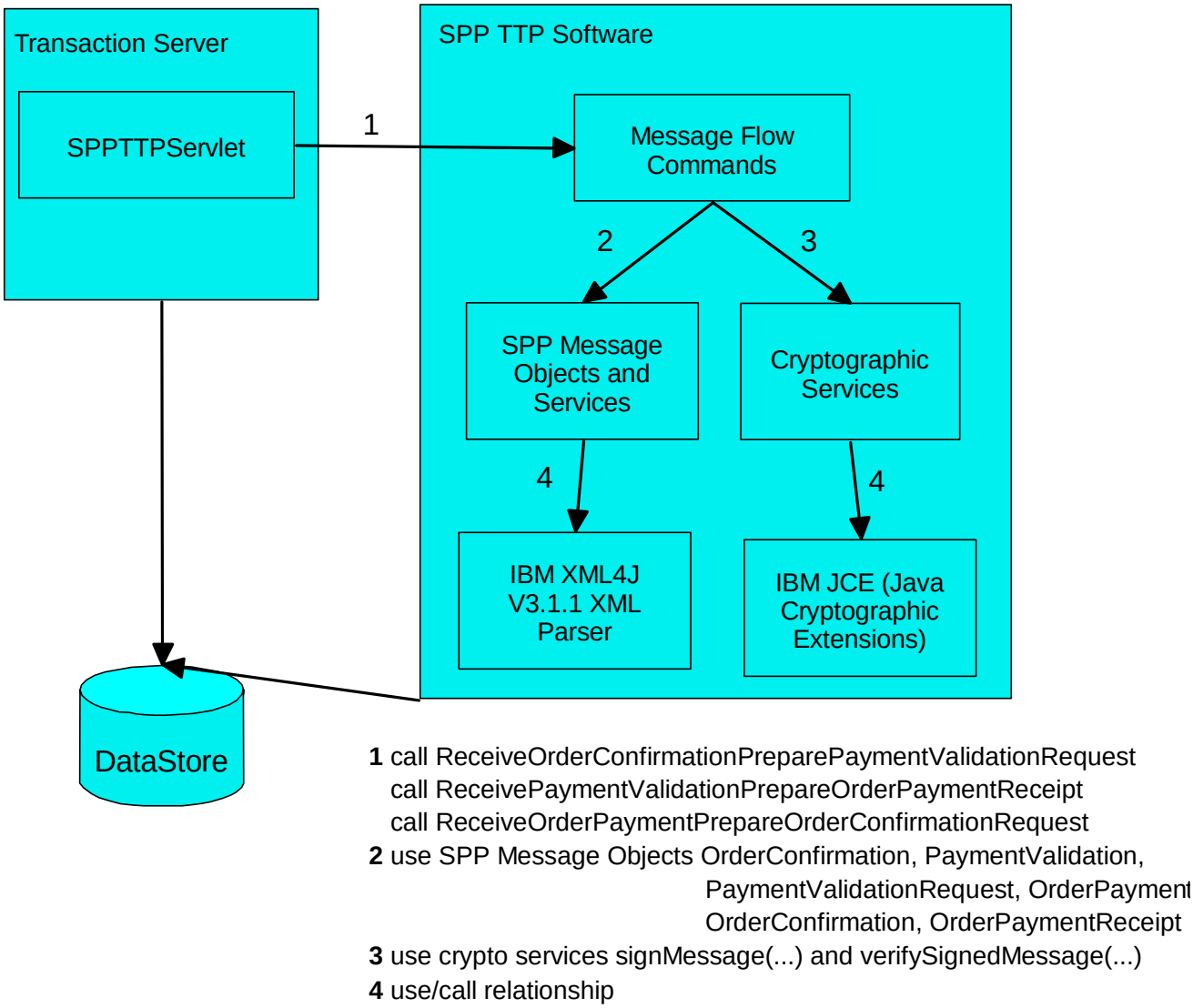

Figure 6-5 High Level Design of SPP TTP Software

A class diagram of the TTP message flow commands is shown in Figure 6-7. These commands extend the abstract base class *MessageFlowBaseCmd* and implement the *MessageFlowInterface.*

