10 Secure Electronic Transactions: Overview, Capabilities, and Current Status

Gordon Agnew

A&F Consulting, and
University of Waterloo, Ontario, Canada

10.1 Introduction

Until recently, there were two primary forms of credit card transactions:

1) Card present and,
2) Card not present or mail order telephone (MOT).

In a typical "in store" transaction, the customer presents their credit card to perform a transaction. The merchant "swipes" the card and the customer's credit card information along with the amount of the transaction is forwarded to a payment gateway. Once the credit information is verified, the payment gateway returns an authorization to the merchant and a receipt is issued to the customer. In the event of fraud on the part of a customer, the merchant is indemnified against loss since the payment gateway authorized the transaction.

In the case of a purchase made via the telephone, the customer's credit card is not physically present for verification. Generally, the merchant simply accepts the customer's card number over the phone and completes the transaction. Since no authorization was issued by the payment gateway, liability for customer fraud rests with the merchant. For some merchants, this risk is acceptable if profit margins were large enough. On the other hand many merchants have found the risks unacceptable.

In 1996, Mastercard and Visa announced their support of a developing standard for electronic credit card transactions. This replaced the competing standards that each company was pursuing independently. In 1997Visa and Mastercard pooled their resources and formed Secure Electronic Transaction LLC (SETCo) to implement the *SET Specification* [10.4].

SETCo manages the Specification, oversees SET product-compliance testing, and promotes the use of SET as a global payment standard.

The secure electronic transaction (SET) protocol, in many ways, mirrors a card-present transaction over the Internet.

In the following sections, we will examine the details of the operation of SET as well as compare its capabilities to other protocols used in electronic commerce. In addition, we will look issues related to SET's adoption and refinements to the protocol.

10.2 Protocol Stack and Capabilities

There are many functions required to implement a large interconnected network such as the Internet. To facilitate this, network functionality is usually divided into a set of layers or the *protocol stack*. Each layer "talks" to a corresponding or "peer" layer at the other end of the communications channel. Each layer works transparently with the other layers in the network. The lowest layer in the network is the *physical* layer that involves the actual means for transporting data, for example, the actual cables or fibre optics that form the network. At the highest level is the *application* layer which are the programs that are run by the user. One major advantage of such a structure is that the user does not have to be concerned with how the lower layers are implemented. The user simply runs the application and information is passed locally down through the various layers to the Physical layer. The user's data is passed to the physical layer at the destination server then back up to the corresponding application layer at the other end of the connection. In most cases, this layer transparency is realized by a process called "encapsulation;" as data is passed from a higher layer to a lower, the lower layer takes the original data and adds header and control information then passes it down to the next layer. At the destination, the process is reversed and the header/control information is stripped off as data is passed to higher layers. This process simplifies implementation of networks but introduces some interesting security issues, as we will explain below.

There are several levels at which security can be introduced to protect Internet connections. The level in which security functions are used has a strong impact on what types of security can be provided. In Fig. 10.1, three security protocols are shown as they fit into the Internet protocol stack. The three we will consider are IPSec, SSL and SET.

