Secure Voice over Public Internet

The project "Secure Voice over Public Internet", SVoPI, investigates new approaches to deliver at the same time secure and user friendly voice services over public Internet. The feasibility of voice communication with key exchange from end to end without the use of a public key infrastructure is to be demonstrated with Softphones. The goal is to start the clients from a Web Browser without the need to install manually a program on the local computer. The advantage of our solution is that the client can be used by non-professionals in the area of security, since no X.509-certificates have to be installed.

Peter Gysel, Christoph Stamm, Markus Zeiter | peter.gysel@fhnw.ch

Voice communication over public internet becomes more and more popular, even security aspects are rarely regarded at all. Nevertheless, secure voice over public internet (VoIP) is technically possible with X.509 certificates. Since handling of these certificates is quite complicate, secure VoIP from client to client is not really wide spread. Several attempts were made to overcome that problem of acceptance. The company *PrivaSphere*, for example, successfully offers a service that enables exchange of e-mails with security from end-user to end-user, which is based on a trust infrastructure. This service also serves as basis for the secure messaging platform *IncaMail* by the Swiss Post [Incma]. However, things are even more complicated with voice, because two channels with different characteristics have to be handled: a control channel based on the Session Initiation Protocol (SIP) and a voice channel with real time conditions.

This paper is based on ongoing research and development on secure VoIP in our institute. The goal is to show how a new combination of well known fundamental techniques results in a secure VoIP infrastructure that aims two main targets:

- *user friendliness:* it means that fingerprints or complicated management of certificates are unnecessary. And it also means that a common IT user, new to the subject, should be able to bring up the service within one or two hours. No software installation by hand should be needed.
- *end-to-end security:* in contrast to hop-to-hop security it means that only the two end users have the key for the encryption of the voice stream.

In our project[1] we use client-server architecture with a SIP proxy and a trust server to validate clients. The SIP proxy handles phone calls and is provided from the company *e-fon* which is a SIP supplier and VoIP provider. It operates virtual private branch exchange (PBX) at the customer or centrex services (the virtual PBX at *e-fon*). Primarily it offers VoIP with hardware phones. *PrivaSphere* provides us a trust server and customizes it to the desired requirements. Our tasks are evaluating a SIP client and developing a so-called trust client, which offers interfaces to interact with a trust server.

The paper is structured as follows: after an introduction to VoIP and to security, we describe our approach of secure VoIP in more detail and explain a timing problem concerned to our approach.

Abbreviation	Description
ARP	Address Resolution Protocol
IPSec	IP Security
MAC	Media Access Control
MAC	Message Authentication Code
MIKEY	Multimedia Internet KEYing
MITM	Man-in-the-middle
PKI	Public Key Infrastructure
PRF	Pseudo-Random Function
PSTN	Public Switched Telephone Network, classical telephony
RTP	Real-time Transport Protocol
RTCP	Real-time Transport Control Protocol
SDP	Session Description Protocol
S/MIME	Security/Multipurpose Internet Mail Extension
SIP	Session Initiation Protocol
SIPS	SIP Security also called SIP over TLS
SRTP	Secure RTP
TEK	Transport Encryption Key
TGK	TEK Generation Key
TLS	Transport Layer Security
VoIP	Voice over IP

Introduction to VoIP

Voice over IP (VoIP) allows phone calls on computer networks on the basis of the *internet protocol* (IP). Compared to the classical telephony offers VoIP a higher voice quality provided that the connection and the available bandwidth are stable, many additional functions, cost reduction of the

1 CTI Project "Secure Voice over Public Internet", 10830.1 PFES-ES

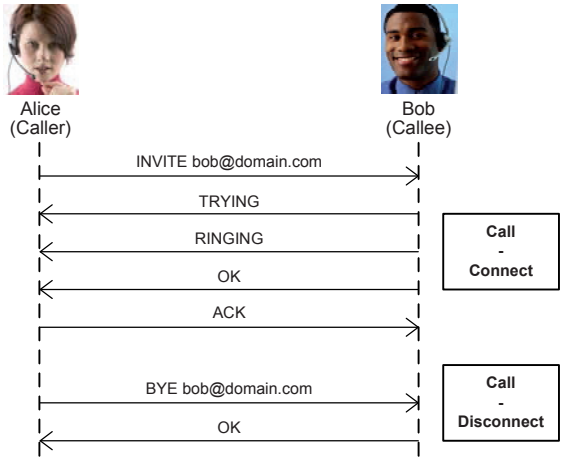

Figure 1: Call setup in a direct connection between two communication participants.

