[MS-SSRTP]:

Scale Secure Real-time Transport Protocol (SSRTP) Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial version
4/25/2008	0.2	Minor	Revised and edited technical content
6/27/2008	1.0	Major	Revised and edited technical content
8/15/2008	1.01	Minor	Revised and edited technical content
12/12/2008	2.0	Major	Revised and edited technical content
2/13/2009	2.01	Minor	Revised and edited technical content
3/13/2009	2.02	Minor	Revised and edited technical content
7/13/2009	2.03	Major	Revised and edited the technical content
8/28/2009	2.04	Editorial	Revised and edited the technical content
11/6/2009	2.05	Editorial	Revised and edited the technical content
2/19/2010	2.06	Editorial	Revised and edited the technical content
3/31/2010	2.07	Major	Updated and revised the technical content
4/30/2010	2.08	Editorial	Revised and edited the technical content
6/7/2010	2.09	Editorial	Revised and edited the technical content
6/29/2010	2.10	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.10	None	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	3.0	Major	Significantly changed the technical content.
11/15/2010	3.0	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	3.0	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	3.0	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	3.0	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	4.0	Major	Significantly changed the technical content.
4/11/2012	4.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	4.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	4.0.1	Editorial	Changed language and formatting in the technical content.
2/11/2013	4.1	Minor	Clarified the meaning of the technical content.
7/30/2013	4.1	None	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
11/18/2013	4.1	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	4.1	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	4.2	Minor	Clarified the meaning of the technical content.
7/31/2014	4.2	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	4.2	None	No changes to the meaning, language, or formatting of the technical content.
3/30/2015	5.0	Major	Significantly changed the technical content.
9/4/2015	5.0	None	No changes to the meaning, language, or formatting of the technical content.
7/15/2016	5.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	5.0	None	No changes to the meaning, language, or formatting of the technical content.
4/27/2018	6.0	Major	Significantly changed the technical content.
7/24/2018	7.0	Major	Significantly changed the technical content.
8/28/2018	8.0	Major	Significantly changed the technical content.
8/17/2021	9.0	Major	Significantly changed the technical content.
8/20/2024	10.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Scale Secure Real-time Transport Protocol (SSRTP) Extensions protocol specifies a proprietary extension to the Secure Real-Time Transport Protocol (SRTP) Extensions protocol, as described in [MS-SRTP].

This protocol provides the same functional capabilities as [MS-SRTP], which includes providing confidentiality, message authentication, and replay protection to the RTP traffic and to the control traffic for RTP. However, this protocol has one key additional motivation – to improve performance in scenarios where the same RTP payload is distributed to hundreds of recipients. To achieve this performance improvement, this protocol defines a new cryptographic transform that differs from [MS-SRTP] in packet format, encryption parameters, and message authentication processing.

This protocol and [MS-SRTP] share common components and constraints. Unless explicitly specified in this document, this protocol by default uses the parameters and algorithms as described in [MS-SRTP].

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

Advanced Encryption Standard (AES): A block cipher that supersedes the Data Encryption Standard (DES). AES can be used to protect electronic data. The AES algorithm can be used to encrypt (encipher) and decrypt (decipher) information. Encryption converts data to an unintelligible form called ciphertext; decrypting the ciphertext converts the data back into its original form, called plaintext. AES is used in symmetric-key cryptography, meaning that the same key is used for the encryption and decryption operations. It is also a block cipher, meaning that it operates on fixed-size blocks of plaintext and ciphertext, and requires the size of the plaintext as well as the ciphertext to be an exact multiple of this block size. AES is also known as the Rijndael symmetric encryption algorithm [FIPS197].

AES Counter Mode: A type of counter-mode encryption that generates encryption key streams by using **Advanced Encryption Standard (AES)** cipher and successive integers.

cryptographic context: A set of cryptographic state information that is maintained in a Secure Real-Time Transport Protocol (SRTP) stream.

dual-tone multi-frequency (DTMF): In telephony systems, a signaling system in which each digit is associated with two specific frequencies. This system typically is associated with touch-tone keypads for telephones.

encryption sequence number (ESN): An explicit sequence number that is embedded in each Scale Secure Real-Time Transport Protocol (SSRTP) packet by SSRTP.

