# Advancement towards secure authentication in the Session Initiation Protocol

**Lars Strand**

*DOCTORAL DISSERTATION*

*for the degree of*

***PHILOSOPHIAE DOCTOR (PhD)***


*Department of Informatics*
*Faculty of Mathematics and Natural Sciences*
*University of Oslo*

*August 2011*

# Abstract

Voice over IP, or *VoIP*, is the delivery of multimedia content over IP networks, such as the Internet. As the popularity of VoIP increases, attackers find VoIP installations more attractive to exploit to their gain – thus security threats and attacks become more prevalent. The deployment of secure VoIP setup is therefore required to minimize risks of successful security attacks. Hence we see the crucial importance of implementing and use of strong security mechanisms in real-world VoIP installations.

Despite this importance, VoIP installations often lack deployment of strong security mechanisms that enforce authentication. Strong authentication in the VoIP signaling protocol SIP is important to ensure the authenticity of the communication peers. However, the most common authentication method used in SIP is weak and vulnerable to security attacks.

This thesis contributes by offering advanced mechanisms for authentication in SIP. The overall goal of the research has been to exploit how identity and authentication is handled in SIP, point out weaknesses, threats and attacks, and try to replace or enhance the currently used authentication mechanisms in such a way that the authentication service becomes more secure. An important question has been how to measure the "security" of new authentication methods over the old one. Since any changes to SIP will require an upgrade and re-deployment of existing VoIP-installation, it was also desirable to keep the changes to the SIP standard to a minimum.

# Preface

This dissertation is submitted to the Department of Informatics, Faculty of Mathematics and Natural Sciences, University of Oslo, in partial fulfillment for the degree Philosophiae Doctor (PhD). My main supervisor has been Wolfgang Leister, Assisting Research Director at Norwegian Computing Center (NR). Josef Noll, Professor at University of Oslo, and Torleiv Maseng, Director of Research at Norwegian Defence Research Establishment (FFI), have been my co-supervisors.

The research has been carried out in the period August 2007 to July 2011 at a 75% capacity, including one semester of course work, at the Norwegian Computing Center (NR). The remaining 25% I worked as a Linux Open Source Consultant at the company Redpill Linpro.

The study was supported by the Research Council of Norway (NFR) via the project EUX2010sec, Enterprise Unified Exchange Security (project 180054).

# Acknowledgments

First, and above all, I would like to thank my primary supervisor, Wolfgang Leister, for his continuous guidance, constructive criticism and support of my work. I would also express huge thanks to my co-supervisor, professor Josef Noll, for his directed guidance when I needed it the most. A sincere 'thank you' also go towards my second co-supervisor, Torleiv Maseng, for suggestions and constructive feedback.

I would like to thank my colleagues at NR for discussions, feedback and support. Especially Arne-Kristan Groven, who acted as a project manager for the EUX2010sec project, Anders Moen Hagalisletto who guided and showed me through my first scientific article and conference, and Lothar Fritsch who asked all the right questions. I would also like to thank the support of the staff at NR, which allowed me to concentrate exclusively on my research.

The industrial sponsors that made the EUX2010sec project possible are also thanked, in particular Bjørn Venn at Buskerud County Municipality (Buskerud fylkeskommune), Knut Arve Hauknes who was CTO at Ibidium, and Stein Øverland, CFO at Redpill Linpro.

Further I would like to thank: Alan Duric, CTO and co-founder at Telio, for several constructive discussions. Asterisk hacker Olle E. Johansson who provided me with an industrial view of SIP and VoIP, and Leif Johansson, active member of the 'kitten' WG in IETF, who suggested I'll have a look at GSS-API.

Also thank to my family and friends who always provide me with support and commitment. Last but not least, I would like to thank my wife Mette Kristine, for her love and patience which helped me through the harder periods of my work.

# Thesis structure

This thesis is divided into three parts. *Part I* introduces the field of SIP security, with particular focus on authentication, within the scope of this research. It includes a summary of the research contributions which have been published in journals or international conferences. *Part II* contains these research contributions. In *Part III* relevant appendices are provided.

**Part I** – Chapter 1 describes the background and motivation, and the *scientific method* of the thesis. Chapter 2 provides an extended *contextual background* that provides an overview of the background and issues that are dealt with in the research. Following this introduction and background is a presentation of the *frame of reference* in Chapter 3 that provides a state-of-the-art literature review and a knowledge base that can be used to enhance the understanding of the research results presented. The *discussion* of the contributions and a summary of each included *research paper* is presented in Chapter 4. The *conclusion* in Chapter 5 summarizes the research contributions and limitations, and provides suggestions for further research.

**Part II** – This part contains the following eight research papers:

- **Paper A** A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project.
- **Paper B** Formal Modeling of Authentication in SIP Registration.
- **Paper C** Designing attacks on SIP call setup.
- **Paper D** A survey of SIP Peering.
- **Paper E** Improving SIP authentication.
- **Paper F** Generic Security Services API authentication support for the Session Initiation Protocol.
- **Paper G** Migration towards a more secure authentication in the Session Initiation Protocol
- **Paper H** Advancement towards secure authentication in the Session Initiation Protocol

A brief summary and a detailed list of the publications and related work is provided in the next chapter.

**Part III** – Two appendices are provided. One appendix with a list of VoIP related acronyms and their meaning, and one appendix with a list of SIP methods and their functions.

# List of publications

Part II of this thesis is composed of papers A-H.

The author of this thesis is the principal contributor and first author of papers D-H. He is the second author of paper B and C, and third author of paper A.

A holistic research approach to VoIP security is presented in paper A. The research then continued with a close examination of the SIP `REGISTER` method which uncovered a weakness in the Digest Access Authentication presented in paper B. Further inspection of the SIP protocol and the SIP `INVITE` method led to paper C, where call-hijack attack was uncovered and presented. An overview of some of the challenges with SIP peering and security challenges is presented in paper D. In paper E the vulnerability presented in paper B is further examined, an attack implemented and a solution presented that counter the attack. In paper F, support for the Generic Security Services API authentication method in SIP is presented. Two new authentication mechanisms, the Password Authenticated Key Exchange and the Simple Authentication and Security Layer, are introduced in paper G. Paper H extends the discussions of Papers E-G with new moments, and summarizes the findings.

## Main contributions

**Paper A** Lothar Fritsch, Arne-Kristian Groven, Lars Strand, Wolfgang Leister and Anders Moen Hagalisletto. "A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project", *International Journal on Advances in Security*, ISSN 1942-2636, vol. 2, no. 2&3, 2009, pages 129-141.

Abstract: *The present paper describes the approach and preliminary results from the research project EUX2010SEC. The project works closely with Voice over IP (VoIP) companies and users. The project aims at providing better security of open source VoIP installations. The work towards this goal is organized by gathering researchers and practitioners around scientific activities that range from security modeling and verification up to testbed testing. The expected outcomes of the project are a solid scientific and practical understanding of the security options for setting up VoIP infrastructures, particular guidance on secure, typical setups of such infrastructure. The project's special focus is on producing results relevant to the practitioners in the project, aiming at the stimulation of innovation and the provision of highest quality in open source based VoIP products and services. The article describes the research-based innovation approach used.*

**Paper B** Anders Moen Hagalisletto and Lars Strand. "Formal modeling of authentication in SIP registration", *The $2^{nd}$ International Conference on Emerging Security Information,*

*Systems and Technologies (SECURWARE)*, pages 16-21, Aug 2008, Cap Esterel, France.

Abstract: *The Session Initiation Protocol (SIP) is increasingly used as a signaling protocol for administrating Voice over IP (VoIP) phone calls. SIP can be configured in several ways so that different functional and security requirements are met. Careless configuration of the SIP protocol is known to lead to a large set of attacks. In this paper we show how 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 PROSA. In particular, we analyze the VoIP architecture of a medium size Norwegian company, and show that several of the well known threats can be found.*

**Paper C** Anders Moen Hagalisletto and Lars Strand. "Designing Attacks on SIP Call Setup", *International Journal of Applied Cryptography*, Volume 2, Number 1, July 2010, pp. 13-22.

Abstract: *Many protocols running over the internet are neither formalised, nor formally analysed. The amount of documentation for telecommunication protocols used in real-life applications is huge, while the available analysis methods and tools require precise and clear-cut protocol clauses. A manual formalisation of the Session Initiation Protocol (SIP) used in Voice over IP (VoIP) applications is not feasible. Therefore, by combining the information retrieved from the specification documents published by the IETF and traces of real-world SIP traffic, we craft a formal specification of the protocol in addition to an implementation of the protocol. In the course of our work we detected several weaknesses, both of SIP call set-up and in the Asterisk implementation of the protocol. These weaknesses could be exploited and pose as a threat for authentication and non-repudiation of VoIP calls.*

**Paper D** Lars Strand and Wolfgang Leister. "A Survey of SIP Peering", *NATO ASI - Architects of Secure Networks (ASIGE)*, May 2010, Genova, Italy.

Abstract: *When placing a call from one SIP Service Provider to another, the call is traditionally routed over PSTN, instead of IP. This can lead to higher costs, reduced quality, and lack of functionality. These issues can be addressed by setting up a SIP peer between providers. A SIP peer is a layer 5 interconnection between two SIP Service Providers for the purpose of routing real-time and quasi-real-time call signalling between their customers. We survey the SIP peering architecture and show security implications.*

**Paper E** Lars Strand and Wolfgang Leister. "Improving SIP authentication", *The $10^{th}$ International Conference on Networks (ICN)*, pages 164-169, Jan 2011, St. Maarten, The Netherlands Antilles.

Abstract: *The digest access authentication method used in the voice over IP signaling protocol, SIP, is weak. This authentication method is the only method with mandatory support and widespread adoption in the industry. At the same time, this authentication method is vulnerable to a serious real-world attack. This poses a threat to VoIP industry installations and solutions. In this paper, we propose a solution that counters attacks on this wide-spread authentication method.*

**Paper F** Lars Strand, Josef Noll and Wolfgang Leister. "Generic Security Services API authentication support for the Session Initiation Protocol", *The 7$^{th}$ Advanced International Conference on Telecommunications (AICT)*, pages 117-122, Mar 2011, St. Maarten, The Netherlands Antilles.

Abstract: *The mandatory and most deployed authentication method used in the Session Initiation Protocol, the Digest Access Authentication method, is weak. Other, more secure authentication methods have emerged, but have seen little adoption yet. In this paper, support for using a generic authentication method, the Generic Security Services API, is added to the Session Initiating Protocol. When using this method, the Session Invitation Protocol does not need to support nor implement other authentication methods, only use the provided API library. This enables the Session Initiation Protocol to transparently support and use more secure authentication methods in a unified and generic way. As the suggested method includes a modification of the Session Initiation Protocol, an initial deployment strategy towards the Generic Security Services API authentication methods is added. To negotiate an authentication service, we use the pseudo security mechanism Simple and Protected GSS-API Negotiation Mechanism.*

**Paper G** Lars Strand, Wolfgang Leister and Alan Duric. "Migration towards a more secure authentication in the Session Initiation Protocol", *The 5$^{th}$ International Conference on Emerging Security Information, Systems and Technologies (SECURWARE)*, Aug 2011, Nice, France.

Abstract: *This paper specifies a two-step migration towards a stronger authentication in the Session Initiation Protocol. First, we add support for a Password Authenticated Key Exchange algorithm that can function as a drop-in replacement for the widely adopted Digest Access Authentication mechanism. This new authentication mechanism adds support for mutual authentication, is considered stronger and can rely on the same shared password used by the digest authentication. A more long-term solution is to replace the authentication scheme with the Simple Authentication and Security Layer. The Simple Authentication and Security Layer separates the authentication mechanisms from the Session Initiation Protocol, and adds support for a range of more secure authentication mechanisms in a generic and unified way. Both methods are presented, discussed, and shown how to integrate into the Session Initiation Protocol.*

**Paper H** Lars Strand and Wolfgang Leister. "Advancement towards secure authentication in the Session Initiation Protocol", Submitted to *International Journal On Advances in Security*, ISSN 1942-2636.

Abstract: *The Digest Access Authentication method used in the voice over IP signaling protocol, SIP, is weak. This authentication method is the only method with mandatory support and widespread adoption in the industry. At the same time, this authentication method is vulnerable to a serious real-world attack. This poses a threat to VoIP industry installations and solutions. In this paper, we propose a solution that counters attacks on this wide-spread authentication method. We also propose a two-step migration towards a stronger authentication in SIP. We add support for a Password Authenticated Key Exchange algorithm that can function as a drop-in replacement for the widely adopted*

*Digest Access Authentication mechanism. This new authentication mechanism adds support for mutual authentication, is considered stronger and can rely on the same shared password used by the digest authentication. A long-term solution is to replace the authentication scheme in SIP with a security abstraction layer. Two such security frameworks are introduced, discussed and evaluated: the Generic Security Services Application Program Interface and the Simple Authentication and Security Layer, which both enable SIP to transparently support and use more secure authentication methods in a unified and generic way.*

# Related work

**R1** Lars Strand. "Securing Open Source Communications Systems", poster presentation at the *VERDIKT programme conference*, Oct 2007, Hell, Norway.

**R2** Lars Strand. "Authentication in SIP", poster presentation at the *VERDIKT programme conference*, Oct 2008, Bergen, Norway.

**R3** Lothar Fritsch, Arne-Kristian Groven and Lars Strand. “A holistic approach to Open-Source VoIP security: Preliminary results from the EUX2010SEC project”, *The $8^{th}$ International Conference on Networks (ICN)*, pages 275-280, Mar 2009, Cancun, Mexico.

Abstract: *This paper describes the approach and preliminary results from the research project EUX2010sec. The project works closely with Voice-over-IP (VoIP) companies and users. It aims at providing better security of open source VoIP installations. The work towards this goal is organized by gathering researchers and practitioners around several scientific activities that range from security modeling and verification up to testbed testing. The expected outcomes of the project are a solid scientific and practical understanding of the security options for setting up VoIP infrastructures, particular guidance on secure, typical setups of such infrastructures, The project's special focus is on producing results relevant to the practitioners in the project, aiming at the stimulation of innovation and the provision of highest quality in open-source based VoIP products and services.*

**R4** Anders Moen Hagalisletto, Lars Strand, Wolfgang Leister and Arne-Kristian Groven. “Analysing Protocol Implementations”, *The $5^{th}$ Information Security Practice and Experience Conference (ISPEC)*, pages 171-182, Apr 2009, Xi’an, China.

Abstract: *Many protocols running over the Internet are neither formalised, nor formally analysed. The amount of documentation for telecommunication protocols used in real-life applications is huge, while the available analysis methods and tools require precise and clear-cut protocol clauses. A manual formalisation of the Session Initiation Protocol (SIP) used in Voice over IP (VoIP) applications is not feasible. Therefore, by combining the information retrieved from the specification documents published by the IETF, and traces of real world SIP traffic we craft a formal specification of the protocol in addition to an implementation of the protocol. In the course of our work we detected several weaknesses, both of SIP call setup and in the Asterisk implementation of the protocol.*

*These weaknesses could be exploited and pose as a threat for authentication and non-repudiation of VoIP calls.*

**R5** Elin Sundby Boysen and Lars Strand. "Security analysis of the SIP Handover Extension", *The 2nd Norwegian Information Security Conference (NISK)*, pages 84-96, Nov 2009, Trondheim, Norway.

Abstract: *With the increased demand for mobility support in VoIP, a SIP Handover Extension has been proposed. This paper discusses and analyses the security threats to this extension. Different usage scenarios for the Handover Extension are identified. For each scenario we identify known threats, and discuss how to counter them using known security mechanisms. The Handover Extension is particularly vulnerable to wiretapping call-hijacking. Further we argue that all identified threats can be countered using known security mechanisms, and propose S/MIME as the best countermeasure, but with an added price of increased complexity.*

**R6** Lars Strand. "VoIP lab as a research tool in the EUX2010sec project", *Norwegian Computing Center, Technical Report DART/08/10*, April 2010, Oslo, Norway.

Abstract: *As part of the research project EUX2010sec at NR, a Voice over IP testbed has been set up. The testbed has a central role in the research project that enables us to do testing, experimentation and measurements of VoIP protocols and VoIP scenarios. This white paper describes the equipment available, both server hardware and UAs (phones), software components and various network configuration setup of the testbed.*

# Other scientific activities

- Technical Program Committee, The International Conference on Networks (ICN) 2010, 2011.
- Technical Program Committee, The International Conference on Emerging Security Information, Systems and Technologies (SECURWARE) 2009, 2010, 2011.
- Session chair, International Conference on Networks (ICN) 2009, 2011
- Session chair, The 2nd International Conference on Emerging Security Information, Systems and Technologies (SECURWARE) 2008

# Contents

# PART I:

# Introduction

# Chapter 1

# Introduction

> "It's appalling how much worse VoIP is compared to the PSTN. If these problems aren't fixed, VoIP is going nowhere."
>
> *— Philip Zimmerman, author of PGP and ZRTP*

VoIP is used as a broad term to describe a family of communication protocols, software applications and hardware appliances where traditionally telephony services are transported over packet-switched networks, usually the Internet. VoIP has been deployed in the telecommunication industry since the late 1990s and is today emerging as a mature technology rapidly replacing the traditional public-switched telephone network (PSTN). The rationale used by the industry for switching from PSTN to VoIP is usually cost savings, increased functionality and less administrative and operational overhead for maintaining both a IP network and a PSTN telephone network.

Telecommunication is a big industry that involves many different vendors, that produce different products which offer VoIP and VoIP-related services. However, since telecommunication devices manufactured by different vendors, must be able to communicate, it requires a common communication protocol. The *de facto* VoIP protocols adopted by the industry are the Session Initiation Protocol (SIP) and the Real-time Transport Protocol (RTP). These two protocols address two different functions – SIP is used for signaling, e.g., setting up, modifying and tearing down multimedia sessions, while RTP transports the actual media stream, e.g., voice.

The VoIP service cannot rely the security on the telecommunication infrastructure, dedicated lines, physically protected switches, and certified telephony equipment as is common with PSTN. Since VoIP normally uses the Internet Protocol (IP) over Internet as carrier, and SIP signaling is such a fundamental component of VoIP, we require and would expect strong security mechanisms to be present and implemented in real-world SIP installations.

## 1.1 Background and motivation

At the end of 2009, 29.1% of the private landline phone market in Norway was using VoIP. There has been a steady increase in the number of VoIP users since 2002, as well as a decrease in PSTN subscriptions [64]. Providing precise VoIP usage and penetration numbers are challenging, due to the broad definition of VoIP and varying methodology in measuring [5] – for

example, do the measurement include regular Skype users, "pure VoIP" (also called "PC-to-PC calls"), and the use of embedded VoIP in online gaming? While the literature diverges on the estimates, survey from other countries show similar increase of VoIP usage trends [70].

One important component of VoIP, is the family of communication protocols that enable real-time communication over the IP networks. These VoIP protocols are open "Internet standards" developed by the Internet Engineering Task Force (IETF)[1] and are freely available. The VoIP protocol standards have been implemented in numerous open source and proprietary software projects. While several open source VoIP servers and clients exists, an often used VoIP server is the "Asterisk" server[2].