Start line:	INVITE sip:8495302002@192.168.2.25 SIP/2.0
Header:	Via: SIP/2.0/UDP 192.168.3.250:5060; branch=1 From: sip:8495305005@192.168.2.25;tag=29ae1249 Max-Forwards: 70 To: sip:8495302002@192.168.2.25 Call-ID: 48c7df2a9b4@myvoip1 Cseq: 1 INVITE Contact: sip:8495305005@192.168.3.250 Content-Length: 202 Supported: 100rel Content-Type: application/sdp
Space line:	
Body:	v=0 o=Anonymous 1234567890 1234567890 IN IP4 192.168.3.250 s=SIGMA is the best c=IN IP4 192.168.3.250 t=0 0 m=audio 6006 RTP/AVP 8 3 0 a=rtpmap:8 PCMA/8000 a=rtpmap:3 GSM/8000 a=rtpmap:0 PCMU/8000

Figure 2: Example of a request message.

infrastructure for business companies and the openness for process and application integration. The data protection as well as the security in general can currently not be guaranteed. With the changeover of classical telephony to VoIP telephony, as a result of the universally usable communication media of IP networks, telephony loses its status of a closed system. Of course, IP telephony obtains not only advantages but also disadvantages as for example spam or viruses etc.

Before two parties can communicate, they need to agree on different parameters such as the audio codec. These parameters as well as error handling and set-up and tear-down of the call are transmitted over a separate signaling channel, while the digitized voice data are transmitted over a voice channel. This concept with a separate signaling channel is not only applicable in VoIP but also in other multimedia applications.

The signaling channel uses the *Session Initiation Protocol* [SIP] and *Session Description Protocol* [SDP]. SIP establishes, terminates or modifies VoIP sessions between two or more IP phones. Compared to the H.232 protocol it offers more flexibility and opportunities for the connection management, and is especially designed for IP connections [Hvss]. SDP is to negotiate and determine various session parameters, such as audio codecs, because not all devices support the same parameters.

As shown in Figure 1, several messages are exchanged for a connection establishment: for example *INVITE* is a request message and *TRYING* is a response message. Usually a caller sends request messages and a callee answers with response messages. Figure 2 shows an example of such a request message. Each message is text based and contains a start line which indicates the specific message type, a header, a blank line, and the body. SDP is included in the message body of SIP. Because reliability is quite important during the connection establishment, these messages are usually transferred with TCP on port 5060 or 5061.

The voice channel – as its name suggests – is used to transmit the digitized voice. It makes use of the *Real-time Transport Protocol* [RTP] and the *Real-time Transport Control Protocol* [RTCP]. While RTP carries the media streams (e.g. audio and video) or out-of-band events signaling, RTCP is used to monitor transmission statistics and *Quality of Service* (QoS) information. Both protocols are based on UDP. In contrast to TCP, UDP is connectionless and usually quicker than TCP, but less reliably, because it does not wait for lost and late arriving packets. Single packet losses are not a big issue, as they contain few data and our language has much of redundancy. The real problem is the loss and delay of several consecutive packets. This can heavily reduce voice quality.

Today's VoIP network topologies are either direct connection or client-server environments. In a direct connection the clients communicate with each other without any server. So they have to know the IP address and port numbers of their counterparts. The session handling is done with the earlier described protocols. In a client-server environment a server handles client registrations, forwards messages, and returns contact address information to the clients. In case such a server creates new SIP messages based on the requirements of the communication session, it is called a SIP proxy.

Introduction to Security

The goal of VoIP encryption is to ensure that no attacker can spy on an encrypted phone call. In a VoIP infrastructure without encoding mechanisms an attacker can listen to a telephone call and can record it with little effort. In Figure 3, for example, the attacker uses *Address Resolution Protocol spoofing* [ARP], which is usually followed by network package analysis (more information is found at [Arpsp], [Caiab] or [Xarp]). The *Man-in-the-middle* (MITM) attacker sends a malicious

