Like Chapter 7, this chapter looks at security protocols that are used in today's networks. But unlike the protocols described in Chapter 7, the protocols discussed in this chapter provide security services for specific applications (such as Simple Mail Transfer Protocol (SMTP), Hypertext Transfer Protocol (HTTP), and so on). Figure 8-1 shows each of these application protocols along with its location in the Transportation Control Protocol/Internet Protocol (TCP/IP) stack. This chapter provides a detailed look at two well-known security protocols—S/MIME and SET—which operate above the application layer.

S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) is a specification for securing electronic mail. S/MIME, which is based on the popular MIME standard, describes a protocol for adding cryptographic security services through MIME encapsulation of digitally signed and encrypted objects. These security services are authentication, nonrepudiation, message integrity, and message confidentiality.
Although S/MIME is most widely known and used for securing e-mail, it can be used with any transport mechanism that transports MIME (such as HTTP). S/MIME can even be used in automated message transfer agents, which use cryptographic security services that do not require human intervention. The S/MIME specification even points out how to use its services to encrypt fax messages sent over the Internet.

The following section describes S/MIME along with the various MIME types and their uses. It explains how to create a MIME body that has been cryptographically enhanced according to Cryptographic Message Syntax (CMS), a formatting standard derived from PKCS #7. Finally, it defines and illustrates how cryptographic signature and encryption services are added to MIME data.

**Overview**

In the early 1980s, the Internet Engineering Task Force (IETF) developed Request for Comment (RFC) 822, which became the specification that defined the standard format of electronic mail messages. Along with RFC 821 (which defined the mail transfer protocol), RFC 822 was the foundation of the SMTP, which was designed to carry textual messages over the Internet.

MIME, also developed by the IETF, was designed to support nontextual data (such as graphics or video data) used in Internet messages. The MIME specification adds structured information to the message body that allows it to contain nontextual information. However, MIME does not provide any security services.

In 1995, RSA Data Security, Inc., led a consortium of industry vendors in the development of S/MIME. After work on the specification was under way, RSA passed it to the IETF for further development. S/MIMEv3 is the current version. Through continued development by the IETF S/MIME working group, the protocol has incorporated a number of enhancements.
S/MIME Functionality

S/MIMEv3 currently provides the following security enhancements to MIME content:

- **Enveloped data**  This function supports confidentiality services by allowing any content type in a MIME message to be symmetrically encrypted. The symmetric key is then encrypted with one or more recipients’ public keys. The encrypted data and corresponding encrypted symmetric key are then attached to the data structure, along with any necessary recipient identifiers and algorithm identifiers.

- **Signed data**  This function provides data integrity services. A message digest is computed over the selected content (including any algorithm identifiers and optional public-key certificates), which is then encrypted using the signer’s private key. The original content and its corresponding signature are then base-64 encoded (base-64 and other encoding methods are described later in “Transfer Encoding”).

- **Clear-signed data**  This function allows S/MIME to provide the same data integrity services as provided by the signed data function, while at the same time allowing readers that are not S/MIME-compliant to view the original data. Following the processes just described, a digital signature is computed over the selected content, but only this digital signature (and not the original data) is base-64 encoded.

- **Signed and enveloped data**  This function supports both confidentiality and integrity services by allowing either the signing of encrypted data or the encrypting of signed data.

Cryptographic Algorithms

S/MIMEv3 implements support for several symmetric content-encryption algorithms. However, some S/MIME implementations still incorporate RC2 with a key size of 40 bits, and by today’s standards, a 40-bit key is too weak. However, in most current S/MIMEv3 implementations, the user can choose from various content-encryption algorithms, such as DES, Triple DES, or RC2 with a key size greater than 40; see Chapter 2.
The specification does, however, spell out all algorithms to be used for security services within S/MIMEv3. Some of them are optional, and others are required. They are as follows:

- **Digest and hashing algorithms**  These must support MD5 and SHA-1; however, SHA-1 should be used.

- **Digital signature algorithms**  Both sending and receiving agents must support DSA and should also support RSA.