Hash-based Message Authentication Code (HMAC): A mechanism for message authentication using cryptographic hash functions. HMAC can be used with any iterative cryptographic hash function (for example, MD5 and **SHA-1**) in combination with a secret shared key. The cryptographic strength of HMAC depends on the properties of the underlying hash function.

master key: A key that provides information for packet encryption and authentication in **Secure Real-Time Transport Protocol (SRTP)** and **Scale Secure Real-Time Transport Protocol (SSRTP)** transactions.

Real-Time Transport Protocol (RTP): A network transport protocol that provides end-to-end transport functions that are suitable for applications that transmit real-time data, such as audio and video, as described in [RFC3550].

RTCP packet: A control packet consisting of a fixed header part similar to that of **RTP packets**, followed by structured elements that vary depending upon the RTCP packet type. Typically, multiple RTCP packets are sent together as a compound RTCP packet in a single packet of the underlying protocol; this is enabled by the length field in the fixed header of each RTCP packet. See [RFC3550] section 3.

RTP packet: A data packet consisting of the fixed RTP header, a possibly empty list of contributing sources, and the payload data. Some underlying protocols could require an encapsulation of the RTP packet to be defined. Typically one packet of the underlying protocol contains a single RTP packet, but several RTP packets can be contained if permitted by the encapsulation method. See [RFC3550] section 3.

RTP payload: The data transported by **RTP** in a packet, for example audio samples or compressed video data. For more information, see [RFC3550] section 3.

RTP profile: A collection that contains payload type codes and mappings to payload formats, such as media encodings. It can also define extensions or modifications to the **Real-Time Transport Protocol (RTP)** that are specific to a particular class of applications. Typically, an application operates under only one profile.

salt: An additional random quantity, specified as input to an encryption function that is used to increase the strength of the encryption.

Scale Secure Real-Time Transport Protocol (SSRTP): A Microsoft proprietary extension to the **Secure Real-Time Transport Protocol (SRTP)**, as described in [RFC3711].

Secure Real-Time Transport Protocol (SRTP): A profile of **Real-Time Transport Protocol (RTP)** that provides encryption, message authentication, and replay protection to the RTP data, as described in [RFC3711].

session: A collection of multimedia senders and receivers and the data streams that flow between them. A multimedia conference is an example of a multimedia session.

Session Description Protocol (SDP): A protocol that is used for session announcement, session invitation, and other forms of multimedia session initiation. For more information see [MS-SDP] and [RFC3264].

session key: A symmetric key that is derived from a master key and is used to encrypt or authenticate a specific media stream by using the **Secure Real-Time Transport Protocol (SRTP)** and **Scale Secure Real-Time Transport Protocol (SSRTP)**.

SHA-1: An algorithm that generates a 160-bit hash value from an arbitrary amount of input data, as described in [RFC3174]. SHA-1 is used with the Digital Signature Algorithm (DSA) in the Digital Signature Standard (DSS), in addition to other algorithms and standards.

SSRTP stream: A sequence of **Scale Secure Real-Time Transport Protocol (SSRTP)** packets from a sender and to a receiver who are identified by the same Synchronization Source (SSRC).

synchronization source (SSRC): The source of a stream of **RTP packets**, identified by a 32-bit numeric SSRC identifier carried in the RTP header so as not to be dependent upon the network address. All packets from a synchronization source form part of the same timing and sequence number space, so a receiver groups packets by synchronization source for playback. Examples of synchronization sources include the sender of a stream of packets derived from a signal source such as a microphone or a camera, or an RTP mixer. A synchronization source could change its data format (for example, audio encoding) over time. The SSRC identifier is a randomly chosen value meant to be globally unique within a particular RTP session. A participant need not use the same SSRC identifier for all the RTP sessions in a multimedia session; the binding of the SSRC identifiers is provided through RTCP. If a participant generates multiple streams in one RTP session, for example from separate video cameras, each has to be identified as a different SSRC. See [RFC3550] section 3.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[MS-RTP] Microsoft Corporation, "Real-time Transport Protocol (RTP) Extensions".

[MS-SRTP] Microsoft Corporation, "Secure Real-time Transport Protocol (SRTP) Profile".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, https://www.rfc-editor.org/info/rfc2119

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003, https://www.rfc-editor.org/info/rfc3550

[RFC3711] Baugher, M., McGrew, D., Naslund, M., et al., "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004, http://www.rfc-editor.org/rfc/rfc3711.txt

1.2.2 Informative References

[MS-DTMF] Microsoft Corporation, "RTP Payload for DTMF Digits, Telephony Tones, and Telephony Signals Extensions".

