Journal of Universal Computer Science, vol. 14, no. 5 (2008), 673-692
submitted: 1/11/07, accepted: 28/2/08, appeared: 1/3/08 © *J.UCS*

Security and Usability Aspects of Man-in-the-Middle Attacks on ZRTP

Martin Petraschek, Thomas Hoeher, Oliver Jung
(ftw (Telecommunications Research Center Vienna), Austria
{petraschek, hoeher, jung}@ftw.at)

Helmut Hlavacs, Wilfried Gansterer
(University of Vienna, Austria
{helmut.hlavacs, wilfried.gansterer}@univie.ac.at)

Abstract: ZRTP is a protocol designed to set up a shared secret between two communication parties which is subsequently used to secure the media stream (i.e. the audio data) of a VoIP connection. It uses Diffie-Hellman (DH) key exchange to agree upon a session key, which is inherently vulnerable to active Man-in-the-Middle (MitM) attacks. Therefore ZRTP introduces some proven methods to detect such attacks. The most important measure is a so called Short Authentication String (SAS). This is a set of characters that is derived essentially from the public values of the Diffie-Hellman key exchange and displayed to the end users for reading out and comparing over the phone. If the SAS on the caller's and the callee's side match, there is a high probability that no MitM attack is going on. Furthermore, ZRTP offers a form of key continuity by caching key material from previous sessions for use in the next call. In order to prevent that a MitM can manipulate the Diffie-Hellman key exchange in such a way that both partners get the same SAS although different shared keys were negotiated, ZRTP uses hash commitment for the public DH value.

Despite these measures a Relay Attack (also known as Mafia Fraud Attack or Chess Grandmaster Attack) is still possible. We present a practical implementation of such an attack and discuss its characteristics and limitations, and show that the attack works only in certain scenarios.

Key Words: ZRTP, Man-in-the-Middle-Attack, Security
Category: K.6.5

1 Introduction

In most Voice over IP (VoIP) environments, media data is sent over the network without protection and can thus easily be eavesdropped. For securing the conversation, SRTP (Secure Realtime Transmission Protocol, RFC 3711 [Baughter et al. 2004]) has evolved as a widely accepted standard which provides encryption and integrity checking as well as key derivation functionality. SRTP does not implement a key agreement procedure, which means that a secret master key has to be provided by some other mechanism. Setting up a shared secret between Alice[1] and Bob in a secure way is not trivial, since they can only

[1] Alice and Bob are commonly used as placeholders for the end users of communication system; Mallory is the malicious attacker.

communicate over the insecure network connection. ZRTP is a key agreement protocol specifically designed for VoIP systems to solve this problem. It was invented by Phil Zimmermann, who is best known for being the creator of the encryption software PGP (Pretty Good Privacy). PGP made it possible to secure email conversation without the need for a PKI (Public Key Infrastructure). Equally, ZRTP is a key agreement protocol which makes it possible to encrypt media data of VoIP connections without relying on a PKI.

ZRTP uses the Diffie-Hellman (DH) [Diffie and Hellman 1976] key exchange to set up a shared secret. Since DH is known to be vulnerable to man-in-the-middle (MitM) attacks, ZRTP implements various mechanisms to detect an ongoing attack. The most important countermeasure is a so called *Short Authentication String* (SAS). This is basically a cryptographic hash value calculated over the shared secret and displayed to the end users through a user interface. The values on Alice's and Bob's side have to be compared by reading them aloud over the phone. If they match, there is a high probability that no MitM attack is going on.

The basic idea for our attack is based on the so called "Chess Grandmaster Attack" (aka Mafia Fraud Attack): Anyone can beat a chess grandmaster by playing postal chess. Just play two grandmasters at once, one as white and the other as black, and relay the moves between them. By that, the grandmasters are effectively playing against each other.

Instead of trying to crack the cryptography behind the DH key exchange, which is believed to be infeasible, Mallory, the malicious attacker, carries out a Man-in-the-Middle (MitM) attack where both Alice and Bob unknowingly connect to her directly. This results in two separate ZRTP sessions, one between Alice and Mallory and another one between Bob and Mallory. Mallory is the "valid" end point for these two sessions, thus she has access to the unencrypted media data and can relay audio data between the two sessions. Naturally, these two session have different encryption keys and different SAS values. Without any further actions, the attack would be detected as soon as Alice and Bob compare their SAS values. However, there are several ways for Mallory to avoid detection by manipulating the SAS procedure which will be discussed in section 4.3.

This article is structured as follows. The underlying technique of ZRTP, the Diffie-Hellman Key Exchange, is introduced in Section 2. This includes the basics of the key agreement algorithm as well as the most important properties. In Section 3 we theoretically consider how an attack on ZRTP could be conducted successfully. Thereby VoIP-related aspects are presented which are decisive to mount a Man-in-the-Middle attack. Concerning ZRTP, we show four possible approaches how to defeat or sidestep the SAS comparison.

2 Signaling and media plane

While media data of a communications session is exchanged using SRTP, the actual session setup [Rosenberg et al. 2002] is done using the Session Initiation Protocol (SIP). SIP is used to establish, take down, redirect or control VoIP connections. Similar to HTTP, it is a purely text based protocol that contains a header consisting of a number of fields, and an optional body. Information about media sessions is carried within this body and described by the Session Description Protocol (SDP) [Handley et al. 2006]. The SDP body is used to exchange information like the used media transport protocols, IP addresses, port numbers, and codecs between the peers.