- **Key encryption algorithms**  Sending and receiving agents must support Diffie-Hellman and should also support RSA encryption.

- **Data encryption (session key) algorithms**  Sending agents should support RC2/40-bit key, RC2/128-bit key, and Triple DES. Receiving agents should support RC2/128 and Triple DES but must support RC2/40.

Which algorithm is best? It’s a simple matter of looking at key length; the bigger the key, the greater the security. However, sending and receiving agents are not always at the same level. For instance, the sending agent may be attempting to encrypt something with RC2/128 for added security; however, the receiving agent may only have the ability to decrypt messages with RC2/40. For this reason, the S/MIME specification defines a process for deciding which algorithm is best when you’re sending S/MIME messages.

The following are the specified rules that a sending agent should use in making its decision:

1. **Known capabilities.** If the sending agent has previously received a list of cryptographic capabilities of the recipient, the sender should choose the first (most preferred) capability listed to encrypt the outgoing message.

2. **Unknown capabilities but known use of encryption.** This rule applies when the sending agent has no idea of the encryption capabilities of the recipient but has received at least one previously encrypted message from that recipient. In this case, the sending agent should encrypt the outgoing message using that algorithm.

3. **Unknown capabilities and unknown version of S/MIME.** This rule applies when a sending agent has had no previous contact with the recipient and does not know its capabilities. The sending agent should use Triple DES because of its strength and because it is required by S/MIMEv3. However, if Triple DES is not used, the sending agent should use RC2/40.
S/MIME Messages

S/MIME messages are made up of the MIME bodies and CMS objects. The latter are derived from PKCS #7 data structures.

Before any cryptographic processing takes place, a MIME entity must be prepared. A MIME entity may be a subpart of a message or the whole message, including all its subparts. The latter type of MIME entity is made up only of the MIME headers and MIME body and does not include the RFC822 headers (To:, From:, and so on). This MIME entity is then converted to canonical form, and the appropriate transfer encoding is applied (both processes are discussed in the following sections).

After the MIME entity has been created and all proper encoding has taken place, the MIME entity is sent to security services, where the chosen security function is provided (enveloping, signing, or both). This process yields a CMS (or PKCS #7) object, which in turn is wrapped up in MIME and placed with the original message, according to the selected S/MIME content type.

Canonicalization

As stated in the preceding section, each MIME entity must be converted to a canonical form. This conversion allows the MIME entity to be uniquely and unambiguously represented in the environments where the signature is created and where the signature will be verified. This same process is performed for MIME entities that will be digitally enveloped as well as signed. Canonicalization provides a standard means by which data from various platforms can be exchanged.

Transfer Encoding

Whenever data is processed by digital equipment, it can be encoded and represented in a variety of ways, such as 7-bit, 8-bit, or binary. Transfer encoding ensures that data is represented properly for transfer across the Internet and ensures reliable delivery. One common method is base-64 encoding, which enables arbitrary binary data to be encoded so that it may pass through a variety of systems unchanged. For example, if 8-bit data is transferred and a 7-bit device (such as a mail gateway) receives it, there is a good chance that before it is forwarded to its final destination, it may be stripped of characters.
NOTE:
As you might expect, if a digitally signed message is altered or stripped of characters, it will be selected as invalid.

Enveloped-Only Data

The process of generating an encrypted MIME entity is called digital enveloping and is provided for by the enveloped-data content type. This content type consists of encrypted content of any type and encrypted content-encryption keys for one or more recipients. For each recipient, a digital envelope (made up of the encrypted content and associated encrypted content-encryption key) is created, ensuring confidentiality for the message while it is in transit. Figure 8-2 illustrates the S/MIME enveloped-data process.

To construct an enveloped-data content type, follow these steps:

1. For a chosen symmetric algorithm (that is, RC2, DES, and so on), generate a pseudo-random content-encryption key.

2. For each recipient, encrypt the content-encryption key. Which encryption to use depends on which key management system is used. The associated key management systems are as follows:

   **RSA key transport**  The content-encryption key is encrypted in the recipient's public key.
Diffie-Hellman key agreement The recipient’s public key and the sender’s private key are used to generate a shared symmetric key, which is then used to encrypt the content-encryption key.

