Formal modeling of authentication in SIP

Anders Moen Hagalisletto
Norwegian Computing Center
anders.moen@nr.no

Lars Strand
Norwegian Computing Center
lars.strand@nr.no

Abstract

The Session Initiation Protocol is increasingly used as a signaling protocol for administrating VoIP phone calls. Careless configuration of the SIP protocol is known to lead to a large set of attacks. SIP can be configured in several ways so that different functional and security requirements are met. Tools exist for attacking Voice over IP protocols, yet most tools target the message traffic aspect.

In this paper we show how different configurations of SIP can be specified in a protocol centric formal language. Both static analysis and simulations can be performed on the resulting specifications by the recently developed tool called PROSA. In particular, we analyze the Voice over IP architecture of a medium size Norwegian company, and show that several of the well known threats can be found including a new attack on the registration subprotocol.

1. Introduction

Voice-over-Packet (VoP) has been a research topic for nearly three decades. It was not until the late 1990s that telecommunications opened up to the Internet. Today Voice-over-IP (VoIP) is the transmission of voice over Internet or other packet switched networks. The underlying protocols providing VoIP functionality are not limited to voice only. Video, image, email, presence and instant messaging are other services that can be provided.

The integration of voice, video and data over the same network reduces the setup and maintenance costs. However, securing a VoIP system is challenging. First since VoIP shares the same infrastructure as traditional data networks, it also inherits the same security problems. Second VoIP does not have a dominant standard that has been scrutinized over the years by security researchers. Third VoIP have high requirements to the network since its a duplex communication with low tolerance for latency, jitter, and packet loss.

Many will likely expect VoIP to meet the same service level as the Public Switched Telephony System (PSTN), the traditional old circuit-switched telephone networks. For instance people have become accustomed to 99,999% availability rate on PSTN [14], which is less than five minutes a year. Another challenge is scalability: to enable VoIP to scale to a global level and at the same meet adequate security requirements. Is this possible to achieve with today's standard? Some claim it's not [15]. Regardless, VoIP is rapidly gaining acceptance in the telecommunication industry.

The Session Initiation Protocol (SIP) [17] is an application layer protocol for handling multimedia sessions between one or more callers. SIP is used to negotiate and establish a *context* for the media flow. Other protocols are used for media transport. The media protocol Real Time Protocol is often used in combination with SIP. SIP is a text based protocol, similar to HTTP, which makes debugging easier.

When SIP was specified focus was on functionality which could provide additional services. Less focus was given to security features [18]. To improve security, adding security mechanisms in SIP is an area receiving more attention today [5], [10], [8], [16].

The paper is organized as follows. In Section 2 we give an introduction to authentication in SIP. In Section 3, we present the VoIP architecture that is investigated (3.1) and how the formal specifications were obtained (3.2). In Section 4, the formalization of the protocol is presented, including a formalization of Digest access authentication (4.1), and the registration protocol (4.2). In Section 5, the a sample of results from the analysis is presented, including an example of a call-hijacking attack on the registration sub-protocol (5.1) and a description of the application and syntax of PROSA (5.2).

2. Authentication in SIP

Whether a given VoIP configuration is secure enough depends on two factors: (1) the requirements specified by the security policy, and (2) whether these requirements are covered by the implemented security mechanisms. The security requirement for telephony connections depends on the application area and policy of the company: for some users might connectivity be enough, while others would require strong confidentiality, integrity and authenticity.

There are three security configurations of SIP authentication: unsecured, weak authentication, and strong authentication. Unsecured provides no authentication. Weak authentication is an adaptation of the HTTP Digest Access Authentication [9] that requires a shared secret between the two participants. Strong authentication uses certificates in the same way as web browsers and servers use them. This paper only discuss Digest Access Authentication.

Authentication in SIP is not mandatory but *may* be used to identify the calling side. HTTP authentication allows both basic and digest authentication. Basic authentication, where the password is sent in plaintext, has been deprecated as of SIPv2 [17]. The Digest authentication is stateless, challenge-based and only authenticate the client. It should only apply to client-to-client or client-to-server authentication. Server-to-server authentication should rely on other security mechanisms like TLS or IPSec.

When reciving a request, the server may challenge the client with a random *nonce*. The client then hash the nonce, secret password, username and other parameters. The server, upon receiving the hash from the client, does the same computation and compare the two results. If the server generated hash equals the one received from the client, the client is authenticated.

The digest authentication provides message authentication and replay protection only. It does not cover message integrity or confidentiality and does not provide strong authentication when compared to Kerberos or public key based mechanism.

3. Analysis

First we got the informal specification of the voice over IP architecture of the company to be investigated. Based on the voice over IP architecture of the company, the documentation of SIP [17], and Digest Access Authentication [9], we tried to specify the particular instance of of SIP formally. Unfortunately, it was not possible to get a reasonable interpretation of the scenario by only using standard documentation. Instead we shifted focus to the implementation by logging and analysing network traffic based on a case study.