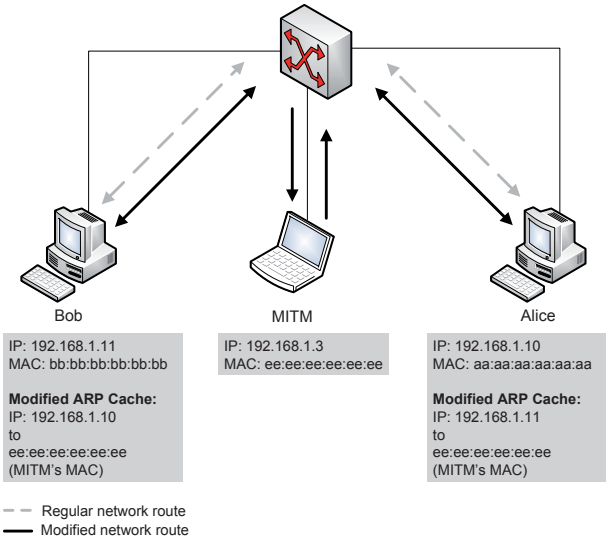

Figure 3: ARP-Spoofing

ARP message to both Alice and Bob. These messages do not include its own, but Bob's or Alice's IP address, respectively. So, Alice's ARP cache contains now MITM's instead of Bob's *Media Access Control* (MAC) address and Bob's ARP cache contains MITM's instead of Alice's MAC address. Thus, Alice will send further packets by mistake to MITM. A good playing MITM will then transmit the received packets from Alice or Bob to the intended recipient in order to not be discovered. It therefore acts as a proxy and can view the contents of each package.

As soon as an ARP spoofing has been completed successfully, the voice communication can be analyzed and recorded with network analysis software, e.g. Wireshark [Wiresh]. For a MITM the following information might be interesting (Figure 4): in SIP packets: SIP-Uniform Resource Identifier, IP address, Ports, SIP proxy, etc.; in RTP packets: audio data.

The fact that with advanced ICT knowledge but little effort conversations can be monitored show us that serious VoIP conversations should be encrypted. To fully secure a VoIP communication, both the signaling and the voice channel must be encrypted. Usually diverse encryption technologies for voice and signaling channels are used because of their different channel structure and requirements. Since a SIP proxy needs to know the information in the header of the SIP message, a hop-to-hop security connection is needed for the signaling channel. In contrast, an end-to-end security connection for the voice channel is preferred, in order to prevent the SIP proxy from voice spoofing.

In the following paragraphs we explain five popular encryption technologies for VoIP. Most of them are application layer technologies, but IPSec works on the network layer of the TCP/IP model. We start with four technologies for the signaling channel:

- *HTTP-Digest-Authentication* [Hda] is a challenge-response technique. The simplest example of a challenge-response technique is password authentication, where the challenge is asking for the password and the valid response is the correct password. A unique number (nonce, also known as *number used once*) is sent as a challenge for identifying the counterpart. The participant responds with an MD5 hash based on the user name, password, nonce, and the address of the SIP server. Except the password all information will be transferred in plain text.
- *S/MIME* [Smime] is a standard for encryption and signing of MIME-encapsulated e-mail through an asymmetric crypto system. It also can be applied to encrypt the signaling channel. Only the MIME body of the SIP message will be encrypted and authenticated. The header of the SIP message will be transmitted in plain text. An end-to-end security can be achieved. With S/MIME tunneling it is possible to encrypt and authenticate the header of the SIP message. This means, first the complete SIP message is encrypted and packed into a body and second a new header is added to this body.
- *Transport Layer Security* [TLS] encrypts the segments of network connections at the application layer to ensure secure end-to-end communication at the transport layer. Only a hop-to-hop encryption makes sense, because at an end-to-end security the involved hops are not

```
⊞ User Datagram Protocol, Src Port: weblogin (2054), Dst Port: sip (5060)
⊟ Session Initiation Protocol
  ⊟ Request-Line: INVITE sip:43@192.168.0.1;user=phone SIP/2.0
      Method: INVITE
    ⊞ Request-URI: sip:43@192.168.0.1;user=phone
      [Resent Packet: False]
  ⊟ Message Header
    ⊞ Via: SIP/2.0/UDP 192.168.0.42:2054;branch=z9hG4bK-jx548jihh6oa;rport
    ⊞ From: "42" <sip:42@192.168.0.1>;tag=2ky5lejx3m
    ⊞ To: <sip:43@192.168.0.1;user=phone>
```

Figure 4: The INVITE message of the SIP protocol supplies information to MITM about a caller and some information about the callee, too.