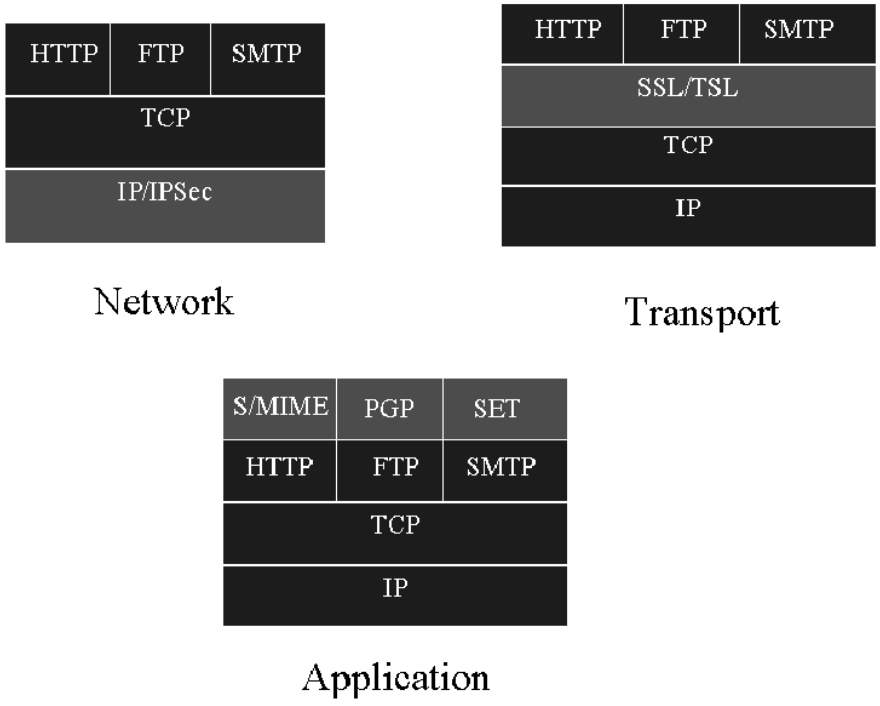

Fig. 10.1 Security layers

10.2.1 Internet Protocol Security (IPSec)

As we see in Fig. 10.1, Internet protocol security (IPSec) is implemented in a relatively low layer. IPSec provides the facilities to encrypt and authenticate user's data (*payload*). If this done, an attacker can see the where the information is going (at least what IP addresses are involved) but not the information itself. In addition, IPSec has the option of taking a standard IP message encrypting it and placing it in a new IP message with a new "disguised" header. This is known as tunnel mode and allows users, for example, to set up private groups over the Internet (virtual private networks).

There are several advantages to providing security at this level:

- Security functions are transparent to the user – the user may not even be aware they are being used.
- The identity of participants can be protected as their IP addresses can be masked.

There are also disadvantages:

- The security functions only protect the IP layer and below – one data is passed to higher layers, it is not protected. If several users are on the same system, information for one user may be visible to other users.
- Identities of users can only be resolved to an IP address. It is common that many users may share an IP address thus, authentication of a particular user may not be possible.

10.2.2 Secure Socket Layer (SSL)

Secure Socket Layer (SSL) was developed by Netscape and is currently in version 3. It is the defacto standard for Internet security at this level and is implemented in most browsers and by most web servers. SSL is designed to provide security functions independent of the application. Since it works at a higher layer than IP-Sec, identities can be resolved to the level of an individual. By the same token, SSL by itself cannot prevent an observer from knowing who is communicating since IP addresses will be added at the lower layers.

SSL designates two types of participants: clients and servers. Clients always initiate a communications session with a server. The server is required to provide authentication information to the client (a certified public key) if requested. The client, however, is not required to provide a certified public key to the server. If this is the case, the applications using SSL may require some other means of authenticating the user (such as a user ID and password/PIN). Once the session has been negotiated, SSL provides a secure (encrypted) and authenticated (data-integrity checks) communications channel between the client and server.

10.2.3 SET

As shown in Fig. 10.1, SET provides security functions at the highest (application) layer of the protocol stack. As in our previous discussion, there are advantages and disadvantages to this. SET is an application and security its functions are not available to other applications. The integrity of SET relies on the ability to resolve identities to a particular individual, merchant or payment gateway (through the use of a full public key infrastructure as we will discuss in Section 10.3) as well as the ability to protect the information exchanged.

As with SSL, even though the information is protected, and observer can still glean information about the participants in a transaction.

10.3 SET Overview

In this section, we will examine the structure of SET and its related security functions[1].

There are two major parts to the SET protocol.

- Registration
- Transaction processing

10.3.1 SET Registration

The security and integrity of transactions are heavily reliant on the use of certified public keys or *public key certificates.* To create a certificate, the user presents unique identification information (ID) and their public key to a c*ertificate authority* (CA). Once the CA is satisfied that the user is authentic (for example, the manager of a bank may authenticate a particular customer), the CA binds the ID and public key of the user together (usually by creating a message digest[2]) then forms a *digital signature*[3] on the result. For another participant to verify the public key of a particular user, they require a trusted copy of the CA's public key in order to verify the certificate. It is assumed that a trusted version of at least on CA's public key is available to the participants.