[MS-SDPEXT] Microsoft Corporation, "Session Description Protocol (SDP) Version 2.0 Extensions".

1.3 Overview

This protocol is a proprietary extension to the **Secure Real-Time Transport Protocol (SRTP)** Extensions protocol, as described in [MS-SRTP]. The new **Scale Secure Real-Time Transport Protocol (SSRTP)** cryptographic transform specified by this protocol extends [MS-SRTP] in the following areas:

- Packet format. See section 2.2. This protocol introduces a new **encryption sequence number (ESN)** field in the SRTP packet.
- **Advanced Encryption Standard (AES)** Counter Mode encryption algorithm. See section 3.1.3.3.1. This protocol generates Initialization Vector (IV) for encryption based on encryption sequence number (ESN) instead of a **Real-Time Transport Protocol (RTP)** sequence number.
- Packet processing and padding in message authentication. See sections 3.1.3.3.2 and 3.1.5.1.3. This protocol rearranges the fields in SSRTP packets and authenticates them in an order different from the on-wire order. It pads buffers to a multiple of 64-bytes for message authentication. (**Note**: not on the wire).

The details of these extensions are specified in sections 2 and 3.

This protocol reuses many components of [MS-SRTP]. These components include:

- Key derivation algorithms and parameters.
- The **master key**, **session key**, and **salt** format and sizes.
- Encryption and authentication primitives.
- RTCP encryption and authentication.

Unless explicitly noted, this protocol uses the same parameters and algorithms as described in [MS-SRTP].

1.4 Relationship to Other Protocols

This protocol is a proprietary extension to [RFC3711]. It shares common components with another SRTP extension described in [MS-SRTP].

This protocol relies on the Session Description Protocol (SDP) Version 2.0 Extensions as described in [MS-SDPEXT] to exchange **master key**s and key parameters.

This protocol encrypts and authenticates **RTP packet**s. It works with other **RTP profile**s, for example RTP Payload for DTMF Digits, Telephony Tones, and Telephony Signals Extensions as described in [MS-DTMF]. It encrypts and authenticates after these profiles on the sending side and authenticates and decrypts before passing RTP and **RTCP packet**s to other profiles on the receiving side.

Secure Real-Time Transport Control Protocol (SRTCP) is considered to be a sub-protocol to SRTP and they are specified together in [RFC3711]. This protocol uses SRTCP to protect RTCP packets.

1.5 Prerequisites/Preconditions

This protocol has the following prerequisites:

- This protocol requires that encryption and authentication algorithms are negotiated through SDP Version 2.0 Extensions as described in [MS-SDPEXT].
- This protocol requires that **master key**s are exchanged through SDP Version 2.0 Extensions and that the keys are configured properly.
- This protocol only provides message confidentiality, authentication, and replay protection for **RTP packet**s and **RTCP packet**s.

1.6 Applicability Statement

This protocol is used in an environment where users require secure RTP traffic and the same **RTP payload** is distributed to multiple receivers. This protocol is required to be used with SDP Version 2.0 Extensions as described in [MS-SDPEXT] to set up the shared **master key** securely.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