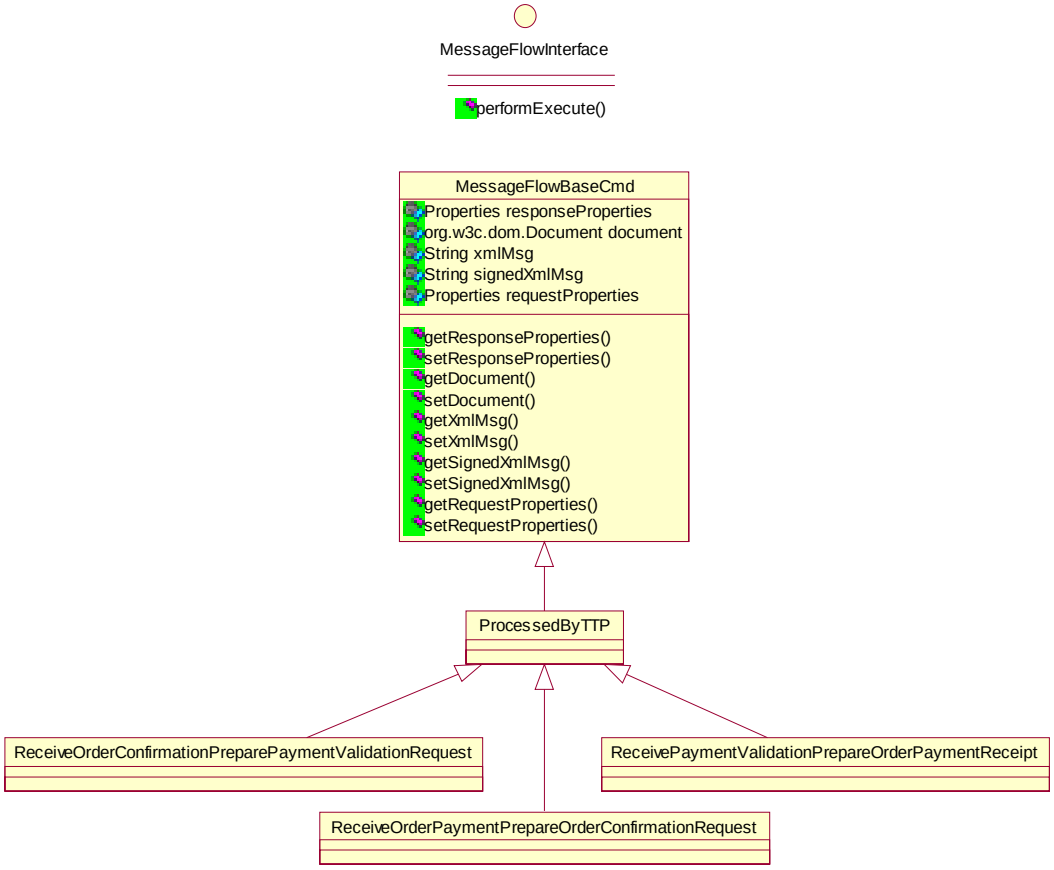

Figure 6-6 TTP Software Message Flow Commands

The SPP Message Objects and Services, the Cryptographic Services, the IBM XML4J Parser V3.1.1, and the IBM JCE V1.3 are the same as those used in the design of the SPP Merchant Software. All messages sent to the TTP are sent to the SPPTTPServlet. The servlet passes the messages onto the SPP TTP Software component where it is processed.

As to the Datastore, Figure 6-8 displays the TTP database schema.

