Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 5, May 2013, pg.289 – 301

SURVEY ARTICLE

A Survey and Analysis of Media Keying Techniques in Session Initiation Protocol (SIP)

Rajesh Kumar[1], Sandip Chauhan[2]
[1]Kalol Institute of Technology affiliated to G.T.U., Ahmadabad, India
[2]Kalol Institute of Technology affiliated to G.T.U., Ahmadabad, India

Abstract— In this paper we are presenting a survey about the key exchange protocols namely ZRTP, DTLS-SRTP, SDES[4] and MSDES[15]. The SIP is used for signaling in VOIP which is completely text based application layer protocol hence there is a lot of security issues and challenges associated with VOIP. A detailed analysis of key exchange protocols and its suitability with SIP has been carried out in this paper.
In VOIP we require security at both signaling and data transfer levels and signaling is text based so confidentiality of signaling parameters and data is a big challenge in VOIP. Even exchanging cryptographic keys to encrypt the media stream in the SIP is difficult task. There is a need for stronger key management protocols which will secure the voice data from various types of attack and which also provides a feasible key exchange mechanism.

Key Terms: - SIP; key exchange; media; security; SDES; ZRTP; DTLS; MSDES; SRTP

I. INTRODUCTION

In VOIP security issues are associated at both session level and media transport level. For establishing session between two parties Session Initiation Protocols (SIP)[1] are involved. But SIP is not the only protocol responsible for session initiation, some another companion protocol such as the Session Description Protocol (SDP) [1][10]is also involved in session initiation. SIP is a text based application layer protocol so it is more prone to various types of attacks during session initiation. After establishing the session, the media transmission starts. For media transmission various protocols like UDP, RTP[2], RTCP[2], and SRTP are involved. But in VOIP media stream is also prone to various attacks. To ensure the security, VOIP has divided in four sections: *signaling*, *session description*, *key exchange* and *secure media (data) transport* as shown in the VOIP protocol stack.[12]

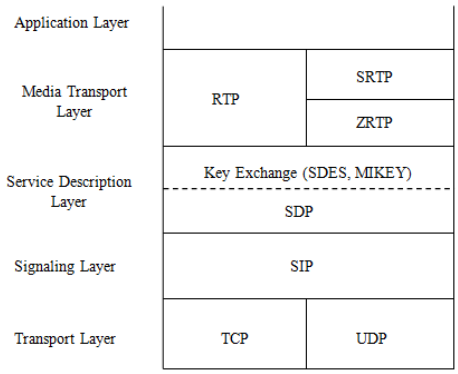

Figure 1: VOIP protocol stack

All these four layers are implemented by using different protocols. Signaling protocols include Session Initiation Protocol (SIP)[10], H.323[10] and MGCP[10]. SDP is the companion protocol for SIP used for describing the media for the session and also used for key exchange. One optional field of SDP is used for key exchange.

Key exchange protocols are used to provide a cryptographically secure way of establishing secret session keys between two or more participants in an untrusted environment. Security of the media (i.e. actual voice datagrams are transmitted) depends on the *secrecy* of session keys and *authentication* of session participants. Since the established key is typically used in a symmetric encryption scheme, key secrecy requires that nobody other than the legitimate session participants be able to distinguish it from a random bitstring. Authentication requires that, after the key exchange protocol successfully completes, the participants' respective views of sent and received messages must *match.* Key exchange protocol for VoIP sessions requires SDP's one optional field for each SDES, ZRTP[6], DTLS-SRTP and MSDES. We will analyze all four in this paper.

Transmission of media using SRTP provides confidentiality, message authentication, and integrity and replay protection to the media stream. Confidentiality means that the encrypted data is indistinguishable by anyone who does not have the key. Message authentication implies that if Alice as a User Agent server receives a datagram apparently sent by Bob as a User Agent Client, then it was indeed sent by Bob. Data integrity implies that any modification of the data during transmission will be detected by the recipient.

II. OUR CONTRIBUTION

VoIP[10] application consists of signaling and media transmission protocols. Signaling protocols are responsible for signaling and media transmission protocols are responsible transmission of media. Together signaling and media transmission protocols are implemented as VoIP stack at different layers.

We have analyzed security of VoIP protocols present at various layer of the VoIP stack. A protocol may be secure when executed in isolation, but the composition of protocols in different layers may be insecure. Let us consider that a call is made from VOIP network to any PSTN lines, if the hardware equipment is compromised then the caller's identity information will be easily available to attacker for future misuse, so call is not secure.

We will analyze how the transport-layer SRTP protocol repeat the key stream used for datagram encryption. We will also analyze how attackers obtain the XOR of plaintext datagrams or even to completely decrypt them. The SRTP [13] key stream is generated by using AES encryption algorithm in a stream cipher-like mode. The AES key is generated by applying a pseudo-random function (PRF) to the session key. SRTP, however, does not add any session-specific randomness to the PRF function. Instead, SRTP assumes that the key exchange protocol, executed as part of RTP [2] session establishment, will ensure that session keys never repeat.

S/MIME-protected SDES [4] key exchange protocols that may be executed prior to SRTP[7], does not provide any replay protection. This paper will analyze how an attacker can replay an old SDES key establishment message, which will cause SRTP to repeat the key stream that it used before, and can create devastating consequences. This attack can be found in analysis of the ***libsrtp*** implementation.

In the ZRTP [6] key exchange protocol the attacker can convince ZRTP session participants that they have lost their shared secret. ZID values, which are used by ZRTP participants to retrieve previously established shared secrets, are *not* authenticated as part of ZRTP.