Once a session is established, media data is transported between the user agents by SRTP or its unsecured version RTP. In contrast to SIP messages that are usually routed over a number of SIP proxies, SRTP data might be transported directly between the user agents. It is therefore common to talk about two different planes: The signalling plane, which comprises SIP and SDP traffic, used to manage calls, and set up session parameters. On the other hand, the media plane denotes the channel that is used to transport data like voice or video.

3 ZRTP

In the following we will explain some important concepts of ZRTP.

3.1 Diffie-Hellman Key Exchange

Diffie-Hellman is a key agreement algorithm used by two parties to agree on a shared secret over an unsecure channel. It works as follows:

1. Alice and Bob agree on two large primes g (generator) and p (residue class), which do not have to be kept secret.
2. Alice generates a random private value a, calculates the public value $A = g^a mod\ p$ and sends A to Bob.
3. Bob generates a random private value b, calculates the public value $B = g^b mod\ p$ and sends B to Alice.
4. Alice calculates the shared $K = B^a mod\ p$
5. Bob calculates the shared key $K = A^b mod\ p$

Alice and Bob now have the same shared key, since $(g^a)^b = g^{ab} = (g^b)^a$.

It is important to note that Bob, who sends his public value B after he received Alice's public value A, can influence the resulting key K by choosing his private value b appropriately, which would lead to security risks (see section 3.2). Therefore ZRTP uses the Diffie-Hellman key exchange *with hash commitment*, where Alice sends a cryptographic hash of A instead of A itself in step 2, thereby *commiting* herself to her public value without sending it. Once Alice received B from Bob she transmits A. By that, neither Alice nor Bob have the possibility to manipulate the resulting shared key K.

Diffie-Hellman has a very important property called *perfect forward secrecy.* This means that disclosure of a session key will not compromise previous and subsequent keys as newly generated session keys are not related to the previous keys.

3.2 Short Authentication String (SAS)

The DH key exchange does not provide authentication and is therefore susceptible to MitM attacks. ZRTP uses a mechanism called Short Authentication String (SAS) to detect such attacks. Using an SAS to prevent MitM attacks is a common mechanism in encrypted voice communications. It is also used by e.g. the GSMK CryptoPhone [Cryptophone] or the Rohde & Schwarz TopSec GSM crypto phone. In ZRTP the SAS is calculated as the HMAC of the shared secret $s0$ (the result of the key agreement procedure) and displayed to Alice and Bob through some user interface. The SAS comparison is done by reading out the SAS value over the phone. If both values match, there is a high probability that no MitM attack is going on.

ZRTP currently provides two distinct methods for generating the SAS value which is negotiated during the ZRTP handshake: If base32 encoding is used, the leftmost 20 bits of the SAS value are rendered in a form called *z-base-32* [O'Whielacronx 2007], which is designed to be as convenient as possible to handle for human users. This results in 4 characters that are displayed to the user. If base256 is negotiated, the leftmost 16 bits of the SAS value are rendered using the *PGP word list* ([Juola and Zimmerman 1996], [PGP Wordlist]). This is a list of 512 distinct words that maps data bytes to verbal words and offers human speakers a convenient way to send them over a voice channel. It works similar to the NATO phonetic alphabet.

Note that due to the use of hash commitment in the Diffie-Hellman key exchange, neither Alice nor Bob can influence the value of the SAS string. Guessing the correct SAS value has a success rate of only 1 in 2^{16}, if base256 representation is used.

3.3 Signaling ZRTP (SDP attributes)

The basic idea of ZRTP is based on the approach to directly negotiate the shared key from endpoint-to-endpoint within the media channel. This means that a ZRTP implementation can act independently from the actual SIP signaling. Zfone, a tool that adds ZRTP support to any SIP client, just sniffs SIP messages in order to learn the negotiated media ports so that afterwards the application can operate on the right RTP traffic. The evolution of the ZRTP draft is still ongoing and in its latest version (originated from July 2007) Phil Zimmermann has complemented a section that considers *Signaling Interaction*. If considering the entire specification then this is the first apparent interface that integrates ZRTP media encryption and SIP/SDP signaling. Two additional SDP attributes have been defined for the purpose of transporting ZRTP elements within signaling:

a=zrtp-zid: zid-id

a=zrtp-sas: sas-string sas-value

The first mentioned attribute a=zrtp-zid has several fields of application as it could be used to generally indicate that ZRTP is supported, or to inform intermediary ZRTP devices that they must not operate as ZRTP endpoint. Additionally, this attribute provides the given hex-encoded ZID which is usually used to index the table containing cached key material for the new ZRTP negotiation (key continuity). As an additional security measure, the SAS value can also be transported in the SIP signaling referred to as a=zrtp-sas. By that it is possible for the SIP client software to compare the SAS on behalf of the user. Note that this check can only be relied on if the signaling path has end-to-end integrity protection. Since the SAS is actually a public value it does not need confidentiality protection when sent over the signaling path. One problem of this approach is that the a=zrtp-sas attribute will never be present in the initial SDP offer/answer exchange. The SAS value can only be calculated after the media session is set up and the ZRTP handshake is carried out. Thus, an additional SIP message after the successful call initiation is required to convey the a=zrtp-sas attribute to the communication partner. Best suited for this task would be the UPDATE message which is used to update parameters of session (re-INVITE). Another possibility would be to delay the SDP answer in order to await the ZRTP handshake as well as the SAS value which could then be added into the SDP answer. For this particular case the VoIP softclients must be able to establish a bidirectional media channel prior the INVITE cycle has been successfully finished.