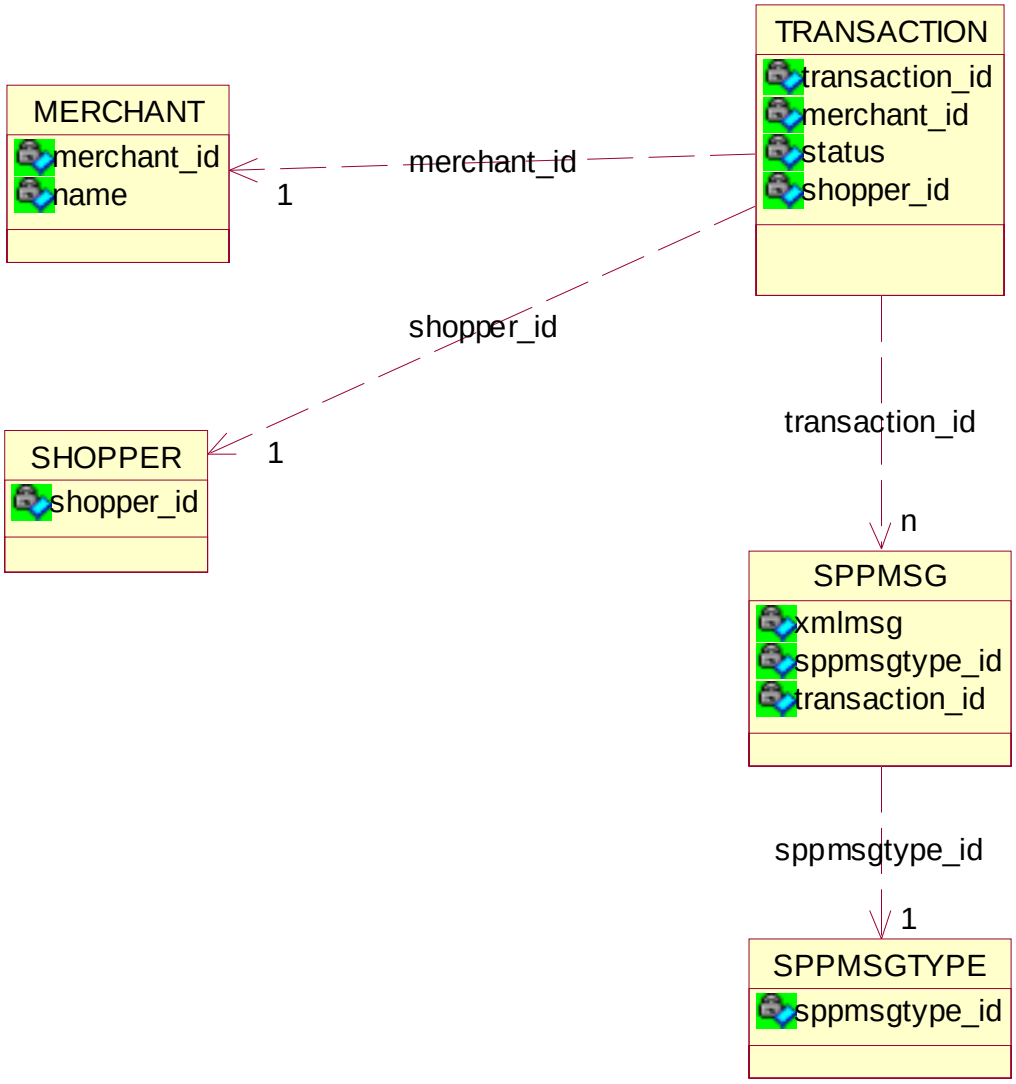

Figure 6-8 TTP Database Schema

The TRANSACTION table stores a separate record for each transaction. Several SPP messages are transmitted in a transaction. These messages are stored in the

SPPMSG table. Each message is of a particular type, e.g., *Order*, *OrderPaymentRequest*, and *OrderPayment* and so on. The SPPMSGTYPE table contains a record for each message type that exists in the protocol. Each record in the SPPMSG table is associated to a record in the SPPMSGTYPE table to identify the type of message. The MERCHANT and SHOPPER tables identify the merchant and shopper respectively in a transaction. These tables can be further extended to store registration information such as name, address, and contact information. Note that the schema shown in Figure 6-8 does not represent the data model required for implementing the protocol in a production environment. For a production environment, the schema would have to include information such as billing information, contract information, policy information, etc. These are outside the scope of this thesis.