Therefore, an attacker can initiate a session with some party *A* under the disguise of another party *B*, with whom *A* previously established a shared secret. As part of session establishment, *A* is supposed to verify that *B* knows their shared secret. If the attacker deliberately chooses values that cause verification to fail, *A* will decide—following ZRTP specification—that *B* has "forgotten" the shared secret. The ZRTP[8] specification explicitly says that the protocol may proceed even if the set of shared secrets is empty, in which case the attacker ends up sharing a key with *A* who thinks she shares this key with *B*.

Even if the participants stop the protocol after losing their shared secrets, but are using VoIP devices without displays, they cannot confirm the computed key by voice and must stop communicating. In this case, the attack becomes a simple and effective denial of service. Our analysis of ZRTP [11] is supported by the AVISPA[12] formal analysis tool.

Although, several real, exploitable vulnerabilities were found in VoIP security protocols, our main contribution is to highlight the importance of analyzing protocols in context rather than in isolation. Therefore, our study does focus upon important lessons for the design and analysis of security protocols in general.

III. SESSION INITIATION PROTOCOL

Session Initiation Protocol (SIP) is well established protocol used for signalling in telephony and VoIP. SIP is a text based application layer signaling protocol used for establishing, managing and terminating the session. SIP uses a HTTP-like request-response mechanism for initiating a two-way communication session. The

protocol itself is modeled on the three-way TCP handshake. The SIP protocol stack was designed as a rendezvous mechanism for the Internet, where it is more efficient to replicate and transmit the session setup request to multiple destinations, collate the responses, and present the best one to the caller. The following figure 2 describes VoIP scenario where signalling and media transmission are in action.

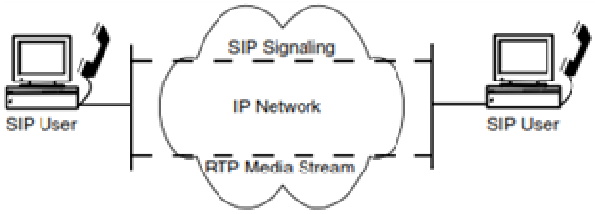

Figure 2: VoIP in Action

SIP network consists of the following components: User Agent Client (UAC), User Agent Server (UAS), proxy server (stateful and stateless), redirect server, location server, and a registrar. UAC and UAS may be a soft phone or hard phone. SIP has six request methods and six response methods are given as follows:

Request Methods:[1]

1. REGISTER: Used by a UA to indicate its current IP address and the URLs for which it would like to receive calls.
2. INVITE: Used to establish a media session between user agents.
3. ACK: Confirms reliable message exchanges.
4. CANCEL: Terminates a pending request.
5. BYE: Terminates a session between two users in a conference.
6. OPTIONS: Requests information about the capabilities of a caller, without setting up a call.

Response Methods:[1]

1. Provisional (1xx): Request received and being processed.
2. Success (2xx): The action was successfully received, understood, and accepted.
3. Redirection (3xx): Further action needs to be taken (typically by sender) to complete the request.
4. Client Error (4xx): The request contains bad syntax or cannot be fulfilled at the server.
5. Server Error (5xx): The server failed to fulfill an apparently valid request.
6. Global Failure (6xx): The request cannot be fulfilled at any server.

SIP user identification is based on a special type of Uniform Resource Identifier (URI) called SIP URI, similar to email addresses. A location server stores the address bindings of users when they register themselves with the registrar. SIP servers can operate in a proxy mode or redirect mode. SIP also supports forking proxies, which receive a single request and forward it to multiple recipients (this makes SIP potentially vulnerable to denial of service attacks).

SIP provides the following services which are implemented in the SIP user agents.

SIP Services[10][16]

1 Call Hold
2 Consultation Hold
3 Music on Hold
4 Transfer - Unattended
5 Transfer - Attended
6 Transfer - Instant Messaging
7 Call Forwarding Unconditional
8 Call Forwarding - Busy
9 Call Forwarding - No Answer
10 3-Way Conference - Third Party Is Added
11 3-Way Conference - Third Party Joins
12 Find-Me
13 Call Management (Incoming Call Screening)
14 Call Management (Outgoing Call Screening)
15 Call Park
16 Call Pickup
17 Automatic Redial
18 Click to Dial

Establishing a SIP Session[1][17]

A SIP system consists of user agents, proxy servers, redirect servers, and registrars. In SIP, who initiates a call is called user agent client(UAC) and who receives a call is called user agent server (UAS). A UAC and a UAS are software programs that may run on a computer system, softphone phone, or a personal digital assistant (PDA). The first step in making a VoIP call is Registration. The SIP registration is the process in which the UA (UAC & UAS) sends a REGISTER request to the Registrar. In the REGISTER request *To* and *From* field are same.

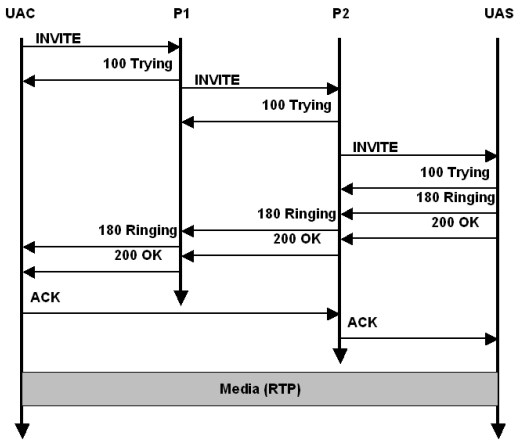

Figure 3: SIP Session Establishment

Figure 3 shows the session establishment between UAC and UAS. UAC sends an INVITE request to P1 (Proxy 1) and P1 routes the call to another proxy server and sands back a 100 Trying (Provisional)response to UAC. From the UAC's reference, P1 understood that the request should be forwarded to P2 (Proxy 2). When the INVITE request arrives at P2,it queries its location server and further proxies the request to the UAS and sends back 100 Trying(Provisional)response to the P1. Upon receiving INVITE request UAS issues a provisional response followed by a final response. The session is established when the UAC receives the final response(OK 200) with tag in *To*: field(contains in the tag field indicates that session has established) and sends out a new request ACK.For the subsequent transmission of request and response the proxy should remain stay in the path. There are two types of proxy servers (i.e. stateful and stateless proxy). Stateful proxies stay in the session path for all the subsequent request and response flow and stateless server just forwards the request and response. In Figure, P2 has done this and thus it receives the ACK request from the UAC. After establishing session the RTP packets (media) media flows directly between the two endpoints.