SET recognizes three types of participants in a transaction.

- The customer (cardholder)
- The merchant
- The payment gateway.

SET then defines a hierarchical approach to creating and distributing public-key certificates for each type of participant. This is shown in Fig. 10.2. Here, the highest member of the hierarchy is the *root certificate authority* maintained by SETCo. The root authority issues public key certificates to the various payment brands. These in turn become Certificate Authorities authorized to issue certificates to their member banks.

---

[1] A full description of SET can be found in SET Specification Books [10.4]

[2] A message digest is a fixed length image of a longer message formed using a transformation that is "one-way" and unpredictable. That is, it is very easy to create but virtually impossible to find a second message that would create the same image. For a more in depth look at cryptographic functions, the reader is referred to [10.2]

[3] A digital signature is formed using the signer's private key. It can be verified using the signer's public key.

Further down the hierarchy are the certificate authorities associated with each type of participant in a transaction. The *payment card issuing certificate authority* issues public key certificates to customers. The *merchant bank* or *acquirer certificate authority* issues public key certificates to the merchants while payment gateways have their own certificate authority.

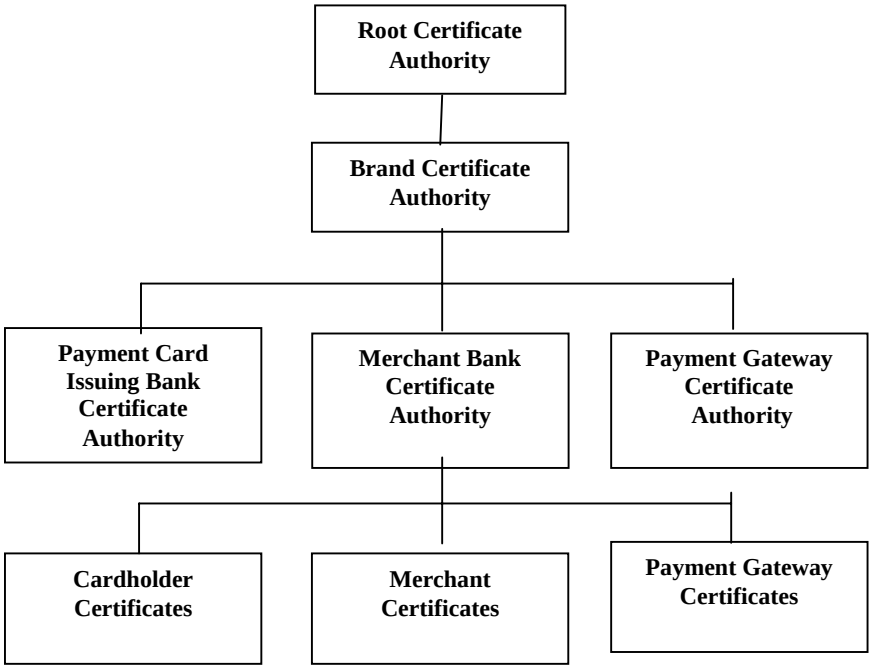

Fig. 10.2 SET certificate hierarchy

In such a hierarchy, a *certificate chain* can be used to verify any member of the hierarchy. For example, for a particular merchant, the certificate chain might include their own public key certificate issued by their acquirer CA, a certificate on the acquirer CA issued by the brand CA and finally the certificate of the brand CA as issued by the root CA. A trusted version of the root CA's public key would allow the chain to be verified. A graphic representation of a certificate chain is shown in Fig. 10.3.

Merchant's Certificate (from Merchant's Bank CA)	Merchant's Bank CA Certificate (from Brand CA)	Brand's CA Certificate (from Root CA)

Fig. 10.3 Example certificate chain for a merchant

10.3.2 Transaction Processing

There are three main phases in a secure electronic transaction:

- Purchase request
- Payment authorization
- Payment capture

An overview of the interaction among the participants in a transaction is shown in Fig. 10.4.

Purchase Request Phase

The details of the purchase request are shown in Fig. 10.5. Within the purchase-request phase, there are 5 basic steps, as we will describe.