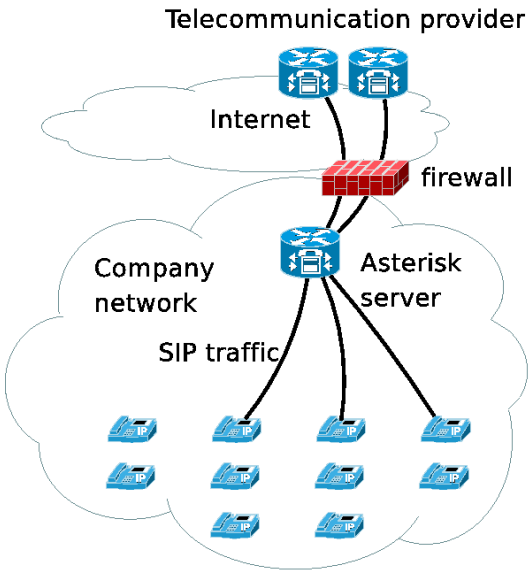

Figure 1. Scenario for VoIP architecture.

3.1. VoIP in a medium size company

Our case study is taken from a medium sized Norwegian company with 98 employees. Most users have a "hard phone" from Cisco, a conventional looking phone using VoIP. Some also have on their computer a "soft phone", an application capable of VoIP. The configuration files count 127 SIP phones, both soft and hard phones. There is one VoIP server running the open source software Asterisk [1] on Linux.

All phone calls destined to the outside, or external phone calls, are handled by an external telecommunication provider. This third party provider handles both VoIP and PSTN calls. The internal Asterisk server communicates with the telecommunication provider over two different connections. Both connections use the Inter-Asterisk eXchange v2 protocol (IAX) [4]. IAX uses a single UDP stream on a static port which simplify network and firewall management internally. See overview of the architecture in figure 1.

The hard phones use DHCP to obtain and configure network settings. This simplifies network management. But since no static IP is used, all the phones need to register to the Asterisk server at startup time. During the registration they also perform Digest authentication.

We obtained from the company the configuration files for the Asterisk server. This configuration was then

used to set up an identical system at our lab. Two X-Lite [3] soft-phones were used to place the SIP calls. All network traffic between the phones and the Asterisk server was then intercepted and logged using a network monitoring tool called "Wireshark" [2].

3.2. Method

It is well known that protocols in general and industrial security protocols can be hard to specify formally [11], [6]. Based on information acquired from sniffing network traffic, we were able to give a precise specification of SIP registration with Digest authentication. This specification was used as input to the PROSA protocol analyzer [13] in order to validate the specification and simulate uncompromized as well as compromised protocol instances. A severe call-hijacking attack was found at the end, which shows that formalization, simulation and automated analysis can reveal potential and real weaknesses with the implementation of VoIP systems.

Several lessons were learned by using Wireshark: The informal specifications in RFC 3261 were incomplete: The actual message interaction was not specified explicitly, neither was the the ordering of elements, and concrete adaptation of Digest Access Authentication.

Moreover the most important lessons were that (i) formalization of complex protocols is much faster using network monitoring tools than without, and (ii) taking batches from such tools give realistic specifications that are close to the implementation.

There have been few attempts to use formal methods and formal technologies to analyze SIP, with the exception of [11]. The security analysis was performed by first obtaining a formal specification that could be studied in itself and by mechanized tool support. We used the protocol analyzer PROSA to specify a register session of SIP. PROSA consists of a formal language based on temporal epistemic logic, a static analyzer that can automatically refine protocol specifications, and a simulator that can execute protocols as well as attacks on protocols. The static analyzer was used to debug specifications: typical errors like misprints of sender/receiver names or incomplete and incorrect message contents was detected instantaneously. Then several simulations were configured, attack-free, eavesdropping attacks and finally a severe call-hijacking attack that breaks integrity of the clients address.

A consequence of the latter is that the attacker receives phone calls intended to the client, but neither the client itself, nor the responder might know that the SIP channel is redirected and corrupt.

4. Formal specification of SIP

In this section we describe the formal specification of the SIP registration and signaling sub-protocols. First we derive specifications in high level protocol specifications in the form used in the literature. A protocol clause is of the form

$$(P) \quad A \longrightarrow B \quad : \quad M$$

meaning "agent A sends a message M to the agent B" The messages in the protocols consist of basic entities as follows:

A, B, C, S, I, $I(A)$	agent terms
K_{AB}	symmetric key shared by A, B
N_A	nonce generated by agent A
W_A^Y	string containing the text Y related to agent A
X_A	miscellaneous entities

There are three composition operators in the notation: concatenation of message content denoted by "," (comma), hashing $\mathsf{H}[M]$, and encryption denoted by $E(K \ : \ M)$, where K denotes a key and M a message content.