7 SPP Performance Evaluation

A performance analysis of the SPP protocol was performed. Since the dominant security technology in the SPP protocol is the use of digital signatures, the digital signatures component will have a significant impact on the performance of the protocol. The performance of digital signatures is determined by the digital signature provider, the signature algorithm used, and the key size. For the SPP protocol prototype implemented for this thesis the security provider was IBM JCE version 1.3. With this provider, the available signature algorithms are MD2withRSA, MD5withRSA, SHA1withRSA and SHA1withDSA. The algorithm used in the prototype implementation was SHA1withRSA. This algorithm uses the SHA1 message digest protocol in combination with the RSA encryption protocol.

We performed experiments to measure the performance of our implementation. These experiments are designed to compare the time it takes to complete a single run of the protocol using a key size of 512 bits, 1024 bits, 2048 bits as well as disabling the digital signatures altogether. Note that the larger the key size the more difficult it is to break the security, but the execution time is expected to be more substantial. For each of the key sizes, we executed a single run of the protocol 10 times and measured the completion time, which is defined to be the time from when the TTP receives the *Order Payment* message to when the TTP sends the merchant the *Transaction Receipt* message. This definition of completion time is used because the process from the reception of the *Order Payment* message to the transmission of the *Transaction Receipt* message would under normal circumstances be performed automatically by

machines. This would exclude, for example, the "think" period from when an *Order Payment Request* message was received by the buyer and the subsequent submission of the *Order Payment* message to the TTP.

Table 6.1 describes the performance results that we have obtained. The machine used for doing the experiments was an IBM Thinkpad A22e with 512 MB of RAM running the Microsoft Windows 2000 operating system. The processor is a Pentium III running at approximately 846 MHz.

Run Number of SPP Protocol	***Digital Signatures Disabled (time in milliseconds)***	***512 Bit Modulus (time in milliseconds)***	***1024 Bit Modulus (time in milliseconds)***	***2048 Bit Modulus (time in milliseconds)***
1	2624	3495	4286	9383
2	2544	3194	4446	9263
3	2543	3898	5032	8903
4	2804	3335	4947	9013
5	2754	3575	4587	9157
6	2643	3365	5227	9414
7	2704	3495	4697	9293
8	3278	3535	4667	9704
9	2724	3405	5167	9433
10	2784	3555	5228	9214
Average	**2740.2**	**3485.2**	**4848.4**	**9277.7**

Table 7-1 Performance Measurement Results of the SPP Protocol

The results shown in Table 7-1 clearly show the tradeoff between key size and the completion time. They also show the impact of the digital signatures component on the completion time. Specifically, the completion time is increased by 21%, 77% and 236% when the key sizes are 512, 1024 and 2048 bits, respectively.

8 Extension of SPP Protocol for a Two Buyer Scenario

In this section, we extend the SPP protocol to a two-buyer scenario. In this scenario, buyer 1 is the party placing an order with a merchant and buyer 2 is the party who is paying for the order. This feature is desirable as it models some real world shopping scenarios such as gift purchases. For example, assuming Sue is buyer 1 and James is buyer 2; on Sue's birthday James would tell Sue to make a purchase at merchant XYZ and James will pay for it.

We first present in subsection 8.1 our protocol design, and then in subsection 8.2 an extension to this protocol to improve buyer operation.

8.1 Design 1

The steps for our protocol design for the two buyer scenario are shown in Figure 8-1 and Table 8.1.