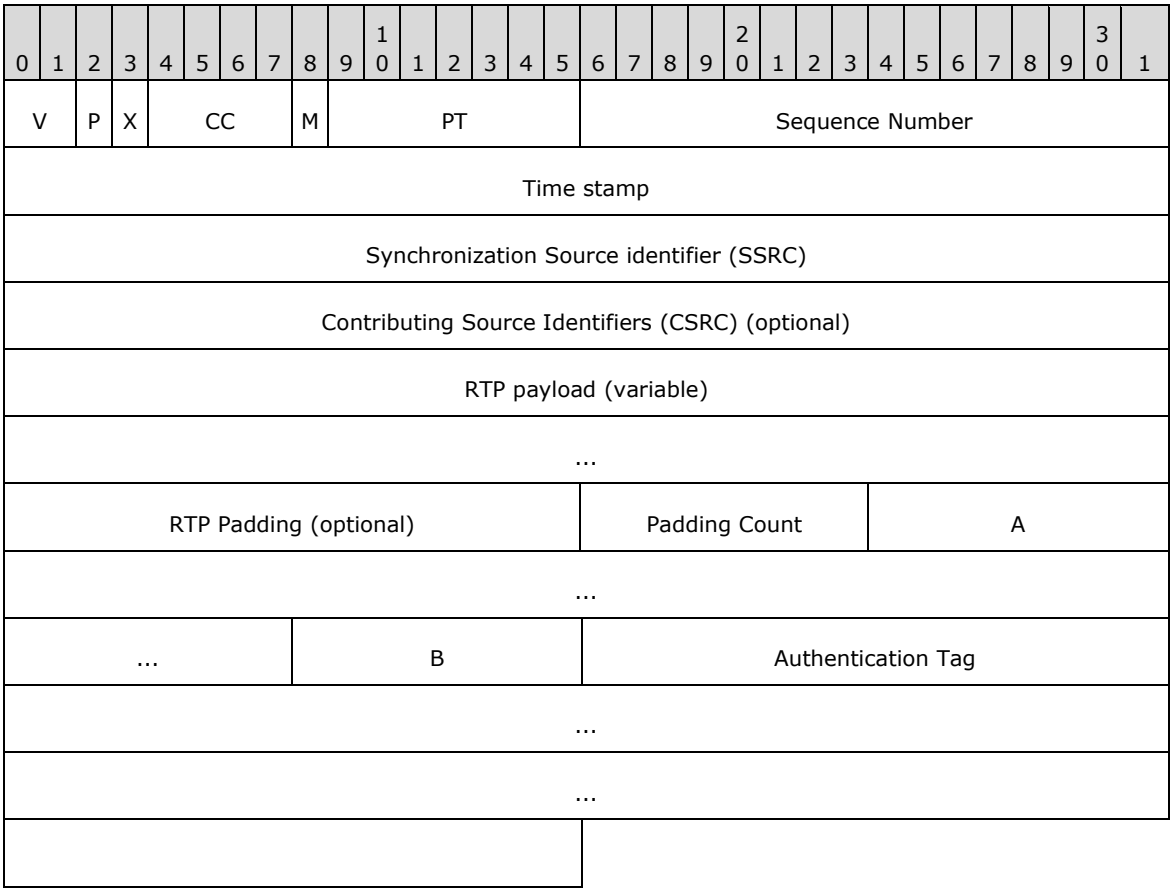
2.1 Transport

The **Scale Secure Real-Time Transport Protocol (SSRTP)** transforms **RTP**/**RTCP packets** only. Refer to [MS-RTP] for transports that RTP uses.

2.2 Message Syntax

2.2.1 Scale Secure RTP Message Syntax

The following bit table shows the packet syntax on the wire for this protocol.



V, P, X, CC, M, PT, Sequence Number, Time stamp, SSRC, CSRC, RTP payload, RTP Padding, Padding Count: Standard **RTP packet** fields. For details, see [RFC3550] section 5.1.

A - Encryption Sequence Number (6 bytes): A 48-bit unsigned integer in network order. Alignment is not needed. This is the explicit sequence number SSRTP uses in encryption and authentication. The **encryption sequence number (ESN)** MUST continuously grow and is not required to be contiguous. Note that the ESN is only used in RTP packets.

B - MKI (1 byte): An unsigned SRTP **master key** identifier in network order. Alignment is not needed. For details, see [RFC3711] section 3.1. **MKI** MUST be used and the **MKI** length MUST be 1 byte.

Authentication Tag (10 bytes): An array of unsigned chars where any element can have a value in the range of 0 to 0xff. The authentication tag size MUST be 10 bytes.

2.2.1.1 Encrypted Portion

This protocol concatenates **RTP payload** fields, RTP padding fields, and the padding count field and then encrypts them. The encryption algorithm is different from SRTP Extensions as specified in [MS-SRTP], and is specified in section 3.1.3.3.1.

2.2.1.2 Authenticated Portion

This protocol authentication rearranges the packet fields for this protocol as shown in the following bit table. The rearranged fields are concatenated to a virtual packet, and the authentication tag is calculated on it. Note that this rearrangement is only done to calculate the authentication tag and does not show up on the wire.

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>20</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>30</td><td>1</td></tr>
<tr><td colspan="32">Contributing Source Identifiers (CSRC) (optional)</td></tr>
<tr><td colspan="32">RTP payload (variable)</td></tr>
<tr><td colspan="32">...</td></tr>
<tr><td colspan="24">RTP Padding (optional)</td><td colspan="8">Padding Count (optional)</td></tr>
<tr><td colspan="32">Encryption Sequence Number</td></tr>
<tr><td colspan="24">...</td><td colspan="8">Padding (variable)</td></tr>
<tr><td colspan="32">...</td></tr>
<tr><td>V</td><td>P</td><td>X</td><td colspan="4">CC</td><td>M</td><td colspan="7">PT</td><td colspan="16">Sequence Number</td></tr>
<tr><td colspan="32">Time stamp</td></tr>
<tr><td colspan="32">Synchronization Source identifier (SSRC)</td></tr>
<tr><td colspan="32">Rollover Counter</td></tr>
</table>

A new field Rollover Counter is included in authentication. It is the same as in SRTP.