Despite VoIP's popularity, some researchers have expressed their concerns, and questioned the current state of VoIP in terms of security [83], and to provide a scalable and reliable service [52, page 2]. These concerns served as the motivation for this thesis' research questions as given in Section 1.3.

The research performed in this thesis has been part of the NFR (The Research Council of Norway) VoIP security research project EUX2010sec[3] which has its roots in the Nordic resource network *Enterprise Unified Exchange (EUX2010)*. The project period was from 2006–2010. The project's overall goal was "to improve both the security level and the security awareness when developing, installing and using open source VoIP/PBX/multimedia solutions" [23]. The project's focus evolved around three main topics *1*) Quality of service (QoS) and the reliability of the VoIP service, *2*) scalability, and *3*) security. A requirement was that all components were to use open source software whenever possible.

The project collaborated closely with VoIP industry partners in Norway, and findings reveal that their largest worries concerning security evolve around identity fraud, fraudulent service usage, and losses due to fraudulent outgoing calls [21]. These worries can be classified in the clear threat taxonomy given by the "VoIP Security Alliance" [97] as *misrepresenting identity*, *interception and modification* and *service abuse*, and became the main threats this research would counter.

The EUX2010sec project emphasized on research results that are applicable to the "real world" so that existing VoIP infrastructure can be improved and secured. To achieve this, we installed a testbed that enabled us have a VoIP infrastructure for experimentation, analysis and testing of VoIP components in various scenarios. That gave us an advantage over a pure theoretical approach, since the performance of a VoIP installation has many deciding factors, like network utilization and congestion, the network architecture, protocols and security mechanisms used [88]. The two main request methods in SIP support authentication (`REGISTER` and `INVITE`), and the authentication mechanism became the initial part of the thesis, since a strong authentication mechanisms is crucial to avoid *misrepresenting of identity*, and, to a certain extent, *interception and modification* and *service abuse*.

[1] The Internet Engineering Task Force homepage: `http://www.ietf.org/`

[2] Asterisk homepage: `http://www.asterisk.org/`

[3] EUX2010sec project homepage: `http://www.nr.no/pages/dart/project_flyer_eux2010sec`

## 1.2 Scope

The discussion of the research background and motivation introduced the importance of authentication in VoIP systems. I have observed (through no formal study) that the most common authentication mechanism used in SIP is the Digest Access Authentication (DAA)[4]. Although other security mechanisms that support authentication have evolved within VoIP, none have seen any widespread adoption as DAA.

The lack of scientific focus on understanding and evaluation of authentication in SIP calls for more research in this area. Current research focus primarily on implementing TLS support [12] in both SIP and RTP (using DTLS [69]), but, as discussed in Paper H and Chapter 3, implementing TLS and certification handling (PKI) is complex and challenging [2]. This research takes a different approach than using TLS to solve the security challenges in SIP.

Following the engineering methodology, presented in the next section, SIP-based VoIP installation was implemented and analyzed in our testlab. After exploring how vulnerable the current solution is to attacks, this research started looking for ways to improved the authentication in SIP. To prevent extensive changes to the SIP standard, any new proposal presented should introduce as small change to SIP as possible.

## 1.3 Method of the thesis

Analyzing the background discussion and the scope of the research, and following the research method used in this thesis, five research questions where identified. These five research questions led to four research goals that had to be addressed.

This thesis follows the scientific research method outlined by Glass (1995) [25] and Comer et al. (1989) [10]. First, the "problem space" was observed, involving examining existing research findings and problem areas (*Question 1*). The problem areas were analyzed and evaluated (*Question 2*, *Question 3*), and new solutions were designed to seek improvement or replace the status quo (*Question 4*, *Question 5*). This research method corresponds closely to what Glass identify as the "engineering method", also called an evolutionary paradigm: Observe existing solutions, propose better solutions, build or develop, measure and analyze, and repeat until no further improvements are possible [25]. Results found by Wainer et al. concludes the most common computer science methodology mainly follows this "engineering epistemology" [98].

By following this research method, the first research questions was to identify the "problem space" and problem areas. Therefore, the first research question was identified as:

- **Question 1:** What are real-world security challenges in SIP?

In order to answer this first research question, this research had to look into the general literature on VoIP and VoIP security (*Goal 1*). After that we identified secure authentication in the SIP protocol as a major challenge and area for further research (*Goal 2*), thus focus in this

[4]This observation has also been confirmed in a private conversation by both Alan Duric, CTO and co-founder of Telio, on the $8^{th}$ October 2009, and by Olle Johansson, VoIP specialist and responsible for the SIP code in Asterisk, on the $13^{th}$ August 2010.

thesis is on authentication in SIP. Rather than a theoretical approach alone, we also focus on real-world applicability and industry deployment, which is formulated in sub-goal *G4 (c)*.

- **Question 2:** Are the current authentication mechanisms used in SIP vulnerable to security attacks and what are the implications?

Our second research question focuses on vulnerabilities in the SIP authentication. We enable a testbed with industry adopted VoIP configuration and traffic, and thus target "real-world" installations. We intend to test, demonstrate and implement security attacks in our testbed where possible (*Goal 2*).

- **Question 3:** What are the challenges in the currently used mitigation strategies in SIP?

In order to counter vulnerabilities and security attacks identified by research question 2, we discuss and analyze current mitigation strategies developed for SIP (*Goal 3)*). Several of the mitigation strategies discussed remains theoretical and lack industry adoption. We therefore include in our approach an evolutionary industry upgrade path to ease the migration to the solution presented in this thesis (*Goal 4*).

- **Question 4:** Which improvements of the SIP authentication mechanisms can be performed applying the engineering methodology?

Our research method implies enhancement and advancement of the currently used authentication mechanism in SIP. The currently used authentication mechanism are analyzed (*Goal 2*) and improved. We also explore how existing standardized authentication mechanisms can be added to SIP (*Goal 3*), with minimal changes to the existing SIP standard.

- **Question 5:** To what degree does the suggested approach improves the SIP authentication in given real-world scenarios?

Our last research question evaluates the authentication mechanisms suggested, and addresses real-world security concerns (*Goal 4*). We identify three SIP scenarios where authentication must be handled, and propose an evolutionary strategy towards the suggested approach.

| **Research Questions** | **Research Goals** |
|---|---|
| Question 1 | Goal 1, Goal 2 |
| Question 2 | Goal 2 |
| Question 3 | Goal 3, Goal 4 |
| Question 4 | Goal 2, Goal 3 |
| Question 5 | Goal 4 |

Table 1.1: Mapping of research questions and research goals.

To address these research questions, keywords were derived from the the background discussion and the research questions themselves to form a goal taxonomy. This diagram assisted the research in defining the topics and direction of the thesis. The keywords where grouped together, as depicted in Figure 1.1, where each group eventuated into a specific research goals. For each research goal identified, the expected outcome is a list of one or more measurable and achievable outcomes. The research goals and the expected outcome are identified as:

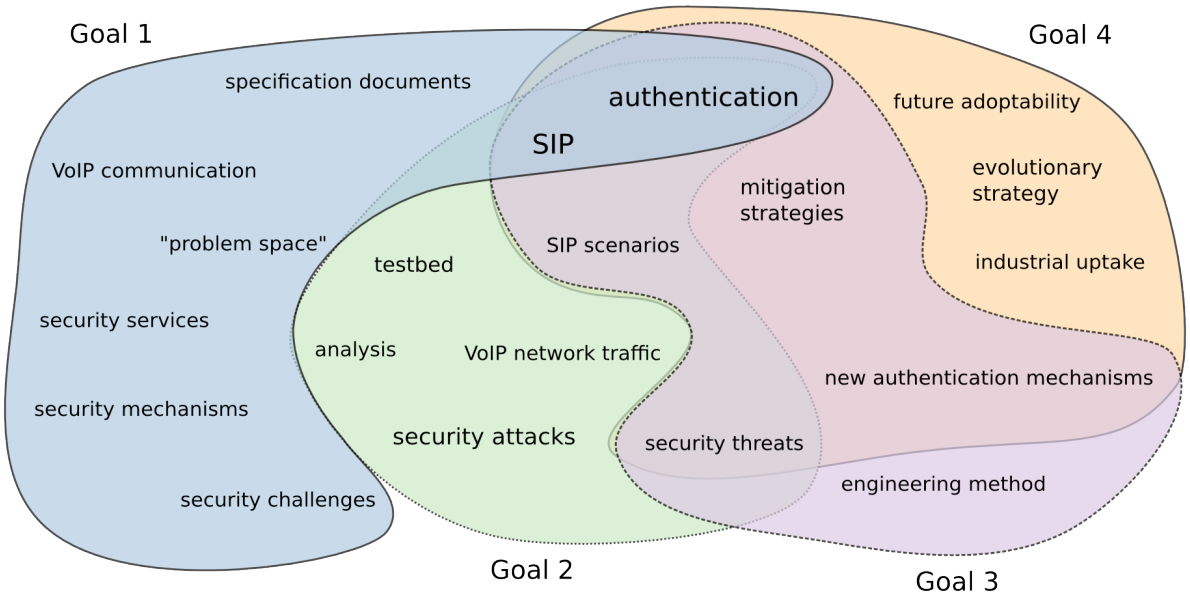



Figure 1.1: A goal taxonomy of relevant keywords derived from the background discussion and research questions which was grouped into and used as basis to formulate the research goals.

- **Goal 1:** Explore VoIP communication and authentication security challenges in SIP.

  The expected outcome of this goal is identified as:

  - **G1 (a):** A list of security challenges in SIP.
  - **G1 (b):** The list given in *G1 (a)* prioritized and ranked as most severe.
  - **G1 (c):** A list of SIP authentication scenarios.

- **Goal 2:** Identify and analyze SIP authentication vulnerabilities and threats by demonstrating security attacks.

  The expected outcome of this goal is identified as:

  - **G2 (a):** Enable a VoIP testbed for security testing of selected VoIP SIP scenarios.
  - **G2 (b):** List of security attacks and their consequences for VoIP SIP communication.
  - **G2 (c):** Analysis, discussion and implementation of at least two security attacks targeting the SIP protocol.

- **Goal 3:** Analyze current mitigation strategies and suggest enhanced secure authentication mechanisms.

  The expected outcome of this goal is identified as:

  - **G3 (a):** A discussion and analysis of current mitigation strategies for SIP authentication.
  - **G3 (b):** Apply engineering methodology to improve SIP authentication.

- **Goal 4:** Evaluate the suggested authentication mechanisms, and address real-world security concerns.

  The expected outcome of this goal is identified as:

  - **G4 (a):** An evaluation of the suggested new authentication mechanisms provided by *G3 (b)*. Discuss against SIP authentication scenarios provided by *G1 (c)*, and security attacks provided by *G2 (b)*.
  - **G4 (b):** Based on *G4 (a)*, recommend a new authentication mechanism.
  - **G4 (c):** Provide an industrial uptake strategy path that addresses real-world security concerns.

The mapping between the research questions and the research goals is shown in Table 1.1. The research goals (Goal 1-4) are expected to be achieved through the scientific contributions of the thesis, which include *Part I* and *Part II*. The details of how the research goals are addressed are discussed in Chapter 4. Table 5.1 provides metrics of the grade of achievement with respect to the research goals.

# Chapter 2

# Contextual background

> "'. . . (with TDM) anyone in the world could call anyone else in the world, any time, any where, and be able to support a good-quality telephonic conversation...' Frankly, this is an elusive, currently-unachievable goal for the VoIP industry."
>
> *— Daniel Minoli, "Voice over IPv6" (2006)*

In this chapter we provide a contextual background which consists of an overview, and major issues dealt with in our research.

## 2.1 Voice over IP

The emergence of VoIP as a technology had experimental beginnings in the 1970s [9], but did not gain industrial penetration until a critical mass of household had broadband connections in the late 1990s and early 2000s [2]. VoIP services like Skype emerged being solely based on the Internet infrastructure, thus forming an alternative to PSTN. In May 2011, the widely popular VoIP application Skype was acquired by Microsoft for $8.5 billion USD[1]. The acquisition was the biggest Microsoft has up to now (2011). This deal put VoIP communication into the headlines and demonstrated quite clear that VoIP today is "big business" with industrial momentum [16].

In the mid 1990s two competing protocol suites gained momentum: *1*) The H.323 [35] developed by the Telecommunication Standardization Sector (ITU-T), and *2*) the SIP protocol [77] developed by the Internet Engineering Task Force (IETF). While H.323 did see some early industry adoption, SIP gained more momentum in the sequel. Today, the discussions whether to prefer H.323 over SIP are considered to be over. Researchers and market analysts claim that all H.323 installations should be considered to be replaced and upgraded to SIP in order to avoid becoming a liability [82]. Today, the protocol pair SIP and Real-time Transport Protocol (RTP) [80] are emerging as the industry standard.

With the migration to VoIP, many will likely expect VoIP to meet the same service level as the Public Switched Telephony System (PSTN), the traditional circuit-switched telephone

[1]Microsoft acquire Skype: `http://about.skype.com/press/2011/05/microsoft_to_acquire_skype.html`

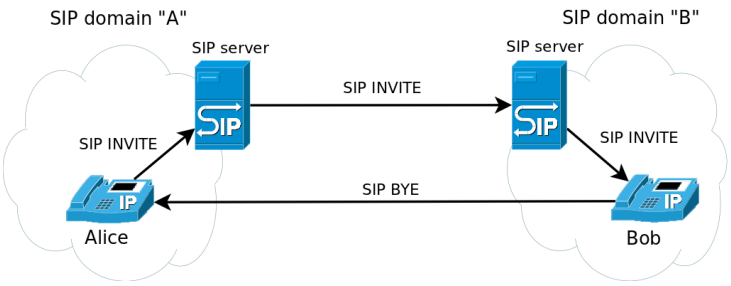



Figure 2.1: When Alice calls Bob, the SIP `INVITE` message goes from Alice's UA through SIP domain A's SIP server, which forwards the call to SIP domain B's SIP server and then to Bob. When the call is ending, the SIP BYE message may be sent directly between the UAs forming a message path trapezoid.

networks. People have become accustomed to high availability on PSTN, and since the PSTN providers trust each others signaling, a reasonably reliable caller identification [43] is provided.

## 2.2 Session Initiation Protocol

The Session Initiation Protocol (SIP) is an open IETF specification. Its core functionality is specified in specification document "RFC 3261" [77]. The RFC is one of the largest (in terms of page number) ever defined by the IETF[2]. Also, the number of SIP-related specification documents is numerous and growing[3] [66], which can make it difficult for developers to find and use the relevant RFCs. There even exists a specification document that list relevant specifications under the SIP umbrella and serves as a guide to the SIP RFC series [75].

SIP is an application layer protocol that is designed to be independent from the lower transport protocol layer. SIP can run over the transport layer protocols TCP, UDP and SCTP, however, it is the experience of this research that UDP is the most common protocol utilized [88]. SIP is a text-based protocol that are modeled after, and bear resemblance to, both SMTP [65] and HTTP [18], two other IETF protocols [82, page 106].

### 2.2.1 SIP architecture and components

The purpose of SIP is to negotiate, establish, change and tear-down the context of a multimedia flow; other protocols, such as the RTP, are used for the media (voice) transport. Illustrated in a simple example, SIP works as follows: When Alice calls Bob, as depicted in Figure 2.1, a SIP `INVITE` message is sent from Alice's phone, via one or more SIP servers, to Bob's phone. Before allowing Alice to send an `INVITE` request to Bob, Alice's SIP server may request Alice to authenticate and do rudimentary SIP header checks before forwarding to Bob's SIP server. When the call is terminated, the SIP `BYE` message might be sent directly between the Alice and

[2]IETF stats, "Distribution of the Number of Pages per Document": `http://www.arkko.com/tools/allstats/pagedistr.html`

[3]The "VoIP RFC Watch" illustrate the enormous growing rate of RFCs which are related to SIP and VoIP in general: `http://rfc3261.net/`

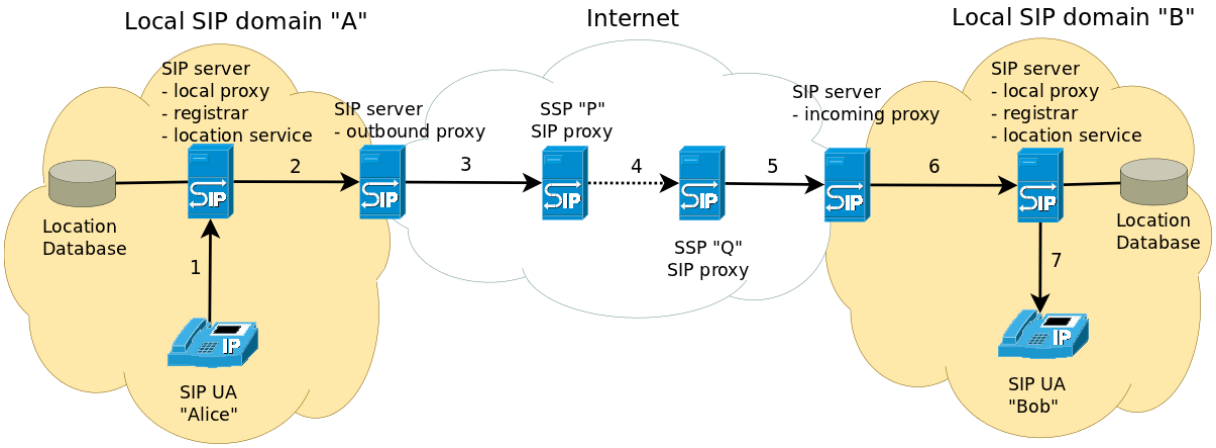



Figure 2.2: A SIP communication network with two UAs ("Alice" and "Bob"), two local SIP domains ($A$ and $B$) and two SIP Service Providers ($P$ and $Q$).

Bob, thus forming a "SIP trapezoid" for the message path. The RTP traffic might also be sent directly between Alice and Bob without going through the intermediate SIP servers.

In a SIP enabled network, there are defined several logical components which handle specific tasks:

- A SIP **User Agent** (UA) is capable of creating and receiving SIP requests, thus representing the functionality present in all nodes connected to a SIP network. In the VoIP literature, most often a UA is used to describe a SIP enabled telephone. The UA can be divided into the part responsible for sending a SIP request and processing replies, called *User Agent Client* (UAC), and the part responsible for processing and answering requests is called *User Agent Server* (UAS).

- The SIP **Proxy server** handles the routing and forwarding of SIP messages between other SIP nodes.

- For any request a **Redirect server** receives, a response is returned that direct the original sender to one or more alternate addresses. The original sender must then re-send the SIP request to the new address received.

- The SIP **Registrar server** binds the permanent "long-term" SIP addresses, called Address-of-Record (AoR), to the currently used IP-address/hostname of the UA, called the Contact URI. Example: The UA having the SIP address `sip:alice@example.com` can be found at `sip:alice@192.168.1.123:5060`. These bindings are stored in the Registrar's user location database. This database is updated whenever a UA update its binding by using the SIP `REGISTER` request.

An example of a SIP network scenario is shown in Figure 2.2. Before Alice can place a call to Bob, Alice's UA must first register to the local SIP Registrar server. The details about this transaction is given in Section 2.2.2 on the following page. The SIP Registrar server stores the binding in a location database, which may be local on the SIP server, or on a dedicated database server.

