CHAPTER 6

MEDIA PROTECTION MECHANISMS

Any multimedia application—such as video, voice, or gaming—uses a distinct set of protocols to set up sessions between end points (for example, SIP, H.323) and a distinct protocol to transmit the media streams. The standard protocol used to exchange media streams is RTP[1] (Real Time Protocol), which is defined in RFC 3550. As discussed in Chapter 3, "Threats and Attacks," RTP streams can be intercepted and manipulated in order to perform various attacks. Although IPSec can be used to protect RTP, its limitations require a more scalable and versatile solution that alleviates the NAT traversal issue, dynamic allocation of sessions,[2] and the need for a PKI. This has led to the development of SRTP[3] (Secure Real Time Protocol). The use of SRTP requires a mechanism to exchange cryptographic keys before sending any media. Therefore, key management protocols such as MIKEY and SDescriptions[4] have been proposed to provide the necessary keying material and management mechanisms to maintain the security of multimedia sessions. Currently, there is not a single key-exchange mechanism considered to be the industry standard because each has strengths and weaknesses. The most logical approach: to combine SRTP with the appropriate key-exchange mechanism is to identify the requirements that need to be supported by the environment and evaluate the applicability of each of the existing key management mechanisms. Alternatives to using SRTP include DTLS (Datagram Transport Layer Security) and IPSec, which were discussed in Chapter 5, "Signaling Protection Mechanisms." The following sections describe SRTP and discuss its strengths and limitations.

1. *H. Schulzrinne, et al. "RTP: A Transport Protocol for Real-Time Applications," IETF RFC 3550, July 2003.*
2. *P. Thermos, T. Bowen, J. Haluska, and Steve Ungar. Using IPSec and Intrusion Detection to protect SIP implanted IP telephony. IEEE GlobeCom, 2004.*
3. *M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. "The Secure Real-time Transport Protocol (SRTP)," IETF RFC 3711, March 2004.*
4. *F. Andreasen, M. Baugher, and D. Wing. Session Description Protocol Security Descriptions for Media Streams, IETF draft draft-ietf-mmusic-sdescriptions-12.txt, 2005.*

SRTP

The Secure Real Time Protocol (SRTP) is a profile for the Real Time Protocol (RTP, IETF RFC 3550) to provide confidentiality, integrity, and authentication to media streams and is defined in the IETF RFC 3711. Although there are several signaling protocols (for example, SIP, H.323, Skinny) and several key-exchange mechanisms (for example, MIKEY, SDESCRIPTIONS, ZRTP), SRTP is considered one of the standard mechanism for protecting real-time media (voice and video) in multimedia applications. In addition to protecting the RTP packets, it provides protection for the RTCP (Real-time Transport Control Protocol) messages. RTCP is used primarily to provide QoS feedback (for example, round-trip delay, jitter, bytes and packets sent) to the participating end points of a session. The RTCP messages are transmitted separately from the RTP messages, and separate ports are used for each of the protocols. Therefore, both RTP and RTCP need to be protected during a multimedia session. If RTCP is left unprotected, an attacker can manipulate the RTCP messages between participants and cause service disruption or perform traffic analysis.

The designers of SRTP focused on developing a protocol that can provide adequate protection for media streams but also maintain key properties to support wired and wireless networks in which bandwidth or underlying transport limitations may exist. Some of the highlighted properties are as follows:

- The ability to incorporate new cryptographic transforms.
- Maintain low bandwidth and computational cost.
- Conservative in the size of implementation code. This is useful for devices with limited memory (for example, cell phones).
- Underlying transport independence, including network and physical layers that may be used, and perhaps prone to reordering and packet loss.

These properties make the implementation of SRTP feasible even for mobile devices that have limited memory and processing capabilities. Similar design properties are found in MIKEY (Multimedia Internet KEYing). Therefore, the use of MIKEY for key exchange and SRTP for media protection is one combination of mechanisms to provide adequate security for Internet multimedia applications, including VoIP, video, and conferencing.