V, P, X, CC, M, PT, Sequence Number, Time stamp, SSRC, CSRC, RTP payload, RTP Padding, Padding Count: Standard **RTP packet** fields. For details, see [RFC3550] section 5.1.

Encryption Sequence Number (6 bytes): A 48-bit unsigned integer in network order. Alignment is not needed. This is the explicit sequence number SSRTP uses in encryption and authentication. The **encryption sequence number (ESN)** SHOULD continuously grow and is not required to be contiguous. The last 8-bit of ESN SHOULD NOT be 0. Note that the ESN is only used in RTP packets.

Padding (variable): Added after the ESN field and the preceding fields are zero-padded to the 64-byte boundary. The RTP header, excluding CSRC, is added after the padding, and then followed by the rollover counter. The whole authenticated portion cannot be a multiple of 64-bytes in size, but field **V** MUST start at the 64-byte boundary.

Rollover Counter (4 bytes): A 4-byte unsigned integer in network order. Alignment is not needed. The rollover counter records how many times the RTP sequence number has been reset to 0 after passing 65535.

2.2.2 Scale Secure RTCP Message Syntax

The Scale Secure **RTCP packet** syntax is the same as the SRTCP packet. See [RFC3711] section 3.4.

3 Protocol Details

3.1 Endpoint Details

This protocol can be used to secure any RTP traffic. It does not have any role-specific behavior, such as for protocol client or server roles. All behavior described here applies to both protocol client and server roles.

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

This protocol is an extension to SRTP as specified in [MS-SRTP]. It keeps all [MS-SRTP] states. Refer to [MS-SRTP] section 3.1.1 for these states.

In addition to states as specified in [MS-SRTP], this protocol keeps the last **encryption sequence number (ESN)** sent and received for each **SSRTP stream**.

3.1.2 Timers

None.

3.1.3 Initialization

This protocol keeps all the states, as specified in [MS-SRTP], and the initialization of these states is the same as in [MS-SRTP]. In addition, this protocol initializes the **encryption sequence number (ESN)** value.

3.1.3.1 Cryptographic Contexts

This protocol requires that each sender and receiver in a **session** maintain cryptographic contexts. A cryptographic context includes the **master key**, key parameters, and run time states.

This protocol maintains two cryptographic contexts per session: one for the send direction and one for the receive direction. This protocol supports multiple media streams sharing the same SSRTP session. Each media stream MUST be uniquely identified by one **Synchronization Source (SSRC)**. This protocol maintains per SSRC transform independent parameters in **cryptographic contexts**, as specified in section 3.1.3.2.

When sending or receiving an SSRTP packet, this protocol first uses the SSRTP session and direction to identify the cryptographic context, then uses the Synchronization Source (SSRC) in the packet to decide the per SSRC transform independent parameters in the cryptographic context.

3.1.3.2 SSRTP Parameter Settings

Where packets for this protocol inherit the state from [MS-SRTP], parameters settings MUST be the same.

For your convenience, see the following list of transform independent parameters. For details, see [MS-SRTP] section 3.1.3.2.

- The encryption algorithm MUST be **AES Counter Mode** and encryption MUST be used.
- The authentication algorithm MUST be **Hash-based Message Authentication Code (HMAC)-SHA-1** and authentication MUST be used.
- The replay list size MUST be 64 entries.
- The **master key** indicator MUST be used.
- The master key indicator length MUST be 1 byte.
- The key derivation rate MUST be 0.
- The master key length MUST be 128-bit.
- The master **salt** key length MUST be 112-bit.
- The encryption **session key** length MUST be 128-bit (AES_128).
- The encryption session salt length MUST be 112-bit.
- The authentication session key length MUST be 160-bit.
- The master key lifetime MUST be 2^{48}-1 packets for RTP and 2^{31}-1 for RTCP.
- SRTCP and SRTP MUST have the same parameter settings with the exceptions specified in [RFC3711] section 3.2.1.

This protocol maintains the following transform independent parameters per **Synchronization Source (SSRC)**.