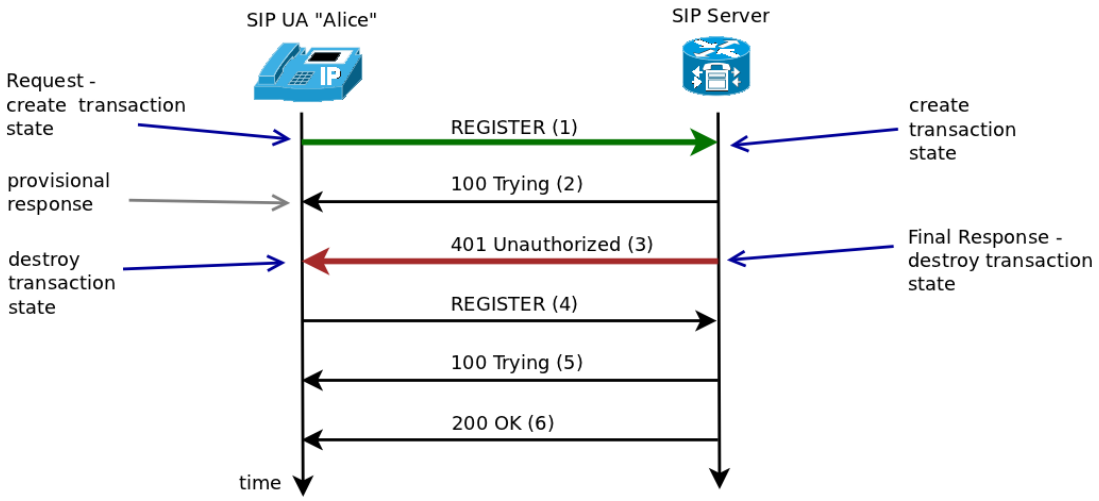



Figure 2.3: The SIP `REGISTER` dialog for authenticated communications.

In the following paragraph, the numbers in parentheses refer to the numbers in Figure 2.2. When Alice places a call destined to Bob, a SIP `INVITE` message is sent first to the local SIP server (1). This proxy server decides that the destination of the SIP request is non-local, and therefore routes the message to the outbound SIP proxy (2). The outbound proxy sends the message to the SIP Service Provider's (SSP) SIP proxy (3). The SSP $P$ routes the SIP message to SSP $Q$ (4). The routing between $P$ and $Q$ might traverse one or several hops, expressed by a dotted line in Figure 2.2. When the request arrives to the SIP server for "SIP domain $B$" (5), the message is routed to the local SIP proxy (6). The local SIP proxy looks up Bob's address in the Location database, and routes the message to Bob's UA (7).

### 2.2.2 SIP dialog and message structure

SIP is heavily influenced by the HTTP request-response model, where each transaction consists of a request that requires a particular response. For a list of SIP request methods and response codes, see Appendix ii. A transaction is a sequence of SIP messages exchanged between SIP nodes. The transaction is started by one SIP request, followed by zero or more informal (provisional) SIP response messages and one (or more) final response messages.

An example is the SIP `REGISTER` transaction, depicted in Figure 2.3. In the following paragraph, the numbers in parentheses refers to the protocol messages there. Alice's UA sends a SIP `REGISTER` message to the SIP Registrar server (1), thus starting a transaction. The server will respond immediately with the provisional response `100 Trying` (2). This informs the UA that the request has been received and that it is being processed. Since the UA is not authenticated, the SIP server responds with a `401 Unauthorized` message (3) and ending the transaction. For the UA, this response means that it needs to retry the SIP `REGISTER` request (and start a new transaction), this time with the necessary authentication credentials (4). The server responds again with a provisional response (5), and if the authentication is successful, the SIP server responds with a `200 OK` message (6). The UA is now authenticated, and the SIP Registrar has recorded the UA's Contact IP-address/hostname in its Location Database.

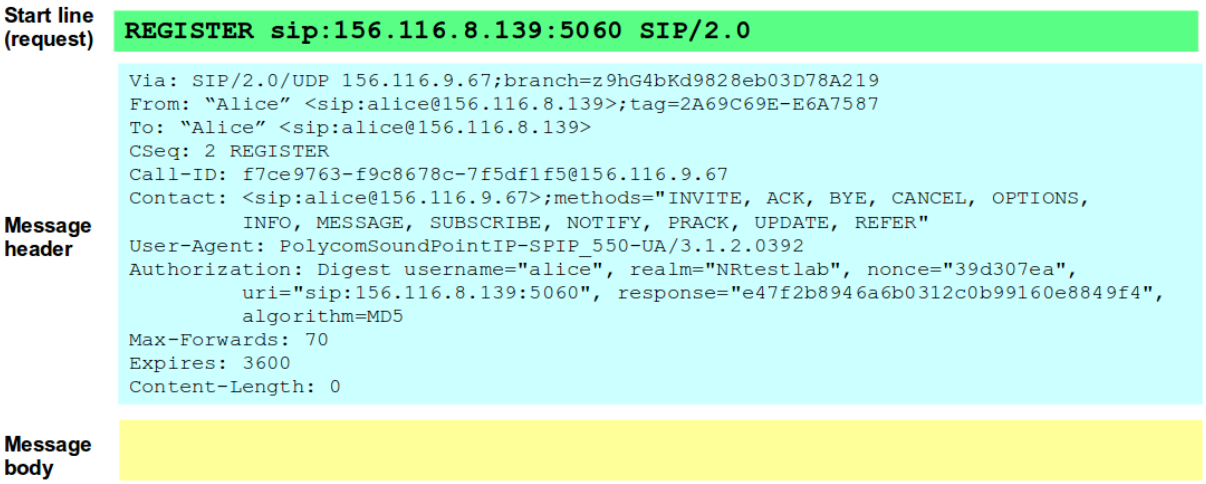



Figure 2.4: The SIP message structure.

Since these two SIP `REGISTER` transactions are related and belong in the same context, they form a SIP *dialog*. A dialog represents a peer-to-peer SIP relationship between two user agents [77]. While messages 1-3 and 4-6 in Figure 2.3 on the facing page each are one transaction, the messages 1-6 constitute a SIP dialog. Messages having the same *Call-ID*, *From-tag* and *To-tag* values belong to the same dialog.

All SIP messages have the same structure as depicted in Figure 2.4, with a start line, message header and message body:

1. The *start line* identifies the type of request and where the request is destined to. In Figure 2.4 the start line indicates that this message is a registration request being sent to IP address 156.116.8.139.

2. The *message headers* are textual, and always in the format

   `<header_name>: <header_value>`

   where the header value can contain one or more parameters. These headers include context information for the SIP transaction.

3. A *message body* is optional, and may include additional information. For the SIP `REGISTER`, there is no message body. However, for SIP `INVITE` message, RTP specific data is embedded in the message body using the Session Description Protocol (SDP) [32].

Having demonstrated the structure and flow of the SIP messages, the next section will introduce the scenarios where SIP is used, which further leads to attack scenarios.

### 2.2.3 SIP scenarios

In a common SIP network architecture, as depicted in Figure 2.2 on page 11, we selected three scenarios where identity in SIP needs to be handled. Scenario *I* between the UA and the local SIP server; Scenario *II* between SIP servers; and Scenario *III* end-to-end between the participating UAs. These three SIP scenarios are depicted in Figure 2.5 on the following page.

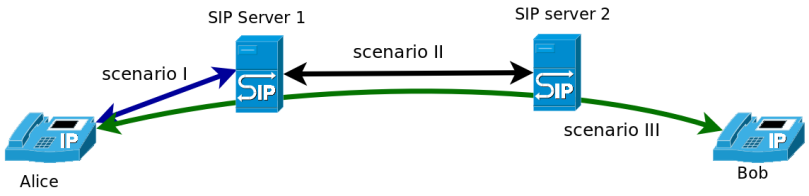



Figure 2.5: Three different usage scenarios where authentication in SIP is desired.

Scenario I between the UA and the local SIP server is relevant when the UA comes online and before any outgoing calls can be placed. Then, the UA must register itself to a local SIP server. During the SIP `REGISTER` dialog, the server usually challenges the UA to authenticate. Before placing a call (sending a SIP `INVITE`), the UA might be challenged again by the server to authenticate. The most common authentication method used between UA and server today is the Digest Access Authentication (DAA) [83, page 77].

Scenario II handles the authentication between SIP servers to achieve trust between SIP servers. It is not desirable to have SIP traffic handled by an unknown or untrusted SIP server that might have malicious intent. However, since most SIP servers today use some kind of SIP peering, discussed in Paper D, the relationships between servers are normally static and pre-defined. Therefore the identities between SIP servers are often predetermined by other security mechanisms than what are offered by SIP (like IPSec, TLS etc.).

Scenario III is about end-to-end (e2e) authentication, which determines the identity of both the caller and the callee across different SIP domains. This is of particular importance and not easily attained in SIP [82, page 355]. There is an increased threat and fear for both VoIP phishing and SPIT (Spam over Internet Telephony), that might seriously affect SIP-based VoIP services. By enforcing end-to-end authentication in SIP, these threats might be mitigated or prevented.

Other SIP scenarios where SIP authentication needs to be applied are identified in the literature [57]. The scenario “end-to-middle security” (e2m) requires that a SIP UA should be able to protect SIP message body and/or headers from SIP intermediates (SIP servers), except those that provide services based on the SIP content. An example scenario could be where SIP is used for Instant Messaging and a SIP server provide logging service that need to read the content of the SIP messages. This research does not consider the e2m scenario, since we found little practical applicability using this scenario.

Two other SIP scenarios are identified and discussed by Barnes [3]. These SIP scenarios complement the e2m scenario by the “middle-to-middle” (m2m), and “middle-to-end” (m2e) scenarios. The m2m scenario is similar to SIP Scenario II, but the focus on these scenarios is to apply confidentiality protection of the SIP messages rather than authentication. Therefore, we restrict this research to the three SIP scenarios (I-III) defined above.

## 2.3 VoIP threat and attack taxonomy

An extensive list of reported and known VoIP vulnerabilities and VoIP security research is given by Keromytis [41, Chapter 3]. A vulnerability can be defined as a flaw or weakness

in the protocol design, implementation or operation. A threat is a possible danger that might exploit a vulnerability [81]. There are several possible ways to categorize security threats. The threat taxonomy given by VOIPSA [97] defines potential security threats to VoIP deployments, services and users categorized as:

**Social threats** represent threats similar to those to email, due to the similarities to the email service (discussed in Paper D). This include misrepresenting of identity or content, identity theft and theft of service. The security attacks presented in Papers C and E can be categorized as theft of service and misrepresenting of identity. To overcome these threats, we discuss various approaches to providing improved authenticated identities and prevent identity theft.

**Eavesdropping** is the result of monitoring the signaling (SIP) and/or the multimedia content (RTP traffic), without altering the content. By eavesdropping on the call, potentially the entire call can be recorded. Contextual information about the participants, like username, contact-address and user equipment can also be obtained. Even if the signaling and/or multimedia content is encrypted, for an attacker it might be of value to know that a call was made at a particular time.

**Interception and modification** listen in on and modify the signaling and/or multimedia content. Threats include conversation impersonation and hijacking. The attacks presented in paper C and E utilize this threat, since the SIP signaling was intercepted and modified.

**Service abuse** describes threats that cover improper use of a VoIP provider's services. Threats include bypassing the VoIP providers authentication mechanism, registration hijacking and exploiting misconfigured equipment. The goals of these threats are usually financial gain. The attack on registration discussed in Paper E, can also be classified as registration hijacking and thus service abuse.

**Interruption of service** include known types of performance latency and denial of service (DoS) threats, but also cover physical intrusion and loss of power. An example of a DoS threat include receiving large amount of illicit network traffic that renders the VoIP service unavailable due to resource exhaustion.

We focus primarily to counter threats that are categorized as *social threats*, *interception and modification* and *service abuse*. In this research, threats to SIP authentication and call-setup are identified and analyzed, and security attacks implemented.

A security attack is an *implemented threat*; defined as an intentional act or assault, on system security that derives from a threat [81]. Successful attacks are often designed by looking at the problem in a completely different way. The number of attacks are increasing for each year, and "Cyber Crime" is growing fast [56]. Specialized and targeted attacks, derived from what is called "Advanced Persistent Threats" (APTs) are emerging [11]. In the literature, security attacks are classified into *passive attacks* and *active attacks* [36, 81]:

**Passive attacks** are derived from the eavesdropping threat. A passive attack does not alter the system resources (network traffic), but learns from the information leakage. This kind of

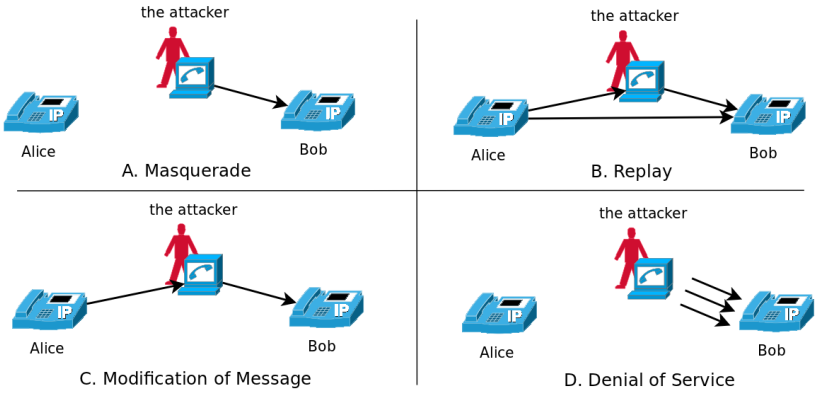



Figure 2.6: The four different active attacks with modification of the network stream ($B$ and $C$) or the creation of a false network stream ($A$ and $D$).

attack can be difficult to detect, therefore the main focus is often to prevent it. Stallings identified two kinds of passive attacks [87, page 7]: The first attack is *release of message contents*, where information can be deducted from the network traffic eavesdropped such as listening in on a VoIP conversation or capturing an email message. The second attack is *traffic analysis*, where the content of the traffic could not be obtained. Even if encryption is used to mask the traffic content, the attacker could observe the length of the message, time of communication, and involved UAs. This observation of communication pattern might be useful information for the attacker, even if the content is not available.

**Active attacks** involve modification of the data, or the creation of false data (network traffic). The following four active attacks are identified, as depicted in Figure 2.6: (*A*) Using *masquerade* the attacker pretends to be a different entity when communicating with Bob. (*B*) In a *replay* the attacker re-sends earlier captured message from Alice to Bob. (*C*) In a *modification of message*, also called man-in-the-middle (MitM) attack, the message from Alice is altered on its way to Bob by the attacker. (*D*) In a *denial of service* (DoS) attack, the purpose is to overload the victim with messages (network traffic) so as to degrade performance.

A successful real-world attack often uses a combination of these attacks. The attack presented in Paper E, is an active "modification of message" attack. In contrast, the attack presented in Paper C, is a combination of passive "release of message content" (capture of Call-ID, username, From- and To-header values) and active "masquerade" attack (sending SIP `CANCEL` or `BYE` message to block out Alice and Bob, and finish the SIP `INVITE` dialog with the SIP proxy).

## 2.4 Industrial objectives and requirements

In the context of this research, we were involved with the formulation of the requirements of the public tenders for a new VoIP solution for Buskerud County Municipality (3000 UAs) and Akershus County Municipality (4700 UAs) [24]. The operational security requirements in these tenders included strong authentication; these are requirements that SIP digest authentication (DAA) does not cover adequately.

The telecommunication industry have had their telephone services engineered for high avail-

ability, aiming for 99.999% uptime [42] which only allows for 5.26 minutes downtime per year. The telecommunication industry has been conservative with a fifteen-year product cycle for PSTN [2]. Moving to VoIP will likely require a faster product cycle, since the IT-industry has a five-year product cycle. This conservative tradition may result in reluctance to adopt new or non-standardized network protocols and security services. In this research, we therefore include an upgrade migration path and points out real-world security concerns (as shown in Paper H and Section 5.1).

# Chapter 3

# State of knowledge

> "For VoIP to succeed in the long term, VoIP services must offer similar security and protection levels to what is available today in the PSTN."
>
> *— Dorgham Sisalem, et al., "SIP Security" (2009)*

This chapter defines authentication and introduces the current state of knowledge for security in SIP with focus on authentication.

The SIP core specification document [77] endorses and recommends the use of security mechanisms to secure the SIP communication. In the context of SIP communication in this research, it must be stressed that the user herself is not authenticated and identified, but the user's phone (UA).

## 3.1 Authentication

Authentication is about verifying claimed identity; that is, the process of establishing confidence in user identities [36]. An identity can be defined as an electronic representation of an entity [6, page 353]. The entity presents a claimed attribute value to the authentication system that "acts as evidence to prove the binding between the attribute and that for which it is claimed" [81]. When the entity authenticate and provide information to the verifier, the information provided can usually be grouped into three categories:

1. *Something the entity knows* (secrets), which include passwords and PIN codes.
2. *Something the entity has* (tokens), which include smart cards and encryption keys.
3. *Something the entity is* (biometrics), which include fingerprints and retina/iris characteristics.

The suggested authentication methods presented in this research can be categorized as "secrets", or, *something the entity knows*. Methods for comparing authentication methods across the three categories [7] exist, but are outside the scope of this research. The authentication systems that use secrets are often based on one way hash functions and shared secrets, such as the

DAA, or public-key cryptography [79, Section 3.2]. Multi-factor authentication systems, utilize two or three of these categories and are not considered in this research.

This research focuses on the technical challenge when authentication involves SIP nodes communicating over a network. It must be stressed that in this research an entity is either a user's UA or a SIP server. The UA is identified by a phone-number/username and the IP-address/hostname pair, denoted as an Address-of-Record (AoR). Equally important is it to establish the identity of the communicating peer, i.e., the SIP server, as discussed in Paper G.

## 3.2 Digest Access Authentication

The Digest Access Authentication (DAA) is currently the most common authentication mechanism for SIP. DAA is simple, but rather insecure. It is the only authentication mechanism that must be mandatorily supported [77, Section 22]. DAA uses the MD5 hash function [49] and a challenge-response pattern, and relies on a shared secret between client and server within a SIP domain [19]. DAA is performed during the SIP `REGISTER` handshake between the UA and the SIP server, as depicted in Figure 2.3 on page 12. The UA receives a nonce value from the SIP server, computes a digest hash value over the nonce, the shared secret and some other SIP header values, and send it back to the SIP server. The SIP server then computes the same digest hash, and compares these. If both digests are identical, the UA is authenticated. The DAA is weak and vulnerable to a serious real-world attack, as described in Papers B and E.

Based on the DAA, Undrey [96] proposed a more flexible use of variables protected by the digest. His paper addresses the shortcomings of DAA and suggests to allow the server to decide which headers it requires to be included and protected by the digest computation. Unfortunately, his approach does not require specific headers fields to be included. His approach is therefore vulnerable to the same vulnerability that is presented and implemented in Papers B and E.

Yang et al. [100] also conclude that DAA is weak. They argue that, since DAA is vulnerable to off-line password guessing attacks, a more secure authentication method would be required. They propose an authentication method based on Diffie-Hellman [13, 68]. Unfortunately, they do not discuss nor add any additional SIP header values in their new authentication scheme. Therefore, their solution is also vulnerable to the same registration attack discussed above.