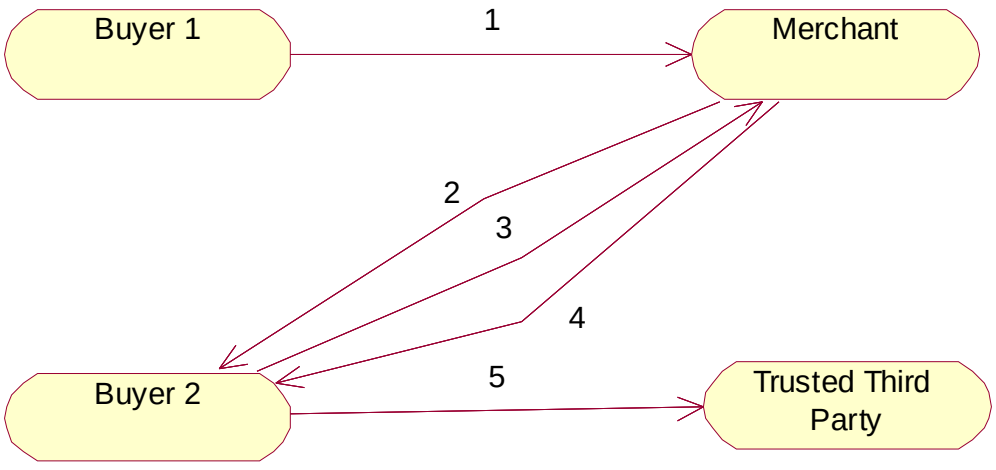

Figure 8-1 Two Buyer SPP Scenario: Protocol Design

Step Number	Step Description
1	*Buyer 1 sends merchant an Order message*
2	*merchant sends buyer 2 an Order Payment Request Notification message*
3	*buyer 2 sends Merchant a Get Order Payment Request message*
4	*merchant sends buyer 2 an Order Payment Request message*
5	*buyer 2 sends TTP an Order Payment message*

Table 8-1 Details of Steps for our Two-Buyer Protocol

8.1.1 Step 1: Buyer 1 Sends Merchant an Order Message

In step 1, buyer 1 sends an *Order* message to the merchant. This message is the same as that described in the single buyer scenario except that the content has been extended as follows:

- items to be purchased
- shipping information
- *previously quoted price
- *timestamp
- email address of buyer 2

- $CERT_{b1}$
- $*S_{b1}$(items to be purchases, shipping information, *previously quoted price, *timestamp, email address of buyer 2, $*CERT_{b1}$)

The additional information includes the email address of buyer 2, the certificate of buyer 1 and the signature of the message by buyer 1. The email address of buyer 2 is needed because the *Order Payment Request Notification* message will be sent by the merchant to buyer 2. The *Order* message is embedded inside the Order Payment Request message. When buyer 2 receives the *Order Payment Request* message, he/she can check the *Order* message and verify the person that actually created the order.
Note that the message signature is optional. If the message is not signed then buyer 2 does not have undeniable proof of who has initiated the order. The only information that buyer 2 would have is the contents of the order which includes some credential information such as name of the person and shipping information plus the certificate of the buyer 1. This might be acceptable in some situations. The advantage of making the signature optional is that buyer 1 is not required to install the SPP Electronic Wallet application on his/her machine.

8.1.2 Step 2: M*erchant Sends Buyer 2 an Order Payment Request Notification Message*

The *Order Payment Request Notification* message is sent to buyer 2 by email. The purpose of this message is to notify buyer 2 that an order has been made and it has to be approved and paid for. The information contained in the email message includes:

- *Contents of the order
- *Price of the order
- *Name of person who created the order
- Transaction Id

The reason why this message is delivered by email is because buyer 2 is most likely not online at the moment when this message is sent and secondly the destination address of buyer 2 is not known until it is provided by buyer 1. The optional fields in this message include contents of the order, price of the order and the name of person who created the order. These fields are optional because for security reasons; one may not wish to include this information in an email, where the data transmitted are not protected by encryption. The only information that is required is the transaction id. A secure protocol can be used in a subsequent message to obtain the other information that is included in the *Order Payment Request Notification* message.

8.1.3 Step 3: B*uyer 2 Sends Merchant a Get Order Payment Request Message*

Buyer 2 sends a *Get Order Payment Request* message to the merchant. To ensure security, this message is delivered using the https protocol. A sample message would look as follows:

https://www.merchantXYZ.com/GetOrderPaymentRequest?transactionId=12345

8.1.4 Step 4: *Merchant Sends Buyer 2 an Order Payment Request Message*

The merchant sends buyer 2 an *Order Payment Request* message. The contents of this message are as follows; they are the same as those for the single buyer scenario except that the *Order* message, which was embedded inside the *Order Payment Request* message, is also included.