The application that implements SRTP has to convert RTP packets to SRTP packets before sending them across the network. The same process is used in reverse to decrypt SRTP packets and convert them to RTP packets. Figure 6.1 depicts this process.

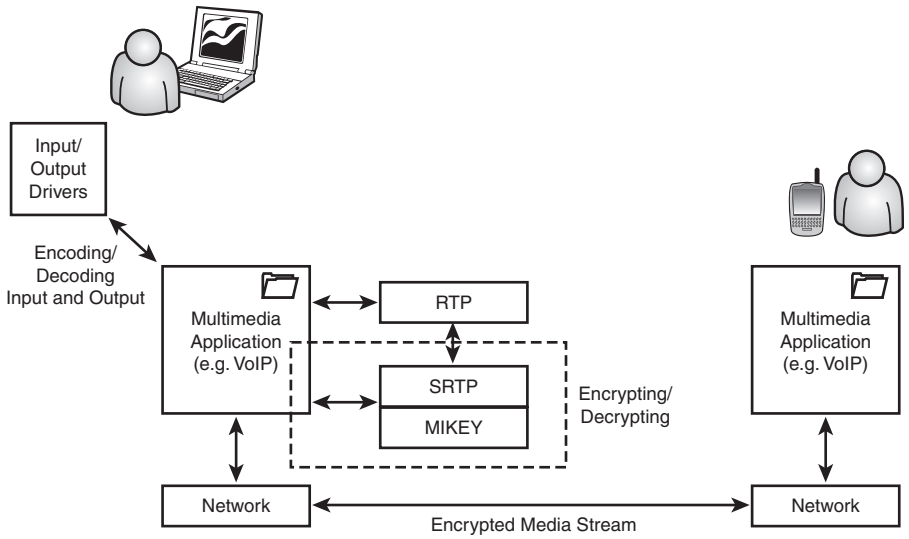

FIGURE 6.1 SRTP encoding/decoding.

After the application captures the input from a device (for example, microphone or camera), it encodes the signal using the negotiated or default encoding standard (for example, G.711, G.729, H.261, H.264) and creates the payload of the RTP packet. Next, the RTP payload is encrypted using the negotiated encryption algorithm. The default encryption algorithm for SRTP is AES (Advanced Encryption Standard) in *counter mode* using a 128-bit key length. This mode, along with the *null* mode,[5] is mandatory for implementations to be considered compliant with the IETF RFC (see RFC 3711 for additional requirements) and interoperate with other implementations. SRTP also recommends the use of AES in f8 mode to encrypt UMTS (Universal Mobile Telecommunications System) data. This mode also uses the same size for the session key and the salt as in counter mode. The use of AES in SRTP allows processing the packets even if they are received out of order, which is a desirable feature for real-time applications.

5. *The NULL mode can be used in cases where confidentiality is not desired.*

In addition to providing data encryption, the SRTP standard supports message authentication and integrity of the RTP packet. The default message authentication algorithm is SHA-1 using a 160-bit key length. The message authentication code (MAC) is produced by computing a hash of the entire RTP message, including the RTP headers and encrypted payload, and placing the resulting value in the *Authentication tag* header, as shown in Figure 6.2.