In Universal Mobile Telecommunications System (UMTS) networks, the Authentication and Key Agreement (AKA) [1] mechanism performs authentication and session key distribution. Work has been performed to map AKA parameters into the DAA [55, 95]. However, AKA is designed to run on an "IMS Identity Module" that usually resides in a tamper-resistant smart card. The specification document also mentions explicit that DAA with AKA is vulnerable to the same security threats as ordinary DAA [55].

## 3.3 Authentication using Secure MIME

Secure MIME (S/MIME) [61] is an end-to-end integrity and authentication mechanism that is presented in the SIP core specification document RFC3261 [77, Section 23.4]. The entire SIP message is encapsulated in a specific SIP message using MIME [20], which is signed and optionally encrypted. The receiving UA checks whether the sending UA's certificate is signed by

a trusted authority. By doing certificate validation, the UA can verify the authenticity of the sending entity. Since S/MIME depends on end-user certificates, the UAs must support multiple root certificates since no consolidated certificate authority exists. Additionally, certificate handling issues, such as revocation and renewal, complicate the use of certificates. There has been rather limited industry support for S/MIME. As an indication for this claim we mention that the international SIP interoperability test conference "SIPit 28"[1] gathered 19 vendors with 40 distinct SIP implementations where none supported S/MIME.

When encapsulating and tunneling the entire SIP message in a S/MIME body, it is the burden of the UAS to determine which header changes in the outer SIP message were legitimate and which might be a security attack. Therefore, developers have worked on finding a way of signing just the needed SIP headers to establish the identity of the sender. The 'message/sipfrag' [84] provides a method to include only a subset of the SIP message in a S/MIME message. Unfortunately, the sipfrag document outlines only the method itself, and does not list the required SIP headers.

The SIP Authenticated Identity Body (AIB) format [60] is an effort to extend the sipfrag method. AIB protects integrity and optionally encrypts part of the SIP message in a S/MIME, as in sipfrag. The identity of a UA is held in the `From`-header field of an AIB, and is thus always included. However, AIB mandates the inclusion of other SIP headers to assist in detection of integrity violations. These headers are `Contact`, `Date`, `Call-ID`, and `From` header values. Since both sipfrag and AIB rely on S/MIME, they suffer from the same certificate issues as S/MIME. As far as we are aware, neither sipfrag nor AIB have seen adoption in the industry.

## 3.4 Transport Layer Security

Transport Layer Security (TLS) [12] support for SIP, called "Secure SIP" and denoted *SIPS*, has gained some industry momentum. At the latest SIPit conference ("SIPit 28"), 50% of the 40 SIP implementation supported TLS with mutual authentication. In the core SIP specification [77], TLS is recommended to be deployed on SIP servers.

TLS is designed to make use of TCP to provide a protected end-to-end communication between two endpoints. The application data, here SIP, are encrypted and integrity-protected. The communicating endpoints authenticate using digital certificate, usually X.509 certificates, require a PKI. According to the annual "CSI Computer Crime and Security Survey" [71], only 35% of the surveyed companies have invested and deployed a PKI. The use of PKI is also riddled with security challenges and fundamental problems, as listed in [17, Chapter 20].

TLS does *not* offer end-to-end confidentiality and integrity protection of SIP messages, since the TLS connection must be terminated and initiated for each hop between intermediate SIP servers. While the use of TLS also restricts SIP to use TCP as transport protocol, we experienced that SIP often is used over UDP in real-life industry scenarios [88]. Work has been done to add support for TLS over UDP, called Datagram Transport Layer Security (DTLS) [69], and extensions for using SIP over DTLS are considered [38].

By using TLS, SIP relies on a lower communication layer protocol to enforce security mechanisms (TCP or UDP). This research focuses on solving authentication issues on the application

[1] The SIPit28 summary is available at: `https://www.sipit.net/SIPit28_summary`

layer.

## 3.5 Other approaches to authentication

Two other methods for handling identity have emerged within the Internet Engineering Task Force (IETF):

1. The *P-Asserted Identity* [37] is intended to work within a trusted environment. An unprotected SIP header is appended by the UAs SIP server that informs the receiving SIP server that the identity of the UA has been checked and thus can be trusted. However, since the SIP header is sent in clear rather than protected by cryptography methods, it can easily be removed by an attacker without any of the communicating peers noticing this.

2. The *SIP Strong Identity* [62] introduces a new SIP service, the "authentication service", which signs a hash over selected SIP header values, and includes the signature as a SIP header along with a URI that points to the sender's certificate. The receiver computes the same hash and compares the results. However, using this method, only the client is authenticated and an attacker can remove these headers without implications.

Note that both "P-Asserted Identity" and "SIP Strong Identity" rely on a successful DAA authentication to be applicable. These are also applied by the SIP servers rather than the clients themselves, and are thus only providing indirect authentication of the client since the server is authenticating on behalf of the client. None of these authentication methods have seen any widespread deployment yet [83, Chapter 6].

Palmieri et al. [58, 59], dismiss DAA as a usable authentication method, and instead craft a new authentication schema with digital signatures based on public-key encryption. But since they rely on certificates, their solution suffers under similar certification handling issues as S/MIME and TLS. They also admit that relying on public key infrastructure (PKI) is both difficult and costly to implement. Liao et at. [45], propose an improved authentication in SIP with self-signed public keys on elliptic curves. However, Liao's proposal uses smart-cards to store authentication data and rely on a trusted third party [44].

From the different authentication mechanisms discussed in this chapter, we see that most of the approaches neither consider nor discuss which SIP header values to integrity protect that identify the SIP client or server. The second group rely on PKI with all the drawbacks and challenges of handling the X.509 certificates. The third group (TLS) has industrial momentum, but only offers hop-by-hop protection and operate on the transport layer. Our research contributes to enhanced security for SIP authentication on the application layer without demanding use of X.509 certificates or PKI.

# Chapter 4

# Contributions and summary of papers

"No security measures are guaranteed; all we can do is reduce the odds."

*– Bruce Schneier, "Crypto-Gram Newsletter" (November 2004)*

The second part of the dissertation consists of research contributions through eight published papers in peer-reviewed international conferences or journals. The author of this thesis is the principal contributor and first author of papers D-H. He is the second author of paper B and C, and third author of paper A.

**Paper A** Lothar Fritsch, Arne-Kristian Groven, Lars Strand, Wolfgang Leister and Anders Moen Hagalisletto. "A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project", *International Journal on Advances in Security*, pages 129-141, Volume 2, Number 2&3, 2009, ISSN 1942-2636.

**Paper B** Anders Moen Hagalisletto and Lars Strand. "Formal modeling of authentication in SIP registration", *The $2^{nd}$ International Conference on Emerging Security Information, Systems and Technologies (SECURWARE)*, pages 16-21, Aug 2008, Cap Esterel, France.

**Paper C** Anders Moen Hagalisletto and Lars Strand. "Designing Attacks on SIP Call Setup", *International Journal of Applied Cryptography*, Volume 2, Number 1, July 2010, pages 13-22.

**Paper D** Lars Strand and Wolfgang Leister. "A Survey of SIP Peering", *NATO ASI - Architects of Secure Networks (ASIGE)*, May 2010, Genova, Italy.

**Paper E** Lars Strand and Wolfgang Leister. "Improving SIP authentication", *The $10^{th}$ International Conference on Networks (ICN)*, pages 164-169, Jan 2011, St. Maarten, The Netherlands Antilles.

**Paper F** Lars Strand, Josef Noll and Wolfgang Leister. "Generic Security Services API authentication support for the Session Initiation Protocol", *The $7^{th}$ Advanced International Conference on Telecommunications (AICT)*, pages 117-122, Mar 2011, St. Maarten, The Netherlands Antilles.

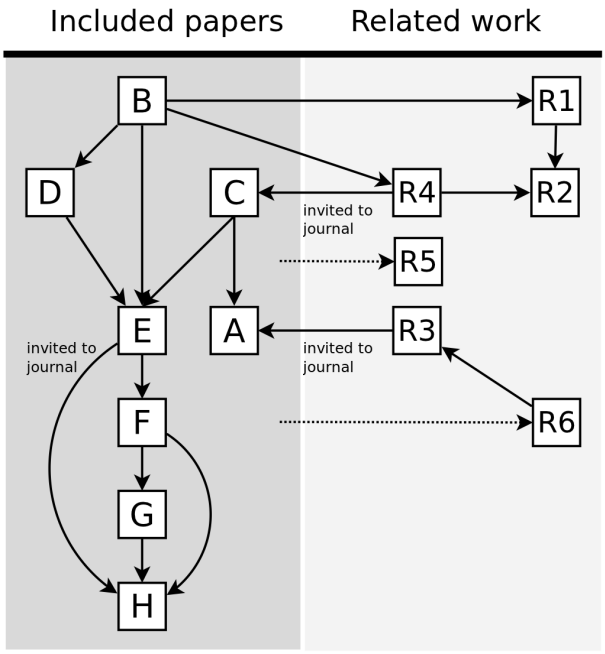



Figure 4.1: The thematic progression between the research contribution (included papers) and related work.

**Paper G** Lars Strand, Wolfgang Leister and Alan Duric. "Migration towards a more secure authentication in the Session Initiation Protocol", *The 5$^{th}$ International Conference on Emerging Security Information, Systems and Technologies (SECURWARE)*, pages 57-62, Aug 2011, Nice, France.

**Paper H** Lars Strand and Wolfgang Leister. "Advancement towards secure authentication in the Session Initiation Protocol", Submitted to *International Journal On Advances in Security*, ISSN 1942-2636.

The thematic progression, and how the different papers are interconnected, is shown in Figure 4.1. The papers are not based on independent research, but are closely interrelated thematically.

We start with journal *Paper A* which gives an overview of the research framework for the EUX2010sec project, which this research is a part of. The project's main objective is to find out from several perspectives if it is possible to use open source software for VoIP in public organizations and in the industry. Besides looking into metrics for open source software quality, Paper A emphasizes security models, the VoIP testbed, and formal protocol analysis. Results from the latter are treated in depth in Papers B and C.

After reviewing VoIP communication and authentication security challenges in SIP, analysis of authentication in a real-world SIP installation was performed and presented in *Paper B*. The findings concluded that the SIP authentication was weak and could be vulnerable to a security attack. A continuation of this work was performed where the call-setup request method (SIP `INVITE`) was analyzed and a weakness detected (call-hijack attack). This work was presented in paper R4, which was invited to journal and included as *Paper C*.

In *Paper D*, real-world security challenges are presented that elaborate and explain the problem domain for papers E-H. In *Paper E*, the vulnerability found in paper B is analyzed, a security attack implemented that confirm the vulnerability, and a mitigation strategy is suggested that close the vulnerability. However, a more flexible and stronger authentication mechanisms was desired. A new approach to authentication is presented in *Paper F*, where a security layer (GSS-API) is added to SIP that is both flexible and offers more secure authentication mechanisms. Another approach is offered in *Paper G*, where a two-step migration to stronger authentication in SIP is presented. Finally, the research is summarized in journal *Paper H*, where the different proposed approaches are discussed against real-world security concern and SIP authentication scenarios.

In Section 1.3, research questions were established, and research goals were identified that would address these research questions. This chapter revisits the research goals and explains how these goals are addressed by the included papers. The research goals documented in Section 1.3 are:

- **Goal 1:** Explore VoIP communication and authentication security challenges in SIP.
  - **G1 (a):** A list of security challenges in SIP.
  - **G1 (b):** The list given in *G1 (a)* prioritized and ranked as most severe.
  - **G1 (c):** A list of SIP authentication scenarios.
- **Goal 2:** Identify and analyze SIP authentication vulnerabilities and threats by demonstrating security attacks.
  - **G2 (a):** Enable VoIP testbed for security testing of selected VoIP SIP scenarios.
  - **G2 (b):** List of security attacks and their consequences for VoIP SIP communication.
  - **G2 (c):** Analysis, discussion and implementation of a least two security attacks targeting the SIP protocol.
- **Goal 3:** Analyze current mitigation strategies and suggest more secure authentication mechanisms.
  - **G3 (a):** A discussion and analysis of current mitigation strategies for SIP authentication.
  - **G3 (b):** Apply engineering methodology to improve SIP authentication.
- **Goal 4:** Evaluate the suggested authentication mechanisms, and address real-world security concerns.

  The expected outcome of this goal is identified as:
  - **G4 (a):** An evaluation of the suggested new authentication mechanisms provided by *G3 (b)*. Discuss against SIP authentication scenarios provided by *G1 (c)*, and security attacks provided by *G2 (b)*.

- **G4 (b):** Based on *G4 (a)*, recommend a new authentication mechanism.
- **G4 (c):** Provide an industrial uptake strategy path that addresses real-world security concerns.

The scientific contributions are mapped to the research goals in Table 4.1. *Part I* of the thesis gives an introduction to the problem area, describe the current "state-of-the-art" research and mitigation strategies within the problem area, and, to some extend, evaluated the proposed theoretical approach. Thus, research goals 1, 3 and 4 are addressed in Part I. In the following sections, each paper is summarized and explained how the contribution match the research goals. *Part II* of the thesis contains the full papers.

| Scientific contributions | Research Goals |
|---|---|
| Part I of the thesis | Goal 1, 3, 4 |
| Paper A | Goal 1, 2 |
| Paper B | Goal 2 |
| Paper C | Goal 2 |
| Paper D | Goal 1, 2 |
| Paper E | Goal 2, 3 |
| Paper F | Goal 3 |
| Paper G | Goal 1, 3 |
| Paper H | Goal 2, 3, 4 |

Table 4.1: Mapping of scientific contributions and research goals.

## 4.1 Paper A: Contributions and summary

## A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project

Paper A [23] provides an overview of the EUX2010sec research project, and is based on conference paper (R3) [22] (which was invited to journal). The journal paper provides an overview of the research project, and states that the overall goal of the project is to improve the level of security and awareness when developing, installing and using open source VoIP solutions. Three different research activities conducted in the research project are presented:

1. The **security model** activity was to gather and analyze VoIP stakeholder requirements. Stakeholders were identified (in the Norwegian market), contacted and interviewed. The stakeholders usage scenarios and requirements concerning VoIP security were classified into pre-defined scenario profiles derived from VoIP literature. The results are presented by Fritsch and Groven [21].

2. **Testbed systems** were built with the same VoIP technology (hardware and software) and configurations as used by our project partners. Real-world VoIP traffic was routed through the various VoIP testbed installations for analysis and testing. Details about the testbed testing is given in [88] (R6).

3. **Formal protocol analysis** was used for initial security analysis of the SIP protocol. This activity would assist in revealing unknown protocol failures and wrongful implementation of protocols. The analysis tool PROSA [28] was used in the project.

In addition to these three research activities, some work was performed on open source and maturity models, which are presented elsewhere [26, 27]. The software maturity models are beyond the scope of this thesis.

The paper presented some initial result, including findings based on requirements elicitation from stakeholders. The largest worries (threats) concerning VoIP security were stated around the topics of identity fraud, fraudulent service usage and losses due to fraudulent outgoing calls into a billed long-distance network. To counter these threats, a strong authentication mechanism was desired. This research contributed with the testbed systems and security testing, thus addressing research Goal G2 (a). The results from the security testing was used as input to the formal protocol analysis. Weaknesses and attacks found from the formal protocol analysis were implemented and verified in the testbed, thus addressing research Goal G2 (b) and (c). By providing background discussion of VoIP and security challenges, Paper A also achieved aspects of research Goal 1 (a).

## 4.2 Paper B: Contributions and summary

## Formal modeling of authentication in SIP registration

Paper B [29] presents the first results of our analysis of VoIP systems, and addresses research Goal 2. In this paper, we found and presented an vulnerability in the DAA. We developed a VoIP testbed for testing of a real-world VoIP system that enabled us to explore and test security challenges, and we identified and analyzed SIP authentication vulnerabilities.

Whether a VoIP configuration is considered secure depends on two factors: (*1*) the requirements specified by the given security policy for a particular installation, and (*2*) whether these requirements are covered by the implemented security mechanisms.

The results presented in this paper are based on a case study taken from a medium sized Norwegian company with 100 employees. The company used VoIP and counted 127 registered SIP phones (including faxes and both soft- and hard-phones). All outbound calls were sent to a VoIP provider which routed the calls further. All phones had to register to an internal SIP server running Asterisk at regular intervals. During the registration they performed DAA. This setup address research Goal 2 (a), since we replicated the VoIP setup in our testbed.

By using network tools, we obtained VoIP traffic of the registration process, and combined by relevant SIP specification documents [77] [19], a precise specification of SIP registration with DAA was generated. Static analysis and simulation was performed on the specification by a locally developed tool PROSA [28].

In our case study, a disgruntled employee could easily exploit vulnerabilities of the company's VoIP system, and obtain, intercept and modify the same network traffic using the same tools as used in this paper. Therefore we are simulating an attacker as powerful as the Dolev Yao attacker [14] which can intercept and modify any message carried over the network. However, the attacker can not brute force or break the underlying hashing or encryption algorithms.

Several trivial denial of service attacks could be launched by the attacker by changing SIP `REGISTER` message headers. This resulted in exclusion of the "client" by the SIP server, since it was considered corrupted. A more serious attack was constructed by hijacking the SIP registration handshake. In the attack, the attacker is able to manipulate the UA to believe it has successfully registered to the SIP registration server. While the SIP registration server is fooled to believe that UA can be contacted using the corrupt address (IP/hostname). The result is that all future phone calls would be routed to the attackers contact address. This attack address research Goal 2 (c).

The paper also concludes that for large and complex protocols like SIP, it is possible to formally specify the details of message exchange on a level that permits security analysis. The use of real-world VoIP traffic has two importance implications: (*1*) formalization of network protocols is much faster using network tools to obtain and analyze network traffic, and (*2*) using traces from such tools give realistic specifications that are close to the implementation.

## 4.3 Paper C: Contributions and summary

## Designing Attacks on SIP Call Setup

Paper C [30] was an invited journal paper based on a conference paper [31] (R4). This paper uses the same method as presented in Paper B, but has its focus on the SIP call-setup (SIP `INVITE`), instead of the SIP registration (SIP `REGISTER`). In this paper we present a call-hijack attack that was detected, analyzed and implemented. The attack address research Goal 2 (c).

Paper C use formal modeling of SIP in order to (*1*) verify whether the Asterisk implementation of SIP follows the specifications, and (*2*) perform attacks exploiting weaknesses in the protocol definitions and its implementations.

Network traffic generated from VoIP sessions in the testbed was used as input data for the formal modeling. The attack found was implemented and tested in the testbed, thus addressing research Goal 2 (a).

A SIP `INVITE` message handshake is used whenever a multimedia session (usually a call) is initiated. Before a UA can initiate a SIP `INVITE` session, it must have completed the SIP registration explained and analyzed in Paper B. Paper C uses the same VoIP system setup as Paper B.

SIP defines distinct functionality for registration, call setup and modification, call control and mid-call signaling. The following SIP methods, called *sub-protocols* in this paper, are analyzed and used: *digest access authentication* (SIP `REGISTER`), *call setup* (SIP `INVITE`) and *call teardown* (SIP `BYE`). We found that the Asterisk implementation diverges from the specification in three ways:

1. Alice can hear the ringing signal in her, the caller's, UA *before* the callee's UA (Bob) is authenticated to the SIP server. Hence Alice is fooled to believe that Bob's UA is calling, which is not the case.