Very simple example

Prime number: $p = 13$
Prime root: $g = 7$

Alice:
$a = 3$
$A = g^a \bmod p = 7^3 \bmod 13 = 5$

Bob:
$b = 6$
$B = g^b \bmod p = 7^6 \bmod 13 = 12$

Key calculation:
$K = B^a \bmod p = (g^b \bmod p)^a \bmod p = g^{ab} \bmod p = (g^a \bmod p)^b \bmod p = A^b \bmod p$
Alice: $K = B^a \bmod p = 12^3 \bmod 13 = 12$
Bob: $K = A^b \bmod p = 5^6 \bmod 13 = 12$

able to process the SIP messages. SIP in combination with TLS is then called *SIPS* [Sips].

- *IPsec* [IPsec] is a security protocol to secure communication over IP networks. IPsec works directly on the network layer of the TCP/IP protocol stack and can therefore be used to encrypt the voice channel. The protocol grants confidentiality, authenticity and integrity of the data.

In some cases a *HTTP digest authentication* is not appropriate, so TLS, IPsec, or S/MIME would be possible candidates. TLS has caught on and nowadays it is mandatory for the encryption between two and more SIP proxies in a hop-to-hop environment, because they exchange more confidential data between each other than between itself and a SIP client. However, TLS between SIP proxy and client is recommended and TLS is already integrated in some hardware- and software-based SIP telephones.

For the encryption of the voice channel *Secure Real-Time Transport Protocol* [SRTP] and again IPSec are common encryption technologies. SRTP offers an opportunity for the transfer of protected real-time payloads based on RTP. For the symmetric encryption of a multimedia connection a so-called *master key* must be distributed in advance between the communication participants on a potentially insecure channel. As a transport medium for this key the SDP payload, a part of the SIP message, is prescribed in the *Request for Comments* (RFC) of SIP. There are basically two standardized key exchange techniques which can be used for VoIP: (i) in the *SDP Security Descriptions* the random generated master key will be added in plain text; (ii) in the *Multimedia Internet KEYing protocol [MIKEY]* the master key can be encrypted and authenticated.

- *SDP Security Descriptions:* The SRTP master key is transmitted in plain text as a SDP parameter in the SIP message. This variant is preferred from commercial vendors such as Cisco, Microsoft etc. because it easily allows lawful inspection. Law enforcement authorities, but unfortunately also unauthorized people, might tap the operated SIP proxy of the VoIP providers and read out the session key in plain text. Thus, they can easily perform monitoring of the voice communication.
- *Multimedia Internet KEYing protocol:* MIKEY allows a real end-to-end security and provides the following key exchange procedures: *Pre-shared Key*, *Public Key Encryption*, *Diffie-Hellman Key Exchange with a digital signature*, and *Diffie-Hellman Key Exchange with Hashed Message Authentication Code*. The Diffie-Hellman key exchange is a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. In Figure 5, firstly, Alice and Bob agree on a prime number p and a primitive root $g \bmod p$ with $2 \leq g \leq (p - 2)$. A primitive root modulo p is a number g with the property that any number coprime[2] to p is congruent to a power of g mod p [Priro]. Secondly, Alice and Bob each consider a secret number a and b, respectively. Alice then computes $A = g^a \bmod p$ and sends g, p and A to Bob. Bob then computes $B = g^b \bmod p$ and sends B to Alice. In order to get the secret key Alice computes key $K = B^a \bmod p$ and Bob computes key $K = A^b \bmod p$. The key K is further used to calculate the master key for SRTP. In the MIKEY protocol the key K is called *TEK Generation Key* (TGK) and the master key is called *Transport Encryption Key* (TEK). The TEK is derived from the TGK trough a *Pseudo-Random Function* [PRF].

The difficulty of Diffie-Hellman is to avert a MITM attack. ARP spoofing allows an attacker to modify data packets and thus changing the Diffie-Hellman messages. Because these Diffie-Hellman messages contain all the parameters g, p, A

2 Two integers **a** and **b** are said to be coprime if they have no common positive factor other than 1.

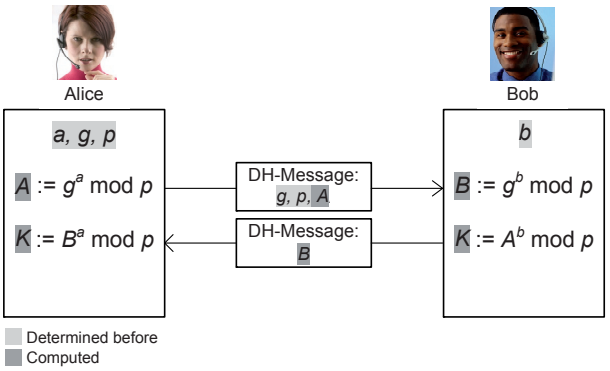

Figure 5: Procedure of the Diffie-Hellman key exchange.