- The rollover counter
- The highest received RTP sequence number
- The highest received ESN on this SSRC
- The replay list

For information about transform dependent parameters, see sections 3.1.3.3.1 and 3.1.3.3.2.

In addition, this protocol MUST initialize the **encryption sequence number (ESN)** to a random number in the range of 0 to 2^{47}-1. The highest bit of the ESN is recommended to be 0 to avoid ESN wraparound too early at the beginning of the stream. ESN is shared among all media streams.

Unless explicitly noted, this protocol follows [RFC3711] to set other mandatory parameters.

3.1.3.3 SSRTP Cryptographic Transform

This protocol defines a new cryptographic transform. The new transform is based on the default SRTP transform with variations specified in this section.

3.1.3.3.1 Message Encryption

The SRTP default encryption algorithms are specified in [RFC3711] section 4.1. The encryption algorithm for this protocol is based on these algorithms, with the difference in packet format and IV calculation.

This protocol requires that the encryption algorithm MUST be **AES Counter Mode**, with the following parameters. See [RFC3711] section 4.1 for details of the parameters.

- n_b (block cipher size) MUST be 128-bit (the AES algorithm's fixed cipher block size).
- n_e (encryption key size) MUST be 128-bit.
- The session **salt** key MUST be used and n_s MUST be 112-bit.
- SRTP_PREFIX_LENGTH MUST be 0.

This protocol requires that the packet MUST be in the format specified in section 2.2.1 and the encrypted fields MUST be arranged in the format specified in section 2.2.1.1.

IV calculation needs the run-time state. See section 3.1.5.1.2 for details.

3.1.3.3.2 Message Authentication and Integrity

The SRTP default authentication algorithm is **Hash-based Message Authentication Code (HMAC)-SHA-1**, specified in [RFC3711] section 4.2. The authentication algorithm in this protocol is the same, but the authenticated fields are different.

This protocol implements Hash-based Message Authentication Code (HMAC)-SHA-1 and requires the following parameters:

- *n_a* (authentication key size) MUST be 160-bit.
- *n_tag* (authentication tag size) MUST be 80-bit.

This protocol requires that the authenticated fields MUST be in the format specified in section 2.2.1.2.

3.1.3.4 Session Key Derivation

This protocol implements the **session key** derivation algorithm specified in [RFC3711] section 4.3. This protocol requires that the key derivation rate MUST be 0.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 SSRTP Packet Processing

3.1.5.1.1 Packet Index Determination and Replay Protection

The **RTP packet** index is used in this protocol for replay protection, as specified in [RFC3711] section 3.3.2. Note that this protocol requires the key derivation rate to be 0, so that the RTP packet index is not used in key derivation.

The **encryption sequence number (ESN)** MUST NOT be used in replay protection because the ESN is not required to be contiguous in one protocol stream. The last received ESN can be used to help estimate the RTP packet index.

3.1.5.1.2 SSRTP AES Counter Mode IV Generation

This protocol requires that the encryption mode MUST be **AES Counter Mode**. With the exception of IV generation, this protocol's AES Counter Mode algorithm is identical to standard SRTP AES Counter Mode as specified in [RFC3711] section 4.1.

This protocol defines IV as:

IV = (k_s * 2^16) XOR ((ESN >> 16) * 2^64) XOR (ESN * 2^16)

Where:

k_s: encryption **salt**, specified in [RFC3711] section 4.1, generated by the key derivation procedure in section 3.1.3.4.

ESN: the **encryption sequence number (ESN)** embedded in protocol packets.

For security reasons, this protocol requires that the ESN MUST be different for any two pieces of different **RTP payload** content protected by the same **master key**.

3.1.5.1.3 Sending and Receiving SSRTP Packet

This protocol requires that **RTP packet**s MUST be encrypted and authenticated. With some exceptions, the protocol implements steps similar to those SRTP uses, as specified in [RFC3711] section 3.3. The process is copied here for convenience and the exceptions are noted.