If relating the *Signaling Interaction* to the introduced MitM attack we can say that an automated SAS comparison would further improve the security and defeat our approach, especially, if the annoying SAS verification is done automatically for each call. If integrity protection of the SIP signalling is used, Mallory is not able to manipulate the a=zrtp-sas attribute without being discovered.

3.4 Significance

ZRTP is one of the candidates for being the future standard for key agreement in VoIP environments. Mainly due to unsolved IPR issues ZRTP is currently not the preferred solution for IETF. Nevertheless, it is essential to examine its security features critically. Our demonstration shows that a MitM attack on ZRTP is quite easy if the end users are not cautious and do not compare SAS values. If they DO compare SAS values, an attack is theoretically possible, but in practice it is quite likely to be detected. Especially when talking to somebody with a familiar voice (including friends, colleagues, and relatives) a MitM is extremely unlikely to succeed.

4 How the attack works

Throughout this section we show how a MitM attack can be set up as well as the possibilities for manipulating the SAS procedure.

4.1 Getting into the Signaling Path

The first challenge for Mallory is to get into the signaling path between Alice and Bob. There are several ways to achieve this, depending on the scenario. In the chosen network environment (depicted in Figure 1), Mallory can carry out an ARP spoofing[2] (aka ARP request poisoning)[Whalen et al. 2001] attack between Alice and the SIP proxy server. All IP packets that would normally go directly between these two hosts are then directed to Mallory, who passes them on to the intended recipient. If the communication is not integrity protected she also has the possibility to *modify* packets in transit.

The attack also works if the SIP proxy server is not located within the LAN. It is then necessary to ARP poison the link between Alice and the default gateway or whatever router that is responsible for the SIP proxy's network. In any case, it requires that the attacker has access to the LAN (or more specific: the broadcast domain) of Alice, otherwise ARP poisoning will not work. In other scenarios it might be more promising to consider approaches like the installation of a rogue WLAN access point to get into the signaling path. Furthermore, it is not necessarily required to insert a bogus component into the signaling path, it is

[2] This is a technique to carry out MitM attacks by which spoofed ARP packets are sent to victims in order to pretend a new IP/MAC association. As a result all IP-packets are redirected to the attacker's MAC-address.

also conceivable to compromise an existing network element such as a SIP proxy server or a router by exploit an existing vulnerability or to cooperate with the operator of the SIP rpoxy.

For carrying out the ARP poisoning attack, we use the open source program *ettercap* [Ettercap]. This tools allows to get into the IP communication between two (or more) hosts in a very simple way. Executing the command

`ettercap -i eth1 -T -q  -M arp:remote` $/IP_X/\ /IP_Y/$

on an attack host is sufficient to listen into the complete conversation between the two hosts with IP addresses IP_X and IP_Y.

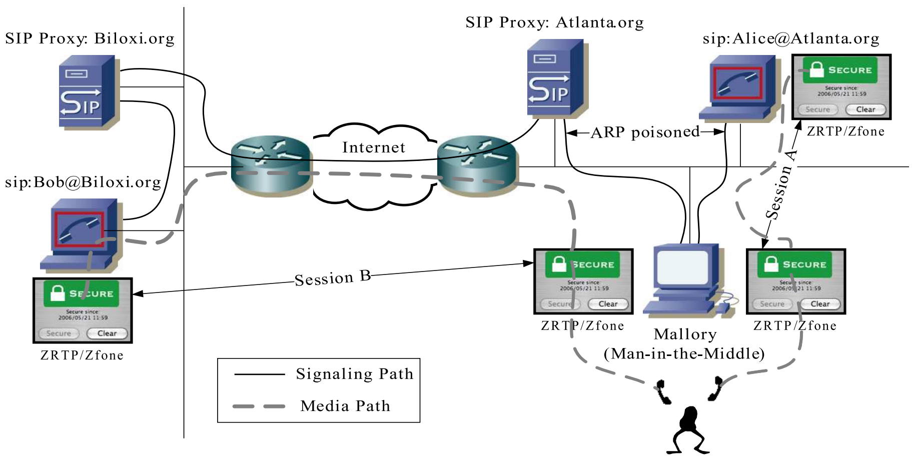

Figure 1: SIP signalling

4.2 Getting into the Media Path

Once Mallory is in the signaling path, she has the ability to modify packets in transit. In our given case she modifies SIP messages so that media data is diverted to her host. Basically, a SIP message consists of several protocol specific headers as well as various payload types. During the SIP session setup the Session Description Protocol (SDP) is used as payload. In particular, SDP contains attributes which signal the communication partner where the host expects to receive media data. Exactly these attributes are modified in transit to the IP address of Mallory and the given port respectively. The corresponding SDP elements are abbreviated with c (= connection information) and m (= media name and transport address).

`c=IN IP4` *IP_address_of_MitM_Host*
`m=audio` *media_port_of_MitM_Host* `0 RTP/AVP 0 101`

This replacement is done using ettercap *filters*, which allow to modify packets in transit when they traverse the MitM host. In general, these filters allow to search for packets matching defined header fields and to add, replace or remove information within these packets based on a simple programming language.