IV. SESSION DESCRIPTION PROTOCOL [1] [4]

The Session Description Protocol (SDP) is defined by the IETF in RFC 2327. The purpose of SDP is to communicate the media capabilities and desired properties between the communicating parties. SDP can be used in connection with many different signaling protocols and media gateway control protocols. The session description in SDP is represented in a text based list of variables and their parameters.

V. SDES [2] [4]

SDP Security Descriptions for Media Streams

The SDP Security Descriptions for Media Streams enables the peers in a two-party unicast communication to exchange security parameters and keys that allow them to set up SRTP[5] cryptographic contexts. SDES is not an authenticated key exchange mechanism because the security parameters and the keys are carried in clear text in form of SDP attributes. SDES relies instead on the SIP signaling to be protected or encrypted by other means, like for instance S/MIME or TLS. Therefore, SDES may be regarded as an enhancement of the "k =" SDP parameter, which was deemed insufficient because the establishment of data security associations in general and SRTP cryptographic contexts in particular require far more parameters than just a key.

Negotiation of the SRTP Cryptographic Parameters

SDES introduces a new SDP attribute, named crypto , with the following structure:

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

One or more crypto attributes may accompany a media line in an SDP offer. A crypto attribute should not be specified at the session level. Multiple crypto attributes indicate that the sender of the SDP offerer (the UAC) supports multiple security schemes, in the given order of preference; this enables the answerer to select one of them (codecs, sampling rate, etc.) in the subsequent SDP answer. The fields in a crypto definition carry the following information:

• The *tag* field contains an integer number and is used in the SDP UAC/UAS process to uniquely identify crypto attributes associated with certain media line given below:

Tag 1: AES_CM_128_HMAC_SHAI_80

Tag 2: AES_CM_128_HMAC_SHAI_32

Tag 3: F8_128_HMAC_SHAI_32

• The *crypto-suite* field encodes the suite of encryption and authentication algorithms used to protect the respective media stream. The crypto-suite available in SDP[4568 rfc] are tabulated in following table.

	Tag 1	**Tag 2**	**Tag 3**
Master key length	128 bits	128 bits	128 bits
Master salt length	112 bits	112 bits	112 bits
SRTP lifetime	2^48 packets	2^48 packets	2^48 packets
SRTCP lifetime	2^31 packets	2^31 packets	2^31 packets
Cipher	AES Counter Mode	AES Counter Mode	AES F8 Mode
Encryption key	128 bits	128 bits	128 bits
MAC	HMAC-SHA1	HMAC-SHA1	HMAC-SHA1
SRTP auth. tag	80 bits	32 bits	80 bits
SRTCP auth. tag	80 bits	80 bits	80 bits
SRTP auth. key len.	160 bits	160 bits	160 bits
SRTCP auth. key len.	160 bits	160 bits	160 bits

Table 1 : Crypto Suites for AES Algorithm

• The *key-params* field specifies one or more sets of keying material that is used to protect the traffic in the sending direction. One such set has the following structure:

inline: base64(key salt)[" " lifetime][" " MKI " : " MKI-length]

This set also contains: the SRTP master key (key) concatenated with the salting key (salt) and encoded in base64, followed by an optional maximum lifetime of the key (lifetime) represented in number of packets protected with the respective keys, and an optional MKI value accompanied by the MKI field length. If missing, the lifetime of the key defaults to the 2^{48} SRTP packets and 2^{31}SRTCP packets (whichever occurs first). Multiple key sets identified by different MKI values can be defined for a media stream to enable MKI-based re-keying. On the other hand, no support for the From – To type of re-keying is offered in SDES. The *session-params* field provides a number of additional SRTP cryptographic context parameters. When missing, default values defined by SRTP are used. These parameters are:

• KDR – the key derivation rate, which is 2 power the number specified by this parameter;

• UNENCRYPTED SRTP, UNENCRYPTED SRTCP, UNAUTHNTICATED SRTP – indicating which type of protection the SRTP/SRTCP packets should (not) be afforded;

• WSH – the window size hint, is used as a hint for the size of the SRTP replay protection window in the absence of other sources of information like for instance SDP bandwidth modifiers;

• FEC ORDER – indicating the order in which Forward Error Correction and SRTP are applied to the media packets

• FEC KEY – the master key and salt to be used for the FEC stream where the FEC stream is sent as part of a different RTP session (e.g. to a different port number than the media stream).

There are two further parameters that characterize an SRTP cryptographic context: the SSRC and the ROC (Roll Over Count). Since SDES operates exclusively in the signaling plane, no explicit correlation between the negotiated SRTP cryptographic contexts and the SSRC (Synchronisation Source Identifier) values of the actual media streams can be established at the signaling time. For this reason a "late binding" mechanism is necessary, which maps the SSRC values to the SRTP cryptographic contexts created by SDES when media traffic is actually received. The ROC is always initialized to zero because in the peer-to-peer unicast scenario that characterizes SDES, both participants join at about the same time. Also, the initial SQN(Sequence Number) value should be generated in the range $0 – 2^{15} -1$ (instead of the full range of $0 – 2^{16} -1$).This reduces the chances of the sender's and receiver's ROCs becoming out of sync to those cases when at least 2^{15} consecutive RTP packets are lost at the beginning of the media session. In such a case the sender will increment the ROC

whereas the receiver will not. However losing 2^{15} consecutive RTP packets corresponds, for instance, to several minutes of conversation, which is unlike to occur in practical scenarios.