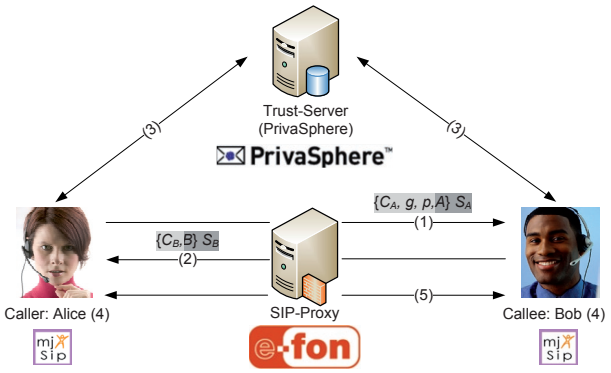

Figure 7: Structure of our product and the communication paths between the individual components

or B, an attacker can intercept them, compute $Z = g^z \bmod p$, where z is a random number, and send Z to Alice and Bob (Fig. 6). So, the Diffie-Hellman key exchange will be done twice: once between Alice and MITM to compute key K_A and once between MITM and Bob to compute K_B.

In order to prevent a MITM attack, the messages must be authenticated using digital signatures or message authentication codes. Digital signatures usually involve certificates issued by a *Public Key Infrastructure* (PKI), which can create, distribute, validate, and revoke digital certificates. Revoking a compromised certificate prevents its further usage. A certificate authority (CA), for example VeriSign [Veris], is part of a PKI and publishes keys bound to a given user. This is done using the CA's own key, so that trust in the user key relies on one's trust in the validity of the CA's key. The purchase of such a certificate is time consuming and expensive. Instead this official X.509 certificates can also self-signed certificates be utilized. A self-signed certificate is an identity certificate that is signed by its own creator. It can immediately be created or renewed for free. The main disadvantage of such self-signed certificates is they are not trustworthy. They cannot be revoked if a private key has been compromised. Therefore an attacker is able to spoof an identity.

Finally we can summarize, *Diffie-Hellman Key Exchange with a digital signature* of MIKEY offers a maximum of security and is a suitable option for a user-friendly application.

Our Approach

In Figure 7 we show our approach of secure VoIP and the interrelationship between the following components: SIP client, SIP proxy, and trust server. As a SIP client we use the open source project MjSIP [Mjsip]. It is a small project and provides only basic functionalities like SIP and RTP stack. The trust server is basically a database that contains all registered users and their reference certificates. Our trust server architecture is based on the established trust infrastructure from *PrivaSphere*. The access to this service is secured by TLS and a login. We use HTML forms to make requests to the server. The trust server has to be physically separated from the SIP proxy, because it should not be possible for the SIP proxy to listen to any call.

We use SIPS to secure the signaling channel because TLS is established and recommended in the RFC of SIP; the voice channel is secured by SRTP. The exchange of the *Master Key* for the SRTP connection is done using the MIKEY method *Diffie-Hellman Key Exchange with a digital signature*. Its digital signature is verified with