2. The acknowledgment message during the SIP `INVITE` handshake from the SIP server to Bob's UA is sent *before* an acknowledgments message has been received from Alice's

UA. As a consequence, Bob's UA is misled to believe that Alice has acknowledged Bob's message.

3. After Alice initiates a teardown (hangup), the SIP `OK` message from the SIP server is sent *before* the related `OK` message is sent from Bob. This breaks the specification, since it would implicate that Bob has received the `BYE` message, which is not the case in the implementation.

The consequences of these implementation deviations were not examined further, and should be an area for future work. A hypothesis is that some or all of these deviations are implemented to "optimize" and enhance the user experience when calling.

By using an attacker as powerful as the Dolev Yao attacker [14], a severe call-setup attack was found. The attacker intercepts the SIP `INVITE` message handshake to learn the dialog identifiers and injects a SIP `CANCEL` to both the caller and the callee to tear down the call prematurely, and redirects the multimedia session (the RTP traffic) to a destination of his choosing. This attack effectively breaks the authenticity of the participants, since we no longer can trust the identity of the UAs involved in the phone call. As a consequence the attacker can set up an arbitrary call, that Alice is billed for. In addition, the CDR logs, that telephony providers are obliged to carry out by legislation, are incorrect.

## 4.4 Paper D: Contributions and summary

## A Survey of SIP Peering

Paper D [89] presents a survey of SIP peering architectures and explains technical and security implications, and thus addressing Research Goals 1 and 2. The original design for SIP communication, the so-called "email model", intended to enable each SIP UA to connect to any other SIP server over the Internet. The idea was to have the same communication model for SIP as for email (SMTP). However, SIP has not seen global reachability for three main reasons: *1*) The telephony providers have traditionally collected termination fees between communication partners (other providers). If everyone directly is able to connect to everyone, no business relationships between providers are necessary. Therefore, the carriers have no economic incentive to switch to a global reachable SIP addressing scheme. *2*) Operators of public telephony services need to comply to a range of legal regulatory requirements. These requirements are applicable for the PSTN with clear boundaries between telephony operators and telephony users. *3*) There are a range of security concerns and challenges to which no simple solution exists. This list of security concerns and challenges fulfill research Goal 1 (a):

(a) Unwanted calls, also known as "Spam over Internet Telephony" (SPIT), are a threat to the VoIP infrastructure. Since there are currently only a few open SIP servers, SPIT has not yet grown to be a widespread problem compared to *email spam*. SPIT is harder to prevent than email spam, since VoIP calls are interactive — the content or the intentions of a call are not identifiable in advance, before the receiver picks up the phone. Therefore, filters for SPIT cannot be applied as easily as spam filters for email. Also social factors are important, since

one usually picks up the phone when it rings, instead of being able to choose when and how often to check email. Providers fear that SPIT could become common if they open up their SIP services to the Internet [74].

(b) Assuring the *identity* of the caller. Signaling in PSTN has traditionally been trusted between carriers, and by end users (caller-id). This trust is not applicable to open SIP servers, since the SIP `INVITE` message can come from any user on the Internet and the be easily spoofed.

(c) *Denial of Service* attacks are threats to availability. The *email model* is particularly vulnerable to DoS attacks, since SIP servers need to accept request from anyone on the Internet. This makes it hard to guarantee a stringent Quality of Service (QoS) agreement to customers and other SIP providers. DoS attacks on SIP have been studied [101], and work has been done to develop specialized security mechanisms that aims to prevent SIP DoS attacks [15].

These security challenges and threats can results in attacks and are discussed, thus addressing research Goal 2 (b). As a result of these concerns, VoIP providers have not deployed open SIP servers. Instead, VoIP calls are either transcoded and sent through the PSTN network, or VoIP providers set up "VoIP peering".

Security considerations related to SIP are also applicable in a peering relationship. But to demand that the VoIP providers use and enforce network security mechanisms like TLS/IPSec and deploy secure and scalable network design everywhere and end-to-end is unrealistic. Thus, focus should be on enhancing existing security mechanisms already present in SIP.

SIP peering mandates a trust relationship between peering SIP proxies, but the available authentication methods do no provide mutual authentication between UAs. The paper concludes that more work are needed on authentication in SIP to solve the presented security challenges.

## 4.5 Paper E: Contributions and summary

## Improving SIP authentication

Paper E [91] extends the work started on in paper B. The vulnerability analyzed in paper B was implemented and tested in our VoIP testbed with a replicated industry partners VoIP configuration, addressing research Goal 2 (a). The attack successfully hijacked a client's (Alice) SIP `REGISTER` handshake, resulting in false client registration data at the SIP location server.

The attack exploits the unprotected SIP header `Contact`, which carries the values of the contact points (hostname/IP-address) for the client to register at the SIP server. The attack was implemented by using an network tool that can change a network stream in *real-time*. By changing the value of the `Contact` header during the SIP `REGISTER` transaction, as depicted in Figure 4.2, all requests, and hence all calls (sessions) to Alice's UA will be diverted to a hostname or IP-address controlled by an attacker. The attack goes undetected by both the client and SIP server, and both peers believe the SIP `REGISTER` was completed successfully without interruption. The explanation and demonstration of this attack addresses Goal 2 (c).

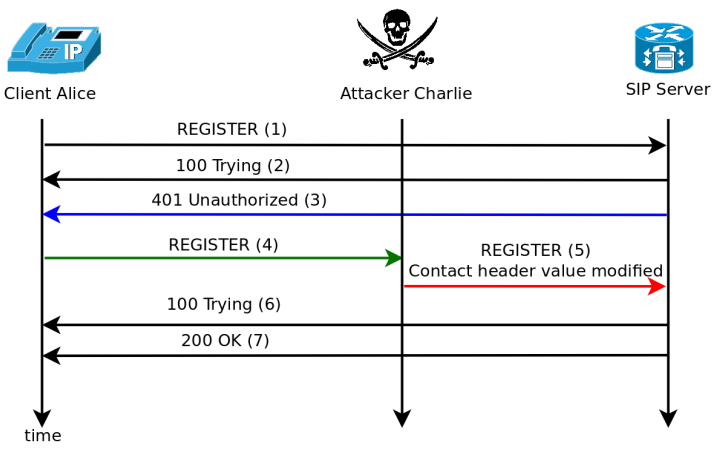



Figure 4.2: The attacker Charlie can modify the `Contact` header value, and thereby have all Alice's calls redirected to him.

To mitigate this attack, the `Contact` header value must also be integrity protected by the DAA digest. We modified the DAA to also include the `Contact` header. Thus any modification of the `Contact` value will be detected. The implemented change only resulted in a negligible computation performance penalty; – for 100.000 authentication requests using DAA, only 0.44 seconds separated the original DAA from our modified DAA. This mitigation strategy and suggestion for an improved DAA addresses Goal 3 (b).

The original SIP designers focused on functionality and compliance at the cost of security. A more thorough investigation of the SIP DAA in the design phase would have revealed the vulnerability and attack presented, and the vulnerability could have been prevented early on.

Still, DAA remains weak and vulnerable to other attacks, even though we have developed an enhanced DAA mechanism. Thus, we are looking in the upcoming papers for enhanced authentication methods to replace the DAA mechanism.

## 4.6 Paper F: Contributions and summary

### Generic Security Services API authentication support for the Session Initiation Protocol

Paper F [93] takes into account that SIP supports a wide range of functionalities that can be utilized, ranging from mobile handsets to high-end servers, each with different security requirements. Different security requirements may use different authentication methods depending on the usage and threat scenario. For example, a mobile handset may have different requirements for authentication than authentication between SIP servers. Additional requirements like power consumption and computational power must be considered. Thus, adding new security services to SIP to improve the security design and meet different security requirements can be challenging.

The GSS-API [46] provides a generic interface for application layer protocols like SIP, with a layer of abstraction for different security services like authentication, integrity or confidential-

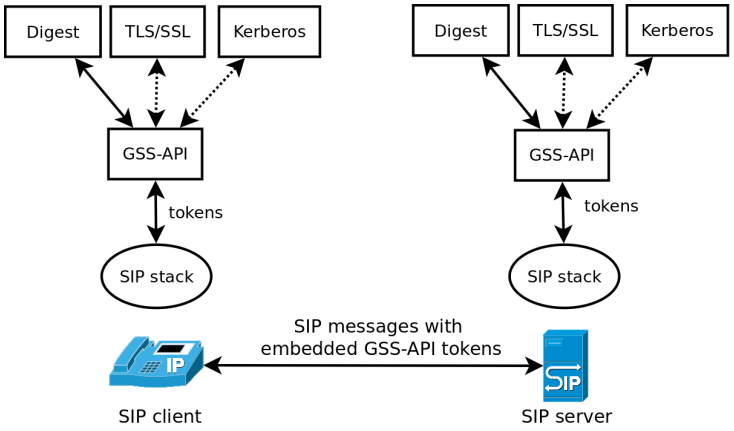



Figure 4.3: The GSS-API interface in SIP.

ity. With the GSS-API, SIP does not need to support or implement every authentication method, but use the provided security API [94, page 302]. The GSS-API is developed by the IETF and has been scrutinized by security professionals over the years. It has been extensively tested, and is now classified as a mature standard by the IETF.

In this paper we show that the use of the GSS-API provides SIP with a wide range of different authentication methods in a uniform and standardized way, as depicted in Figure 4.3, thus addressing research Goal 3 (b). The GSS-API is not a communication protocol in itself, but relies on SIP to encapsulate, send, and extract data messages called "tokens" between the client and server. The tokens' content are opaque from the viewpoint of the calling application, and contain authentication data, or, once the authentication is complete, portion of data that the client and server want to sign or encrypt.

We reuse the DAA SIP headers for GSS-API support, and instead of encapsulating DAA data, we send the GSS-API tokens. An example of both DAA `Authorization` header and the new `Authorization` header with GSS-API data is depicted in Figure 4.4.

Different authentication methods can be used depending on the different security requirements for each SIP installation. This adds to the flexibility of SIP, like adding a new authentication method, without requiring further changes to the SIP standard, once the GSS-API is supported.

## 4.7 Paper G: Contributions and summary

## Migration towards a more secure authentication in the Session Initiation Protocol

Paper G [92] proposes a two-step migration towards secure authentication in SIP. First, we introduce an authentication method based on the Password Authenticated Key Exchange (PAKE) [34], which provides mutual authentication based on a shared secret, and can function as a drop-in replacement of the digest authentication currently analyzed. The second authentication method proposed is the Simple Authentication and Security Layer (SASL) [50], which enables SIP to transparently support and use more secure authentication methods in a unified

DAA

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response="ccbde1c3c129b3dcaa14a4d5e35
    519d7",uri="sip:CompanyA",algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

GSS-API

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: GSSAPI ttype="context"
    token="0401000B06092A864886F712010202DACD139402AAF44350CDE32"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Figure 4.4: A SIP `REGISTER` message with the original DAA `Authorization` header on the top, and the same header carrying GSS-API data below.

and generic way.

In the paper, three SIP scenarios were identified where identity in SIP needs to be handled, as explained in Section 2.2.3 and depicted in Figure 2.5. These SIP scenarios address research Goal 1 (c).

PAKE was chosen, due to the following attractive features: *1*) PAKE provides mutual authentication between UA and the SIP server, and thus a rogue SIP server can not claim that the authentication succeed without knowing the shared password. PAKE assures the UA that the SIP server knows the UA's encrypted password. *2*) Reuse of the shared password used by DAA as the UA's credential, which enables our approach to easily replace DAA used within a local SIP domain (scenario I). *3*) PAKE offers strong protection of the shared secret if the communication is eavesdropped, that prevents brute-force attacks, including dictionary-based off-line attacks, to which the DAA is vulnerable to.

In the second migration step support for SASL in SIP is added. SASL has the same properties as the GSS-API, explained in the previous paper, but SASL is designed specifically for communication protocols. SASL is also supported in several popular communication protocols applications like IMAP, SMTP and LDAP adopted by the industry.

As with GSS-API, SASL does not provide any authentication mechanisms by itself, but support different underlying authentication mechanisms through a standardized interface[1], as depicted in Figure 4.5.

SASL does not provide a transport layer and thus relies on the application to encapsulate,

[1]A list of registered SASL mechanisms is maintained by IANA: `http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml`

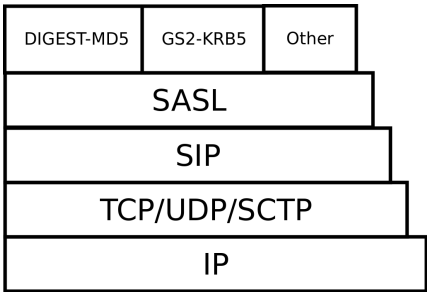



Figure 4.5: The SIP protocol stack with SASL and underlying security mechanisms.

send and extract SASL messages between client and server. The SASL messages sent between client and server contain authentication data, and are opaque from the viewpoint of the calling application. SIP only needs to add support to a SASL software library implementation, and thus have support to a range of underlying authentication mechanisms the library supports.

Both PAKE and SASL re-use the SIP DAA headers to carry authentication data. A SIP `REGISTER` message with SASL authentication data is depicted in Figure 4.6. By adding support for PAKE and SASL in SIP, we address research Goal 3 (b).

## 4.8 Paper H: Contributions and summary

## Advancement towards secure authentication in the Session Initiation Protocol

Paper H [90] is based on a conference article [91] (Paper E) where we analyzed and implemented an attack on the Digest Access Authentication used in the Session Initiation Protocol (SIP) and proposed a correction to mitigate this attack. Since there is a need for better authentication methods in SIP, we add support for a security abstraction layer in SIP [93] (Paper F) and propose a migration strategy towards a secure authentication in SIP [92] (Paper G). Paper H includes a summary of Papers E, F and G.

The current mitigation strategies are analyzed and new authentication mechanisms are introduced, thus addressing Research Goal 3 (a) and (b). The authentication mechanisms in SIP and their support in the three SIP scenarios were identified, is listed in Table 4.2 on the facing page.

In addition to the implemented attack and mitigation strategy published in Paper E, addressing Research Goal 2 (a) and (c), DAA was still vulnerable to offline brute-force dictionary attack. We therefore implemented and tested "Password Based Key Derivation Function version 2" (PBKDFv2) [39] to counter this threat. By using PBKDFv2 on the shared secret and the nonce value received from the SIP server, the computation overhead of computing the DAA digest is significant compared to the original DAA as shown in Figure 4.7. The increase in computation time to compute a DAA digest with PBKDFv2 does increase the cost of an exhaustive brute-force search without a significant impact of computing a single DAA digest to authenticate a UA to the SIP server.

While DAA with PBKDFv2 reduces much of the risk of a brute-force dictionary attack,

DAA

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response=
    "ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",
    algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

SASL

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: SASL mechanism="DIGEST-MD5"
    data="YAzgusSGGeRFGw9nfUvOAxcEDzZCBmKY1H2ElnegaccBcx3DUSkGNW
    Y4qfiSwcXwjLtoqW0eBNog7ixHN"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Figure 4.6: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying SASL data to the right.

| **Authentication mechanisms** | Supported authentication scenarios | | | Supported SIP methods | |
|---|---|---|---|---|---|
| | scenario I | scenario II | scenario III | `REGISTER` | `INVITE` |
| Digest Access Authentication (DAA) | yes | no | no | yes | yes |
| Secure MIME (S/MIME) | no | no | yes | yes[a] | yes |
| Secure SIP (SIPS) using TLS | yes | yes | no[b] | yes | yes |
| P-Asserted Identity | no | yes | no | no | yes[c] |
| SIP Strong Identity | no | yes | no | no | yes[d] |
| Password Authenticated Key Exchange (PAKE) | yes | no | no | yes | yes |
| Generic Security Service API (GSS-API) | yes | yes | yes | yes | yes |
| Simple Authentication and Security Layer (SASL) | yes | yes | yes | yes | yes |

Table 4.2: List of SIP authentication mechanisms and their support.

[a] Not intended to be used with SIP `REGISTER`, however there are no constrains in the SIP specification for using S/MIME in addition to DAA.

[b] SIPS only offers hop-by-hop confidentiality and authentication protection and thus no end-to-end protection.

[c] Does not provide an authentication method *per se*, but provide identity authentication in a trusted environment.

[d] The authentication service is handled by intermediate SIP servers to verify UAs across SIP domains.

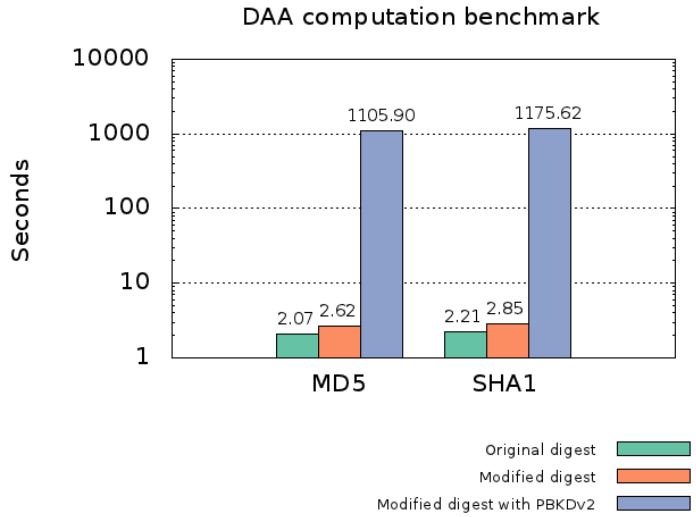



Figure 4.7: The computation overhead for 100.000 iterations for original DAA, our modified DAA, and modified DAA with PBKDFv2 for both MD5 and SHA1.

it does not provide us with means to authenticate the SIP server. Therefore, we introduce an authentication method based on the Password Authenticated Key Exchange (PAKE) [34], which provides mutual authentication based on a shared secret, and can function as a drop-in replacement of the digest authentication currently used (Paper G).

However, a more flexible authentication method is desired. Different security requirements may require different authentication mechanisms. Instead of adding support for many different authentication mechanisms in SIP, we introduce support for a security abstraction layer. A flexible SIP authentication has been requested and desired "to accommodate a variety of authentication mechanisms used to authenticate SIP requests" [47]. The two security programming interfaces GSS-API (Paper F) and SASL (Paper G) are introduced and discussed, thus addressing research Goal 4 (a). Both methods enables SIP to transparently support and use more secure authentication methods in a unified and generic way.

Support for the GSS-API/SASL security layer in SIP, have the following attractive properties that address real-world security concerns, and thus address Research Goal 4 (b) and (c):

1. Mature, stable and industry-adopted standards: The industry might be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Both the GSS-API and SASL are stable, mature standards that have been adopted by the industry. Thus, implementing GSS-API or SASL should not be considered a drastic nor radical change by the relevant standardizing bodies (the IETF) nor the VoIP industry.