As a result, Alice and Bob unknowingly send their media data directly to the attacker Mallory. Note that both SDP messages (the initial one contained in - SIP - INVITE or '200 OK' as well the repyling one in the subsequent '200 OK' or ACK message) have to be modified in order to receive the audio stream from Alice to Bob as well as Bob to Alice.

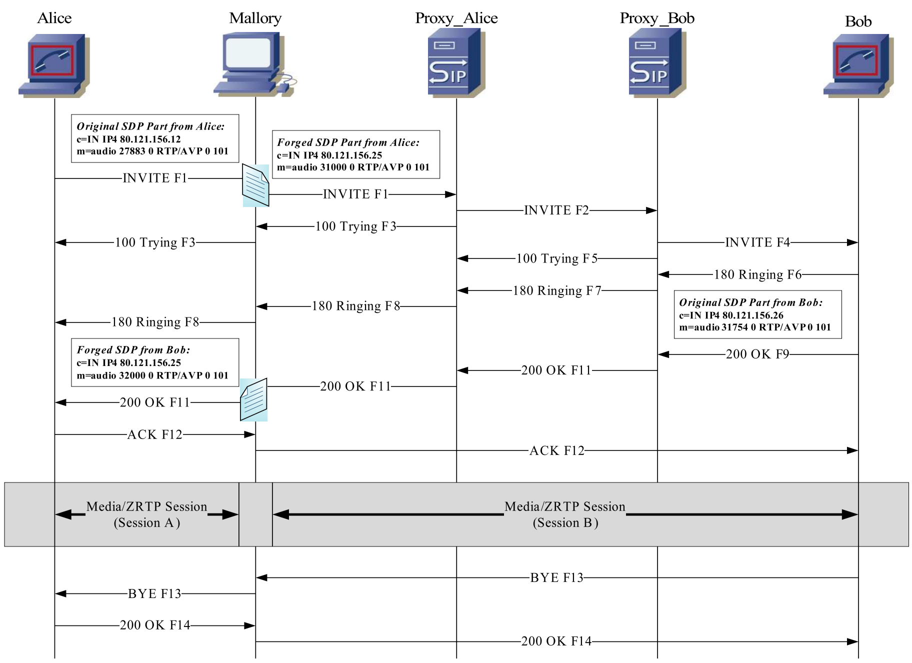

Figure 2: SIP signalling

4.3 Defeating the Short Authentication String (SAS)

Due to the tampering with the signaling, all media data from Alice and Bob is sent to Mallory. If no media encryption is used, Mallory simply forwards the RTP packets between Alice and Bob and can listen into the conversation.

In case Alice wants to start a ZRTP session, Mallory will receive the initial ZRTP message. Instead of forwarding this message to Bob, Mallory responds to Alice by herself and subsequently sets up a ZRTP session S_{Alice} (referred to as Session A). At the same time, Mallory initiates a second ZRTP session S_{Bob} (Session B) to Bob. In the end, two ZRTP sessions (S_{Alice} and S_{Bob}) are set up, with Mallory sitting in the middle and having access to the unencrypted audio. The session keys K_{Alice} and K_{Bob} are different since they are the result of two independent Diffie-Hellman key exchanges.

There are now a number of possibilities for Mallory how to relay the media data between S_{Alice} and S_{Bob}, which will be investigated in the following.

4.3.1 Direct Relay

The simplest method for solely eavesdropping an ongoing call is to take the decrypted RTP payload from one session, and to feed it into the other session where it gets encrypted again. During the handover between S_{Alice} and S_{Bob} Mallory can easily wiretap the unencrypted media.

Advantages:

- Little delay
- Normal interaction quality
- Attack can be fully automated, no human interaction by Mallory is required
- Communication partners hear each others *real* voice

Disadvantage:

- If the SAS comparison is carried out, the attack will be discovered

As long as Alice and Bob do not compare the SAS value, they are unable to find out that an attack is going on. However, since the two sessions are using different session keys and the SAS is derived from the session keys, an SAS comparison between Alice and Bob will fail and the MitM attack is uncovered.

4.3.2 Direct Relay + Imitating SAS

Similar to the previous method, but when the SAS comparison takes place during the conversation, the attacker replaces the spoken value with the one appropriate for the specific session. This is not an easy task, since the real voice has to be imitated. A different voice during SAS verification will most likely raise suspicion. Furthermore the timing is also challenging, since the insertion must be done in real time and must affect only the SAS value. The following methods are conceivable:

- *A human imitator tries to speak the SAS as authentic as possible.* This requires an attacker who has skills in imitating other people's voices. A training phase will be necessary for the attacker to become familiar with the victim's voice. Some voices will be easier to imitate than others. A male attacker will only be able to imitate male voices. To make this difficult attack more likely to be successful, the attacker can *whisper* the SAS, pretending this is done for security reasons. Whispered voices are almost not identifiable. However, this attack is very risky and quite likely to be detected by an attentive user.
- *Having a computer synthesize the SAS.* The authors are not aware of an existing voice synthesizing system that can *imitate* a particular human voice. Such a system will likely have to be trained in advance, which could be done by making bogus calls to the victim thereby collecting audio samples.
- *Collecting audio samples of all SAS values from the victim and replaying them in the correct order* [Rescorla et al. 2007]. If z-base-32 encoding is used, the attacker has to collect 32 different audio samples (24 letters and 8 numbers) to have a complete set of characters [O'Whielacronx 2007]. If the PGP word list is used for rendering the SAS, a number of 512 distinct audio samples has to be acquired.