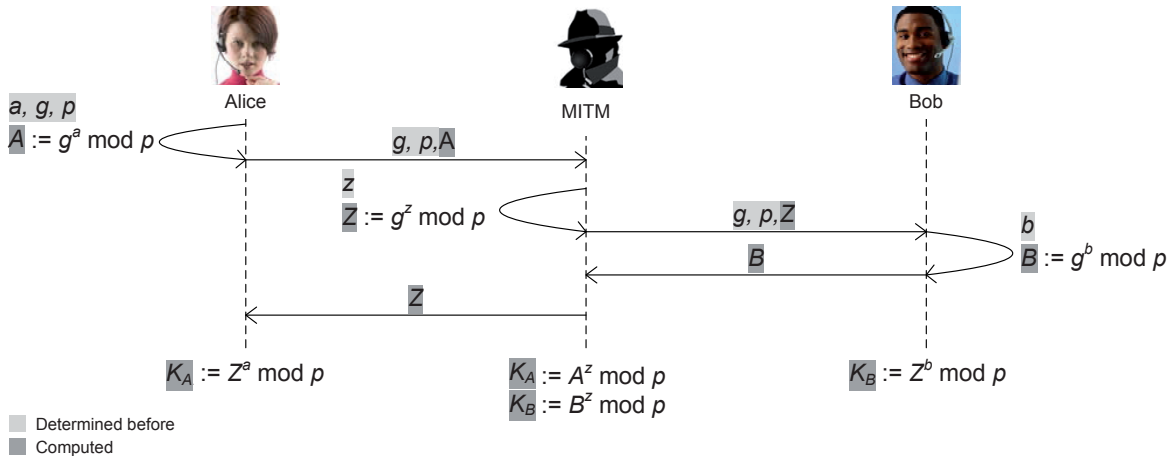

Figure 6: MITM attack on Diffie-Hellman key exchange

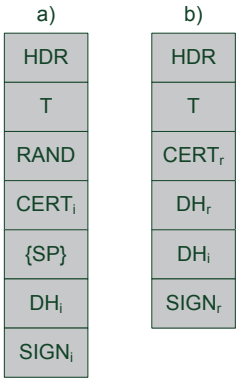

Figure 8: a) shows the structure of the initiator message of the presented MIKEY protocol, and b) the response message

a self-signed X.509 certificate, which itself has been authenticated at the trust server.

1. Now, that we have explained the general structure of our approach, let us have a closer look how two registered users, Alice and Bob, initiate a phone call (Figure 7):
2. The caller Alice starts the key exchange with the initiator message of MIKEY. In the CERT payload of this message we transfer the self-signed certificate from Alice, which has also been deposited at the trust server. This message (Fig. 8a) is part of the body of the SIP INVITE message (precisely in the SDP, Fig. 9) and it is base64 encoded.
3. The callee Bob replies with the response message from MIKEY. It is similar structured as the initiator message and it contains the self-signed certificate from Bob in the certificate payload, which has also been deposited at the trust server. This message (see Figure 8b) is part of the body of the SIP OK message.
4. After receiving the MIKEY messages the validation process starts: Alice and Bob both validate her/his counterpart at the trust server. Therefore, he or she needs the login name and the certificate of his or her counterpart, and additionally the own login data. With the own login data the trust server is able to authenticate the requester without setting up a session. The actual validation of the counterpart occurs then by comparing login name and certificate with the stored data. The trust server responds with the result of the validation and the status of relationship. When the conversation participants have already built a trustful relationship, then the trust server sends back a confirmation of it. Otherwise it generates a so-called voice unlock code (VUC), i.e. a one-time-password, and sends that back to the requester. The requester then communicates this VUC out-of-band (e.g. SMS, Fax etc.) to his/her conversational participant, while he/she sets up a session to the trust server and transfers the received VUC to it. Finally, the trust server creates a trustful relationship between both conversational participants.
5. Alice and Bob calculate the master key for the SRTP protocol with the Diffie-Hellman parameters. This calculation is independent of the validation process and can be performed in parallel with the validation.

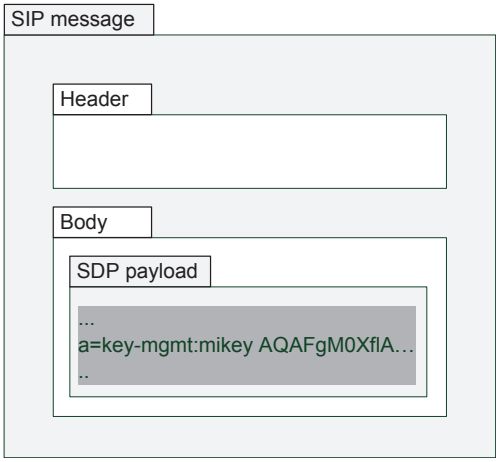

Figure 9: The MIKEY message „mikey AQ ...“ is part of a session description with the prefix „key-mgmt:“. This MIKEY message „AQ ...“ is the base64 encoded initiator message.