Known symmetric key The content-encryption key is encrypted using a previously distributed symmetric key.

3. For each recipient, create a block of data containing the recipient information. This information includes the encrypted content-encryption key and other recipient-specific information (such as version and algorithm identifiers).

4. Encrypt the message using the content-encryption key.

5. Prepend the recipient information to the encrypted content, and base 64-encode the result to produce the enveloped-data value.

When the digital envelope is received, the process is reversed to retrieve the original data. First, the enveloped data is stripped of its base-64 encoding. Then the appropriate content-encryption key is decrypted. Finally, the content-encryption key is used to decrypt the original content.

Signed-Only Data

The S/MIME specification defines two methods for signing messages:

- Application/pkcs7-mime with signed-data (usable only by S/MIME-compliant mailers)
- Multipart/signed, also known as clear signing (usable by all mailers)

S/MIMEv3 doesn’t mandate which method to use, but the specification mentions that the multipart/signed form is preferred for sent messages because of its readability by any mailer. The specification states that receiving agents should be able to handle both kinds.

Signed Data An S/MIME application/pkcs7-mime message with signed data may consist of any MIME content type, in which any number of signers in parallel can sign any type of content. Figure 8-3 illustrates S/MIME data signing.

The following steps apply to constructing a signed-data content type:

1. For each signer, select a message digest or hashing algorithm (MD5 or SHA-1).
2. Compute a message digest or hash value over all content to be signed.
3. For each signer, digitally sign the message digest (that is, encrypt the digest using the signer’s private key).

4. For each signer, create a signer information block containing the signature value and other signer-specific information (such as version and algorithm identifier).

5. Prepend the signed content with signer information (for all signers), and then base-64-encode it to produce the signed data value.

After it is received, the signed data content type is stripped of its base-64 encoding. Next, the signer's public key is used to decrypt and reveal the original message digest. Finally, the recipient independently computes the message digest and compares the result with that of the one that was just decrypted.

**Clear-Signed Data** It is possible that data you have digitally signed might be received by a recipient that is not S/MIME-compliant, rendering the original content unusable. To counter this problem, S/MIME uses an alternative structure, the multipart/signed MIME type.

The body of the multipart/signed MIME type is made up of two parts. The first part, which can be of any MIME content type, is left in the clear and placed in the final message. The contents of the second part are a special case of signed data, known as a detached signature, which omits the copy of the plaintext that may be contained within the signed data. Figure 8-4 illustrates the S/MIME clear-signed data process.
Signing and Encrypting

S/MIME also supports both encryption and signing. To provide this service, you can nest enveloped-only and signed-only data. In other words, you either sign a message first or envelope the message first. The decision of which process to perform first is up to the implementer and the user.

NOTE:
The S/MIMEv3 specification (RFC2633) describes security risks involved with each technique (envelope first or signing first).

Registration Request

In addition to security functions, S/MIME defines a format for conveying a request to have a public-key certificate issued. A MIME content type, application/x-pkcs10, is used to request a certificate from a certification authority.
NOTE:
The specification does not mandate the use of any specific technique for requesting a certificate, whether it is through a certificate authority, a hardware token, or manual distribution. The specification does, however, mandate that every sending agent have a certificate.

Certificates-Only Messages

A certificates-only message is an application/pkcs7-mime and is prepared in much the same way as a signed-data message. This message, which is used to transport certificates to an S/MIME-compliant end entity, may be needed from time to time after a certification authority receives a certificate request. The certificates-only message can also be used for the transport of certificate revocation lists (CRLs).

Enhanced Security Services

Currently there are three optional enhanced security services that can be used to extend the current S/MIMEv3 security and certificate processing services.

- **Signed receipts**  A signed receipt is an optional service that allows for proof of message delivery. The receipt provides the originator a means of demonstrating to a third party that the recipient not only received but also verified the signature of the original message (hopefully, this means that the recipient also read the message). Ultimately, the recipient signs the entire message and its corresponding signature for proof of receipt. Note that this service is used only for signed data.