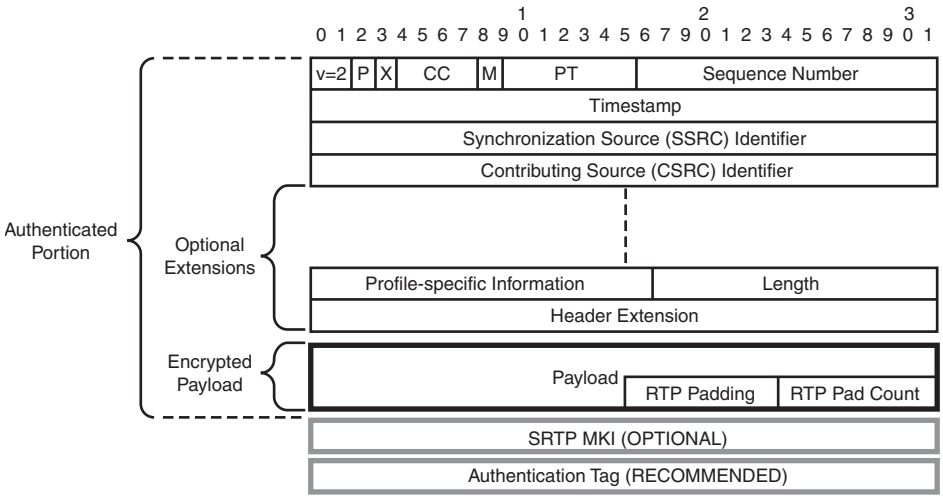

FIGURE 6.2 Format of the SRTP packet.

You might note that the SRTP message resembles the format of an RTP message with the exception of two additional headers: the MKI and the Authentication tag. The *MKI* (Master Key Identifier) is used by the key management mechanism (for example, MIKEY), and its presence is *optional* in implementations according to the SRTP standard (RFC 3711). The MKI can be used for rekeying or to identify the master key from which the session keys were derived to be used by the application to decrypt or verify the authenticity of the associated SRTP payload. The key-exchange mechanism generates and manages the value of this field throughout the lifetime of the session. The use of the Authentication tag header is important and provides protection against message-replay attacks.[6] In VoIP deployments, it

6. *J. Bilien, et al. Secure VoIP: Call Establishment and Media Protection. Royal Institute of Technology (KTH). Stockholm, Sweden, 2004.*

is recommended that message authentication be used at a minimum if encryption is not an option. Use of both is the optimal approach.

Note that the message headers are purposefully not encrypted (for example, sequence number, SSRC) to support header compression and interoperate with applications or intermediate network elements that might not be required to support SRTP but need to process the RTP headers (for example, billing). This limitation allows an attacker to perform traffic analysis by collecting information from the RTP headers and extensions, along with information from underlying transports (for example, IP, UDP). One area of interest is the future protocol extensions that will be developed for RTP and the sensitivity of the information that these extensions will carry.

Figure 6.3 shows an example of an application using SDescriptions (Security Descriptions) to transmit a cryptographic key for use with SRTP. The key is transmitted within the SDP portion of a SIP message. The SDP media attribute *crypto* defines the type of algorithm, the encryption mode, and the key length (AES_CM_128), along with the message digest algorithm and its length (SHA1_32).