So far, we have seen an undisturbed phone call initialization. Now, let us show what could happen if a MITM attacks? In Figure 10 we have depicted this situation. The first part is the MIKEY method *Diffie-Hellman Key Exchange with a digital signature* and looks like Figure 6. Then in the second part are both conversational participants verified by the trust server. In addition to the Diffie-Hellman messages the certificates and the digital signatures will be transmitted in MIKEY. As soon as the key exchange has been completed, each conversational participant verifies the digital signature. There are two different scenarios, how a MITM attacker can change the MIKEY messages:

- Scenario 1: MITM replaces Alice's certificate C_A and the Diffie-Hellman value A by its data C_M and Z. Then it signs the MIKEY message with its own certificate and sends the signed message $\{C_M, g, p, Z\}S_{M1}$ to Bob. Because both a certificate and the associated private key are required to sign a message, the MITM cannot sign the message with Alice's certificate. Bob verifies the signature of the received MIKEY message using the received certificate C_M. The signature is accurate and Bob can validate Alice with her login name and the received certificate C_M. Then the trust server compares login names and certificates with the data in its database, but they do not match. Hence, the verification fails.
- Scenario 2: MITM can also leave Alice's certificate in the MIKEY message and just replaces the Diffie-Hellman value A by its own value Z. As in the first scenario, MITM signs the MI-

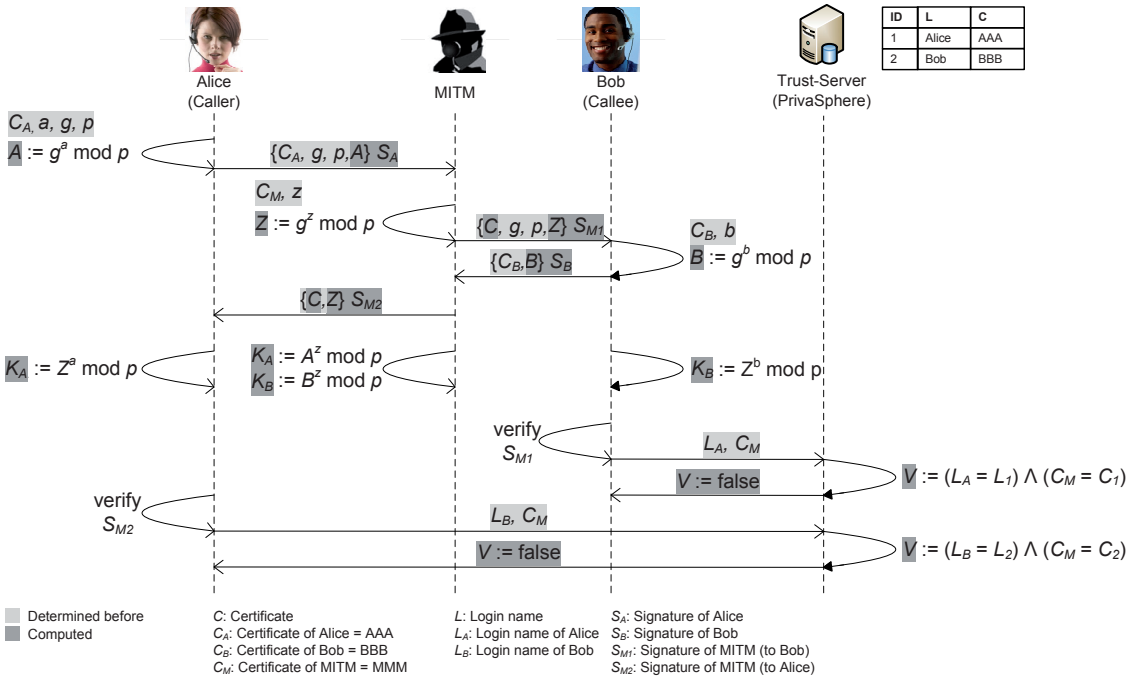

Figure 10: MITM attack on MIKEY

KEY message and sends now $\{C_A, g, p, Z\}S_{M2}$ to Bob. Then Bob verifies the signature of the received MIKEY with the certificate contained in the message. Because it receives Alice's certificate and the signature was done with MITM's certificate, the verification fails. So, the validation at the trust server can be omitted.

In both scenarios no voice communication will be established, because the verification of the signature or the validation fails. This means, as long as an attacker has no access to the trust server, a secure communication can be guaranteed.

Timing Problem

In the approach described above we have a timing problem with the delays in the computation of Diffie-Hellman parameters and the validation. The calculation time of Diffie-Hellman parameters depends on computing capacity and available entropy[3] and takes between 100 and 500 ms. The validation of the counterparts at the trust server takes about 2 s. Since no voice communication should be allowed before both parties are validated, a considerable and disturbing delay of more than two seconds occurs between the acceptance of a call and the establishing of voice communication.