- **Security labels**  Security labels can be used in a couple of ways. The first and probably most easily recognizable approach is to describe the sensitivity of data. A ranked list of labels is used (confidential, secret, restricted, and so on). Another technique is to use the labels to control authorization and access, describing which kind of recipient should have access to the data (such as a patient’s doctor, medical billing agents, and so on).

- **Secure mailing lists**  When S/MIME provides its services, sending agents must create recipient-specific data structures for each
recipient. As the number of recipients grows for a given message, this processing can impair performance for messages sent out. Thus, *mail list agents* (MLAs) can take a single message and perform the recipient-specific encryption for every recipient.

**Interoperability**

Since the S/MIME standard first entered the public eye, a number of vendors have made efforts to incorporate it. However, a lack of interoperability is one pitfall that end users should take into account. For example, many vendors are still S/MIMEv2-compliant, whereas others have moved to S/MIMEv3 without supporting backward compatibility. Other problems include limits on the certificate processing available in various products.

To help promote product interoperability, the RSA Interoperability Test Center was established. This S/MIME test center allows vendors to perform interoperability testing on their products and to have the results published. The following Web address provides interoperability information as well as products that have been found to be S/MIME-compliant: http://www.rsasecurity.com/standards/smime/interop_center.html.

**Secure Electronic Transaction (SET)**

The Internet has made it easier than ever for consumers to shop, money to be transferred, and bills to be paid over the Internet at the press of a button. The price we pay for this ease of use, however, is increased opportunity for fraud. For example, Figure 8-5 illustrates how easy it is for those with very little character to fraudulently generate credit cards used for online payment, known in the industry as *payment cards*.

The *Secure Electronic Transaction* (SET) specification provides a framework for protecting payment cards used in Internet e-commerce transactions against fraud. SET protects payment cards by ensuring the confidentiality and integrity of the cardholder’s data while at the same time providing a means of authentication of the card. The current version of the specification (SETv1) was initiated by MasterCard and Visa in February 1996 and was completed in May 1997.

SET is defined in three books. The first book, *Business Description*, describes the specification in business terms (that is, goals, participants,
The specification defines the business requirements of SET as follows:

- To provide confidentiality of payment information and enable confidentiality of the associated order information
- To ensure the integrity of all transmitted data
- To provide authentication that a cardholder is a legitimate user of a branded payment card account
To provide authentication that a merchant can accept branded payment card transactions through its relationship with an acquiring financial institution

To ensure the use of the best security practices and system design techniques to protect all legitimate parties in an electronic commerce transactions

To create a protocol that neither depends on transport security mechanisms nor prevents their use

To facilitate and encourage interoperability among software and network providers

**SET Features**

To meet its stated business requirements, SET defines the following necessary features:

- **Confidentiality of information**  Confidentiality provides a secure channel for all payment and account information, preventing unauthorized disclosure. SET provides for confidentiality through the use of the DES algorithm.

- **Integrity of data**  Data integrity ensures that the message content is not altered during transmission. This feature is provided through the use of digital signatures using the RSA algorithm.

- **Cardholder account authentication**  Cardholder authentication provides merchants a means of verifying the cardholder as legitimate. Digital signatures and X.509v3 certificates are used to implement this function.

- **Merchant authentication**  Merchant authentication gives cardholders a means of verifying that the merchant not only is legitimate but also has a relationship with a financial institution. Again, digital signatures and X.509v3 certificates are used to implement this service.

- **Interoperability**  Interoperability allows the use of this specification in hardware and software from various manufacturers, allowing their use by cardholders or other participants.
SET Participants

Various participants use and interact with the SET specification. Figure 8-6 illustrates a simplified overview of the participants' interactions.

Following are these participants and their roles in the transactions governed by SET:

- **Issuer** The issuer is the bank or other financial institution that provides a branded payment card (such as a MasterCard or Visa credit card) to an individual. The card is provided after the individual establishes an account with the issuer. It is the issuer that is responsible for the repayment of debt, for all authorized transactions placed on the card.
Cardholder   The cardholder is the individual authorized to use the payment card. The SET protocol provides confidentiality services for the cardholder's transactions with merchants over the Internet.