- transaction ID,
- amount,
- *Order message*,
- validity period,
- $CERT_m$
- *purchase agreement
- S_m(transaction ID, amount, *Order*, validity period, $CERT_m$, *purchase agreement)

The *Order* message contains the email address of buyer 2 and the certificate of buyer 1. The certificate of buyer 1 identifies to buyer 2 the person who initiated the order. Buyer 2 may approve the purchase and continue with the transaction. Alternatively, buyer 2 may terminate the transaction by not proceeding. In this case, the timeout capability of the protocol will take effect and the transaction will be aborted.

8.1.5 Step 5: Buyer 2 Sends TTP a Order Payment Message

This step is exactly the same as that for the single buyer scenario. All subsequent steps are also the same as those for the single buyer scenario.

8.2 Properties of Our Two-Buyer Protocol

Our two-buyer protocol has the following properties:

- Buyer 1 does not have to register with the merchant.
- Buyer 2 does not have to register with the merchant.
- Buyer 1 has to optionally install the SPP Electronic Wallet application. If the wallet is installed then buyer 1 can sign the *Order* message and buyer 2 would have undeniable proof of the person who has initiated the order. Note that if buyer 1 does not sign the message buyer 2 still has sufficient information to validate the message and maintain a reasonable amount of security.
- Buyer 1 can view the status of the order at the merchant's storefront.
- Buyer 2 must approve the order made by buyer 1 in order to complete the transaction.

8.3 Extension to Improve Buyer Operation

The steps for our extended protocol are shown in Figure 8-2 and Table 8.2.

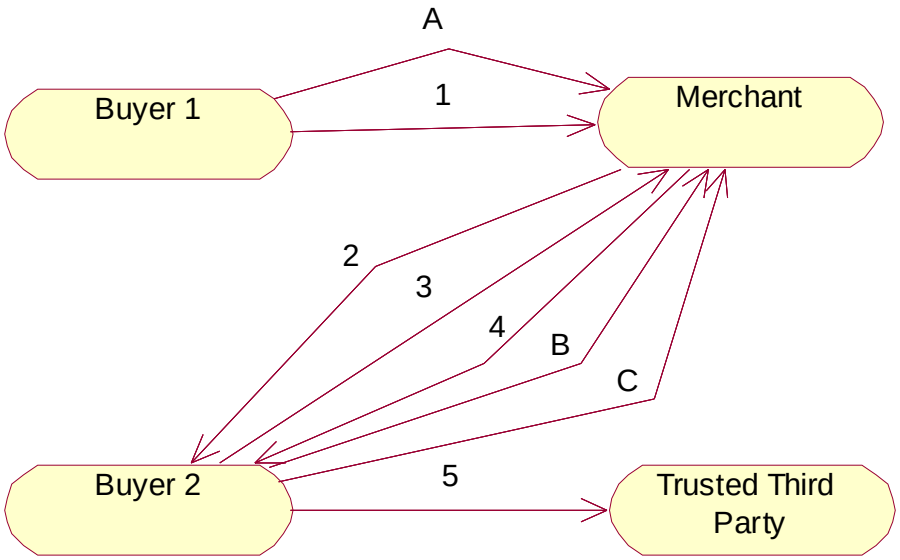

Figure 8-2 Two Buyer SPP Scenario: Extended Protocol

Step Number	**Step Description**
A	*Buyer 1 registers with the merchant*
B	*Buyer 2 registers with the merchant*
C	*Buyer 2 sends a Customer Order Submit Permission List message to the merchant*

Table 8-2 Extra Steps for the Extended Protocol

The motivation for the extension is to solve the problem that no restrictions have been placed on the number of order request messages that may be given to buyer 2.