3.1.5.1.3.1 Sending an SSRTP Packet

1. Determine which cryptographic context to use, as described in section 3.1.3.1.
2. Determine the **encryption sequence number (ESN)** value as the last sent ESN incremented by 1. After the increment, if the last 8-bit of the ESN is 0 increment the ESN by 1 again. ESN is shared by all media streams using the same cryptographic context and it is not **Synchronization Source (SSRC)**-specific.
3. Determine the rollover counter per SSRC.
4. Determine the **master key** and master **salt**. This is done using the current MKI in the cryptographic context.
5. Determine the **session key**s and session salt as specified in [RFC3711] section 4.3, using the master key, master salt, key_derivation_rate, and session key-lengths in the cryptographic context with the ESN, determined in steps 2 and 3.
6. Encrypt the **RTP payload** to produce the encrypted portion of the packet (see section 2.2.1.1). This step uses the encryption algorithm specified in section 3.1.3.3.1, the session encryption key, and the session salt found in step 4, together with the ESN found in step 2.
7. Append the ESN to the packet.
8. Append the MKI to the packet.
9. For message authentication, compute the authentication tag for the authenticated portion of the packet, specified in section 2.2.1.2. This step uses the current rollover counter, the authentication algorithm indicated in the cryptographic context, and the session authentication key found in step 5. Append the authentication tag to the packet.
10. If the RTP sequence number wraps around, update the rollover counter.
11. Record the current ESN as the last sent packet's ESN in cryptographic context.

3.1.5.1.3.2 Receiving an SSRTP Packet

1. Determine which cryptographic context and per **Synchronization Source (SSRC)** transform independent parameters to use, as described in section 3.1.3.1.

2. Determine the current **encryption sequence number (ESN)** from the packet. Estimate the packet index and rollover counter using the last ESN received on this SSRC, the current ESN, the highest received RTP packet sequence number of the media stream and the previous rollover counter of the media stream.

3. Determine the **master key** and master **salt** using the MKI in the packet.

4. Determine the **session key**s and session salt, as defined in [RFC3711] section 4.3, using the master key, master salt, key_derivation_rate, and session key-lengths in the cryptographic context with the ESN, as determined in steps 2 and 3.

5. For message authentication and replay protection, first check whether the packet has been replayed, as specified in [RFC3711] section 3.3.2, using the Replay List and the index as determined in step 2. If the packet is judged to be replayed, the packet MUST be discarded and the event SHOULD be logged.

6. Perform verification of the authentication tag, using the rollover counter from step 2, the authentication algorithm indicated in the cryptographic context, and the session authentication key from step 4. If the error audit message is "AUTHENTICATION FAILURE," as specified in [RFC3711] section 4.2, the packet MUST be discarded from further processing and the event SHOULD be logged.

7. Decrypt the Encrypted Portion of the packet using the decryption algorithm specified in section 3.1.3.3.1, the session encryption key, and salt found in step 4, with the ESN from step 2.

8. Update the rollover counter and highest sequence number, s_l, in the cryptographic context for this media stream, as specified in [RFC3711] section 3.3.1, using the packet index estimated in step 2. If replay protection is provided, also update the Replay List, as specified in [RFC3711] section 3.3.2.

9. Update the last received ESN in cryptographic context.

10. Remove the ESN, the MKI, and authentication tag fields from the packet.

3.1.5.2 SSRTCP Packet Processing

This protocol processes **RTCP packet**s the same way as [MS-SRTP]. For details, see [MS-SRTP] section 3.1.5.2.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

The following annotations present examples of test vectors for this protocol. Binary data is described in format (length in bytes, HEX value) and only the **RTP packet** transform is included. The **RTCP packet** transform is identical to SRTCP, as described in [RFC3711]. An RTCP transform example is not provided.

4.1 Key Derivation

This section provides an example of test vector for key derivation.

Input:

Master key:

(16, CB4A3C93F3D587ABA1AB0BDF8C6AA0FB)

Master salt:

(14, 53EF4F4594296D0EB286D9CC96E4)

Derived keys:

SSRTP encryption key:

(16,C3FCC67BFBF17CFA2DC69F4B4CFC59CD)

SSRTP authentication key:

(20,23B8B2D911CF8C6416F4AAB94083E0CC32615694)

SSRTP session salt:

(14,929B3AD0FDB565FDBEAA50412C8D)

SSRTCP encryption key:

(16,122E3C94A0D945242AF0B79C6EDCE0BB)

SSRTCP authentication key:

(20,999BDAC078DBC12E7677AD05B9B2B54CBFDCBAA6)

SSRTCP session salt:

(14,839D270762975E43F6351493434E)

4.2 RTP Packet Transform

This section provides an example of test vector for encryption of a RTP packet. It shows a sample input RTP packet and the output of the **RTP packet** transform.