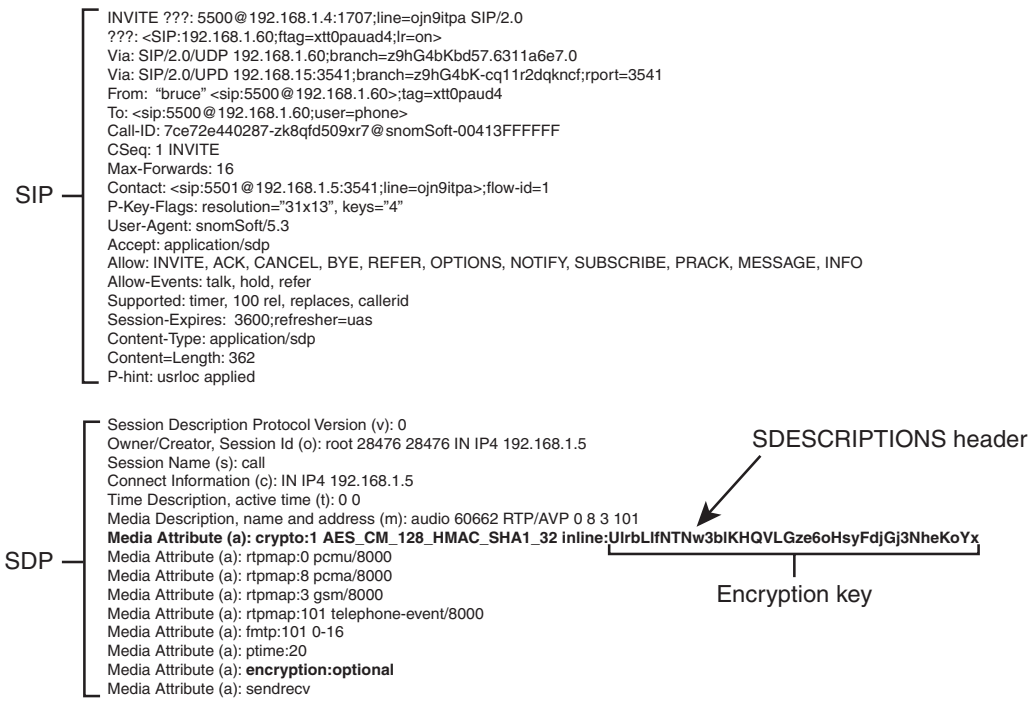

FIGURE 6.3 Key negotiation using SDescriptions in SIP.

The "inline" method indicates that the actual keying material is captured in the key-info field of the header. The syntax of the header is defined as follows:

```
a=crypto:<tag> <crypto-suite> <key-params> [<session-params>]
```

<crypto-suite> identifies the encryption and authentication algorithms (in this case, AES in counter mode using a 128-bit key length and SHA-1).

The next attribute is <key-params>, where

```
key-params = <key-method> ":" <key-info>
```

In this case the <key-method> is inline

```
<key-info> = UlrbL1fNTNw3blKHQVLGze6oHsyFdjGj3NheKoYx
```

Another mechanism of exchanging cryptographic keys is through the use of MIKEY, as discussed in further detail in Chapter 7, "Key Management Mechanisms." Figure 6.4 shows a SIP INVITE that announces the use of MIKEY in the SDP portion of the message. The following message is a capture from communications that use the *minisip* implementation.[7]

The attribute header key-mgmt in the SDP indicates that MIKEY should be used to encrypt media during this session.

If the signaling message (in this case, SIP) is transmitted in the clear, the encryption key can be intercepted and the contents of the media streams can be decrypted by an adversary. Therefore, it is necessary that signaling messages that carry encryption keys are also encrypted using protection mechanisms discussed in Chapter 5. In this case, the SIP signaling was performed using UDP to exchange keying material. UDP does not offer any protection and thus the keying material are exposed to eavesdropping

After the keys have been negotiated, the application encrypts the RTP payload and sends the SRTP packets to the remote end. Figure 6.5 shows an example of the SRTP packet.

7. *Israel Abad Caballero. Secure Mobile VoIP. Master's thesis, Department of Microelectronics and Information Technology, Royal Institute of Technology, June 2003.*

```
Internet Protocol, Src: 192.168.1.35, Dst: 192.168.1.20                    IP

User ??? Protocol/ Src Port: 5060, Dst Port: 5060                          UDP

INVITE sip:bob@192.168.1.20 SIP/2.0                                        SIP
Route: <sip:192.168.1.20:5060;transport=UDP;lr>
From: <sip:slice@192.168.1.35>;tag=2029
To: <sip:bob@192.168.1.20>
Call-ID: 5872@192.1368.1.35
CSeq: 301 INVITE
Contact: <sip:alice@192.168.1.35:5060;transport=UDP>;expires=1000
Content-Type: application/sdp
Via: SIP/2.0/UDP 192.168.1.35:5060;branch=z9hG4bK19718
Content-Length: 3542

v: 0                                                                       SDP
o: - 3344 3344 IN IP4 192.168.1.35
s: Minisip Session
c: IN IP4 192.168.1.35
t: 0 0
a: key-mgmt:mikey AQQFgAATBcCAAAAAHK/AAAAAAAAAAAAAAAAAAAAAoAx9bH01P3ztk
                                        LAAAAJwABAQEBEAIBAQMBFAQBDgUBAAYBAAcBAQgBAQkBAAoBAQs
                                        BCgwBAAcQrp33V4S04/yprsxz2nytcQMCBpMwggaPMIIEd6ADAge
                                         CAgkA8+z1SAxBJE4wDQYJKoZIhvcNAQEFBQAwgYsxCzAJB
```

FIGURE 6.4 Use of MIKEY in SIP for key negotiation.