Initiate Request

The process starts with the customer shopping, and selecting an item or items. The customer has a completed order form and has selected a particular payment card. The customer's (cardholder's) computer running the cardholder's software package (hereafter called just the *cardholder*) sends an *initiate request* (*P INIT REQ*) message to the merchant requesting the certified public key of the payment gateway.

Initiate Response

Once the merchant receives the initiate request, it assigns an unique transaction ID to the message and returns a signed version of the transaction ID, its own certifi-

cate and the appropriate (for the particular brand) payment gateway's certificate to the cardholder.

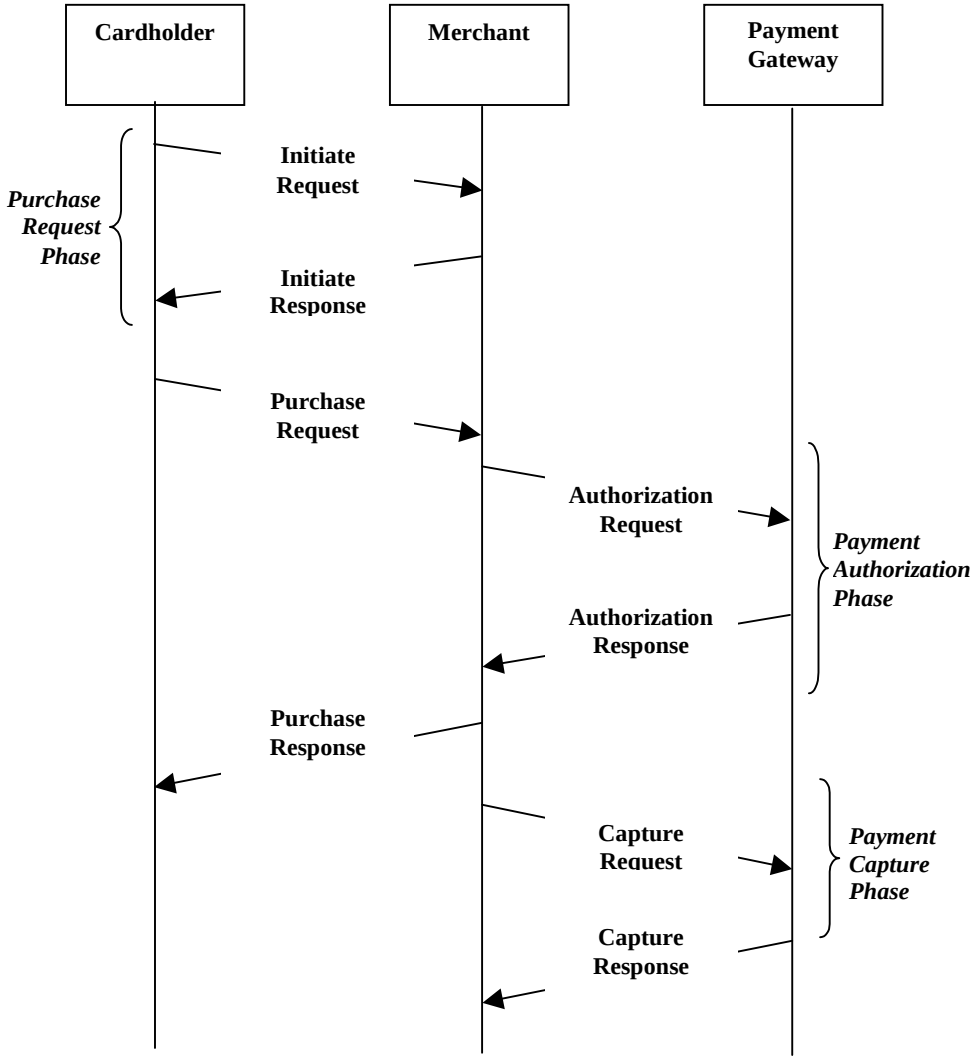

Fig. 10.4 SET overview

Cardholder Purchase Request

Once the response is received, the cardholder verifies the certificates of the merchant and gateway as well as the merchant's digital signature on the transaction

information. Once this is complete, the cardholder creates two messages: an *order information* (OI) message intended for the merchant and a *payment information* (PI) message intended for the payment gateway. The PI message information such as the credit card number of the cardholder and will be concealed from the merchant. These messages both contain the unique transaction ID that the merchant assigned. This is done so that the two messages can be linked to one another.