4.1. Digest access authentication

Digest access authentication use hashing, where nonces are used to protect against cryptoanalysis. We let $\mathsf{H}[\mathrm{C}]$, denote the hashing of content C. Digest access authentication is then given by

$$\begin{aligned} H_1 &= \mathsf{H}[W_C^{\mathrm{uname}}, W^{\mathrm{realm}}, K_{CR}^{\mathrm{pwd}}] \\ H_2 &= \mathsf{H}[W^{\mathrm{meth}}, W_C^{\mathrm{URI}}] \\ \text{response} &= \mathsf{H}[H_1, N_R, X_{\mathrm{nc}}, N_C, W^{\mathrm{qop}}, H_2] \end{aligned}$$

Written out explicitly the response yields:

$$\mathsf{H}[\mathsf{H}[W_C^{\mathrm{uname}}, W^{\mathrm{realm}}, K_{CR}^{\mathrm{pwd}}], N_R, X_{\mathrm{nc}}, N_C, W^{\mathrm{qop}}, \mathsf{H}[W^{\mathrm{meth}}, W_C^{\mathrm{URI}}]]$$

A typical application is then given by a challenger R requesting a client C to authenticate as described in the following protocol skeleton:

$$\begin{aligned} &(\mathrm{D}_1) \quad R \longrightarrow C : N_R \\ &(\mathrm{D}_2) \quad C \longrightarrow R : W_C^{\mathrm{uname}}, W^{\mathrm{realm}}, N_R, W_C^{\mathrm{URI}}, X_{\mathrm{nc}}, N_C, \\ &\qquad\qquad W^{\mathrm{qop}}, \mathsf{H}[H_1, N_R, X_{\mathrm{nc}}, N_C, W^{\mathrm{qop}}, H_2] \end{aligned}$$

Both parties, share a common secret, a password K_{CR}^{pwd}, playing the role of a symmetric key. Initially the challenger R sends a nonce N_R to the client C. The client responds by sending the all the basic entities in the response in plaintext, except the password, and then

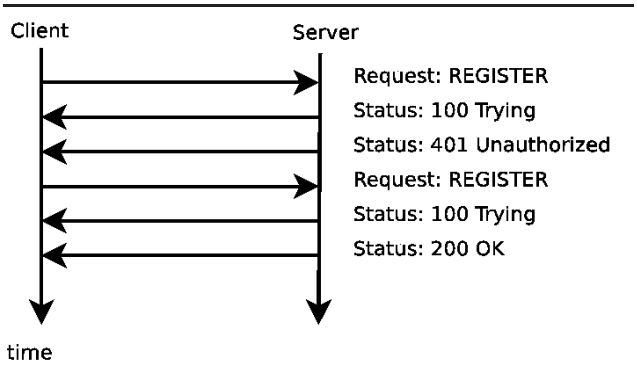

Figure 2. Digest authentication in SIP.

at the end the response itself. The challenger R can then perform hashing according to the response scheme above on the received entities and the password to check whether the response corresponds with the verification. The entities involved in digest access authentication can be explained as follows:

W_C^{uname}	authentication username of C
W^{realm}	defines a protective domain
K_{CR}^{pwd}	shared password between client C and challenger R
W^{meth}	main method of message (like HTTP)
W_C^{URI}	Digest URI for client C
N_R	nonce of the challenger R
X_{nc}	nonce counter
N_C	client C's nonce
W^{qop}	quality of protection

A "realm" is a protection domain on the server which is globally unique. Each realm on the server are partitioned into a set of logical protection spaces, each with its own authentication scheme.

In case of the SIP protocol, the URI is interpreted as the SIP URI, which have the same construction like an email address $<$ sip : user@domain $>$.

The authentication is one-way: The client C is authenticated to the challenger R. Authenticity of the client C is guaranteed by the secrecy of the shared key K_{CR}^{pwd}: Agent R can be certain that the message comes from C, since C is the only agent except C that possesses the key. Integrity of the message entities involved is provided by the fact that the hash could only be generated by C and freshness of the message is provided by the challenger nonce N_R.