Merchant   The merchant is any entity that provides goods and/or services to a cardholder for payment. Any merchant that accepts payment cards must have a relationship with an acquirer.

Acquirer   The acquirer is a financial institution that supports merchants by providing the service of processing payment cards. In other words, the acquirer pays the merchant, and the issuer repays the acquirer.

Payment gateway   The payment gateway is the entity that processes merchant payment messages (for example, payment instructions from cardholders). The acquirer or a designated third party can act as a payment gateway; however, the third party must interface with the acquirer at some point.

Dual Signatures

The SET protocol introduced dual signatures, a new concept in digital signatures. Dual signatures allow two pieces of data to be linked and sent to two different entities for processing. For example, within SET a cardholder is required to send an order information (OI) message to the merchant for processing; at the same time, a payment instructions (PI) message is required by the payment gateway. Figure 8-7 illustrates the dual signature generation process.

The dual signature process follows these steps:

1. A message digest is generated for both the OI and the PI.
2. The two message digests are concatenated (hashed) to produce a new block of data.
3. The new block of data is hashed again to provide a final message digest.
4. The final message digest is encrypted using the signer's private key, producing a digital signature.

A recipient of either message can check its authenticity by generating the message digest on its copy of the message, concatenating it with the message digest of the other message (as provided by the sender) and computing the message digest of the result. If the newly generated digest

Figure 8-7 illustrates the dual signature generation process.
matches the decrypted dual signature, the recipient can trust the authenticity of the message.

**SET Certificates**

The SET protocol provides authentication services for participants through the use of X.509v3, and has revocation provisions through the use of CRLv2 (both X.509v3 and CRLv2 are described in Chapter 6.). These certificates are application-specific; that is, SET has defined its own specific private extensions that are meaningful only to SET-compliant systems. SET contains the following predefined profiles for each type of certificate:

- **Cardholder certificates** function as electronic representations of payment cards. Because a financial institution digitally signs these certificates, they cannot be altered by a third party and can be generated only by the financial institution. A cardholder certificate does not contain the account number and expiration date. Instead, the account information and a secret value known only to the cardholder's software are encoded using a one-way hashing algorithm.

- **Merchant certificates** function as electronic substitutes for the payment card brand decal that appears in a store window; the decal
itself is a representation that the merchant has a relationship with a financial institution allowing it to accept the payment card brand. Because the merchant’s financial institution digitally signs them, merchant certificates cannot be altered by a third party and can be generated only by a financial institution.

- **Payment gateway certificates** are obtained by acquirers or their processors for the systems that process authorization and capture messages. The gateway’s encryption key, which the cardholder gets from this certificate, is used to protect the cardholder’s account information. Payment gateway certificates are issued to the acquirer by the payment card brand organization.

- **Acquirer certificates** are required only in order to operate a certification authority that can accept and process certificate requests directly from merchants over public and private networks. Those acquirers that choose to have the payment card brand organization process certificate requests on their behalf do not require certificates because they are not processing SET messages. Acquirers receive their certificates from the payment card brand organization.

- **Issuer certificates** are required only in order to operate a certification authority that can accept and process certificate requests directly from cardholders over public and private networks. Those issuers that choose to have the payment card brand organization process certificate requests on their behalf do not require certificates because they are not processing SET messages. Issuers receive their certificates from the payment card brand organization.

### Certificate Management

The SET specification states that certificates must be managed through a strict certificate hierarchy, as shown in Figure 8-8 (certificate hierarchies are explained in Chapter 6).

In the case of SET, each certificate is linked to the signature certificate of the entity that digitally signed it. By following the trust tree to a known trusted party, a person can be assured that the certificate is valid. For example, a cardholder certificate is linked to the certificate of the issuer (or the brand organization on behalf of the issuer). The issuer’s certificate is linked back to a root key through the brand organization’s certificate. The public signature key of the root is known to all SET software and can be used to verify each of the certificates in turn.
Payment Processing

To provide for secure payment processing over the Internet, the SET specification defines multiple transaction types, as shown in Table 8-1.