2. Minimal changes to the SIP standard required: The authentication data re-use the existing SIP DAA headers, so minimal changes to the SIP *message contents* are required. Also, minimal changes are required to the SIP *message flow*, since the authentication handshake is just extended by a number of required SIP message round-trips to complete the new authentication exchange.

3. Flexible and adaptive to new requirements and future changes: Instead of adding numer-

ous different authentication mechanisms to SIP-based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. By adding support to a security layer in SIP, adding new or modifying existing underlying authentication mechanisms does not need any redesign of the SIP specification standard. In this case, only the GSS-API/SASL software library needs to be updated. Thus, authentication in SIP becomes adaptive to future extensions.

# Chapter 5

# Conclusion

> "So far as we know, organized crime has not targeted VoIP because it's not yet big enough, but that will change as VoIP steps into its role as the heir apparent to the PSTN."
>
> *– Philip Zimmerman in January 2007 edition of VON (Voice On the Net) magazine*

This chapter concludes *Part I* of the thesis. It gives a summary of the research and highlights the key contributions of the thesis. It also provide some considerations of the research and suggestions for future research.

## 5.1 Summary of the research

The overall goal of this research has been to enhance the security in the authentication in the SIP protocol. From this overall goal, several research questions were identified that resulted in specific research goals, including analysis and improvements of authentication mechanisms for the SIP protocol. The conducted research evaluated real-world SIP installations, and pointed out real-world attacks. In particular the Digest Access Authentication (DAA) mechanism, the most deployed authentication mechanism in SIP, suffers from a vulnerability that, if left uncorrected, is a serious flaw. Enhancements to the DAA, as well as other secure authentication mechanisms, listed below, have been proposed to replace the original DAA.

The scientific method used in this research, called the engineering method and discussed in Section 1.3, suggests an empirical as well as a theoretical approach. The problem areas and security challenges were identified. Security attacks were implemented and their consequences discussed. Based on the consequences of these attacks, new improved solutions were proposed and discussed. If any security challenges were found, the proposed solution was either improved or replaced until no further improvements were necessarily or possible. This methodological approach thereby corrects and integrate previous knowledge with an empirical orientation. Therefore, we focus on both a theoretical approach and also work with industrial partners and technology to improve status quo.

The following aspects are the key contributions of this research:

**Analysis of the Digest Access Authentication in SIP:** The DAA is currently the most common authentication mechanism for SIP, since it is the only authentication mechanism that

must be mandatorily supported. This research have shown and confirmed that the DAA is weak and vulnerable to a registration attack (during the SIP `REGISTER` handshake). The need for an improvement of the DAA to counter this vulnerability is emerging.

**Analysis of the call-setup message flow in SIP:** We analysed the situation where after a phone (UA) has registered successfully, a SIP `INVITE` message can be sent to initiate a session (e.g., set up a call). We showed and implemented a call-setup hijack attack, where attackers can take over a SIP session. The attackers intercept and modify the SIP `INVITE` handshake and throw out the calling parties by issuing fake SIP `CANCEL` messages. Since this attack is hard to detect by SIP servers, the call is wrongfully registered as an ordinary call, thus effectively faking call detail records (CDRs). To counter this attack, further analyses must be conducted on the particulars of the SIP call-setup handshake, and thus is an area for future work.

**Show, implement and mitigate a real-world attack on DAA in SIP:** The DAA vulnerability analyzed earlier (registration attack) was verified and confirmed by implementing this attack on the SIP protocol in real-time. A solution to counter this serious registration attack was proposed and implemented. However, a replacement for the DAA was desired to counter other vulnerabilities and add a more flexible authentication service to SIP.

**Introduce new secure authentication in SIP:** By following the engineering methodology, four different alternative authentication methods have been introduced and analyzed:

1. The modified DAA with the SIP header value *ContactURI* included in the digest can be used to prevent the registration hijack attack found earlier. However, the modified DAA was still vulnerable to offline dictionary attack.
2. The "Password-Based Key Derivation Function version 2" (PBKDFv2) on the shared secret to make dictionary- and brute-force attacks significant harder to be executed on the DAA. This method does not authenticate the SIP server, only the client.
3. The replacement of the DAA with a modified "Password Authenticated Key Exchange" (PAKE) introduces mutual authentication and re-use the shared secret used by the DAA. These properties make PAKE a preferred mechanism over the DAA and PBKDFv2. While PAKE seems to be efficient, it does neither open up for future extensions nor allow modification of authentication in SIP once implemented.
4. We introduce a GSS-API/SASL security layer which enables SIP to transparently support and use more secure authentication methods in a unified and generic way without later change to the SIP protocol specification. The method takes the limitations of the previous mechanisms into consideration, and is seen as the most viable solution towards an evolutionary security enhancement of authentication in SIP.

Combination of GSS-API/SASL in SIP is an evolutionary breakthrough in SIP to satisfy the need of current and upcoming security threats and requirements. Support for the GSS-API/SASL security layer in SIP was found to have the following attractive properties that address real-world security concerns:

**A) Mature, stable and industry adopted standards:** The industry might be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Both GSS-API and SASL are stable, mature standards that have been adopted by the industry, including Postifx-, Sendmail- and Exim-SMTP servers, OpenLDAP, PostgreSQL database, and the Apache web server. Thus, implementing GSS-API or SASL can be seen as an evolutionary update by the VoIP industry and can thus be relevant for the relevant standardizing bodies (the IETF).

**B) Minimal changes to the SIP standard required:** The authentication mechanisms suggested re-use the existing `WWW-Authenticate`and `Authorization` SIP header values. The original header value carrying DAA authentication data content is replaced by authentication data content for the proposed authentication mechanism. Thus, only two lines need to be changed in the SIP *message header*, as the example given in Figure 4.6 on page 35 shows. No changes to the SIP message body are required. Only minimal changes are required to the SIP *message flow*, since the authentication handshake is just extended with a number of required SIP message round trips to complete the new authentication exchange. The number of extended round-trips needed depend on the underlying authentication mechanism used to complete the exchange.

**C) Flexible and adaptive to new requirements and future changes:** By adding support to a security layer in SIP, adding new or modifying existing underlying authentication mechanisms does not need any redesign of the SIP specification standard. Only the GSS-API/SASL software library needs to be updated. Thus, authentication in SIP becomes adaptive to future extensions, or local requirements set forth by companies/authorities (military, e.g. NATO). As a consequence, the suggestion of GSS-API/SASL support is seen as a viable solution to authentication in SIP.

## 5.2 Contributions to VoIP security

The outcome of the research has been published in conference proceedings and international journals; three have been published in journals, eight in international conference proceedings. One technical report has been published, and ten VoIP presentations/talks has been given, including two poster presentations[1].

The research goals, which were presented and discussed in Section 1.3, have been addressed. How the goals, with sub-goals, have been fulfilled is outlined in Table 5.1.

In addition to addressing the defined research goals, industrial applicability and relevance have been provided in terms of VoIP security discussions, advise and consultancy to real-world VoIP providers (Telio[2], Telekompetanse[3], Ibidium[4]) and customers (Buskerud County Municipality[5], Akershus County Municipality[6]) during this research period.

[1] A complete list of presentations found here: `http://www.nr.no/~strand/publications.html`
[2] Telio Holding ASA: `http://www.telioholding.no/`
[3] Telekompetanse AS: `http://www.telekompetanse.com/`
[4] Ibidium AS: `http://www.ibidium.no`
[5] Buskerud Fylkeskommune: `http://www.bfk.no`
[6] Akershus Fylkeskommune: `http://www.akershus.no/`

| The research goals | | | Fulfilled |
|---|---|---|---|
| Goal 1 | Explore VoIP communication and authentication security challenges in SIP. | | ✔ |
| | G1 (a) | A list of security challenges in SIP. | Part I, Paper A, D |
| | G1 (b) | The list given in *G1 (a)* prioritized and ranked as most severe. | Part I |
| | G1 (c) | A list of SIP authentication scenarios. | Part I, Paper G |
| Goal 2 | Identify and analyze SIP authentication vulnerabilities and threats by demonstrating security attacks. | | ✔ |
| | G2 (a) | Enable VoIP testbed for security testing of selected VoIP SIP scenarios. | Paper A, B, C, E, H, (Paper R6 [88]) |
| | G2 (b) | List of security attacks and their consequences for VoIP SIP communication. | Paper A, D |
| | G2 (c) | Analysis, discussion and implementation of a least two security attacks targeting the SIP protocol. | Paper A, B, C, E, H |
| Goal 3 | Analyze current mitigation strategies and suggest more secure authentication mechanisms. | | ✔ |
| | G3 (a) | A discussion and analysis of current mitigation strategies for SIP authentication. | Part I, Paper H |
| | G3 (b) | Apply engineering methodology to improve SIP authentication. | Paper E, F, G, H |
| Goal 4 | Evaluate the suggested authentication mechanisms, and address real-world security concerns. | | ✔ |
| | G4 (a) | An evaluation of the suggested new authentication mechanisms provided by *G3 (b)*. Discuss against SIP authentication scenarios provided by *G1 (c)*, and security attacks provided by *G2 (b)*. | Paper H |
| | G4 (b) | Based on *G4 (a)*, recommend a new authentication mechanism. | Part I, Paper H |
| | G4 (c) | Provide an industrial uptake strategy path that addresses real-world security concerns. | Part I, Paper H |

Table 5.1: The fulfillment of the research goals and sub-goals (the expected measurable outcome).

The results of this research has also led to discussions with the Norwegian Defence Research Establishment (FFI), with the intention to form a Working Group within NATO to continue the work in this thesis[7]. This research has also explored and laid the foundation for further standardization work within the IETF, as described in the next Section.

[7]Discussion with the "Norwegian Defence Research Establishment" (FFI) on $11^{th}$ July 2011.

## 5.3 Considerations and suggestions for further research

This thesis analyzed real-world VoIP installations and improved authentication in the SIP protocol. The proposed research argues that the currently used DAA is weak and a new improved authentication mechanisms is desired. While analyzing and applying new authentication mechanisms to SIP, the following limitations were discovered:

- Not all underlying authentication mechanisms supported by GSS-API/SASL provide better integrity protection over the DAA. In fact, one of the supported underlying authentication mechanisms supported by SASL earlier was the DAA ("Digest-MD5"). Support for DAA in SASL was withdrawn in April 2011 by the IETF due to a number of security considerations [51]. Further analysis of the 30 official supported SASL mechanisms[8] needs to be performed in order to generate a preferred and prioritized list of mechanisms.

- A deployment of GSS-API/SASL enabled VoIP solutions, will still require a software update if a new underlying authentication mechanism is changed or added. Since the GSS-API/SASL software library must be updated to reflect any changes, a software update must be performed on the UA and SIP servers that handle authentication. However, since GSS-API/SASL introduce an abstraction layer the SIP standard remain unchanged after such an update.

- The use of Transport Layer Security (TLS) as a security mechanism for SIP has received research focus and some industry momentum lately. TLS provides security services between the Transport Layer (TCP/UDP) and SIP to encrypt the application data. This research has focused on solving security challenges within the SIP protocol itself without relying on security protocols that operate on lower levels in the protocol stack, such as TLS or IPSec. Also, relying on TLS requires a Public Key Infrastructure (PKI) for certificate handling. In most cases, this is complex and expensive to implement. The use of TLS might not be suited for all SIP scenarios, like in mobile handset with limited bandwidth and computation power.

- The SIP scenarios presented in this research often require and are built upon a client-server model: The client (UA) connects to an SIP server or another client (UA) and they authenticate each other. How the proposed authentication methods presented in this research can be integrated in a peer-to-peer or decentralized distributed VoIP system is not considered and thus a topic for future research.

This research analyzed and improved the authentication in SIP. We see the following research topics to extend and continue our research:

- To complete an implementation and development of a standalone SIP client and server with working support for the proposed security mechanisms, is the goal of the ongoing work. The implementation is envisaged to be based on existing open source SIP client/server and GSS-API/SASL libraries.

[8] A list of registered SASL mechanisms is maintained by The Internet Assigned Numbers Authority (IANA): `http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml`

- Evaluate and test different underlying authentication mechanisms supported by GSS-API/SASL to ensure that pre-defined security requirements are met regardless of SIP scenarios and communication environments ("dynamic environments"). The security requirements must be deduced from a given security policy, or pre-defined to met a minimal set of requirements if no such policy exists. The test should include benchmark measurement following the performance metrics defined by Malas and Morton [48].

- Standardization must be seen in a long-term time-frame, and we can only initiate such a process that needs to be continued. This work includes that we collaborate with the Working Groups *sipcore*[9] and *kitten*[10] within the IETF to develop and promote PAKE, GSS-API and SASL support in SIP as Internet Standards. The starting point would be to create IETF drafts for each approach according to recommendations set forth by the IETF[11].

- After the above standardization work has been completed (or is well underway), work with VoIP vendors to ensure industry co-operation and uptake of the proposed security mechanisms in SIP. Technical discussions have been initiated with Cisco (Tandberg)[12] and Huawei Norway[13] on this topic.

- The Security Assertion Markup Language (SAML) [78] tries to solve single sign-on (SSO) by standardizing an open standard for exchange of authentication and authorization data between security domains. A competing standard, that addresses the same problem and currently have some industrial momentum, is the OpenID protocol [67]. Some work is currently ongoing within the IETF to add support for SAML to SIP [63], and also add SAML as a underlying SASL mechanism [99]. However, further research should evaluate and explore how SAML and OpenID solve authentication in a decentralized manner, and how this approach can be applied to SIP.

This thesis has contributed with publications in international journals, conference proceedings and presentations. The research goals defined in Chapter 1 have been achieved. The results in this thesis have industrial relevance and provide an evolutionary mitigation strategy for industrial uptake. This research has also laid the groundwork for future extension of the SIP protocol in the area of security.

[9] IETF WG "Session Initiation Protocol Core" (sipcore): `http://datatracker.ietf.org/wg/sipcore/`

[10] IETF WG "Common Authentication Technology Next Generation" (kitten): `http://datatracker.ietf.org/wg/kitten/`

[11] To follow the guidelines set by the IETF in their document "Guidelines to Authors of Internet-Drafts": `http://www.ietf.org/id-info/guidelines.html`

[12] TANDBERG ASA: `http://http://www.tandberg.com/`

[13] Huawei Technologies Co. Ltd.: `http://www.huawei.com`

# Bibliography


[1] 3rd Generation Partnership Project. Security architecture (Release 10), 2010. [Online]. Available (30. August 2011): `http://www.3gpp.org/ftp/Specs/html-info/33102.htm`.

[2] Ross J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, 2 edition, April 2008.

[3] Mary Barnes. IETF DRAFT: a mechanism to secure SIP identity headers inserted by intermediaries. Technical report, IETF, December 2004.

[4] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986 (Standard), January 2005.

[5] Philipa Biggs. GSR Discussion paper: "Voice over Internet Protocol: Enemy or Ally?" (updated chapter from: "Status of Voice over Internet Protocol (VoIP) Worldwide, 2006"). Technical report, International Telecommunication Union (ITU), Nov 2009 (updated).

[6] Matthew Bishop. *Computer Security: Art and Science*. Addison Wesley, Dec 2002.

[7] William E. Burr, Donna F. Dodson, and W. Timothy Polk. Electronic authentication guideline. Technical Report NIST Special Publication 800-63, National Institute of Standards and Technology, April 2006.

[8] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle. Session Initiation Protocol (SIP) Extension for Instant Messaging. RFC 3428 (Proposed Standard), December 2002.

[9] D. Cohen. Specifications for the Network Voice Protocol (NVP). RFC 741, November 1977.

[10] D. E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. Computing as a discipline. *Commun. ACM*, 32:9–23, January 1989.

[11] Michael K. Daly. The advanced persisten threat. USENIX, LISA'09, Baltimore, USA, November 2009.

[12] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878.

[13] Whitfield Diffie and Martin E. Hellman. Multiuser cryptographic techniques. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*, AFIPS '76, pages 109–112, New York, NY, USA, 1976. ACM.

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

[15] Sven Ehlert, Ge Zhang, Dimitris Geneiatakis, Georgios Kambourakis, Tasos Dagiuklas, Jirí Markl, and Dorgham Sisalem. Two layer Denial of Service prevention on SIP VoIP infrastructures. *Computer Communications*, 31(10):2443–2456, 2008.

[16] Steve Evans. "Good to talk?". *Unified Communication Industry News*, July 2011.

[17] Niels Ferguson and Bruce Schneier. *Practical Crypthography*. Wiley Publishing, Jan 2003.

[18] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. Updated by RFCs 2817, 5785.

[19] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication. RFC 2617 (Draft Standard), June 1999.

[20] N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2045 (Draft Standard), November 1996. Updated by RFCs 2184, 2231, 5335.

[21] Lothar Fritsch and Arne-Kristian Groven. VoIP stakeholder profiling: Public stakeholders and infrastructure owners. Technical Report DART/06/2009, Norwegian Computing Center, Department of Applied Research in Information Technology, December 2009.

[22] Lothar Fritsch, Arne-Kristian Groven, and Lars Strand. A holistic approach to open-source VoIP security: Preliminary results from the EUX2010sec project. In *Proceedings of the Eight International Conference on Networking (ICN2009)*, pages 275–280. IEEE Computer Society, March 2009.

[23] Lothar Fritsch, Arne-Kristian Groven, Lars Strand, Wolfgang Leister, and Anders Moen Hagalisletto. A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project. *International Journal on Advances in Security*, 2(2&3):129–141, 2009.

[24] Akershus fylkeskommune Bygg og innkjøp. Åpen anbudskonkurranse – rammeavtale på levering av telefoniløsning til Akershus fylkeskommune (Public tender for delivering new VoIP service to Akershus County Municipality), 2010. [Online]. Available (30. August 2011): `http://www.doffin.no/search/show/search_view.aspx?ID=MAY145946`.

[25] Robert L. Glass. A structure-based critique of contemporary computing research. *J. Syst. Softw.*, 28:3–7, January 1995.

[26] Ruediger Glott, Arne-Kristian Groven, Kirsten Haaland, and Anna Tannenberg. Quality models for free/libre open source software - towards the "silver bullet". *Software Engineering and Advanced Applications, Euromicro Conference*, 0:439–446, 2010.

[27] Arne-Kristian Groven, Haaland, Kirsten Haaland, Ruediger Glott, and Anna Tannenberg. Security measurements within the framework of quality assessment models for free/libre open source software. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, ECSA '10, pages 229–235, New York, NY, USA, 2010. ACM.

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

[29] Anders Moen Hagalisletto and Lars Strand. Formal modeling of authentication in SIP registration. In *Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08*, pages 16–21. IEEE Computer Society, August 2008.

[30] Anders Moen Hagalisletto and Lars Strand. Designing attacks on SIP call set-up. *International Journal of Applied Cryptography*, 2(1):13–22(10), July 2010.

[31] Anders Moen Hagalisletto, Lars Strand, Wolfgang Leister, and Arne-Kristian Groven. Analysing protocol implementations. In Feng Bao, Hui Li, and Guilin Wang, editors, *The 5th Information Security Practice and Experience Conference (ISPEC 2009)*, volume LNCS 5451, pages 171–182. Springer Berlin / Heidelberg, April 2009.

[32] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. RFC 4566 (Proposed Standard), July 2006.

[33] C. Holmberg, E. Burger, and H. Kaplan. Session Initiation Protocol (SIP) INFO Method and Package Framework. RFC 6086 (Proposed Standard), January 2011.