Advantages:

- Little delay
- Normal interaction quality
- Communication partners hear each others *REAL* voice

Disadvantages:

- Independent of the applied method manual intervention is required
- Insertion of faked SAS comparison has to be timed very accurately
- In case Mallory collects SAS audio samples, she must have collected the SAS symbols of callee and caller in advance.

4.3.3 Direct Relay + Sidestepping SAS

After salutation, Mallory tries to be the first to actively request SAS comparison from Alice and Bob, acting as Bob or Alice respectively. By that, she can avoid having to say the SAS herself. She only has to confirm the "correctness" of the SAS of the communication partners. This attack is similar to the previous one, since Mallory has to imitate the voice of Alice and Bob when requesting the SAS comparison. However, an audio sample containing a request for an SAS comparison is much easier to obtain than a sample of all possible SAS values. This attack is also difficult in timing. During SAS comparison, the two audio sessions have to be isolated: The SAS responses coming from Alice or Bob must not be relayed to the corresponding communication partner. Once Mallory has confirmed the correctness of the SAS to both Alice and Bob, she has to switch to "Direct Relay" mode. If it takes Alice and Bob different amounts of time to say the SAS, this leads to a gap in the conversation that might rise suspicion. Also the situation right after Mallory has confirmed both SAS values could cause a strange interaction behavior as probably both victims are waiting for the other one to start communication.
Advantages:

- Little delay
- Normal interaction quality
- Communication partners hear each others *REAL* voice
- Mallory does not need any SAS value (she just confirms the read out)
- No audio samples are required which map the whole set of possible SAS values

Disadvantages:

- Manual intervention must be timed very accurately
- For both victims two audio samples are required where they are actively requesting the SAS read out as well as a kind of confirmation

4.3.4 Masquerade

In this case Mallory listens to Alice on S_{Alice} and almost concurrently repeats everything she heard for Bob in S_{Bob}. Moreover, Mallory can even control the conversation in an arbitrary way as she is also able to modify, omit or insert words. Once one side utters the SAS, Mallory has to react and to provide the appropriate SAS value for each call leg instead of just repeating it. Logically, this adaption has to be done for both directions.

Advantages:

- SAS comparison is not broken
- The voice speaking the SAS and the voice in the rest of the conversation is the same

Disadvantages:

- Works only if communication partners do not know each other's voice
- Rather large latency, due to delay when repeating spoken words
- As a result of the above: bad interaction quality
- Mallory has to listen and talk at the same time, which is a challenging task, especially when oral fluency has to be maintained. However, if Mallory has a partner, either direction can in principle be handled by a separate person in order to improve responsiveness.

Since this attack works only in scenarios where the communication partners do not know each other's voice, it is very limited in its use. An important scenario where this attack might be feasible is telephone banking. Since the voice of the teller is most likely unknown to the caller, this limitation of the attack is not a problem here.

4.4 Premise to break ZRTP: New ZID

Every instance of ZRTP-aware software or device has a unique, random 96-bit ZRTP-ID (ZID), which is generated at installation time. It is used as a reference for storing a number of values (such as a cryptographic hash of the used shared secret) into a local database. At the next call from/to the same ZID, these cached shared secrets are incorporated into calculation of the shared secret which leads to a form of *key continuity*. Mallory, who does not know these shared secrets, is not able to eavesdrop on this connection.

It is argued in the ZRTP draft that this mechanism should make it more difficult for Mallory to mount an attack:

> The continuity of the cached shared secrets make it possible for us to detect the MitM when he inserts himself into the ongoing relationship, as well as when he leaves. Also, if the attacker tries to stay with a long lineage of calls, but fails to execute a DH MitM attack for even one missed call, he is permanently excluded. He can no longer resynchronize with the chain of cached shared secrets.