Let us now survey how the SDP payloads in a SDES exchange where the offer contains two crypto attributes for the media, one of which specifies two MKIs.

SDP -UAC

v=0
o=alice2890844526290842807INIP410.47.16.5
s=Confidentialconversation
i=SecurityparametersexchangedwithSDES
u=http://www.example.com/abc.html
e=alice@example.com
c=INIP4168.2.17.12
t=28733974962873404696
m=audio49170RTP/SAVP0
a=crypto:1AESCM128HMACSHA180
inline:WVNfX19zZW1jdGwgKCkgewkyMjA7fQp9CnVubGVz|2^20|1:4
session-params
a=crypto:2F8128HMACSHA180 inline:MTIzNDU2Nzg5QUJDREUwMTIzNDU2Nzg5QUJjZGVm|2^20|1:4; inline:QUJjZGVmMTIzNDU2Nzg5QUJDREUwMTIzNDU2Nzg5|2^20|2:4
session-params

SDP - UAS

v=0
o=bob2569084480708426 34INIP410.47.16.5
s=Confidentialconversation
i=SecurityparametersexchangedwithSDES
u=http://www.example.com/abc.html
e=bob@example.com
c=INIP4168.2.17.11
t=28733975262873405696
m=audio32640RTP/SAVP0
a=crypto:1AESCM128HMACSHA180
inline:PS1uQCVeeCFCanVmcjkpPywjNWhcYD0mXXtxaVBR|2^20|1:4

The UAS picks the first proposal (tag = 1) and responds with the same cipher suite and the SRTP master and salting keys for his own sending direction. As already mentioned, the SRTP master and salting keys provided by each endpoint in the a = crypto attribute of the SDP are used to derive the SRTP session keys employed by the respective endpoint to encrypt and generate the MAC for the RTP and RTCP streams that it sends. This differs conceptually from most of the other SDP attributes (like for instance transport address and codecs), where the entity providing them offers information related to the media that it receives. Providing the SRTP keys for the sending – rather than receiving – direction is meant to avoid a two-time pad situation occurring in case of forking scenarios when the SSRC from distinct UASs collide: if the SRTP keys for the receiving directionwereprovidedbytheUACinstead,thenalltheUASswouldhaveusedthem, which combined with the SSRC collision would have led to two-time pad. With the current design, each UAS will pick its own keys. As a final observation, an endpoint must provide SRTP key seven if it only acts as a media receiver in this scenario the receiver's key will be used to protect the RTCP receiver reports that he sends.

VI. **ZRTP [6] [8] [9] [11]**

ZRTP (Zimmermann et al. 2008) is a key agreement protocol that enables the participants in a one-to-one multimedia session to establish a shared secret and agree on the security parameters for setting up SRTP sessions. ZRTP is essentially an authenticated ephemeral Diffie – Hellman exchange, with the mutual authentication being performed by means of a Short Authentication String (SAS). In this way ZRTP can operate without support from a PKI. It may however take advantage of the existence of a PKI by providing an alternative option for the key exchange to be mutually authenticated using digital signatures. ZRTP also combines a number of security mechanisms that reduce its vulnerability to man-in-the-middle attacks, even if the key exchange goes unauthenticated.

The ZRTP exchange takes place over the same port numbers used by the multimedia session for the RTP traffic; the initial messages are sent as each participant learns the port number of the peer. There will be as many distinct ZRTP exchanges as RTP sessions are defined in the multimedia session.

The ZRTP header uses a common format with the RTP header. It specifies an RTP version number of zero to differentiate it from the regular RTP traffic (which uses the version number 2) and includes a specific ZRTP magic cookie to differentiate it from STUN packets.

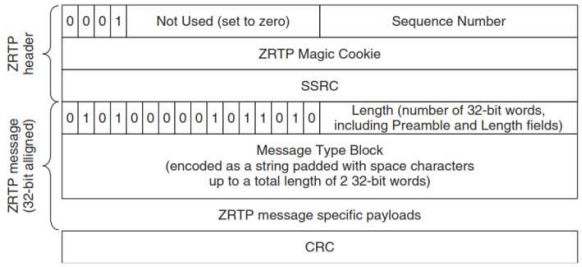

Figure 4: ZRTP Packet Format

The ZRTP packet header as shown in figure 4 also contains:

- A sequence number that is initialized to a random value and is incremented with each ZRTP packet sent. The sequence number is used to estimate losses and detect out-of-order message arrival.
- The SSRC value of the RTP stream with which the ZRTP packet shares the transport endpoints.
- A ZRTP packet carries a ZRTP message, which starts itself with a preamble, followed by the message length, message type and the payloads that are specific to that particular ZRTP message type. The ZRTP packets end with a 32-bit CRC to detect transmissions errors as a supplementary mechanism to the UDP checksum.

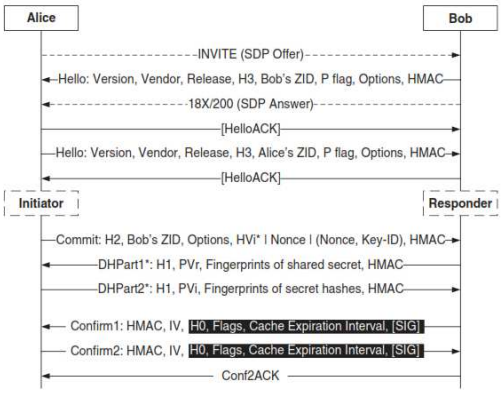

Figure 5: ZRTP Key Exchange

The ZRTP exchange, illustrated in Figure 5, has been designed to be robust to packet losses because the practice has shown that the initial media packets that follow the setup of a session are affected by higher loss rates. Therefore, ZRTP employs an aggressive retransmission scheme (20 Hello retransmissions are performed in about 4 seconds). Acknowledgment messages (HelloACK, Confirm2ACK) are used to stop the retransmissions of a message when no explicit response to that message is received.