[34] International Organization for Standardization and ISO. ISO/IEC 11770-4:2006: Information technology – Security techniques – Key management – Part 4: Mechanisms based on weak secrets, 2006.

[35] International Telecommunication Union. H.323: Packet based multimedia systems. ITU-T Recommendation H.323, 2006.

[36] International Telecommunication Union (ITU). Security Architecture For Open Systems Interconnection (OSI). X.800 Recommendation X.800, The International Telegraph and Telephone Consultative Comittee (CCITT), 1991.

[37] C. Jennings, J. Peterson, and M. Watson. Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks. RFC 3325 (Informational), November 2002. Updated by RFC 5876.

[38] Cullen Jennings and Nagendra Modadugu. IETF DRAFT: session initiation protocol (SIP) over datagram transport layer security (DTLS). Technical report, IETF, October 2007.

[39] B. Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0. RFC 2898 (Informational), September 2000.

[40] V. Kalusivalingam. Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6). RFC 3898 (Proposed Standard), October 2004.

[41] Angelos D. Keromytis. *Voice over IP Security - A Comprehensive Survey of Vulnerabilities and Academic Research*, volume 1. Springer New York, New York, NY, 1st edition, 2011.

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

[43] Otmar Lendl. VoIP peering: Background and assumptions. Technical report, IETF, October 2008.

[44] Yi-Pin Liao. *Secure password authenticated key exchange protocols for various enviroments*. PhD thesis, Tatung University, December 2009.

[45] Yi-Pin Liao and Shuenn-Shyang Wang. A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves. *Computer Communications*, 33(3):372–380, February 2010.

[46] J. Linn. Generic Security Service Application Program Interface Version 2, Update 1. RFC 2743 (Proposed Standard), January 2000. Updated by RFC 5554.

[47] J. Loughney and G. Camarillo. Authentication, Authorization, and Accounting Requirements for the Session Initiation Protocol (SIP). RFC 3702 (Informational), February 2004.

[48] D. Malas and A. Morton. Basic Telephony SIP End-to-End Performance Metrics. RFC 6076 (Proposed Standard), January 2011.

[49] G. Malkin and A. Marine. FYI on Questions and Answers Answers to Commonly asked "New Internet User" Questions. RFC 1325 (Informational), May 1992. Obsoleted by RFC 1594.

[50] A. Melnikov and K. Zeilenga. Simple Authentication and Security Layer (SASL). RFC 4422 (Proposed Standard), June 2006.

[51] Alexey Melnikov. IETF DRAFT: moving DIGEST-MD5 to historic. Technical report, IETF, April 2011.

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

[53] Harry Newton. *Newton's Telecom Dictionary: Telecommunications, Networking, Information Technologies, The Internet, Wired, Wireless, Satellites and Fiber*. Flatiron Publishing, 26th edition, August 2011.

[54] A. Niemi. Session Initiation Protocol (SIP) Extension for Event State Publication. RFC 3903 (Proposed Standard), October 2004.

[55] A. Niemi, J. Arkko, and V. Torvinen. Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA). RFC 3310 (Informational), September 2002.

[56] Gordon M. Snow Assistant Director Cyber Division Federal Bureau of Investigation. Cybersecurity: Responding to the threat of cyber crime and terrorism – statement before the senate judiciary committee, April 2011.

[57] K. Ono and S. Tachimoto. Requirements for End-to-Middle Security for the Session Initiation Protocol (SIP). RFC 4189 (Informational), October 2005.

[58] Francesco Palmieri. Improving authentication in voice over IP infrastructures. In Khaled Elleithy, Tarek Sobh, Ausif Mahmood, Magued Iskander, and Mohammad Karim, editors, *Advances in Computer, Information, and Systems Sciences, and Engineering*, pages 289 – 296. Springer Netherlands, 2006.

[59] Francesco Palmieri and Ugo Fiore. Providing true end-to-end security in converged voice over IP infrastructures. *Computers & Security*, 28(6):433–449, September 2009.

[60] J. Peterson. Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format. RFC 3893 (Proposed Standard), September 2004.

[61] J. Peterson. S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP). RFC 3853 (Proposed Standard), July 2004.

[62] J. Peterson and C. Jennings. Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP). RFC 4474 (Proposed Standard), August 2006.

[63] Jon Peterson, James Polk, Douglas Sicker, Jeff Hodges, and Hannes Tschofenig. IETF DRAFT: SIP SAML profile and binding. Technical report, IETF, October 2010.

[64] Post- og teletilsynet (The Norwegian Post and Telecommunications Authority). Det norske markedet for elektroniske kommunikasjonstjenester 2009 (The Norwegian market for electronic communication services 2009). [Online]. Available (30. August 2011) `http://www.npt.no/ikbViewer/Content/119027/Ekomrapport_2009_.pdf`.

[65] J. Postel. Simple Mail Transfer Protocol. RFC 821 (Standard), August 1982. Obsoleted by RFC 2821.

[66] Archana Rao and Henning Schulzrinne. Automated SIP interoperability testing. The 10th International SIP Conference, Paris, France, January 2009.

[67] David Recordon and Drummond Reed. OpenID 2.0: A platform for user-centric identity management. In *DIM '06: Proceedings of the second ACM workshop on Digital identity management*, pages 11–16, New York, NY, USA, 2006. ACM.

[68] E. Rescorla. Diffie-Hellman Key Agreement Method. RFC 2631 (Proposed Standard), June 1999.

[69] E. Rescorla and N. Modadugu. Datagram Transport Layer Security. RFC 4347 (Proposed Standard), April 2006. Updated by RFC 5746.

[70] TeleGeography Research. Telegeography report & database - executive summary. Technical report, TeleGeography, 2010.

[71] Robert Richardson. 2010/2011 CSI Computer Crime and Securty Survey. Annual Report, Computer Security Institute, 2011.

[72] A. B. Roach. Session Initiation Protocol (SIP)-Specific Event Notification. RFC 3265 (Proposed Standard), June 2002. Updated by RFCs 5367, 5727.

[73] J. Rosenberg. The Session Initiation Protocol (SIP) UPDATE Method. RFC 3311 (Proposed Standard), October 2002.

[74] J. Rosenberg. SIP Peering: What It Does and Doesn't Mean. *SIP Magazine Speaking SIP*, mar 2006.

[75] J. Rosenberg. A Hitchhiker's Guide to the Session Initiation Protocol (SIP). RFC 5411 (Informational), February 2009.

[76] J. Rosenberg and H. Schulzrinne. Reliability of Provisional Responses in Session Initiation Protocol (SIP). RFC 3262 (Proposed Standard), June 2002.

[77] 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, 5393, 5621, 5626, 5630, 5922, 5954, 6026.

[78] Assertions and protocols for the OASIS security assertion markup language (SAML) v2.0, March 2005.

[79] Bruce Schneier. *Applied Crypthography*. Wiley, second edition, 1996.

[80] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Updated by RFCs 5506, 5761, 6051.

[81] R. Shirey. Internet Security Glossary, Version 2. RFC 4949 (Informational), August 2007.

[82] 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.

[83] Dorgham Sisalem, John Floroiu, Jiri Kuthan, Ulrich Abend, and Henning Schulzrinne. *SIP Security*. WileyBlackwell, March 2009.

[84] R. Sparks. Internet Media Type message/sipfrag. RFC 3420 (Proposed Standard), November 2002.

[85] R. Sparks. The Session Initiation Protocol (SIP) Refer Method. RFC 3515 (Proposed Standard), April 2003.

[86] R. Sparks and T. Zourzouvillys. Correct Transaction Handling for 2xx Responses to Session Initiation Protocol (SIP) INVITE Requests. RFC 6026 (Proposed Standard), September 2010.

[87] William Stallings. *Network Security Essentials: Applications and Standards*. Prentice Hall, 4 edition, March 2010.

[88] Lars Strand. VoIP lab as a research tool in the EUX2010SEC project. Technical Report DART/08/10, Norwegian Computing Center, Department of Applied Research in Information Technology, April 2010.

[89] Lars Strand and Wolfgang Leister. A Survey of SIP Peering. In *NATO ASI - Architects of secure Networks (ASIGE10)*, May 2010.

[90] Lars Strand and Wolfgang Leister. Advancement towards secure authentication in the session initiation protocol. *Submitted to International Journal on Advances in Security*, 2011.

[91] Lars Strand and Wolfgang Leister. Improving SIP authentication. In *Proceedings of the Tenth International Conference on Networking (ICN2011)*, pages 164 – 169. Xpert Publishing Services, Jan 2011.

[92] Lars Strand, Wolfgang Leister, and Alan Duric. Migration towards a more secure authentication in the session initiation protocol. In *The Fifth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE2011)*, pages 57–62. Xpert Publishing Services, Aug 2011.

[93] Lars Strand, Josef Noll, and Wolfgang Leister. Generic security services API authentication support for the session initiation protocol. In *Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT2011)*, pages 117–122. Xpert Publishing Services, Mar 2011.

[94] Dobromir Todorov. *Mechanics of User Identification and Authentication: Fundamentals of Identity Management*. Auerbach Publication, 1 edition, June 2007.

[95] V. Torvinen, J. Arkko, and M. Naslund. Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA) Version-2. RFC 4169 (Informational), November 2005.

[96] James Undery. Ieft draft: SIP authentication: SIP digest access authentication. Technical report, IETF, July 2001.

[97] VoIPSA. VoIP security and privacy threat taxonomy. Public Realease 1.0, October 2005.

[98] Jacques Wainer, Claudia G. Novoa Barsottini, Danilo Lacerda, and Leandro Rodrigues Magalhães de Marco. Empirical evaluation in computer science research published by acm. *Inf. Softw. Technol.*, 51:1081–1085, June 2009.

[99] Klaas Wierenga and Eliot Lear. IETF DRAFT: A SASL mechanism for SAML. Technical report, IETF, July 2010.

[100] Chou-Chen Yang, Ren-Chiun Wang, and Wei-Ting Liu. Secure authentication scheme for session initiation protocol. *Computers & Security*, 24(5):381–386, August 2005.

[101] Ge Zhang, Simone Fischer-Hübner, and Sven Ehlert. Blocking attacks on SIP VoIP proxies caused by external processing. *Telecommunication Systems*, 45(1):61–76, 2010.

[102] L. Zhu, P. Leach, K. Jaganathan, and W. Ingersoll. The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism. RFC 4178 (Proposed Standard), October 2005.

# PART II:

# Scientific contributions

# Paper A:
# A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project

Lothar Fritsch, Arne-Kristian Groven, Lars Strand, Wolfgang Leister, Anders Moen Hagalisletto

In
International Journal on Advances in Security, vol 2, 2009
ISSN: 1942–2636

# A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project

Lothar Fritsch, Arne-Kristian Groven, Lars Strand, Wolfgang Leister, and Anders Moen Hagalisletto Norsk Regnesentral
Oslo, Norway
email: {lothar.fritsch, groven, lars.strand, wolfgang.leister, anders.moen}@nr.no

***Abstract*— The present paper describes the approach and preliminary results from the research project EUX2010SEC. The project works closely with Voice over IP (VoIP) companies and users. The project aims at providing better security of open source VoIP installations. The work towards this goal is organized by gathering researchers and practitioners around scientific activities that range from security modeling and verification up to testbed testing. The expected outcomes of the project are a solid scientific and practical understanding of the security options for setting up VoIP infrastructures, particular guidance on secure, typical setups of such infrastructure. The project's special focus is on producing results relevant to the practitioners in the project, aiming at the stimulation of innovation and the provision of highest quality in open source based VoIP products and services. The article describes the research-based innovation approach used.**

***Index Terms*— VoIP, SIP, security model, security requirements, testbed testing, formal protocol analysis.**

## I. Introduction

This article provides overview of the VoIP security research project EUX2010SEC[1] which has its roots in the Nordic resource network *Enterprise Unified Exchange (EUX2010)*. The project is partly funded by the Norwegian Research Council, and runs from 2007 until 2011. The project provides a forum for researchers[2], user representatives from Norwegian public administration[3], and small and medium sized companies representing both the VoIP and open source software industry in Norway[4]. The current work is based on a conference article on the International Conference on Networking in 2009 [1].

### A. Research-based innovation in Norway

The EUX2010SEC project is placed in Norwegian Research Council's technological programme "Kjernekompetanse og verdiskapning i IKT" (VERDIKT), a public funding scheme for user-driven, research-based innovation which targets Norwegian industry and research institutions. The principal tool in the VERDIKT programme is the user-driven project.

The EUX2010SEC project aims at the analysis and development of open source technologies used in VoIP infrastructures. As means towards the goal we implemented a testbed laboratory for the industrial users, and applied user-need based research and problem-solving activities for the VoIP stakeholders in the project. The outcomes shall widen understanding of VoIP, promote secure infrastructures, and strengthen the competitiveness of the Norwegian industry partners in the project. It uses the *Empathic Design* [2] approach and rapid prototyping strategies among other innovation strategies. In addition to industry research work and publication, the project educates a PhD student in the field. Thus EUX2010SEC uses the three most successful industry-oriented innovation strategies considered by MIT researchers [3].

### B. Project goals

The overall research goal of the project is to improve the level of security and awareness when developing, installing, and using open source VoIP solutions, such as the open source Asterisk PBX[5]. The main objectives of VoIP-oriented security are to preserve the availability of VoIP services, to protect VoIP transmissions and stored information from disclosure and theft, to prevent fraudulent usage of voice communication, so called toll fraud with financial losses, and to preserve the integrity of the VoIP system, e.g., that the system logs to be stored by the providers on behalf of the authorities are correct.[6]

As one of the fastest growing Internet technologies today, Voice over IP (VoIP) can provide a number of additional services compared to traditional telephony. These services include conferencing, events notification, presence, instant messaging, video telephony and other multimedia transmissions, and location independence (location mobility). Such wide flexibility imposes challenges on how security is handled [4], [5].

Our experience from work with the industry partners is that in many cases the security model applied to VoIP networks is a model of isolation, physically separating voice and data or using virtual LANs or VPNs to separate VoIP traffic from any other IP traffic. This separation sacrifices many of the benefits of VoIP and makes the integration of communication

This paper is based on the conference article "A holistic approach to Open Source VoIP security: Results from the EUX2010SEC project", presented at the ICN 2009 conference.

[1]Project homepage: http://eux2010sec.nr.no

[2]Norwegian Computing Center, UNU-MERIT

[3]Buskerud County Municipality (Buskerud fylkeskommune).

[4]Redpill Linpro AS, Freecode AS, Nimra Norge AS, Ibidium Norden AS

[5]Asterisk is a central component in the VoIP networks we are interested in. Asterisk homepage: http://www.asterisk.org

[6]In many countries the telephony providers must store the connection logs of for a specified time, typically several months.

applications hard or even impossible. Hence, the potential of VoIP systems is often not utilized. One goal of the project is to look into other possible VoIP network topologies and approaches to security. This would enable the adoption of innovative functions, such as mobile software phones on laptops and PDAs being used on open Public IP networks, much easier.

When analyzing VoIP security and vulnerability different perspectives are used in the project:

- Analysis at device level, focusing on a particular device, e.g., a PBX (Private Branch Exchange);
- Analysis at system level, focusing on the VoIP infrastructure components and VoIP topologies, or;
- Analysis focusing on the flow of data and signals in VoIP systems.

Vulnerabilities in VoIP have many causes [6] which may be related to weaknesses in the applied protocols, the software, or the configurations of the various VoIP applications and equipment in use. EUX2010SEC provides analysis, testing and guidance of many possible options to the suppliers and users of VoIP services, and in addition researches the security consequences.

The EUX2010SEC project aims at transferring innovation to the market by supporting the practitioners with scientific security knowledge. This knowledge is provided by analysis of topologies and usage patterns of VoIP systems; analysis of the systems using both formal methods and testbed testing; the collection of realistic security requirements from practitioners and users; and the development and testing of secure configurations, which will be recommended as base configurations for various basic VoIP setups.

### C. State of knowledge

This section provides an overview of general VoIP security literature. The following sections on verification, testing and security modeling might introduce more specialized background references where needed.

Security of VoIP systems has received much attention in national security bodies and in academia. Analysis focused on technical security issues, and availability considerations of VoIP-based critical communications services. The VOIPSA taxonomy is our starting point for a systematic exploration of known VoIP security threats [6]. The VOIPSA taxonomy is less detailed in the description of problems and fixes, but it is superior in its taxonomic description over many of the hands-on guidebooks such as or [4]. Various governments information security institutions or standards institutes have issued warnings or guidance, for example the U.S-National Institute of Standards and Technology [7] and others [8].

Some scientific publications overlook the topic, but mainly discovered classic attack patterns such as man-in-the-middle attacks, the exploitation of misconfigurations, and reachability control issues [9], [10]. Some work has been done to analyze security vulnerabilities in VoIP implemented technologies [11]. Among others, the SIP protocol [12], [13] has the important role of connection establishment and management. SIP is vulnerable to authentication and hijacking problems [14], and others [15], [16].

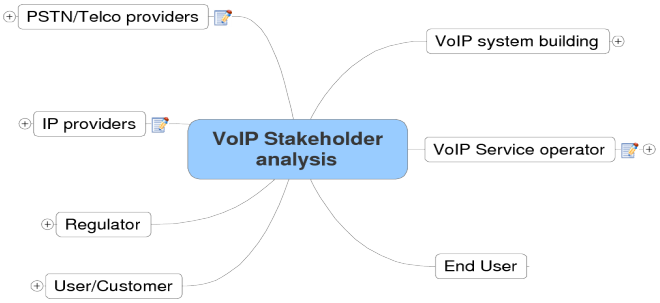



Fig. 2. VoIP Stakeholder analysis

## II. Methodology and approach

The research activities in EUX2010SEC focus on three areas of activity, as shown in Fig. 1

The ***security model*** activity analyzes stakeholders' requirements towards security and stability of VoIP systems. Its goal is to derive typical requirements' profiles, and to provide security models and default configurations for them. This is shown in the right part of Fig. 1.

***Testbed systems*** with the partners' technology, and real user requirements: These testbed systems will have VoIP traffic routed through them for testing the system properties and the consequences of configuration options. They will additionally be used for the deployment of a set of attacks and attack tools. The testbed activity is depicted in the middle part of Fig. 1.

***Formal protocol analysis:*** The function, usage and real configuration and implementation of security-relevant protocols used in the Asterisk family of VoIP systems is formalized and then tested with a protocol verification tool that attacks the protocol model. This approach can reveal unknown protocol failures, and wrongful implementation of protocols. The formal analysis approach is shown in the left part of Fig. 1.

### A. Requirements & security model

The stakeholder and requirements gathering approach is inspired by the privacy design process outlined in [17], and was used in [18]. It is modified in EUX2010SEC to find and elaborate VoIP security requirements for the identified basic scenarios of VoIP usage.

The security model activity is carried out in consecutive steps. A basic stakeholder model and initial scenario profiles is derived from the state of the art literature. Various VoIP project partners and possibly their customers are contacted for empiric research. Steps to be carried out are as follows:

***Stakeholder Analysis:*** The stakeholders are identified and contacted, and their main interests in the VoIP market be captured by means of a stakeholder analysis [19].