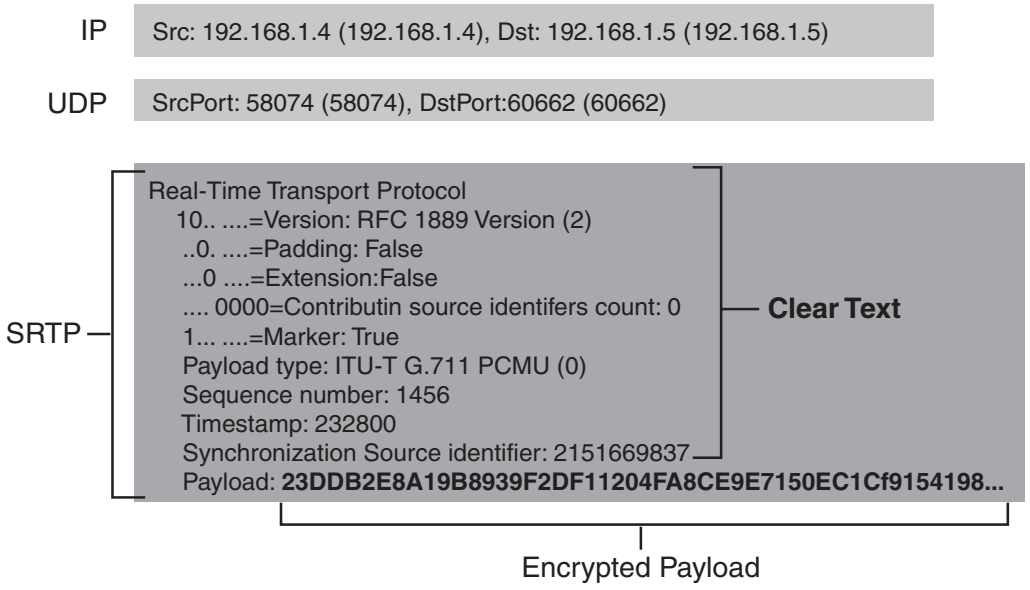

FIGURE 6.5 Contents of an SRTP packet.

All headers in the RTP packet are sent in the clear except for the payload, which is encrypted. Because SRTP uses AES by default, it provides protection against DoS attacks that aim to corrupt the encrypted media content. Typically, stream ciphers that rely on previous blocks to decrypt the next block (cipher block chaining) can be attacked by corrupting the data of one block and thus crippling the ability to successfully reassemble and produce the original content. AES does not suffer from this limitation because it can decrypt each block without requiring knowledge of previous blocks.

The use of authentication and integrity in SRTP messages is an important way to protect against attacks, including message replay and disruption of communications. For example, an attacker may modify the SRTP messages to corrupt the audio or video streams and thus cause service disruption. Another attack can be performed by sending bogus SRTP messages to a participant's device, thus forcing the device to attempt and decrypt the bogus messages. This attack forces the device application to impact the legitimate session by diverting resources to process the bogus messages. In cases where applications do not maintain session state, these attacks might not be as effective compared to stateful applications. Therefore, it is recommended that VoIP implementations use SRTP using SHA-1 with a 160-bit key length (and producing an 80-bit authentication tag) for message authentication and integrity to protect against such attacks. In some scenarios (for example, wireless communications) where bandwidth limitations impose restrictions, the use of a short authentication tag (for example, 32-bit length) or even zero length (no authentication) is an option.

Table 6.1 lists the parameters and corresponding values associated with key management in SRTP.