Anyone can place an order and ask buyer 2 to approve and pay for it. This problem is not necessarily a security exposure as buyer 2 must approve each and every transaction he/she receives. It is more of an inconvenience and a disturbance. To prevent this situation from occurring, steps A, B and C shown in Figure 9-4 are performed prior to the start of a two buyer transaction.

Step A involves buyer 1 registering with the merchant. Buyer 1 would obtain a logon id after the registration is complete. Step B involves buyer 2 registering with the merchant as well. Step C involves buyer 2 sending a *Customer Order Submit Permission List* message to the merchant. The contents of this message are:

- List of customer logon ids that can submit an order message that buyer 2 may pay for and an upper limit on the dollar amount for an order for each customer on the list.
- Logon id of the person creating the list

For example, assuming that Bob and Jack registered with the merchant with the respective logon id's of bob123 and jack456. A *Customer Order Submit Permission List* message contents submitted by fred789 would look as follows:

```
bob123, $100 USD
jack456, $500 USD
fred789
```

The merchant would store this information in its database and only those users on the list would be allowed to initiate an order where payment will be requested from fred789.

9 Conclusions and Future Work

9.1 Summary

In this thesis, a security analysis of the SPP protocol was performed. We have investigated existing online payment protocol domain to determine a list of desirable features of a secure payment protocol. This includes dispute resolution, confidentiality, anonymity, non repudiation, message integrity, and availability and reliability.
The SPP protocol was then analyzed to see if these features are indeed supported.

We next considered a proof analysis of a secure payment protocol. The goal was to list all of the proofs required by each participant in a payment transaction in order to feel secure. The protocol was analyzed to determine if each proof is supported. For the SPP protocol, the participants are the buyer, the merchant, the TTP and the Payment Centre. An example of a proof required by the merchant is that the credit card information supplied by the buyer really belongs to that buyer and not someone else.

We found that SPP supports a majority of the desirable features of a secure payment protocol and a majority of the proofs listed in the proof analysis. Even though the protocol provides a high level of security, there are further enhancements that can be made. The protocol analysis has led to suggestions that can be used to further enhance security. For example, it was determined that using the cryptographic technique of enveloping messages would help increase security. Also using signed timestamps from a timestamp authority would further secure the validity of digital signatures. There is however a cost to implementing the extra security enhancements. In most cases the cost is overhead that leads to performance degradation. Further research is required to

determine if the overhead would justify the extra security in each of the security enhancements.

A prototype of the SPP payment protocol was implemented. The prototype consists of three software components, the SPP electronic wallet used by a buyer, SPP merchant software and SPP TTP software. The software architecture of the prototype was presented in this thesis. We have gained significant knowledge about the ideal technologies to use to implement such a system. Also, the prototype is useful for performance measurements. From our experiments, we found that processing in connection with digital signatures contributes significantly to the completion time of a transaction. We also found that a larger key size provides more security but also requires more processing power.

We have also presented an extension of SPP to support a two buyer scenario. Such an extension would allow merchants to offer additional services such as gift services where one buyer places an order and another buyer pays for the order.

9.2 Future Work

The current SPP protocol description mainly discusses the messages that are exchanged between the various parties in a SPP transaction. The description is at a very high level and it only discusses ideas at a conceptual level. A complete description of the protocol would be a valuable piece of documentation. The resulting document would essentially be a Software Requirements Specification document. It would contain more specific information such as:

- Buyer registration requirements.
- Merchant registration requirements.

- Description on how the buyer and merchant obtain the digital certificates.
- Discussion about the Certificate Authority (CA).
- Policies concerning digital certificates.
    - What specific information needs to be included in the digital certificate? For example, name, address, possibly a one way hash of the credit card number.
    - Certificate expiration policy.
- Cryptographic algorithms used for digital signatures and digests (RSA, PGP)
- Selecting a certificate revocation schema for validating digital certificates.
- Description of the validation algorithms performed for each message by each participant in a transaction.
- Database model for the TTP.
- Use cases describing the various main line scenarios as well as dispute resolution scenarios.