4.2. The registration sub-protocol

If a SIP client is roaming or, like in our case, use DHCP to obtain network configuration, the client must register itself to a registration server. The SIP registration sub-protocol accomplish this task. A registration server should be connected to a location server that handles the bindings between the user's URI and contact addresses. SIP registration without any security mechanisms configured is given by the following specification:

$$
\begin{array}{lllll}
(\text{P}_1) & C \longrightarrow R & : & W^{\text{REGISTER}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(\text{P}_2) & R \longrightarrow C & : & W^{\text{Trying}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(\text{P}_3) & R \longrightarrow C & : & W^{\text{OK}}, W_C^{\text{Contact}}, N_C^{\text{callid}}
\end{array}
$$

Initially in message (P_1), agent C sends a registration request to the registrar server R. Agent R gives a receipt that the registration has been received in (P_2), and then finally if C's request is accepted by R, then a notification message is replied in (P_3). The message entities involved in the protocol scheme are:

N_C^{callid}	The session identifier for the current registration session
W^{Contact}	Contact host for client
W^{REGISTER}	REGISTER method that indicate a registration session
W^{Trying}	100 Trying, receipt to a previous SIP message
W^{OK}	200 OK method notifies successful registration

The N_C^{callid} is the session identifier for the current instance of the protocol. In the realization N_C^{callid} is built up by the host address of the client C and a nonce generated by C. In the context of the registration protocol each registration session from one client to one particular registration server should use the same CALLID N_C^{callid}, in order to prevent a delayed REGISTER request to arrive out of order [17, p. 58]. Note that the available contact hosts can be more than one, hence several potential bindings for the client can be specified by replacing W_C^{Contact}, with a sequence of potential hosts $W^{\text{Contact}} = W_C^{\text{Contact}_1}, \ldots, W_C^{\text{Contact}_n}$.

If digest access authentication is used, as shown in figure 2, then the registration sub-protocol can be specified as follows:

$$
\begin{array}{lll}
(\text{P}_1^{\text{D}}) & C \longrightarrow R : & W^{\text{REGISTER}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(\text{P}_2^{\text{D}}) & R \longrightarrow C : & W^{\text{Trying}}, N_C^{\text{callid}} \\
(\text{P}_3^{\text{D}}) & R \longrightarrow C : & W^{\text{Unauth}}, W^{\text{auth}}, W^{\text{realm}}, N_R, N_C^{\text{callid}} \\
(\text{P}_4^{\text{D}}) & C \longrightarrow R : & W^{\text{REGISTER}}, N_C^{\text{callid}}, W_C^{\text{uname}}, W^{\text{realm}}, \\
 & & N_R, W_C^{\text{URI}}, X_{\text{nc}}, N_C, W^{\text{qop}} \\
 & & \mathsf{H}[\mathsf{H}[W_C^{\text{uname}}, W^{\text{realm}}, K_{CR}^{\text{pwd}}], N_C, X_{\text{nc}}, \\
 & & N_R, W^{\text{qop}}, \mathsf{H}[W^{\text{REGISTER}}, W_C^{\text{URI}}]] \\
(\text{P}_5^{\text{D}}) & R \longrightarrow C : & W^{\text{Trying}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(\text{P}_6^{\text{D}}) & R \longrightarrow C : & W_{\text{OK}}, W_C^{\text{Contact}}, N_C^{\text{callid}}
\end{array}
$$

The protocol can be explained as follows: as the unsecured registration, the user (or client) requests to register at a registrar server R, then R gives a receipt of this message. The challenger R then demands an authentication ($\mathrm{P}_3^{\mathrm{D}}$) by combining message ($\mathrm{D}_1$) with the appropriate SIP method, information attributes and CALLID to the client. The meaning of the latter message is to request the client to authenticate itself to the registrar R. The client does this by essentially combining messages (P_1) and (D_2) in constructing message ($\mathrm{P}_4^{\mathrm{D}}$). A receipt from the registrar is given for the authentication trial in ($\mathrm{P}_5^{\mathrm{D}}$), and if the server R is able to verify message ($\mathrm{P}_4^{\mathrm{D}}$) by hashing the received plaintext elements and the previously known shared secret K_{CR}^{pwd} according to the response scheme, then the registrar notifies the client about the successful registration in the final message ($\mathrm{P}_6^{\mathrm{D}}$). Note that ($\mathrm{P}_6^{\mathrm{D}}$) is is interpreted stronger than (P_3), the former means that the client has successfully registered and is authenticated to R.

In the digest registration protocol there are thus three new messages $\mathrm{P}_3^{\mathrm{D}}$, $\mathrm{P}_4^{\mathrm{D}}$, and $\mathrm{P}_5^{\mathrm{D}}$ and consequently additional four message entities. These entities are described below:

W^{Unauth}	401 Unauthorized method request for authentication
W^{auth}	WWW-authenticate message request
N_B	Challenger nonce for DAA
N_A	Client nonce for DAA

4.3. The call setup sub-protocol

After registration, the user A, wants to call agent B. In the particular scenario explored in this paper, the registrar server acts as a proxy R. For convenience we suppose that the call is made internal, that is, the agents A and B are using a SIP phone call on the internal network (and therefore the call does not involve any extra hops over proxies on external network).