To illustrate how SET provides security of payment processing within e-commerce transactions, we next discuss each of the following transaction types in depth:

- Purchase request
- Payment authorization
- Payment capture

Purchase Request

The purchase request transaction is made up of four messages that are exchanged between the cardholder and the merchant:

1. Initiate request. When the cardholder has selected a purchase and decided which payment card to use, the cardholder is ready to initiate the request. To send SET messages to a merchant, the cardholder must have a copy of the merchant’s and payment
<table>
<thead>
<tr>
<th>Transaction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cardholder registration</td>
<td>Allows the cardholder to register with a CA.</td>
</tr>
<tr>
<td>Merchant registration</td>
<td>Allows a merchant to register with a CA.</td>
</tr>
<tr>
<td>Purchase request</td>
<td>Message from the cardholder containing order information (OI) and payment information (PI) and sent to the merchant and bank.</td>
</tr>
<tr>
<td>Payment authorization</td>
<td>Message between the merchant and payment gateway requesting payment authorization for a transaction.</td>
</tr>
<tr>
<td>Payment capture</td>
<td>Message from the merchant to the payment gateway requesting payment.</td>
</tr>
<tr>
<td>Certificate inquiry</td>
<td>A CA may send this message to either cardholders or merchants to state that more processing time is needed. <em>or</em> A cardholder or merchant may send this message to a CA to check the current status of a certificate request, or to receive the certificate if the request has been approved.</td>
</tr>
<tr>
<td>Purchase inquiry</td>
<td>Allows the cardholder to check the status of authorization, capture, or credit processing of an order after the purchase response has been received.</td>
</tr>
<tr>
<td>Authorization reversal</td>
<td>Allows a merchant to reverse an authorization entirely or in part.</td>
</tr>
<tr>
<td>Capture reversal</td>
<td>Allows a merchant to correct errors in previous capture requests, such as those caused by human error.</td>
</tr>
<tr>
<td>Credit</td>
<td>Allows a merchant to issue credit to a cardholder's account for various reasons (such as for returned or damaged goods).</td>
</tr>
<tr>
<td>Credit reversal</td>
<td>Allows a merchant to correct errors in a previous credit request.</td>
</tr>
<tr>
<td>Payment gateway certificate request</td>
<td>Allows a merchant to request a current copy of the payment gateway's certificates.</td>
</tr>
<tr>
<td>Batch administration</td>
<td>Message between merchant and payment gateway regarding merchant batches.</td>
</tr>
<tr>
<td>Error message</td>
<td>Indicates that a responder rejects a message because it fails tests of format or content verification.</td>
</tr>
</tbody>
</table>
gateway’s key-exchange keys. The SET order process begins when the cardholder software (software that runs with your browser) requests a copy of the gateway’s certificate. The message from the cardholder indicates which payment card brand will be used for the transaction.

2. **Initiate response.** When the merchant receives an initiate request message, a unique transaction identifier is assigned to the message. The merchant then generates an initiate response message containing its certificates and that of the payment gateway. This information is then digitally signed with the merchant’s private key and transmitted to the cardholder.

3. **Purchase request.** Upon receipt of the initiate response message, the cardholder software verifies the certificates of both the merchant and gateway. Next, the cardholder software creates a dual signature using the OI and PI. Finally, the cardholder software generates a purchase request message containing a dual-signed OI and a dual-signed PI that is digitally enveloped to the payment gateway. The entire purchase request is then sent to the merchant.

4. **Purchase response.** When the merchant software receives the purchase request message, it verifies the cardholder’s certificate contained within the message, as well as the dual-signed OI. The merchant software then begins processing the OI and attempts to gain authorization from the payment gateway by forwarding the PI. Finally, the merchant generates a purchase response message, which states that the merchant received the cardholder’s request.