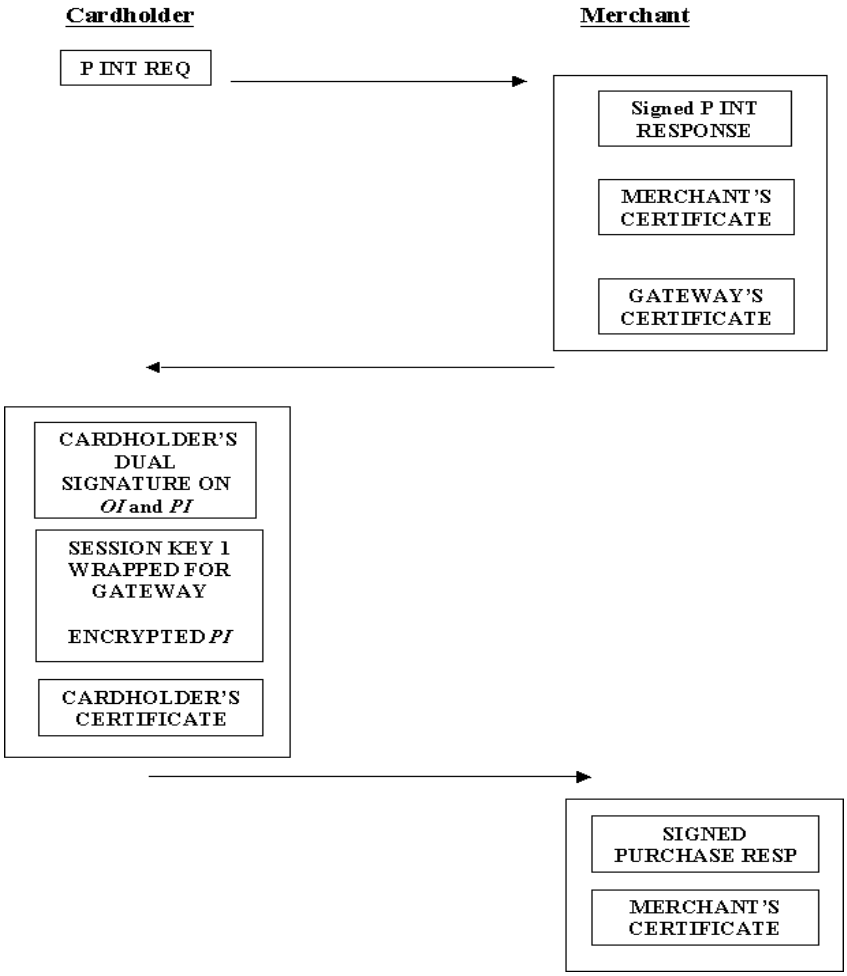

Fig. 10.5 Purchase request phase

At this point, a very elegant method is used bind the two messages together. The cardholder forms message digests of both the OI and PI. These digests are

concatenated, then a third message digest is formed. This final digest is then digitally signed by the cardholder. This forms the *dual signature* on OI and PI.

The next step is used to hide the PI information from the merchant. The cardholder generates a random session key (to be used with a conventional encryption algorithm) that is used to encrypt the PI. To transport this information to the payment gateway, the cardholder combines the random session key and their account information into a message then encrypts it using the payment gateway's public key (so that only the PG can recover the account information and the session key that can decrypt the PI).

Merchant then is forwarded a message containing the PI and OI digest, the dual signature, the "wrapped" version of the PI, session key and account information and the cardholder's certificate.

The reason for the dual signature scheme is as follows: the payment gateway will only have a digest of the order information and not the order itself. The payment gateway cannot determine the purchase from that information. If a dispute arises, between the merchant and customer, the OI can be produced and the payment gateway with knowledge of the PI can regenerate the message digests and verify whose claim is correct. This is an important element in security of SET.