$$
\begin{array}{lll}
(\mathrm{P}_1) & A \longrightarrow R & : W^{\mathrm{INVITE}}, A, B, N_A^{\mathrm{callid}} \\
(\mathrm{P}_2) & R \longrightarrow B & : W^{\mathrm{INVITE}}, A, B, N_B^{\mathrm{callid}} \\
(\mathrm{P}_3) & R \longrightarrow A & : W^{\mathrm{Trying}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_4) & B \longrightarrow R & : W^{\mathrm{Trying}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_5) & B \longrightarrow R & : W^{\mathrm{Ringing}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_6) & R \longrightarrow A & : W^{\mathrm{Ringing}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_7) & B \longrightarrow R & : W^{\mathrm{OK}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_8) & R \longrightarrow A & : W^{\mathrm{OK}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_9) & A \longrightarrow B & : W^{\mathrm{ACK}}, N_A^{\mathrm{callid}} \\
 & A \longleftrightarrow B & \text{Media session (RTP or SRTP)} \\
(\mathrm{P}_{10}) & B \longrightarrow A & : W^{\mathrm{BYE}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_{11}) & A \longrightarrow B & : W^{\mathrm{ACK}}, N_A^{\mathrm{callid}}
\end{array}
$$

The additional entities involved in the session are the following SIP methods:

W^{INVITE}	The INVITE method that indicates a request for phone call
W^{ACK}	An acknowledgement method

By sniffing on the call setup sub-protocol that succeeds a registration session, the following specification was obtained:

$$
\begin{array}{lll}
(\mathrm{P}_1) & A \longrightarrow R & : W^{\mathrm{INVITE}}, A, B, N_A^{\mathrm{callid}} \\
(\mathrm{P}_2) & R \longrightarrow A & : W^{\mathrm{Unauth}}, W^{\mathrm{auth}}, W^{\mathrm{realm}}, N_R, A, B, N_A^{\mathrm{callid}} \\
(\mathrm{P}_3) & A \longrightarrow R & : W^{\mathrm{ACK}}, B, W_A^{\mathrm{Contact}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_4) & A \longrightarrow R & : W^{\mathrm{INVITE}}, A, B, N_A^{\mathrm{callid}} \\
 & & N_R', W_B^{\mathrm{URI}}, X_{\mathrm{nc}}, N_A', W^{\mathrm{qop}} \\
 & & \mathsf{H}[\mathsf{H}[W_A^{\mathrm{uname}}, W^{\mathrm{realm}}, K_{CR}^{\mathrm{pwd}}], N_A', X_{\mathrm{nc}}, \\
 & & N_R', W^{\mathrm{qop}}, \mathsf{H}[W^{\mathrm{INVITE}}, W_B^{\mathrm{URI}}]] \\
(\mathrm{P}_5) & R \longrightarrow B & : W^{\mathrm{INVITE}}, A, B, N_A^{\mathrm{callid}} \\
(\mathrm{P}_6) & R \longrightarrow A & : W^{\mathrm{Trying}}, N_A^{\mathrm{callid}}
\end{array}
$$

$$
\begin{array}{lll}
(\mathrm{P}_4) & B \longrightarrow R & : W^{\mathrm{Trying}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_5) & B \longrightarrow R & : W^{\mathrm{Ringing}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_6) & R \longrightarrow A & : W^{\mathrm{Ringing}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_7) & B \longrightarrow R & : W^{\mathrm{OK}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_8) & R \longrightarrow A & : W^{\mathrm{OK}}, N_B^{\mathrm{callid}} \\
(\mathrm{P}_9) & A \longrightarrow B & : W^{\mathrm{ACK}}, N_A^{\mathrm{callid}} \\
 & A \longleftrightarrow B & \text{Media session (RTP or SRTP)} \\
(\mathrm{P}_{10}) & B \longrightarrow A & : W^{\mathrm{BYE}}, N_A^{\mathrm{callid}} \\
(\mathrm{P}_{11}) & A \longrightarrow B & : W^{\mathrm{ACK}}, N_A^{\mathrm{callid}}
\end{array}
$$

We found typographical error in the documentation, the message (P_6) had opposite direction of the implementation ([19] page 162).

5. Threats

SIP communication is vulnerable to several types of attacks, including network layered attacks like denial of service or eavesdropping and SIP specific attacks like registration hijacking or call redirection. In our case study, a disgruntled employee could easily exploit the vulnerabilites of the company's VoIP system. By plugging in his laptop with VoIP attacker tool instead of his phone, he can easily launch attacks[1]. Once the attacker has access to the infrastructure, eavesdropping on phone calls are easy since VoIP-related communication in the company were transmitted unencrypted.