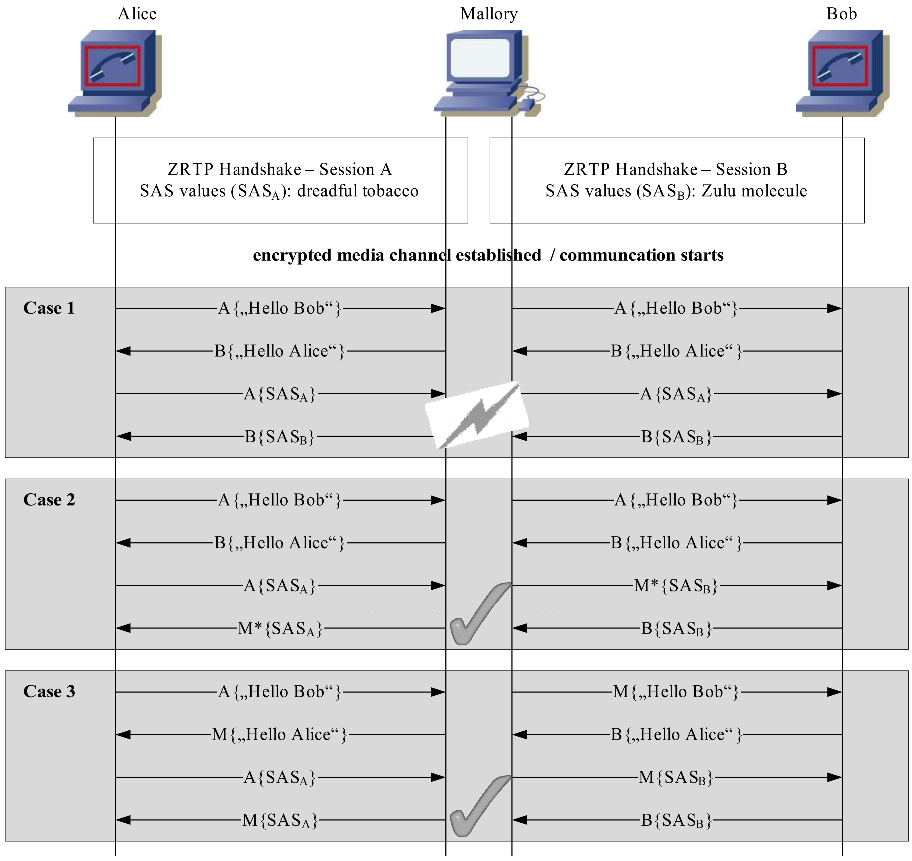

Figure 3: Attack on SAS comparison

To by-bass this security-measure, Mallory generates a new ZID for every single connection. As a consequence, each call is effectively treated as a "first call". Since the draft does not foresee any matching between a SIP-URI and the ZID, it will not raise suspicion if calls from an already known SIP-URI contain a different ZID. In fact there are several occasions where the relationship between a SIP-URI and ZID can change legitimately: Reinstalling the Zfone software or calling from a different device also results in a change of the ZID.

By that, Mallory can avoid that any historical ZRTP key material from previous calls between Alice and Bob is used for calculation of the shared secret. This means that with every new call the communication between Alice and Bob is reset in the initial state which enables Mallory to carry out a successful ZRTP handshake.

However, this tampering can be detected by an attentive user in certain cases. If Alice carried out a successfull SAS procedure and set the 'verified' flag in the Zfone application at a time where Mallory was not present, she might become suspicious as the checkbox went back to the 'unverified' state. This is caused by the fact that for the new ZID that Mallory presents the 'verified' flag is not set. However, this could also happen legitimately for the reasons mentioned above.

Complementarily, it would be conceivable that Mallory stores the negotiated ZRTP credentials for S_{Alice} and S_{Bob} so that she can continue wiretapping with already established key material. For our demonstration this is out of scope.

5 Practical Implementation

We implemented a prototype application to demonstrate the feasibility of a MitM attack on ZRTP. To avoid reinventing the wheel, we used as much readily available software as possible. For the end user application we used the open source SIP client *minisip* [Minisip]. It provides a command line interface, which is very convenient for testing purposes. Since minisip itself does not yet support ZRTP, we installed *Zfone* [ZFone], which is an application that adds ZRTP support to any existing SIP client. It does so by monitoring the network traffic of the host computer. If it detects a SIP call, it tries to carry out a ZRTP handshake with the communication partner in order to subsequently encrypt and decrypt the media stream. In fact, the Zfone application is deployed as a kind of security shim between the VoIP softphone and TCP/IP stack.

One of our attack tools is based on the publicly available ZRTP library (lib_zrtp module) provided by Phil Zimmermann and consists of several components. This library is a command line tool written in C++ that can carry out a ZRTP handshake with a communication partner. It is capable of listening on a given network socket, decrypting incoming ZRTP data and sending out the decrypted RTP data on to another predefined network socket, typically a local one. Conversely, it can also receive RTP data from one network socket, and send it out in encrypted form on another socket.

A Perl-script is used to receive the unencrypted RTP data from a network socket and to play it on a sound device. It also reads audio data from a sound source, constructs RTP packet from it, and sends it out on a network socket.

A simple GUI application also programmed in Perl is used to glue the above mentioned parts together. It listens on the network interface, and as soon as it sees a SIP invite message and the corresponding '200 OK' it starts the required processes. The GUI shows both generated SAS values of the two ZRTP sessions which are used by the attacker when doing the SAS procedure. The tool allows Mallory to select one of four "audio modes" which are varying in the manner how earphones and microphone are combined to handle both directions:

send to A : everything Mallory says is sent to Alice

send to B : everything Mallory says is sent to Bob

put through : audio from Alice is directly passed on to Bob and vice versa

mute : the microphone is deactivated

To make it possible to mount the attack with a conventional laptop which has only one audio input (microphone) and one audio output (headphone) and still make it possible to distinguish the two audio channels (one from Alice and one from Bob), we did the following: We configured the audio devices in such a way that Alice can be heard only in the left earphone and Bob only in the right earphone. By that, Mallory can always distinguish between the two voices, even if she is not familiar with them. This would not be possible if we simply mixed them together into one audio stream that can be heard on both ears.

5.1 Premise for a Repeating Attack: Unknown Voice

In the Repeating Attack, Alice and Bob are not hearing each other's voice. In fact, both are hearing Mallory's voice who is relaying everything that is said in both directions. As a result, both parties must not know the voice of each other. If any of them does, the attack is discovered right at the beginning. This requires that either Mallory must catch the first call between both parties or the attacker assumes that they forgot the partners sound of voice.

As already mentioned earlier, a classical but very critical example for a call where you normally not know the voice of your communication partner is if you call your bank for telephone banking. The teller picks up the phone and guides you through your transaction including the authentication with your individual credentials. If Mallory mounts a successful MitM attack she gets hold of these credentials.