Merchant's Purchase Request Processing

When the purchase request is received at the merchant, it verifies the cardholder's certificate. This is then used to verify the dual signature on the OI and digest of the PI to ensure no tampering of the OI has occurred.

Once this has been verified, the merchant generates a digitally signed *purchase response* message that is returned to the cardholder.

Purchase Response

In the final step in this phase, the cardholder uses the merchant's certified public key to verify the purchase response. This is stored for future reference.

Payment Authorization Phase

This part of the protocol involves the merchant and the payment gateway. The objective is for the merchant to acquire authorization for the transaction. There are three basic steps, as shown in Fig. 10.6.

Merchant Authorization Request

The merchant starts by creating a digitally signed authorization request that includes the amount to be authorized, the transaction ID, and other details about the transaction.

The merchant generates a random session key that is used to encrypt this message. The session key is then wrapped using the payment gateway's public key.

This information is sent along with the cardholder's PI information and wrapped session key, cardholder's certificate and merchant's certificate.

Payment Gateway Processing

When the gateway receives the authorization request, it uses its private key to recover the wrapped session key. This is then used to decrypt the request. The merchant's certificate is verified then used to verify the signature on the request.

Next, the second session key and customer account information are recovered. The session key is then used to recover the PI. The cardholder's certificate is verified and the digital signature on the OI and PI is verified. As a further check, the Transaction ID's on both parts of the message are compared to ensure that they are the same.

The next operation involves the payment gateway creating a message for the issuing bank. This is done over the private financial network.

If the purchase is authorized, then a digitally signed response message is generated by the payment gateway. This message is encrypted with a new random session key that is wrapped using the merchant's public key, then forwarded to the merchant.

Merchant Response Processing

When the response is received by the merchant, the payment authorization is recovered and the signature is verified. A copy of this authorization is kept by the merchant.