SIP calls routed externally to remote hosts, through several Internet domains might be subject to attacks

1 In our case, the attacker would have and easy target. All the phones in the company used the last three digits of the phones phone number as the shared secret. This would make a a brute force attack trivial.

as powerful as the Dolev Yao attacker [7]. The Dolev Yao models means that

(DY-1) cryptography is assumed to be perfect

(DY-2) the attacker controls the entire network

Since the underlying cryptographic operations works perfect (assumption DY-1) the attacker I never can use brute force to break the underlying hashing or encryption algorithms and I can not extract secret entities or decrypt encrypted messages if I does not possess the required secret entities. But the Dolev Yao attacker is assumed to know every cryptographic operation, like public key schemes, symmetric encryption and decryption, concrete hashing algorithms, ways of using HMAC's, etc.

Assumption DY-2 means that the attacker acts like a router that all messages pass through. The attacker has the power to intercept any message, and fake any message. *Interception* means that the attacker knows what every honest agent is doing. Interception is specified formally as

$$(a) \quad A \longrightarrow I(B) \quad : \quad M,$$

which reads "the intended message $A \longrightarrow B \ : \ M$, is picked up by the attacker I". The attacker's capability of *faking* means that it can:

- (i) impersonate as any other agent, that is assumed to be a honest agent,
- (ii) inject any compromised entity into the message (limited to the entities that can be obtained by assumption DY1).

To conclude, an intercepted message can be exploited by the the attacker in five ways, (b) is can be forwarded (eavesdropping), (c) the message can be routed to another agent, (d) the content M can be modified in any way $\underline{M}$ except by the limitations given by assumption DY1, (e) the message content can be modified and routed to another agent (the combination of (c) and (d)), and finally (f) the message can be stalled.

$$\begin{array}{lll} (b) & I(A) \longrightarrow B & : \ M \\ (c) & I(A) \longrightarrow C & : \ M \\ (d) & I(A) \longrightarrow B & : \ \underline{M} \end{array}$$

The cases $(a-f)$ cover man-in-the-middle attacks, the Dolev Yao attacker can also (g) initiate a protocol session as itself or by impersonation, or (h) send messages as a normal "honest" agent.

The attacks that can be launched all target the specified communication patterns of the protocol, not potential weaknesses in the cryptography nor error-prone implementation. Thus without any security mechanism configured, SIP is vulnerable to any imaginable severe attack.

5.1. Attacks on the registration protocol

The first attack on registration is eavesdropping, which is the basis of most of the succeeding attacks. Several trivial denial of service attacks can be launched by the attacker, by changing plaintext strings, the session nonce N_C^{callid}, or replay an old digest authentication response used between another client C' and the registrar server R. On the server side the latter behavior of the "client" must be considered to be corrupt, and the honest client might be excluded from the service. From the eavesdropping attack, we can construct the following call-hijacking attack on registration that includes digest authentication:

$$\begin{array}{ll}
(R^{\text{D}}_{1.1.a}) & C \longrightarrow I(R) : W^{\text{REGISTER}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.1.b}) & I(C) \longrightarrow R : W^{\text{REGISTER}}, W_I^{\text{Contact}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.2.a}) & R \longrightarrow I(C) : W^{\text{Trying}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.2.b}) & I(R) \longrightarrow C : W^{\text{Trying}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.3.a}) & R \longrightarrow I(C) : W^{\text{Unauth}}, W^{\text{auth}}, W^{\text{realm}}, N_R, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.3.b}) & I(R) \longrightarrow C : W^{\text{Unauth}}, W^{\text{auth}}, W^{\text{realm}}, N_R, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.4.a}) & C \longrightarrow I(R) : W^{\text{REGISTER}}, N_C^{\text{callid}}, W_C^{\text{uname}}, \\
 & \quad W^{\text{realm}}, N_R, W_C^{\text{URI}}, X_{\text{nc}}, N_C, W^{\text{qop}} \\
 & \quad \mathsf{H}[\mathsf{H}[W_C^{\text{uname}}, W^{\text{realm}}, K_{CR}^{\text{pwd}}], N_C, W^{X_{\text{nc}}}, \\
 & \quad N_R, W^{\text{qop}}, \mathsf{H}[W^{\text{REGISTER}}, W_C^{\text{URI}}]] \\
(R^{\text{D}}_{1.4.b}) & I(C) \longrightarrow R : W^{\text{REGISTER}}, N_C^{\text{callid}}, W_C^{\text{uname}}, \\
 & \quad W^{\text{realm}}, N_R, W_C^{\text{URI}}, X_{\text{nc}}, N_C, W^{\text{qop}} \\
 & \quad \mathsf{H}[\mathsf{H}[W_C^{\text{uname}}, W^{\text{realm}}, K_{CR}^{\text{pwd}}], N_C, X_{\text{nc}}, \\
 & \quad N_R, W^{\text{qop}}, \mathsf{H}[W^{\text{REGISTER}}, W_C^{\text{URI}}]] \\
(R^{\text{D}}_{1.5.a}) & R \longrightarrow I(C) : W^{\text{Trying}}, W_I^{\text{Contact}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.5.b}) & I(R) \longrightarrow C : W^{\text{Trying}}, W_C^{\text{Contact}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.6.a}) & R \longrightarrow I(C) : W_{\text{OK}}, W_I^{\text{Contact}}, N_C^{\text{callid}} \\
(R^{\text{D}}_{1.6.b}) & I(R) \longrightarrow C : W_{\text{OK}}, W_C^{\text{Contact}}, N_C^{\text{callid}}
\end{array}$$