Bob calculates its Diffie-Hellman parameters and validates Alice after he received the SIP message INVITE. On the other side, the validation of Bob does not start before Alice has received Bob's certificate. She receives Bob's certificate in the SIP message OK. The SIP standard defines voice connection establishment immediately after the OK message, although Alice could not validate Bob quickly enough. The conversational participants cannot be sure that they have contacted the desired party. If the validation is not correct, no voice communication will be established.

The SIP standard allows the insertion of optional parameters in the SIP messages RINGING and OPTIONS. A SIP message RINGING confirms a caller that the connection wish has been signaled to callee. This message is sent immediately before the response to the call invitation, thus the time gain is not large enough. Additionally, the transmission of a SIP message RINGING is not reliable, because no confirmation is sent after it has been received. In contrast a SIP message OPTIONS allows a SIP client to query another client about its communication capabilities before a call invitation. Such an OPTIONS message is acknowledged with an OK response. Hence, the call invitation does not start before the acknowledgement of the OPTIONS message.

Our timing problem can be resolved with an additional and confirmed OPTIONS message before Bob's OK answer to the call invitation. Bob's MIKEY message is inserted as an optional parameter in the SIP message OPTIONS. In case of non-existing or unregistered certificates, the OPTIONS method could be aborted with a non OK message. Only if a correct peer credential setup is detected, the caller proceeds with the INVITE request.

3 Entropy is the randomness collected by an operating system or application for use in cryptography. Mostly it is collected from hardware sources, either pre-existing ones such as mouse movements or specially provided randomness generators

Conclusion and Outlook

The result of our project is a user friendly application for secure voice communication over public internet. Our application is based on a SIP client, which allows key exchange from end-to-end in just a few steps. A trust server allows a registered user to communicate securely with unknown and unregistered participants. Our service combines important advantages:

- user friendliness,
- key exchange from end-to-end without a public PKI, and
- platform independence.

Platform independence is achieved by Java based client software, and user friendliness by using of Java WebStart for simplified client software deployment and installation. So far, we have already implemented the MIKEY library in Java and a trust client for the validation of a conversational participant at the trust server. In addition, we also could successfully integrate the SRTP implementation of the SIP client SIP-Communicator [Sipco] in MjSIP. There are still ongoing tasks before the new extensions can be integrated in the infrastructure of e-fon and PrivaSphere:

- The trust client and the MIKEY library have to be integrated into the SIP client.
- For the encryption of the signaling channel we want to make use of the TLS client from the cryptography provider Bouncy Castle [Bouca].

References

[ARP] ARP RFC: http://tools.ietf.org/html/rfc826

[Arpsp] Arpspoof, a part of DSniff: http://en.wikipedia.org/wiki/DSniff

[Bouca] Bouncy Castle cryptography provider: http://www.bouncycastle.org/

[Caiab] Cain and Abel: http://www.oxid.it/cain.html

[Incma] IncaMail: http://www.incamail.ch/de/

[Ipsec] IPSec RFC: http://tools.ietf.org/html/rfc4301

[Hda] HTTP-Digest Authentication RFC: http://tools.ietf.org/html/rfc2069

[Hvss] H.323 vs. SIP: http://www.cs.umd.edu/~pavlos/papers/unpublished/papageorgiou01comparison.pdf

[MIKEY] MIKEY RFC: http://tools.ietf.org/html/rfc3830

[Mjsip] SIP-Client MjSIP: http://www.mjsip.org/

[PRF] Pseudo-Random Function is described in [TLS]

[Priro] Primitive root: http://en.wikipedia.org/wiki/Primitive_root_modulo_n

[RTP] RTP RFC: http://tools.ietf.org/html/rfc3550

[RTCP] RTCP RFC: http://tools.ietf.org/html/rfc3550#page-19

[SIP] SIP RFC: http://tools.ietf.org/html/rfc3261

[Sips] SIP over TLS is described in [SIP]

[Sipco] SIP-Client SIP-Communicator: http://sip-communicator.org/

[Smime] S/MIME RFC: http://tools.ietf.org/html/rfc2633

[SDP] SDP RFC: http://tools.ietf.org/html/rfc2327

[SRTP] SRTP RFC: http://tools.ietf.org/html/rfc3711

[TLS] TLS RFC: http://tools.ietf.org/rfcmarkup/5246

[Veris] VeriSign: http://www.verisign.com/

[Wiresh] Wireshark: http://www.wireshark.org/

[Xarp] XArp: http://www.safe-install.com/programs/xarp.html