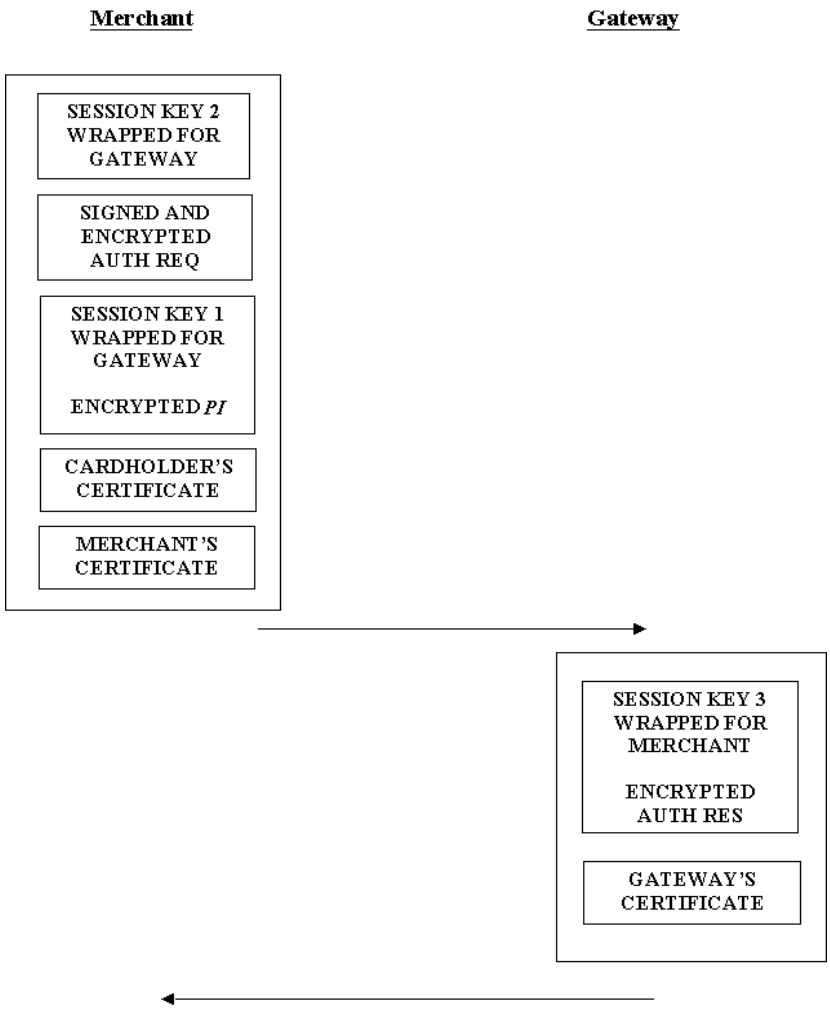

Fig. 10.6 Payment authorization phase

Payment Capture Phase

The final phase in the SET protocol is payment capture. In this phase, the Merchant requests payment from the payment gateway. This phase may occur sometime after the transaction has occurred and involves three basic steps, as shown in Fig. 10.7.

Merchant Payment Capture Request

The merchant creates a digitally signed payment request that includes the final transaction amount, the transaction ID, and other transaction information. This is encrypted using a new random session key that is wrapped using the payment gateway's public key. The encrypted message is sent to the payment gateway along with the merchant's certificate.

Payment Gateway Capture Processing

Upon receipt, the payment gateway recovers the session key, capture request then verifies the merchant's certificate and signature on the request

The payment gateway generates a digitally signed and encrypted response message that is forwarded to the merchant along with the gateway's certificate.

Merchant Processing of Response

This is the final step in the protocol. The merchant recovers the session key and the capture message and verifies the gateway's certificate as well as the digital signature on the message. This is stored by the gateway for reconciliation for payment from the issuer.

10.4 SET Performance

From the description of the SET protocol, it is apparent that SET provides a high level of security and privacy for the participants. This is mainly due to the extensive use of public key certificates and digitally signed and verified messages. This has several important implications. Trust in the system relies on the deployment of a full public key infrastructure. If SET is to be used on a wide-scale basis, certificates have to be issued to all users. This is an enormous and expensive task. On the other hand if the PKI is not in place, then SET will not be used by a large number of users.

In version 1.0 of SET, RSA is specified to implement the public key operations. At present a minimum of 768-bit RSA is required for security, preferably 1024-bit. Public key operations (signing/verifying, wrapping/unwrapping) are computationally intensive, and certificates are large in size and require significant bandwidth to transmit.