Table 6-1 SRTP Key Management

Parameter	Mandatory to Support	Default
SRTP/SRTCP cryptographic transforms	AES_CM, NULL	AES_CM, AES_F8 for UMTS
SRTP/SRTCP authentication transforms	HMAC_SHA1	HMAC_SHA1

Parameter	Mandatory to Support	Default
SRTP/SRTCP authentication parameters	80-bit authentication tag	80-bit authentication tag
Key derivation Pseudo Random Function	AES_CM	AES_CM
Session encryption key length	128 bit	128 bit
Session authentication key length	160 bit	160 bit
Session salt value length	112 bit	112 bit
Key derivation rate	0	0
SRTP packets max key-lifetime	2^{48}	2^{48}
SRTCP packets max key-lifetime	2^{31}	2^{31}
MKI indicator	0	0
MKI length	0	0

In addition, the following parameters are included in the crypto context for each session SSRC value: ROC (Roll Over Counter), SEQ (RTP sequence), SRTCP index, transport address, and port number.

Key Derivation

Although implementations may use a variety of key management mechanisms to manage keys, the SRTP standard requires that a native derivation algorithm be used to generate session keys. The use of the derivation algorithm is mandatory for the initial session keys.

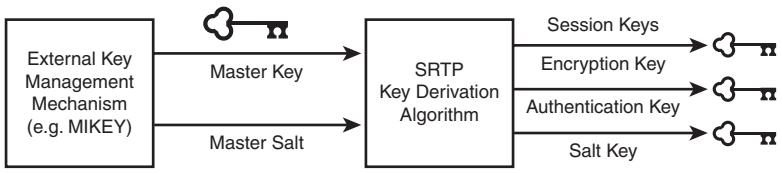

FIGURE 6.6 Key derivation algorithm.

The ability to derive keys through SRTP instead of using an external mechanism reduces additional computing cycles for key establishment. Typically, each session participant maintains a set of cryptographic information for each SRTP stream, which is referred to as the *cryptographic context.* For each cryptographic context, there are at least one encryption, one salt, and one authentication key for SRTP and SRTCPs respectively. Therefore, the SRTP key derivation algorithm can request only one master key and one salt value, when required, to derive the necessary session keys. Figure 6.6 shows this process. The derivation algorithm can be used repetitively to derive session keys. The frequency of session key generation is based on the value of the *key_derivation_rate*, which is predefined. This can be thought of as a key-refreshing mechanism that can be used to protect against cryptanalysis (which might otherwise be possible if a single master key is used). For example, an attacker can collect large amounts of session data and attempt to perform cryptanalysis. If the same key is used for the entire data, when that key is discovered all data can be recovered. If multiple keys are used, however, successful cryptanalysis will recover only data associated with the respective key (not the entire session). Therefore, multiple session keys can support perfect forward secrecy. Although frequent session key generation may be desirable and applicable for unicast sessions (for example, between small groups of two or four participants), it is not applicable for large multicast communications because each participant would have to maintain several hundred keys (which, in turn, deplete resources and impact processing and performance). One way to manage multiple SRTP and SRTCP keys is to refresh only the SRTP session keys on a specific interval and use only one key for SRTCP (for example, SRTCP key_derivation_rate = 0). Note that rekeying is necessary in cases where participants may join or leave during a group session (for example, conference calls). The determination of when such rekeying needs to occur is typically left up to the implementation, as long as there is a mechanism to alert all the participants to the expiration of the current key and the issuance of a new one. For example, the application might automatically trigger rekeying each time a participant joins the discussion or departs from the discussion. Either way, rekeying can be a costly computation depending on the number of participants and resource capabilities available on each participant's device.

Issues with Early Media

In some cases, media is transmitted to the remote ends before completing the signaling messages exchange and establishing a session. This is called early media, and it is a required condition in converged environments (for example, VoIP/PSTN). For example, when a VoIP subscriber calls a number that resides in a SS7 network (for example, PSTN), it might be necessary for the PSTN gateway to provide the signal progress by sending inband tones or announcements before the call is set up. This scenario introduces challenges as to how the media can be protected. Currently, there are discussions in IETF to use MIKEY with EKT (Encrypted Key Transport) to solve this issue.