In the attack, the malicious agent I is able to manipulate the client C to believe that she has successfully registered the additional contact location W_C^{Contact}, while the registration server is fooled to believe that C should be contacted using the corrupt address W_I^{Contact}, which is a location that the attacker I controls. In future deployment of SIP signaling and phone calls, the call is routed to the attackers contact address W_I^{Contact}.

The attacker I is passive in the protocol clauses $(\text{P}_2^{\text{D}} - \text{P}_4^{\text{D}})$, the part of the protocol where authentication occurs, while I is active and injecting the corrupt contact address in the clauses $(\text{P}_1^{\text{D}}, \text{P}_4^{\text{D}}, \text{P}_5^{\text{D}})$. A timely question is what kind of authentication or integrity guarantees are given by the application of Digest Access Authentication. The shared secret does not prevent the attacker to compromize the contact address. The problem with the deployment of digest Authentication is the way it is combined with SIP registration. The attack can be prevented by changing the

Digest response to include the contact address(es):

$$\mathsf{H}[\mathsf{H}[W_C^{\text{uname}}, W^{\text{realm}}, K_{CR}^{\text{pwd}}], W_C^{\text{Contact}}, N_R, X_{\text{nc}}, N_C, W^{\text{qop}}, \mathsf{H}[W^{\text{REGISTER}}, W_C^{\text{URI}}]]$$

When the registrar receives the response, the integrity of the contact address is preserved and fake contact addresses might be discovered.

5.2. Analysis of SIP in PROSA

The translation from the standard protocol specifications in Section 4 is direct as shown in the translation Figure 3. Specification of the registration subprotocol could be performed rapidly. It took approximately 6 man hours of work to have a executable specification of the concrete setup of SIP, thanks to the advanced validation mechanism of PROSA [12]. A test-scenario with the user Alice and the server Asterisk was configured: Each agent possessed the protocol and capability of constructing appropriate nonce. Three configurations of the scenario were tested, simulations with:

(A) with perfect network and no attacker

(B) an eavesdropper sniffing all messages

(C) an active call-hijacking attack as described in Section 5.

In each case Digest authentication was used, with assumption (DY-1). The integrity of the contact address was not preserved in the final simulation C: A query on the final state shows that the attacker successfully had compromised the address and could thereby redirect all calls through its own device.

6. Conclusion

We showed that even with large and complex protocols like SIP it is possible to formally specify the details of the message exchange on a level that permits automated security analysis. The formal specification and simulation of industrial protocols reveals several potential and real deficiencies that are not easily spot reading informal descriptions like RFCs.

A severe attack on registration was discovered in this paper, that is an instance of the general call-hijacking attacks. Both the discovery as well as the repair relied on the formal specification of the case study that was investigated. The approach taken in the paper can be used to rapidly specify and analyse different aspects and scenarious of SIP, closely related to the implemetations. Various configuration Call setup, Secure SIP, routing over several proxies can be explored with the same techniques as proposed in the paper.