5.2 Experimental Results

We carried out the "Repeating" attack in our lab in order to get an understanding of how this attack can work in practice. For that purpose, we placed three people (Alice, Bob and Mallory), each one equipped with a laptop and headphones, in three different rooms. The computers of Alice and Bob had the SIP client minisip as well as Zfone installed. Mallory, acting as the Man-in-the-Middle carried out the "Repeating" attack as described earlier. In this section we focus on issues regarding the interaction quality of the conversation that is allegedly going on between Alice and Bob. In reality, both of them of course only talk to Mallory.

After the call setup is completed, Alice's Zfone application shows a large green box with the label "Secure - AES 128" and a closed padlock. Her SAS values are for example "goldfish inferno". Bob sees the same green label, but since a MitM attack is going on, he gets a different pair of words, for example "scenic universe". The user interface of Mallory's attack tool (see Section 5) displays both word pairs. Everything that Alice says can be heard by Mallory on her left headphone, everything Bob says she hears on the right, which makes it easier for her to distinguish between the two talkers. When Alice starts talking, Mallory presses the "a" key on her computer and speaks everything she hears into her microphone. Bob listens to Mallory talking, thinking that she is Alice. As soon as Bob starts to talk, Mallory presses the "b" button and repeats everything, which is now heard only by Alice. Mallory has to be very careful to press the correct button, otherwise the communication partner will get suspicious.

When Alice wishes to do SAS authentication from Bob ("What is your SAS?"), Mallory repeats this request to Bob. Bob will answer by saying "My SAS is *scenic universe*". Instead of repeating this, Mallory says to Alice: "My SAS is *goldfish inferno*".

A major problem of the "Repeating" attack is that it adds latency to the conversation, since Mallory needs some time for listening and repeating. Recommendation G.114 of the International Telecommunication Union (ITU) considers an end-to-end delay of 150ms as "acceptable" for voice applications. This value of course also includes other delays introduced by the coder, packetization, serialization, encryption, buffering, the network etc. In our tests, this value was exceeded significantly. We do not state any absolute values, since they are highly dependent on the skills of the speaker. We expect that people which are trained in listening and speaking at the same time (like simultaneous interpreters) will reach much better results. Another improvement is to have actually *two* people carrying out the attack. One is responsible for repeating everything Alice says, the other one only for repeating what Bob says.

Problems can also arise if the caller expects to talk to a female person, but the attacker's voice is male or vice versa. Mallory can try to explain this fact to the caller ("Mrs. X is on holiday, can I help you?"), but this can easily impede the synchronization between the two conversations.

6 Zfone Usability

As mentioned previously, *Zfone*, the reference implementation of ZRTP, was deployed as a shim security layer collaborating with the VoIP softphone *minisip*. Both applications represent individual graphical interfaces to the users so that SIP signalling as well as security can be treated separately. The general usability of minisip and unsecure call establishment is out of scope. Hence this section deals exclusively with Zfone and security flaws based on its usability weaknesses.

6.1 SAS Verified Flag

The oral SAS comparison is a measure that requires active human interaction. The two decisive benefits of this approach are that the comparison happens actually "out-of-band" and the human behavior generates additional randomness concerning when and if the SAS is compared. In order to avoid that users have to carry out the SAS procedure at every single call, the ZRTP draft defines the so called *SAS verified flag*. This flag can be set by the user to indicate that the SAS comparison has been successfully carried out and hence all subsequent calls will be secure based on the key continuity property of ZRTP. This means that once a connection is successfully approved as indicated by the SAS verified flag, the security of future calls to the same recipient is guaranteed and further SAS comparisons are unnecessary. The SAS verified flag is stored as a property value coupled with the ZID and the retained cached key material used for key continuity. Zfone implements the SAS verified flag as a checkbox which must be confirmed by clicking into the box. A set checkmark signals that the channel to this unique ZID/partner has been approved and is secure. However, the checkmark disappears if the same user calls from another communication device. This is due to the fact that a user can have several communication devices with different ZIDs but using the same SIP identifier (SIP-URI). In such cases Zfone relies only upon the ZID as well as the corresponding established security state and disregards the SIP-URI.

The usability problems with this kind of representation are the following:

1. After some time of regular Zfone usage the check box indication will increasingly blur, meaning the perception if the checkbox is marked or not will disappear, especially if recognizing that an already verified friend or its SIP-URI is calling. Based on this psychological fact MitM attacks can be mounted relatively easy. Assume Mallory gets into a call between Alice and Bob, which have already set the verified flag in a previous session. During this MitM attack the SAS checkmark disappears because the attacker's ZID, which is used for identification of the Zfone instance, is unknown to Alice and Bob. However, the question is if this incident will be detected and raises suspicion. The likeliness that such an attack remains uncovered increases with the number of daily calls (negligence), the overall period of usage (by habit), and the relationship between the calling parties (familiarity).

2. There is no correlation between ZID and SIP-URI. Using the same SIP-URI coupled with an unverified ZID is treated as if it was a first call. The security measures are "bid down" which results in a situation where both Zfone instances do not share any secrets. The usability problem in this scenario is that the user is not informed that an apparently known person uses a new ZID. This situation can be caused by a new communication device or in the

worst case by an ongoing MitM attack. Indeed, Zfone indicates the usage of an unverified ZID by an unset checkbox, however if the calling SIP-URI has been already verified the user is tempted to neglect the missing checkmark. Because of this we recommend to additionally notify the user about the current circumstances and to request an aware acknowledgment.