For this thesis, a prototype of the payment protocol was implemented and integrated with a simulation of a merchant commerce server. Integrating this protocol with a production commerce server such as Websphere Commerce Business Edition [IBMWS] would be very valuable proof of concept.

Security enhancements to protocol were discussed in Section 5.4. As future work the problems discussed in that section should be analyzed with a goal of obtaining solutions that can result in improved security.

A further performance evaluation of the protocol would also be very useful. Every security feature comes at a cost and the cost is performance. The following criteria could be considered as part of a performance study:

- Validation of signed messages using Certificate Revocation Lists (see section 4.3.4).
- Validation of signed messages without using Certificate Revocation Lists.
- Validation of signed messages using other revocation mechanisms such as Certificate Revocation Trees.
- Using digital envelopes (see section 4.3.1).
- TTP storing encrypted messages in database (see section 4.3.2).
- Using timestamps generated by a timestamp authority (see section 4.3.3).

Bibliography

[AsoS] N. Asokan, P. Janson, M. Steiner, M. Waidner, "State of the Art in Electronic Payment System," IEEE Computer, Vol 30, No. 9, Sept. 1997, pp. 28-35.

[Cybercash] Cybercash homepage : http://www.cybercash.com.

[CYBP] D. Eastlake, "CyberCash Credit Card Protocol Version 0.8" http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1898.html#sec-1.1.

[DigiStamp] Digistamp homepage : http://www.digistamp.com/.

[IBM DB2] IBM DB2 Database : http://www-3.ibm.com/software/data/db2/udb/.

[IBMWS] IBM WebSphere Commerce Suite : http://www-3.ibm.com/software/webservers/commerce/wcs_pro/.

[JavaPlugin] Java Plugin : http://java.sun.com/products/plugin/.

[LeviR]A. Levi, C. K. Koc, "Reducing Certificate Revocation Cost Using NPKI," IEEE Journal on Selected Areas in Communications , vol. 18, no. 4, pp. 561-570, April 2000.

[MacN] A. MacCullagh, W. Caelli, "Non-repudiation in the Digital Environment" August 2000, http://www.firstmonday.dk/issues/issue5_8/mccullagh/

[PGP] PGP website : http://www.pgpi.org/.

[PKCS7] B. Kaliski, J. Staddon, "Public-Key Cryptography Standards" http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html.

[RSA] RSA homepage : http://www.rsa.com/.

[RSASEC] RSA Security : http://www.rsasecurity.com/rsalabs/technotes/bernstein.html.

[Set] SET Electronic Transaction homepage: http://www.setco.org/.

[SET1] "SET Secure Electronic Transaction Specification : Book 1: Business Description" Version 1.0 May 31, 1997. http://www.setco.org/download/set_bk1.zip.

[SET2] "SET Secure Electronic Transaction Specification, Book 2: Programmer's Guide" Version 1.0 May 31, 1997. http://www.setco.org/download/set_bk2.zip.

[SPP] Wong, J.W., Mirlas, L., Kou, W. and Lin, X. "Credit-Card Based Secure On-line Payment," Payment Technology for E-Commerce, edited by W. Kou, Springer-Verlag, 2003, 227-243.

[SSL] SSL Protocol : http://www.netscape.com/eng/ssl3/ssl-toc.html.

[SSLIntro] Introduction to SSL: http://developer.netscape.com/docs/manuals/security/sslin/index.html.

[Ver] Verisign homepage : http://www.verisign.com.

[VER1] "Online Payment Processing: What You Need To Know" : http://www.verisign.com/resources/gd/enablePayment/enablePayment.pdf.

[VER2] "Fraud Prevention: What Every Merchant Should Know About Internet Fraud" : http://www.verisign.com/resources/gd/internetFraud/FraudPrevention.pdf.

[Xerces] Xerces XML Parser: http://xml.apache.org/

[XmlSchema] XML Schema homepage : http://www.w3.org/XML/Schema

[ZhouO] J. Zhou, R. Deng, "On the Validity of Digital Signatures" April 2000, http://www.acm.org/sigs/sigcomm/ccr/archive/2000/april00/Zhou_final2.pdf.