$$
\begin{aligned}
&\mathsf{protocol}[\text{SIP} - \text{register}, 0,\\
&\quad \mathsf{role}(A) \wedge \mathsf{role}(R),\\
&\quad \mathsf{role}(A) \wedge \mathsf{role}(R), \mathsf{role}(A),\\
&\\
&\mathsf{Bel}_A(\mathsf{isKey}(\mathsf{key}(\mathsf{s}, A, R)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (A, R, \mathsf{Text}(\text{REGISTER}) \wedge\\
&\quad \mathsf{Text}(\text{CONTACT}) \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (R, A,\\
&\quad \mathsf{Text}(100\ \text{TRYING}) \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (R, A,\\
&\quad \mathsf{Text}(401\ \text{Unauthorized}) \wedge \mathsf{Text}(\text{WWW-Authenticate}) \wedge\\
&\quad \mathsf{Text}(\text{Asterisk}) \wedge \mathsf{isNonce}(\mathsf{n}(\text{MD5}, R)) \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (A, R,\\
&\quad \mathsf{Text}(\text{REGISTER}) \wedge \mathsf{Text}(\text{Username}) \wedge \mathsf{Text}(\text{Asterisk})\\
&\quad \wedge\ \mathsf{isNonce}\ (\mathsf{n}(\text{MD5}, R)) \wedge \mathsf{Agent}(R) \wedge \mathsf{isNonce}(\mathsf{n}(\text{MD5A}, A)) \wedge\\
&\quad \mathsf{Text}(\text{Nonce Counter}) \wedge \mathsf{Text}(\text{Auth}) \wedge\\
&\mathsf{Hash}[\mathsf{Hash}[\mathsf{Text}(\text{Username}) \wedge \mathsf{Text}(\text{Asterisk}) \wedge \mathsf{isKey}(\mathsf{key}(\mathsf{s}, A, R))] \wedge\\
&\quad \mathsf{isNonce}(\mathsf{n}(\text{MD5}, R)) \wedge \mathsf{Text}(\text{Nonce Counter}) \wedge\\
&\quad \mathsf{isNonce}(\mathsf{n}(\text{MD5A}, A)) \wedge \mathsf{Text}(\text{Auth})\\
&\wedge\ \mathsf{Hash}\ [\mathsf{Text}(\text{REGISTER}) \wedge \mathsf{Agent}(R)]] \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (R, A, \mathsf{Text}(100\ \text{TRYING}) \wedge\\
&\quad \mathsf{Text}(\text{CONTACT}) \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \mathsf{Transmit}\ (R, A, \mathsf{Text}(200\ \text{OK}) \wedge\\
&\quad \mathsf{Text}(\text{CONTACT}) \wedge \mathsf{isNonce}(\mathsf{n}(\text{CALLID}, A)))\\
&\mathscr{B}\ \varepsilon]
\end{aligned}
$$

Figure 3. SIP registration specified in PROSA.

Acknowledgments

Thanks to Wolfgang Leister, Lothar Fritsch, Arne-Kristian Groven, and Bjarte M. Østvold for comments on earlier drafts of this paper.

References

[1] Asterisk: The Open Source PBX & Telephony Platform `http://www.asterisk.org/`.

[2] Wireshark: Go deep. `http://www.wireshark.org/`.

[3] X-Lite from CounterPath Corp. `http://www.counterpath.com/x-lite.html`.

[4] IAX: Inter-Asterisk eXchange Version 2. Internet-Draft v04, mar 2008.

[5] J. Arkko, V. Torvinen, G. Camarillo, A. Niemi, and T. Haukka. Security Mechanism Agreement for the Session Initiation Protocol (SIP). RFC 3329 (Proposed Standard), January 2003.

[6] Steve Bishop, Matthew Fairbairn, Michael Norrish, Peter Sewell, Michael Smith, and Keith Wansbrough. Rigorous specification and conformance testing techniques for network protocols, as applied to TCP, UDP, and sockets. *SIGCOMM Comput. Commun. Rev.*, 35(4):265–276, 2005.

[7] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983.

[8] David Endler and Mark Collier. *Hacking Exposed VoIP: Voice over IP Security Secrets and Solutions*. McGraw-Hill Osborne Media, Nov 2006.

[9] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authen-

tication: Basic and Digest Access Authentication. RFC 2617 (Draft Standard), June 1999.

[10] D. Geneiatakis, G. Kambourakis, T. Dagiuklas, C. Lambrinoudakis, and S. Gritzalis. SIP Security Mechanisms: A state-of-the-art review. In *Proceedings of the Fifth International Network Conference (INC 2005)*, pages 147–155, July 2005.

[11] Prateek Gupta and Vitaly Shmatikov. Security Analysis of Voice-over-IP Protocols. In *Computer Security Foundations Symposium, 2007. CSF '07. 20th IEEE*, pages 49–63, 2007.

[12] Anders Moen Hagalisletto. Validating Attacks on Authentication Protocols. In *Proceedings of the 12th IEEE Symposium on Computers and Communications - (ISCC 2007), July 1-4, Aveiro, Portugal.*

[13] Anders Moen Hagalisletto. *Automated Support for the Design and Analysis of Security Protocols.* PhD thesis, University of Oslo, December 2007.

[14] D.R. Kuhn. Sources of failure in the public switched telephone network. *Computer*, 30:31–36, 1997.

[15] Daniel Minoli. *Voice Over IPv6 - Architecture for Next Generation VoIP Networks.* Newnes, May 2006.

[16] Thomas Porter. *Practical VoIP Security.* Syngress, Mar 2006.

[17] 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. Updated by RFCs 3265, 3853, 4320, 4916.

[18] S. Salsano, L. Veltri, and D. Papalilo. SIP security issues: The SIP authentication procedure and its processing load. *Network, IEEE*, 16:38–44, 2002.

[19] Henry Sinnreich and Alan B. Johnston. *Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol.* John Wiley & Sons, Inc., New York, NY, USA, second edition, August 2006.