H3, H2,
H1, H0: Hash images
ZID: A unique ZRTP endpoit identifier
Options: List of supported/selected hash, encryption and HMAC algorithms, key agreement types, SAS format
HVi: Ahash over the initiator's DHPart2 and the responder's Options

PV: Ephemeral Diffie–Hellman keys
SIG: Digital signature over the exchanged SAS

IV: Initial Value

ZRTP implements three key agreement modes:[8]

• Diffie – Hellman mode
• Preshared mode
• Multistream mode

Discovery and Capabilities Negotiation

The ZRTP discovery phase consists of the ZRTP endpoint exchanging Hello messages. This enables each participant to discover whether the peer implements ZRTP and which cryptographic algorithms, key agreement types and SAS formats are supported. Hello messages may be sent at any time after the peers learn each other's RTP transport address from the SDP payloads. The sender of the first Hello message is in general the one that learns the transport parameters of his peer first, namely the receiver of the SDP offer.

The Diffie – Hellman Key Exchange and the Key Derivation[8][9][17]

The DHPart messages exchange ephemeral Diffie – Hellman keys, which are used to derive a shared secret (denoted as DHResult). ZRTP also implements a key continuity mechanism, that uses cached shared secrets generated during previous sessions established with a certain peer as input for deriving the ZRTP master secret of the current session with the same peer. In order to enable the peers to check the consistency of their shared secrets cache, fingerprints calculated over the cached shared secrets are exchanged in the DHPart messages.

The ZRTP master secret s0 calculation is done according to the following formula:

s0 = hash(counter | DHResult | "ZRTP-HMAC-KDF" | ZIDi | ZIDr | total hash| len(s1) |s1 | len(s2) | s2 | len(s3) | s3)

where the "i" and "r" suffixes denote the initiator and the responder, and counter is always set to 1, DHResult represents the Diffie – Hellman secret which is calculated by each party after receiving the peer's public key, total hash represents a hash calculated over the concatenation of the responder's Hello, the initiator's Commit and the two DHPart messages of the ZRTP exchange , s1 , s2 and s3 represent placeholders for the cached shared secrets, which are either derived from the ZRTP master secrets of previous sessions established between the two parties or represent preconfigured shared secrets set up using an out-of-bound mechanism, the lengths of the s1 , s2 and s3 keys – a length of zero indicates that the corresponding key is missing from the s0 calculation.

The s1, s2 and s3 are obtained in the following way:

• s1 represents one of the "retained secrets" rs1 and rs2 .

• s2 is an optional auxiliary shared secret auxsecret that may be established or pre-configured using any available out-of-bound mechanism.

• s3 represents an optional pbxsecret typically used in the "trusted man-in-the-middle" scenario.

Figure illustrates the key derivation procedure used by the Diffie – Helman key agreement mode. The ZRTP master secret s0 is used as input to a HMAC function, which is applied to a set of distinct character string constants to derive the following session keys:

1. The SRTP master and salting keys: srtpkeyi, srptkeyr, srtpsalti and srtpsaltr. The initiator uses the srtpkeyi, srtpsalti keys to derive the SRTP session keys to encrypt and integrity protect the outgoing SRTP/SRTCP streams, while the responder uses the same keys to validate the authentication tag and decrypt the incoming streams. The srtpkeyr, srtpsaltr keys are used in a similar manner for the SRTP/SRTCP streams flowing in the opposite direction.

2. The shared keys zrtpkeyr and zrtpkeyi, used by the responder to encrypt the Confirm1message and by the initiator to encrypt the Confirm2 message. The same keys are used by the ZRTP peer to decrypt the respective messages.

3. The shared keys hmackeyr and hmackeyi , used by the responder to integrity protect the Confirm1 message and by the initiator to integrity protect the Confirm2 message. They are also employed to integrity protect the GoClear messages. The same keys are used by the ZRTP peer to validate the authentication tag of the respective messages.

4. A ZRTP session key ZRTPSess, which is used to further derive the following session keys:

(a) the sashash – used to generate the SAS

(b)the pbxsecret – used in trusted man-in-the-middle scenarios

(c) a set of keys, one per media stream used to generate the set of session keys described at points 1, 2 and 3 above in multi-stream mode. Key derivation[8][17] procedure as shown in figure 6.

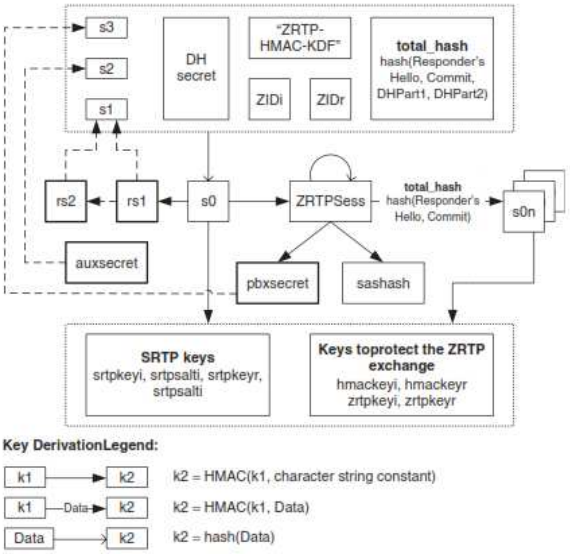

Figure 6: ZRTP Key Exchange

If the encryption is turned off during a session's lifetime, a fresh ZRTPSess key is generated by applying a hash function to itself. This ensures that former ZRTPSess secret keys cannot be easily compromised retroactively if a certain ZRTPSess is compromised at some time. The new ZRTPSess key will be used to generate the new ZRTP master secret using the multistream mode in case the encryption is turned on again during the session's lifetime.