Requiring user acknowledgement for the situation where a known SIP-URI calls using an unknown ZID would solve the above mentioned problems: The correlation between ZID and SIP-URI would be stronger, and the weak "SAS verified" indication would be strengthened. From the usability point-of-view, an asynchronous notification by a popup-window provides more information content compared to a set/unset checkbox.

Furthermore, the ZRTP draft claims:

> "If the SAS matches, it means no attacker was present for any previous session since we started propagating cached shared secrets, because this session and all the previous sessions were also authenticated with a continuous lineage of shared secrets."

From our point-of-view this statement is somewhat weak as it is unreproducible if during the previous sessions which were not verified a Man-in-the-Middle was present. Principally, such an attack could happen as described above when using an unknown ZID that causes security bid down.

6.2 Misleading "Secure" indication

In the middle of Zfone's graphical user interface a status box informs the user of the current ZRTP state. It indicates whether ZRTP is currently used or not and during the key negotiation it displays the progress. A successful ZRTP handshake is represented by a green box which contains a closed padlock and the label "Secure". For the user this form of representation creates a misleading perception of security. For sure, the connection between this ZRTP endpoint and the corresponding one is encrypted by a negotiated shared key, however the communication is not authenticated and consequently the "Secure" indication is improper. In case the SAS comparison has been carried out successfully then Zfone can correctly inform that this and all following sessions are "Secure", but prior to this verification only the genuine current state should be expressed using for instance the labelling "Encrypted/Unauthenticated".

7 Conclusion

Because of its peer-to-peer nature, ZRTP solves the problem of key agreement between VoIP endpoints very elegantly without relying on a centrally managed

Public Key Infrastructure (PKI). Any two endpoints can communicate securely just by setting up ZRTP-aware softphones or hardware devices. "The network" in between does not have to provide any means, it as actually ZRTP-agnostic. This simplicity sets the entry barrier very low. Since the key agreement is truly end-to-end, the communication partners do not have to trust any infrastructural component in between. We showed that it is extremely important to carry out a SAS authentication when contacting somebody for the first time and checking the SAS verified flag for connection with which the SAS has already been carried out. Without doing this, a MitM attack is quite simple to carry out. We also demonstrated that in certain scenarios, especially when the communication partners are not familiar with each other's voice, attacks on the SAS procedure are possible. However, these attacks are difficult to carry out and risk of detection is quite high.

If recapitulating the three presented cases in terms of "How to handle SAS authentication?" then Repeating has currently both practical and technological the best chances of success. From the theoretical point-of-view Direct Relay + Imitating SAS presents also a promising approach, but today we are still faced with problems concerning the timing as well as the SAS imitation/synthesizing. In principle, the third case Direct Relay has only academical character when users are essentially respecting the security guidelines.

Ultimately, ZRTP is a brilliant way to solve the key exchange topic within the media channel respecting the peer-to-peer paradigm. However to guarantee optimal and continued security the user is asked to bring along attention and awareness when deploying ZRTP. This includes protocol aspects like the compelling SAS comparison for the first call, implementational details such as the usability of the SAS verified flag as well as a kind of misbehavior sensitivity.

References

[Baughter et al. 2004] Baugher, M., McGrew, D., et al.: RFC3711, The Secure Real-time Transport Protocol (SRTP), IETF, March 2004

[Diffie and Hellman 1976] Diffie, W., Hellman, M. E.: New directions in cryptography, IEEE Transactions on Information Theory, IT-22(6):644-654, November 1976

[Ettercap] Ettercap: A Suite for Man in the Middle Attacks on LAN, http://ettercap.sourceforge.net

[Cryptophone] GSMK CryptoPhone, http://www.cryptophone.de/

[Handley et al. 2006] Handley, M., Jacobson, V., Perkins, C.: RFC 4566, SDP: Session Description Protocol, IETF, July 2006

[Juola and Zimmerman 1996] Juola, P., Zimmermann, P.: Whole-Word Phonetic Distances and the PGPfone Alphabet, Proceedings of the International, Conference of Spoken Language Processing, 1996

[Minisip] Minisip: A Multi-Platform Open Source SIP client, www.minisip.org

[PGP Wordlist] PGP Wordlist, http://en.wikipedia.org/wiki/PGP_Words

[Rescorla et al. 2007] Rescorla, E., McGrew, D., Fischl, J., Tschofenig, H.: DTLS-SRTP, Proceedings of the 68th Internet Engineering Task Force, Prague, Czech Republic, 2007

[Robin and Schwartz 2006] Robin, J., Schwartz, A.: CS 259 Security Protocols - Final Project Report, Analysis of ZRTP, Standford University, March 2006

[Rosenberg et al. 2002] Rosenberg, J., Schulzrinne, H., et al.: RFC3261, SIP: Session Initiation Protocol, IETF, June 2002

[ZFone] The Zfone Project, www.zfoneproject.org

[Whalen et al. 2001] Whalen, S., et al.: An Introduction to ARP Spoofing, rootsecure.net, April 2001

[O'Whielacronx 2007] O'Whielacronx, Z.: Human-oriented base-32 encoding, `http://zooko.com/repos/z-base-32/base32/DESIGN`

[Zimmerman 2008] Zimmermann, P., Johnston, A., Callas, J.: ZRTP: Media Path Key Agreement for Secure RTP, draft-zimmermann-avt-zrtp-06, March 2008