Input:

RTP header:

(12, 80728001AE773346DE1A3236)

Raw RTP payload:
(142,3F68B92587D38C18D22AFA3FCF30B63098BDB1213F30F91054911E0521EE3A8EE386794C5B5F

D4B9A6477719F27937B6A0C7E8221250A57C5A42E8A99565F7559F21998F2555003F4677DB4AFCD3
59738B51D538B4BE1780CC618E686E9862343F0C65D5A86C334B1915B48D99FCAD8E39E9C8F9BD6
915FD7CBBFFD94A73F373615C5CC8C827B2E4C33EEB492D38)

Encryption sequence number (ESN):

(6, 5E1A32368001)

Rollover counter:

(4, 00000002)

Output:

Encrypted portion (raw data for encryption):

(142,3F68B92587D38C18D22AFA3FCF30B63098BDB1213F30F91054911E0521EE3A8EE386794C5B5F
D4B9A6477719F27937B6A0C7E8221250A57C5A42E8A99565F7559F21998F2555003F4677DB4AFCD3
59738B51D538B4BE1780CC618E686E9862343F0C65D5A86C334B1915B48D99FCAD8E39E9C8F9BD6
915FD7CBBFFD94A73F373615C5CC8C827B2E4C33EEB492D38)

Encrypted payload:

(142,C1D49FFD5B845AAC755FCE604A2B9225D672DDB5A3C4664447F3D39D841B6C84373437FAED0
11C30AD1D91FB9CC7CF1796A97D99886EBB694E6C050ED100073D2526C9FC56AB08555B3A1A2589
D1491D0402EB79C1C1C6E439C815B4AB83421F57293008B70AB296DAFFD7E6E2E67E6A93FF89FE8C
DE14C49FBAB13E233793B1934AA8A5BDBC3BD6B0A91D520EC9)

Authenticated portion:

(208,C1D49FFD5B845AAC755FCE604A2B9225D672DDB5A3C4664447F3D39D841B6C84373437FAED0
11C30AD1D91FB9CC7CF1796A97D99886EBB694E6C050ED100073D2526C9FC56AB08555B3A1A2589
D1491D0402EB79C1C1C6E439C815B4AB83421F57293008B70AB296DAFFD7E6E2E67E6A93FF89FE8C
DE14C49FBAB13E233793B1934AA8A5BDBC3BD6B0A91D520EC95E1A32368001000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000080728001AE
773346DE1A323600000002)

Authentication tag:

(10,2FA5BAC13AC58423BE4A)

5 Security

5.1 Security Considerations for Implementers

- Master keys are randomly generated. The same master key is not used for the send and receive directions in the same SRTP session.
- Master key exchange is done through external mechanisms in **Session Description Protocol (SDP)**. SDP is transferred on a secure transport, such as TLS.
- The initial RTP sequence number is randomly generated. But it does not use a value close to 65535, because this could cause a rollover counter mismatch if there is packet loss at the beginning of **session** startup.
- SRTP does not terminate the connection when a replay attack is detected. Some **RTP profile**s intentionally send the same packet multiple times and the duplicated packets fail replay check. An example is **dual-tone multi-frequency (DTMF)**, as described in [MS-DTMF].

5.2 Index of Security Parameters

Security parameter	Section
Encryption algorithm	3.1.3.2
Authentication algorithm	3.1.3.2
Replay list size	3.1.3.2
Master key indicator length	3.1.3.2
Session key derivation rate	3.1.3.2
Master key length	3.1.3.2
Master **salt** length	3.1.3.2
Encryption session key length	3.1.3.2
Encryption session salt length	3.1.3.2
Authentication session key length	3.1.3.2
Master key lifetime	3.1.3.2
Encryption sequence number value	3.1.3.2
Advanced Encryption Standard (AES) cipher block size	3.1.3.3.1
SRTP cipher prefix size	3.1.3.3.1
Authentication tag size	3.1.3.3.2

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Office Communications Server 2007
- Microsoft Office Communications Server 2007 R2
- Microsoft Office Communicator 2007
- Microsoft Office Communicator 2007 R2
- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Lync Server 2013
- Microsoft Lync Client 2013/Skype for Business
- Microsoft Skype for Business 2016
- Microsoft Skype for Business Server 2015
- Microsoft Skype for Business 2019
- Microsoft Skype for Business Server 2019
- Microsoft Skype for Business 2021
- Microsoft Skype for Business LTSC 2024

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Description	Revision class
6 Appendix A: Product Behavior	Updated list of supported products.	Major

8 Index

T

V