ZRTP Packet Injection Mitigation[8]

Packet injection represents a significant class of DoS attacks that can be mounted by attackers that are not located on the media path. One mechanism to protect a ZRTP exchange against packet injection is the use of total hash.

Man-in-the-middle Attack Mitigation[8][17]

The SAS represents the primary mechanism used by ZRTP to authenticate the ephemeral Diffie – Hellman exchange and therefore protect against man-in-the-middle attacks. It consists of the validation of a short human-readable string over a secure out-of-bound channel, with the purpose of ensuring that the peers of the ZRTP exchange have obtained the same ZRTP master secret and therefore no man-in-the-middle attack occurred.

The SAS value is a human-readable string that usually consists of four "human-oriented" base32 symbols. The SAS value is obtained by truncating and converting to base32 the SAShash.

VII. SRTP [5] [8] [17]

In VoIP usually session are established by SIP and data transported by using the Real-time Transport Protocol (RTP). SRTP is a profile of RTP which provides confidentiality, message authentication, and replay protection to RTP data and control traffic. SRTP uses two types of keys, namely a session key and a master key. These keys are provided by a key management protocol. The master key, salts, and other parameters in the cryptographic context are also provided by the key management protocol, such as SDES, ZRTP, or DTLS-SRTP. Each RTP packet consists of a 16-bit sequence number (SEQ) which is a randomly generated integer value increment monotonically for each packet. The rollover counter is maintained by the receiver and is incremented by 1 every time the sequence number wraps around. In RTP, for a multicast stream with multiple senders, a synchronization source identifier (SSRC) is unique 32 bit number which identifies a sender within a session. A cryptographic context for SRTP is identified by the SSRC, destination network address and destination port. For data encryption, SRTP uses AES (Advanced Encryption Standard), in one of the following two modes: (i) Segmented Integer Counter mode, or (ii) f-8 mode. AES require key, SSRC and SEQ as input, where "key" is the encryption key, SSRC is the synchronization source identifier (chosen randomly for each synchronization source; with the intent that no two synchronization sources in the same session have the same SSRC) and SEQ is the sequence number of the packet incremented by 1 for each packet sent. Instead of using AES as a block cipher, SRTP uses it as if it were a stream cipher and encrypts datagrams by XOR'ing them with the output of AES applied to (key, SSRC, SEQ).

SRTP key derivation

SRTP uses pseudo-random function (PRF) to generate encryption and authentication session keys from the master key, salt and the RTP packet sequence number. Both master key and salt are derived deterministically by applying HMAC, keyed with the material received during the key exchange protocol, to a known plaintext (as defined by the key exchange protocol). The determinism of key derivation is a fatal flaw since it makes an unwarranted assumption about the key exchange protocol used to create the master key. Session key derivation involves an 8-bit label, master salt ms, the key derivation rate, as determined by the cryptographic context, and the index (48-bit ROC||SEQ). Let || denote string concatenation. Let x = (_label_||r) xor ms, where r is the integer quotient obtained by dividing the index by the key derivation rate. Let mk denote the master key and PRF(k, x) denote a pseudorandom function family such that for the secret random key k, given m-bit x, the output is an n-bit string computationally indistinguishable from a truly random n-bit string. The session keys are generated as PRF(mk, x), using different labels for encryption, authentication and salting keys, respectively.

An important point to note is that there is no receiver generated randomness in the session key derivation process. This will allow us to break the protocol because security of the stream cipher-like encryption used in SRTP depends critically on the keystream never repeating.

For the keystream never to repeat, the PRF output must never repeat, because the other inputs into AES (i.e. SSRC and SEQ) are not required to be globally unique and can repeat from session to session. SSRC and SEQ values are public and can be eavesdropped by the attacker. The PRF input must be unique for every session. Master key and salt must be unique in each session. If the attacker ever succeeds in tricking an SRTP session into re-using previously used key material, the master key will repeat. The figure 7 shows key derivation mechanism in SRTP.

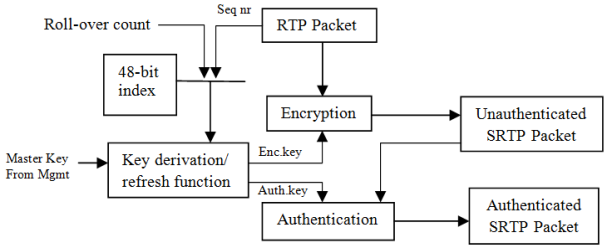

Figure 7: SRTP Key Derivation

VIII. DTLS-SRTP [17] [9]

DTLS-SRTP [12][8] is a DTLS-based extension of SRTP, to reap benefits of performance and security flexibility within SRTP along with the key and association management of DTLS. DTLS-SRTP can be viewed as a key management method for SRTP. The DTLS extension is used to negotiate SRTP algorithms. The DTLS handshake is used to establish keying material, algorithms, and parameters for SRTP. The application data in DTLS is protected using SRTP. Other DTLS record layer content types are protected using the ordinary DTLS record format.

The general pattern of DTLS-SRTP is as follows. For each RTP or RTCP flow, the peers do a DTLS handshake on the same source and destination port pair to establish a DTLS association. SIP determines which side is the DTLS client and which side is the DTLS server. The keying material from that handshake is fed into the SRTP stack. Once that association is established, RTP packets are protected (becoming SRTP) using that keying material.

DTLS-SRTP [9] is defined for point-to-point media sessions, in which there are exactly two participants. Each DTLS-SRTP session contains a single DTLS association, and either two SRTP contexts (if media traffic is flowing in both directions on the same host/port quartet) or one SRTP context (if media traffic is only flowing in one direction). All SRTP traffic flowing over that pair in a given direction uses a single SRTP context. A single DTLS-SRTP session only protects data carried over a single UDP source and destination port pair in a single direction.