SRTCP

Similar to SRTP, the format of the SRTCP packet has the authentication tag and MKI headers, but it also has two additional headers: SRTCP index and "encrypt-flag. Figure 6.7 shows the format of the SRTPC packet. The sensitive information that needs to be protected in an RTCP message includes the originating party of the report and the contents of the report. Therefore, these headers are encrypted.

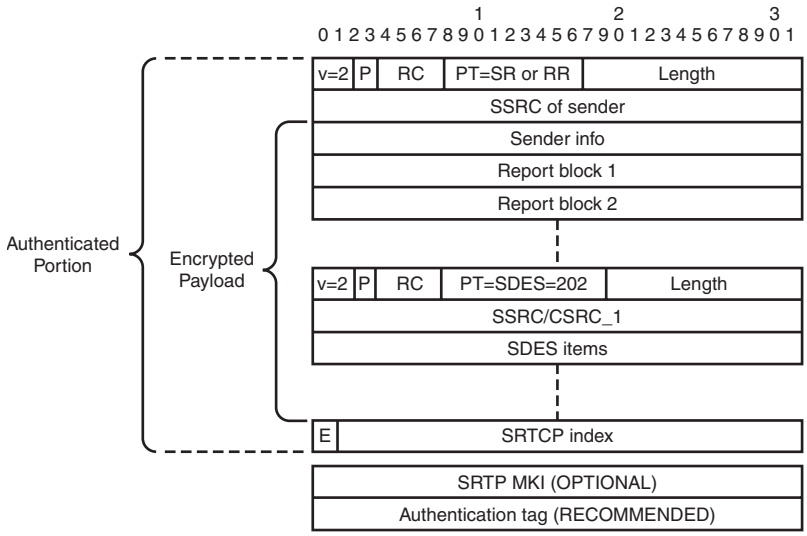

FIGURE 6.7 Format of an SRTCP packet.

The authentication tag, SRTCP index, and encrypt-flag headers are mandatory for SRTCP. For the most part, the processing of SRTCP packets is similar to SRTP packets, including the use of cryptographic algorithms and key lengths.

SRTP provides several properties to protect media streams in multimedia communications. The following list summarizes the strengths and limitations that should be considered when evaluating or implementing SRTP in a network to support multimedia communications.

SRTP strengths

- Provides confidentiality, integrity, and authentication of the message payload (media content).
- Provides protection against replay attacks for both RTP and RTCP.
- Support of AES allows for out-of-order packet reception and processing.
- Minimizes computation and resource consumption for generating cryptographic keys through an external key management mechanism by using a native key derivation algorithm.
- Key derivation algorithm helps protect against certain cryptanalytic attacks and provide perfect forward secrecy.

SRTP limitations

- Lack of RTP header encryption allows for traffic analysis by collecting information from the RTP headers and extensions.
- Cannot maintain end-to-end message integrity and authentication as the media stream is sent from an IP network to an SS7 network (PSTN).
- The key refresh and key management impact processing and resource consumption in large multicast groups. This is not desirable for mobile devices with limited computing resources.

Summary

Currently, SRTP is the standard protocol to provide protection of media streams. It supports authentication, confidentiality, and integrity of media messages to help protect against attacks such as eavesdropping, message replay, call hijacking, and various DoS attacks. One of the long-term challenges and an area for further research remains the key exchange and management in large multicast groups. At the moment, for a variety of reasons, SRTP is not considered a standard practice in VoIP implementations. One reason is the late adoption of SRTP by VoIP vendors in their products and the associated cost to have such functionality available. Another is negligence or lack of expertise to deploy SRTP in VoIP enterprise environments by corporations. Whatever the reason, it will take additional effort to educate users (and, perhaps, disclosure of security incidents [for example, eavesdropping, disruption]) to convince organizations to deploy SRTP as a standard practice.