Upon receipt of the purchase response from the merchant, the cardholder software verifies the merchant certificate as well as the digital signature of the message contents. At this point, the cardholder software takes some action based on the message, such as displaying a message to the cardholder or updating a database with the status of the order.

**Payment Authorization**

During the processing of an order from a cardholder, the merchant attempts to authorize the transaction by initiating a two-way message exchange between the merchant and the payment gateway. First, an authorization request is sent from the merchant to the payment gateway; then an authorization response is received from the merchant by the payment gateway. The request and response are described as follows:
1. **Authorization request.** The merchant software generates and digitally signs an authorization request, which includes the amount to be authorized, the transaction identifier from the OI, and other information about the transaction. This information is then digitally enveloped using the payment gateway's public key. The authorization request and the cardholder PI (which is still digitally enveloped to the payment gateway) are transmitted to the payment gateway.

2. **Authorization response.** When the authorization request is received, the payment gateway decrypts and verifies the contents of the message (that is, certificates and PI). If everything is valid, the payment gateway generates an authorization response message, which is then digitally enveloped with the merchant's public key and transmitted back to the merchant.

Upon receipt of the authorization response message from the payment gateway, the merchant decrypts the digital envelope and verifies the data within. If the purchase is authorized, the merchant then completes processing of the cardholder's order by shipping the goods or performing the services indicated in the order.

**Payment Capture**

When the order-processing portion is completed with the cardholder, the merchant then requests payment from the payment gateway. Payment capture is accomplished by the exchange of two messages: the capture request and the capture response. This process is described as follows:

1. **Capture request.** The merchant software generates the capture request, which includes the final amount of the transaction, the transaction identifier, and other information about the transaction. This message is then digitally enveloped using the payment gateway's public key and transmitted to the payment gateway.

2. **Capture response.** The capture response is generated after the capture request is received and its contents verified. The capture response includes information pertaining to the payment for the transaction requested. This response is then digitally enveloped using the merchant's public key and is transmitted back to the merchant.
Upon receipt of the capture response from the payment gateway, the merchant software decrypts the digital envelope, verifying the signature and message data.

NOTE:
The merchant software stores the capture response and uses it for reconciliation with payment received from the acquirer.

Summary

Security protocols located at the application layer work slightly differently from those that operate on the IP (network) and TCP (transport) layers. Whereas IPSec (see Chapter 7) is used to provide security for all data being transferred across an IP network, S/MIME and SET are used solely to provide security for certain applications.

In 1995, a consortium of application and security vendors, led by RSA Data Security, Inc., designed the S/MIME protocol. Since then, the IETF S/MIME working group has taken control of S/MIME to continue its growth. S/MIME provides security not only for e-mail but also for any data that is transferred via the MIME protocol. Since its creation, S/MIME has continued to grow and improve its security services, adding support for mailing lists, signing receipts, and security labels.

SET is an open specification that provides a framework for protecting payment cards that are used in e-commerce transactions. Initiated by Visa and MasterCard in 1996, SET was completed in 1997, with the help of various other application developers and security vendors. The specification is described in three books totaling more than 900 pages.

Note that this chapter and Chapter 7 discuss only four selected protocols. Numerous others are available today, each of them supporting a specific security task.
Real-World Examples

Both S/MIME and SET have been incorporated in various applications. For secure e-mail, many companies and individuals have chosen to use S/MIME instead of a proprietary system such as PGP. In fact, many users have S/MIME-enabled mailers that they have not taken advantage of. S/MIME is incorporated in Microsoft’s Outlook and Outlook Express e-mail applications as well as Netscape’s Messenger software.

SET has also gained widespread use. Many of the vendors that visitors shop with daily across the Internet are SET-enabled. Currently, the merchants worldwide who use SET number in the hundreds. SET products are available not only for consumers but also for merchants, payment gateways, and SET certificate authorities. For a list of current SET-enabled products as well as the merchants that use them, visit http://www.setco.org/.

For both of these protocols, many security vendors also provide cryptographic APIs (application programming interfaces, or toolkits), which developers can use to produce secured applications. RSA Security, Inc., is one such company.