Between a single pair of participants, there may be multiple media sessions. There must be a separate DTLS-SRTP session for each distinct pair of source and destination ports used by a media session. However, for efficiency, it is recommended that such sessions share a single DTLS session and hence amortize the initial public key handshake. This is done by deriving separate DTLS-SRTP master keys for each DTLS- SRTP session from the same DTLS output.

The security of data exchange done by DTLS-SRTP is dependent on the integrity of the public key certificates possessed by the communicating parties. Ideally, it is maintained by a PKI, but with this solution high costs are associated. An alternative approach is to assign some of the responsibility to the SIP layer. For example, parties may exchange hashes of their public keys in the SIP layer. Then, if the SIP layer is secured,

this provides sufficient guarantees; if it is not, this serves just as an additional hurdle for the attacker, and the combined protocol is still prone to attacks.

IX. **MSDES [15] [17] [5]**

SDES is a keying exchange mechanism in SIP.A new attribute in SDP, known as "crypto", is used to negotiate cryptographic parameters for the SRTP. But, it does not provide sufficient security. Further security requirements may be TLS or IPSec. But TLS and IPSec creates a serious overhead and cost at SIP agents. In particular, TLS has a high cost related to the need of Certificate Authority (CA). Because SIP communicates in a clear-text format so the keying exchange protocol such as SDES provides insufficient security, as seen in the following attribute:

a=crypto: <tag><crypto-suite> inline: <key||salt> [session-parms]

The *tag* field is a unique numeric value used to indicate which crypto-suite UAC is using and UAS indicates which crypto attribute is acceptable. The *crypto-suite* field is described in encryption and authentication transforms, which is to be used for SRTP media streams. The *key||salt* are key information deriving from concatenating master key and salt, and then converting to a base64 format. The *session-parms* are optional session parameters (i.e. master key lifetime, etc). The crypto-suites are defined by SDES, using AES and SHA1 to provide encryption and authentication, respectively.

The crypto-suites are given below:

AES_CM_128_HMAC_SHA1_80,

AES_CM_128_HMAC_SHA1_32, and

F8_128_HMAC_SHA1_32.

Even same crypto-suite is used by both the UAC and UAS, the same key and salts are not used by each side. Therefore, each side will generate and pass these parameters using SDP.

The "crypto" attribute in SDES transports all its parameters in a clear-text format. Due to this reason, to make more secure key exchange mechanism this paper has been carried out using MSDES security technique. It encrypts the information transported using User Agent Password (UAP), which is used in the SIP proxy registration. The key-info (key||salt) encryption for the keys is performed using UAP as follows:

a=crypto: <tag><crypto-suite> inline: E(uap, <key||salt>)

[Session-params]

In the experiment, SIP proxy[15] performed the key management using the UAP (User Agent Password) of the User Agent Client (UAC) and the User Agent Server (UAS), as seen in Figure 8.

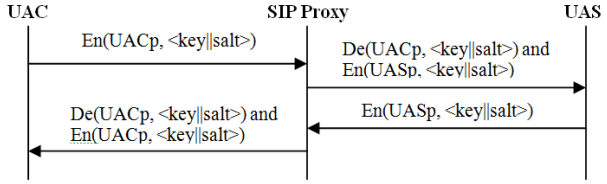

Figure 8: Key Encryption in MSDES at each proxy

As described in above figure 8, the UAC starts encryption by using UAC password, and <*crypto-suite*> encryption algorithm (i.e. AES 256 bits). After that, the SIP proxy decrypts this information and again encrypts the <*crypto-suite*> with UAS password for the information transportation. The encryption is also performed once when the information has reached the UAS. Same steps are performed in reverse order while response coming from UAS. This process facilitates the exchange of cryptographic parameters to make the SRTP more secured. So, it becomes more robust from MitM although the SIP is transported in a clear-text format.

X. **ATTACKS ON VOIP**

The security threats against VoIP deployments, services, and end users are given below:[18][17][12]

1. **Social threats** are those that are directly related to humans. For example, misconfiguration of VOIP equipment, bugs or bad protocol interactions in VoIP systems may invite attacks by misrepresenting the identity of malicious parties as genuine users. Such attacks may lead to attacks such as phishing, spam and theft of service.

2. **Eavesdropping** covers a situations where a third party can illegally (i.e. without any consent from either party) and without authorization from the either parties, listen the data of a VoIP session during the signalling (call setup), and possibly modify session parameters. Such attacks include call re-routing and interception of unencrypted RTP sessions.

3. **Denial of service threats** is an attempt to make a machine or network resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS attack may vary; it generally consists of efforts to indefinitely interrupt services of a host, connected to the Internet access, to VoIP services. DoS attack affects a user's or organization's all communication capabilities (i.e., when all VoIP and data communications are multiplexed over the same network which can be targeted through a DoS attack). This type of attacks may be exploiting flaws in the call setup or the implementation of services in VOIP. This attack is also possible with physical components (e.g., physically disconnecting or severing a cable) or through computing or other infrastructures (e.g., disabling the DNS server, or shutting down power).

4. **Service abuse threats** VoIP offers several value added services which are offered in a commercial setting. Examples of service abuse threats include toll fraud and billing avoidance.

5. **Physical access threats** refer to unauthorized physical access to VoIP equipment (i.e. Gateways, router and other equipment that may be compromised), or to the physical layer of the network.

6. **Interruption of services threats** refer to the problems which occurred non-intentionally rendering VOIP services unavailable. Examples of such threats include loss of power due to inclement weather, resource exhaustion due to over-subscription, and performance issues that degrade call quality.