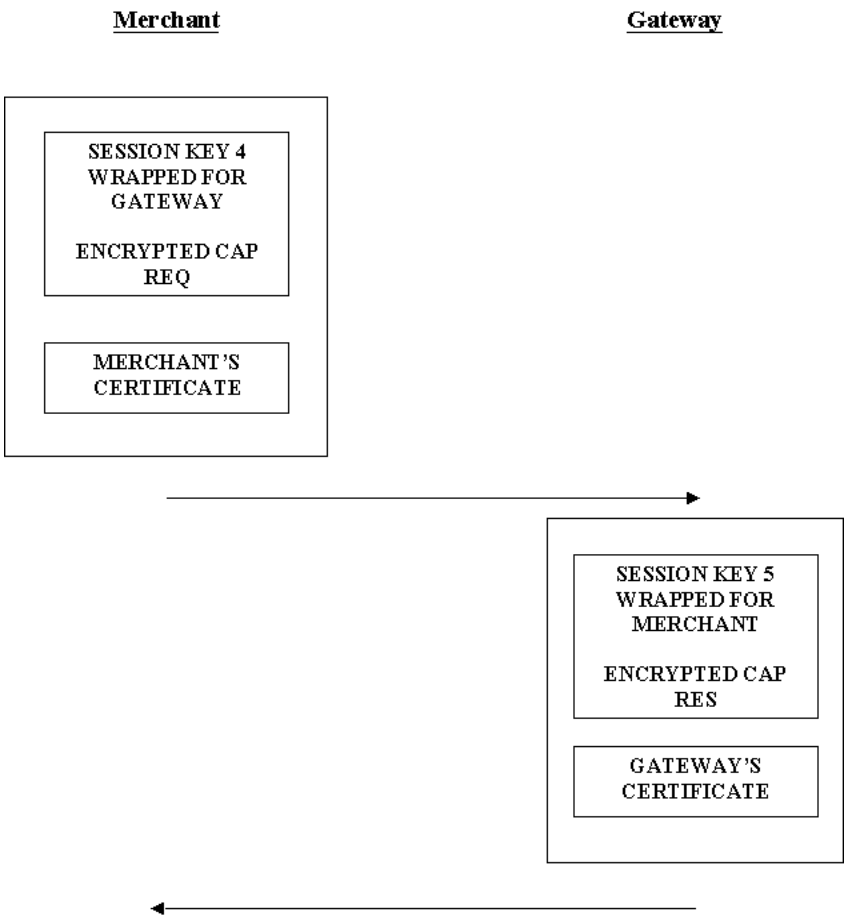

Fig. 10.7 Payment capture phase

In the case of the cardholder using a typical desktop computer, the computational load is not significant. If, on the other hand, the cardholder is not bound to a particular machine, then the cryptographic functions may be implemented in a portable token, such as a smart card. Implementing RSA on smart cards usually requires the smart card to have a cryptographic co-processor that raises the cost of the card.

There is also the issue of conducting e-commerce transactions using wireless handheld devices, such as cell phones or PDAs. In these situations bandwidth and processing power are at a premium and supporting SET may be difficult.

The GartnerConsulting Group did an extensive evaluation of the performance of SET [10.1]. In the study, it was anticipated that merchants could expect in the

order of 10,000 transactions per day while a large payment gateway may approach ½ million transactions per day. In this case, software implementations of the public-key system may not be able to perform operations quickly enough; hardware accelerators may be required (adding to the cost of the infrastructure). They also examined the advantages of using other public key cryptographic systems. In their report, *elliptic curve cryptosystems*[4] (ECC) were considered and shown to have significant advantages in terms of bandwidth and processing overhead.

Sans and Agnew [10.3] present the results of an extensive study of the communications and processing overhead for SET. They show some alternative methods for processing transactions that reduce the overhead incurred using SET.

10.5 What Lies Ahead

There are a number of companies currently offering support for SET. These include IBM, Verisign, CyberTrust, Verifone, Sterling Commerce, Terisa, Netpay and GlobeSet.

SETCo lists more than 40 countries that have adopted SET in one form or another [10.4].

A proposal for SET 2.0 incorporates alternative asymmetric key cryptographic systems (specifically, elliptic curves) and SET 2.0 will also support the use of debit cards by allowing personal identification numbers (PINs) to be encrypted and included in the payment message [10.5]. In addition, a smart-card-based version known as chip-secured SET (C-SET) is being developed to allow smart cards to perform cardholder authentication and transaction security functions (encryption and signatures).

10.6 Summary

In this chapter, we have presented a detailed outline of the SET protocol. The capabilities and shortcomings of SET have been compared to other Internet security protocols.

Currently, SSL is the most widely deployed and used security protocol. It is relatively fast and provides transparent security to the user. It does not, however

[4] The reader is referred to www.certicom.com for a more complete review of ECC technology.

provide the mutual authentication and digital signature capabilities that are required for truly secure e-commerce.

SET, on the other hand, is a very robust protocol that provides a high level of security and trust. The major impediments to widespread deployment and use of SET are the current lack of a comprehensive public key infrastructure and the large overhead required to run the SET protocol. Improvements in processing power and the use of alternative public key cryptosystems such as elliptic-curve-based systems (ECC) may help to overcome some of these obstacles.

10.7 References

[10.1] GartnerConsulting: SET comparative performance analysis. www.setco.org/download/setco6.pdf
[10.2] A. J. Menezes, P. Oorschot, S. Vanstone (1997) Handbook of applied cryptography. CRC Press, New York.
[10.3] O. Sans, G. Agnew (2001) An efficient multiple merchant payment protocol for secure electronic transactions based on purchase consolidation. In: W. Kou, et al. (eds.) Electronic commerce technologies – ISEC2001, LNCS 2048. Springer, Berlin Heidelberg New York.
[10.4] www.setco.org/set_specifications.html
[10.5] www.setco.org/extensions.html