XI. EXPERIMENTAL EVALUATION

MSDES [15][17] is a keying protocol for the keying exchange protocol to be used for SRTP by enhancing the SDES. The efficiency of MSDES was evaluated and summarised in table 2, in comparison to SDES, ZRTP, and DTLS-SRTP.

Indicators	SDES	ZRTP	DTLS-SRTP	MSDES
MitM	Attacks possible	Mitigated through shared secrets and SAS	Mitigated through certificate fingerprints	Attacks must rely on SIP hash brute force
Sig. Conf.	Yes	No	No	No
Forking	Key Leak	Yes	Yes	No Key Leak
Media Clippoing	Yes	No	No	Yes
Shared Key Conferencing	Yes	No	No	Yes
Session Recording	Yes	No	No	Yes
Using PKI	No	No	Yes	No

Table 2: Key Exchange Protocol Summary

XII. CONCLUSION

We have analysed four media key management protocols i.e. SDES, ZRTP, DTLS-SRTP and MSDES. SDES is the most widely used key exchange protocol. MSDES with respect to key leakage problem was found to be more secured media keying protocol. The SIP security solution cannot provide complete protection against attacks. Even the MSDES is not 100% safe from Man-in-the-middle attack which relies on SIP hash brute force. MSDES creates less overheads cost and hence its performance is faster than ZRTP and DTLS-SRTP.

ACKNOWLEDGEMENT

I would like to thank the anonymous reviewers for their valuable remarks that helped me to carry out this analysis.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session initiation protocol," RFC 3261 (Proposed Standard), June 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," RFC 3550 (Standard), July 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt

[3] M. Handley, V. Jacobson, and C. Perkins, "SDP: Session description protocol," RFC 4566 (Proposed Standard), July 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4566.txt

[4] F. Andreasen, M. Baugher, and D. Wing, "Session description protocol (SDP) security descriptions for media streams," RFC 4568 (Proposed Standard), July 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4568.txt

[5] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman, "The secure real-time transport protocol (SRTP)," RFC 3711 (Proposed Standard), Mar. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3711.txt

[6] P. Zimmermann, A. Johnston, and J. Callas, "ZRTP media path key agreement for secure RTP, IETF internet-draft, work in progress, draft-zimmermann-avt-zrtp-15," Mar. 2009. [Online]. Available: http://tools.ietf.org/html/draft- zimmermann-avt- zrtp-15
[7] D. Mc-Grew and E. Rescorla, "Datagram transport layer security (DTLS) extension to establish keys for secure real-time transport protocol (SRTP), IETF internet-draft, work in progress," Feb. 2009. [Online]. Available: http://tools.ietf.org/html/draft-ietf- avt- dtls-srtp-07
[8] John Wiley & Sons Publications: SIP Security by Dorgham Sisalem, John Floroiu, Jiri Kuthan, Ulrich Abend, Henning Schulzrinne. ISBN 978-0-470-51636-2
[9] Addison-Wesley : Securing VoIP Networks by Peter Thermos and Ari Takanen. ISBN 0-321-43734-9
[10] McGrawHill : Career Grade Voice over Internet Protocol by Daniel Collins. ISBN-13: 978-0071406345
[11] R. Bresciani, "The ZRTP protocol: Analysis on the diffie-hellman mode," Computer Science Department Technical Report TCD- CS-2009-13, Trinity College Dublin, 2009. [Online]. Available: http://zfoneproject.com/docs/TCD-CS-2009- 13.pdf
[12] P. Gupta and V. Shmatikov, "Security analysis of voice-over-IP proto- cols," in Proc. Comput. Security Foundations Symp. IEEE, July 2007
[13] S. Garg, N. Singh, T. Tsai, "SRTP+, an efficient scheme for RTP packet authentication", available at http://www.pubs.research.avayalabs.com/pdfs/ALR-2004-001-paper.pdf
[14] D. Geneiatakis, A. Dagiouklas, S. Ehlert, G. Kambourakis, C. Lambri- noudakis, D. Sisalem, and S. Gritzalis, "Survey of security vulnerabilities in SIP," IEEE Commun. Tuts. Surveys, vol. 8, no. 3, Oct. 2006.
[15] S. Pangpronpitag and P. Kasabai, "MSDES:More SDES Key Agreement for SRTP", in I.J.C.T.E., Vol 4, No. 5, Oct 2012
[16] Session Initiation Protocol Service Examples, available at http://tools.ietf.org/html/rfc5359
[17] Vijay K. Gurbani and Vladimir Kolesnikov "A Survey and analysis of Media Keying Techniques in the SIP" IEEE COMMUNICATIONS SURVEY & TUTORIALS,VOL.13,NO.2 SECOND QUARTER 2001.
[18] Angelos D. Keromytis " A Survey Of VOIP Security Research" Springer-Verlag Berlin Heidelberg 2009.

Authors Bibliography

Rajesh Kumar is pursuing Masters of Engineering at Computer Science and Engineering Department, from Kalol Institute of Technology &Research Centre, Gujarat, India affiliated to Gujarat Technological University, Ahmadabad. He has received B.E. (Computer Science & Engg.) degree in 2003 from Bangalore University. He is actively involved in academic field for more than five years. His professional experiences include working on Linux Internals, Device Drivers & Kernel Programming, Mobile Application Development, SIP Testing. His research interests span across RTP and SRTP.

Prof. Sandip Chauhan working as Assistant Professor at Kalol Institute of Technology, Kalol affiliated to Gujarat Technological University. He has received B.E. I.T. in 2006, and M.Tech. CSE in 2009 from NIT Trichy. He is actively involved in academic field since last 3 years. He has guided several academic projects and he is presently teaching PG students. His research areas are Distributed and Parallel computing, Cloud computing, and Network Security. He is a reviewer of various national and international journals.