***Requirements Elicitation:*** The stakeholders are interviewed concerning their usage scenarios and requirements concerning VoIP security.

- The interviews collect anecdotic accounts of problems and requirements.
- The interviewees are presented with scenarios and use cases to single out their typical scenarios.

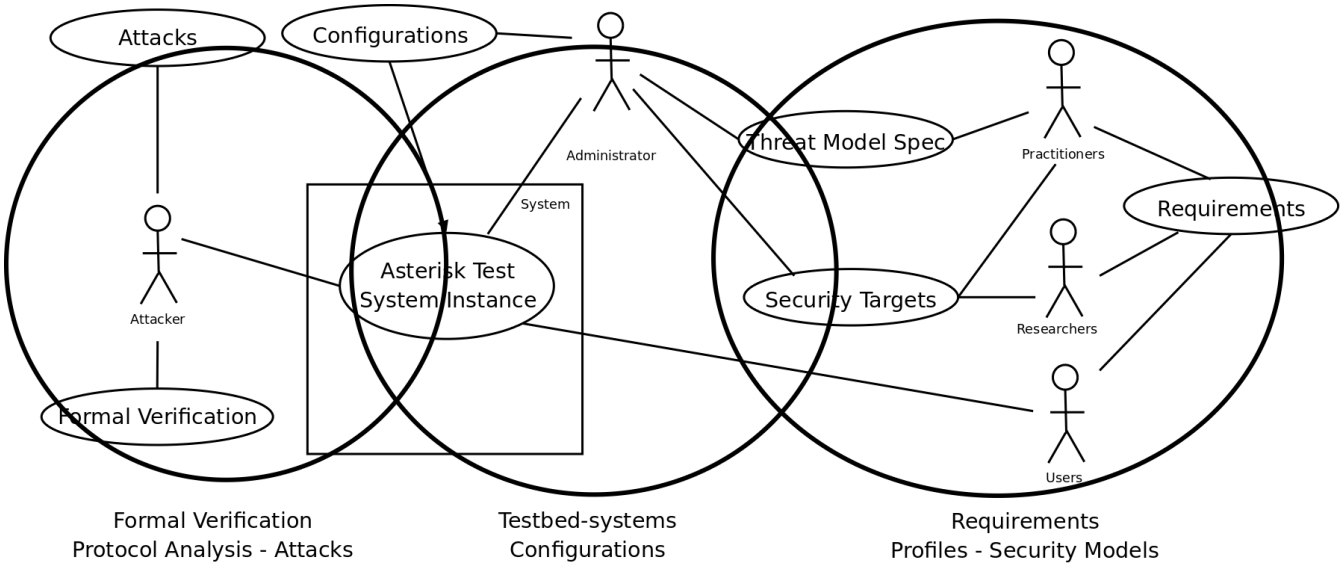



Fig. 1. EUX2010SEC research approach

***Scenario Profiles:*** From the steps above, one or more profiles for typical VoIP usage scenarios will be generated. The profiles should create the basis for further analysis, testbed creation, and verification activities.

- A profile is based on a use case description.
- A profile contains a description of security, reliability, quality-of-service and scalability needs.

***Multilateral Security Analysis:*** For each of the profiles, a multilateral security analysis is performed [20] to ensure that all stakeholders' views and needs are contained. Its goal is to gather security and privacy requirements for the infrastructure in question, and to make suggestions for improvement of the requirements specification. Multilateral security analysis takes into account all stakeholders' requirements relevant to security and privacy issues.

***Security Models:*** Finally, security models are developed for the VoIP profiles. A security model is based on security goals, and a trust model. It contains a description of:

- Subjects
- Objects
- Rules and policies
- Security functions

It is hard to retrieve stable, unified requirements from interviews with stakeholders. Therefore, it is necessary to have several cycles of interaction with the stakeholders to verify the requirements, profiles and models. Our approach to this problem is similar to rapid prototyping in software development: a fast, parallel development of requirements, to be presented and discussed with the stakeholders in several loops of interaction, such as *Maieutik* [21] and *Empathic Design* [2].

### *B. Configurations testbed and attacking*

For testing VoIP configurations and security profiles from our project partners we have developed a dedicated VoIP testbed. Testbeds as a research approach enable us to do prospective analysis of VoIP technology and to effectively gain knowledge about VoIP capabilities, limitations and benefits in different conditions [22]. This provides us with an advantage over a theoretical approach alone, since VoIP is tested in different contexts. The testbed is used as a controlled environment using strict configuration management to ensure scientific measurements. Specifically, we test various VoIP installations, where we launch predefined, reproducible attacks to uncover security vulnerabilities.

Real life VoIP has many deciding factors that have an impact on performance and security, such as the network topology, network congestion, and the protocols used. A theoretical approach alone cannot be employed to consider all these factors because of their complex relationships. Simulation is often used to study computer networks, since it offers a convenient combination of flexibility and controllability. The disadvantage of using simulations is that results may not be applicable to the reality, since often an inappropriate level of abstraction has been applied. The testbed creates an environment where the project researchers can experiment with different VoIP configurations in a low-risk environment, prior to real-world testing and deployment.

We pursue the following goals with the VoIP testbed:

(1) Given VoIP configurations are validated in the testbed against security requirements resulting from the previous analysis steps outlined above in Section II-A. Specifically, the experiments in the testbed shall show conformance between a given VoIP installation, configuration or architecture, and specified security requirements defined by the stakeholders.
While the testbed can be used in various ways, our work hypothesis is as follows: VoIP-specific security mechanism are deployed and tested to see if they are in accordance with the stakeholder's security policy. In this environment, the deployment of attacks will be launched to uncover

potential vulnerabilities. Data gathered from these tests will be used as input to formal modeling and verification, as outlined below in Section II-C.

(2) We use an automated VoIP testbed attack tool to scan a given VoIP installation for known vulnerabilities according to the threat model, and to launch VoIP related attacks.

(3) To be able to re-use a given testbed configuration as a reference configuration management is an important aspect of testbed testing. Especially the handling of a wide range of configuration files is considered as a challenge.

(4) Using the results from the tests we create VoIP configurations that are arguable more secure, based on our findings in the preceding three goals. These configurations, along with recommended best practices, are then presented to the stakeholders for discussion and further refinement.

Various VoIP configurations containing Asterisk PBXs as one of the components are used as target test systems in this testbed. These configurations are copies of real systems deployed in different organizations. When performing tests tests real traffic data are provided by mirroring data traffic into the testbed.

### C. Formal analysis of protocols

Formal protocol analysis is an important part, in addition to extensive security testing of real-world VoIP systems and traffic in the project's experimental testbed. We perform formal protocol analysis in combination with experiments in the testbed using the following methodological approach:

- Real-world production systems are installed and configured in the testbed.
- Network traffic from the testbed is recorded/logged (at a certain level of detail).
- Based on the logged network traffic and additional information, like RFCs, formal specifications are constructed.
- These specifications are further analyzed in a formal analysis tool, capable of identifying potential attacks and vulnerabilities affecting system security.
- In order to validate the results from the formal protocol analysis, attempts are made to reconstruct in the testbed on real-world systems the error conditions found in the formal analysis.

In Fig. 3 the work approach and data flow of our formal protocol analysis is illustrated in more detail. So far, the formal protocol analysis has been looking into the properties of SIP [12], [13]. SIP is used for signaling and is working together with other protocols that take care of the media stream, using, e.g., RTP (Real-time transfer protocol) [23]. SIP is a text-based protocol that needs to be strengthened to enhance security. We have been looking into SIP with digest authentication when analyzing SIP-based traffic [14].

In order to gain initial knowledge of the behavior of the SIP implementation of Asterisk, traffic is recorded from real phone sessions going through an Asterisk server. This is done by using VoIP-targeted IP network monitoring and interception tools such as *Wireshark*[7]. The traces of sessions produced by

[7]Wireshark web page: *www.wireshark.com*

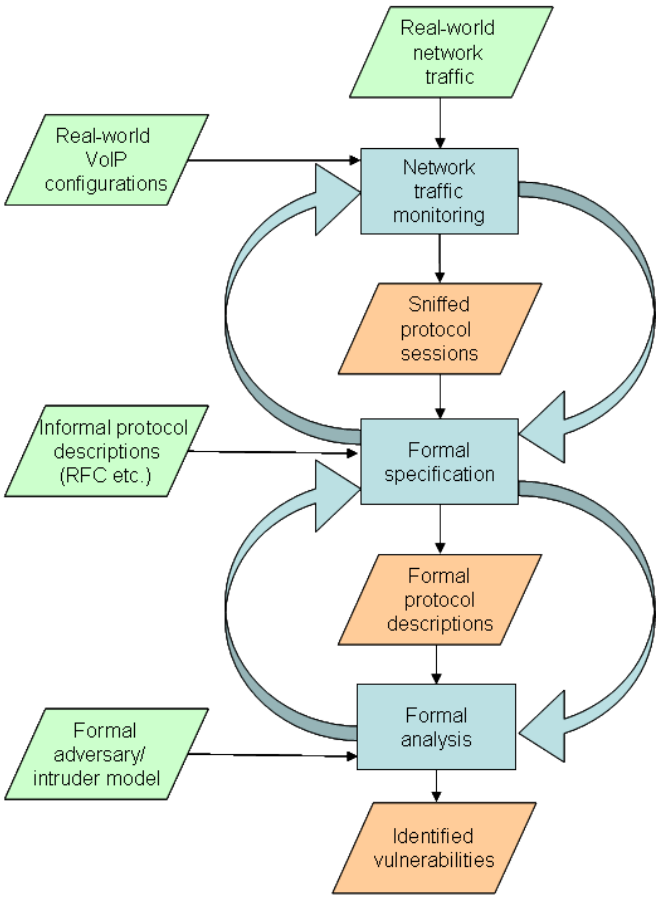



Fig. 3. Formal analysis of VoIP systems

*Wireshark* can be presented both textually and as interaction diagrams, at various levels of detail.

Based on the output from *Wireshark*, formal models/formal specifications are then produced. In this process, the SIP RFC specifications are used as additional guidelines and references. This transformation from the traces of SIP-sessions to formal specifications of the same sessions requires manual intervention, and several rounds of quality assurance.

Having produced the formal models from the SIP traces these are further analyzed in a formal protocol analyzer. We used a locally developed experimental tool for formal protocol analysis, PROSA [24], in the analysis so far. PROSA is based on temporal epistemic logic, and includes a module for automated refinement and validation of protocols.

PROSA is not the sole alternative, and different types of formal protocol analysis tools and methods are today available, of which some are listed below: Process calculus with probability and complexity [25], symbolic execution models/multiset rewriting [26], Protocol logics like BAN logic [27], model checking, either symbolic analysis like strand spaces or (exhaustive) finite state analysis like Murphi [28] or CASPER/FDR [29], and finally search using symbolic representation of states, e.g., the NRL analyzer [29]. Tools similar to PROSA include OFMC [30] and Scyther [31].

The purpose of formal protocol analysis is to look for non-intuitive attacks, omissions in specifications, or errors in different products implementation of protocols. During the analysis of VoIP protocols, networks are assumed to be hostile, in that they may contain intruders that can read, modify, or delete traffic, and that may have control of one or more network principals. Many of these attacks do not depend upon flaws or weaknesses in the underlying cryptographic algorithm, but can be exploited by an attacker. The results of the formal protocol analysis are validated in the testbed.

PROSA is a tool developed for the specification, static analysis and simulation of security protocols. PROSA consists of three main modules: (a) a specification language based on temporal epistemic logic; (b) a static analysis module; and (c) a simulator for executing intended protocols and attacks on protocols.

The language in the PROSA tool contains constructs for specification and reasoning about message transmission, cryptographic operations, and agent beliefs. Below is listed an excerpts of the PROSA language to be used later in this article Here $\mathscr{L}_P$ is the smallest language such that:

| | | |
|---|---|---|
| $(i)$ | Each of the following atomic formulas are in $\mathscr{L}_P$ | |
| | $\varepsilon$ | the empty sentence |
| | $a = b$ | equality |
| | $\mathsf{Agent}(a)$ | $a$ is an agent |
| | $\mathsf{isKey}(k)$ | $k$ is a key |
| | $\mathsf{isNonce}(\mathsf{n}(N, a))$ | $\mathsf{n}(N, a)$ is a nonce |
| | $\mathsf{playRole}(a, x, \mu)$ | $a$ plays the $x$-role in protocol $\mu$ |
| | $\mathsf{role}(a)$ | $a$ is a role in a protocol |
| $(ii)$ | If $\varphi$, $\psi$, $\xi^{\mathcal{T}}$, $\xi^{\mathcal{A}}$, $\xi^{\mathcal{S}} \in \mathscr{L}_P$, then so are; | |
| | $\neg\varphi$, $\varphi \rightarrow \psi$ | propositional logic |
| | $a \longrightarrow b : \varphi$ | $a$ sends the message $\varphi$ to $b$ |
| | $\mathsf{Bel}_a(\varphi)$ | $a$ believes $\varphi$ |
| | $\mathsf{Hash}[\varphi]$ | hash $\varphi$ |
| | $\mathsf{Enforce}_{t^A}(\varphi)$ | enforce agent $t^A$ to do $\varphi$ |
| | $\mathsf{protocol}[\mu, N, \xi^{\mathcal{T}}, \xi^{\mathcal{A}}, \xi^{\mathcal{S}}, \Phi]$ | protocol operator |

In addition there are constructs for, e.g., time stamps, quantifiers, encrypt, decrypt, and constructs that explain succession.

The static analysis module consists of algorithms for *automated refinement* of both protocol specifications and attack descriptions. The automated refinement results in an explicit specification that contains assumptions local to each agent participating, i.e. pre- and postconditions, for each transmission clause. Refined specifications can then be *validated*.

The validation process of a trace specification is performed in two steps in PROSA: First, a tool-supported refinement of the specification is generated. This will give a specification that contains information about the agents beliefs and construction of credentials, like the generation of nonces, timestamps, assumptions about keys, and cryptographic operation like encryption, decryption and hashing. Secondly, the refined specification is validated to check whether a participant in the protocol setting possesses any beliefs that have not been legally obtained through communication or cryptography.

The PROSA language is defined to be close to practical protocol specification and design, understandable for both software developers as well as system architects. The same language is also the metalanguage for reasoning about the protocol specification. In this way it differs from state-the-art tools like OFMC [30] and Scyther [31]. Here a specification is written in one language that is later preprocessed to an intermediate language serving as input to the reasoning tools.

The PROSA language has similarities with, e.g., BAN logic, yet there are some significant differences. The meaning of the belief operator is defined by the detailed definition of the protocol machine, which is a central part of the operational semantics of PROSA. Hence the belief operator is interpreted as part of the execution of protocols. Contrary to a purely logical explanation of abstract security properties and mechanisms, the belief construct is given a concrete operational meaning. Belief means possession, there is no other operator for reasoning about beliefs and data-content. Other logics, e.g., BAN logic, have several operators.

Although beliefs in PROSA are rather complex, in the way they are explained by many rules in the operational semantics, it is still possible to have a rather standard logical understanding of beliefs.

## III. Results and progress

In this section, we summarize the results and the progress so far with an emphasis on the areas of *formal protocol analysis*, *security modeling*, and *laboratory security testing*. Since the project will continue to work into 2011 we expect more results during its course.

### A. Formal protocol analysis

The SIP protocol specification, as described in RFC 3261 [13], is implemented differently in the various VoIP systems. We explored how Asterisk implements the SIP protocol by using formal protocol analysis. Real-world Asterisk configurations originating from an industrial partner were used as basis for our analysis.

Traffic was then monitored and recorded as a basis for the formal analysis, hence capturing the specifics of how SIP is implemented in Asterisk. The fact that Asterisk is implementing a B2BUA (a back to back user agent) instead of a SIP proxy became clear to us during the analysis.

Transforming a representation of a session from network traffic monitoring tool trace to a formal model in standard notation requires manual intervention. We identified the need of a tool that is able to export the traces representing real data traffic from *tcpdump* or *Wireshark* into a formal specification readable for the protocol analysis tool. Until such a tool is developed the transformation must be performed carefully in order to avoid errors in that process.

The PROSA syntax is using a standard Alice-Bob notation, [32] and standard notations for describing security protocols [33]. Hence the PROSA formulas presented in this section should be readable to those familiar with the above mentioned notations. We explain a few constructs using Fig. 4 as an example: The header of a protocol specification consists of a protocol name, then a session number – since there might be several instances – followed by specification of all roles, the

$\mathsf{protocolSIP}, 0,$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B),$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B),$
$\quad \mathsf{role}(A),$

$\mathsf{Enforce}_A(\mathsf{Bel}_A(\mathrm{startProtocol}(\mathrm{SIP-CANCEL},$
$\quad \mathsf{playRole}(B, C, \mathrm{SIP-CANCEL}) \wedge$
$\quad \mathsf{playRole}(S, T, \mathrm{SIP-CANCEL}) \wedge$
$\quad \mathsf{playRole}(A, D, \mathrm{SIP-CANCEL}),$
$\quad \mathsf{Text}(\text{Refer to session}))))$

$A \longrightarrow S : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge$
$\quad \mathsf{Text}(\text{Contact}, A) \wedge \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow A : \mathsf{Text}(\text{Proxy Authentication Required}) \wedge \mathsf{Text}(\text{Username}, A) \wedge$
$\quad \mathsf{Text}(\text{Realm}) \wedge \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$A \longrightarrow S : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$A \longrightarrow S : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge$
$\quad \mathsf{Text}(\text{Contact}, A) \wedge \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A)) \wedge$
$\quad \mathsf{Hash}[\mathsf{Hash}[\mathsf{Text}(\text{Username}, A) \wedge \mathsf{Text}(\text{Realm}) \wedge \mathsf{isKey}(\mathsf{key}(\mathsf{s}, A, S))] \wedge$
$\quad\quad \mathsf{isNonce}(\mathsf{n}(DIGESTRESPONSE, A)) \wedge$
$\quad\quad \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad\quad \mathsf{Hash}[\mathsf{Text}(\text{INVITE}) \wedge \mathsf{Text}(\text{URI}, B)]]$

$S \longrightarrow A : \mathsf{Text}(\text{100 TRYING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow B : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge$
$\quad \mathsf{Text}(\text{Contact}, A) \wedge \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$B \longrightarrow S : \mathsf{Text}(\text{100 TRYING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$B \longrightarrow S : \mathsf{Text}(\text{180 RINGING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$S \longrightarrow A : \mathsf{Text}(\text{180 RINGING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$B \longrightarrow S : \mathsf{Text}(\text{200 OK})$

$S \longrightarrow A : \mathsf{Text}(\text{200 OK})$

$A \longrightarrow S : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow B : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$A \longrightarrow B : \mathrm{start}(\mathrm{MediaTrans},$
$\quad \mathsf{playRole}(\mathrm{Alice}, A, \mathrm{MediaTrans}) \wedge$
$\quad \mathsf{playRole}(\mathrm{Bob}, B, \mathrm{MediaTrans}))$

Fig. 4. Specification of the SIP call setup sub-protocol

agent specific roles, and the start role. The Enforce construct builds instances of tear down subprocesses within each agent making them able to listen for CANCEL messages. In SIP a CANCEL message can appear whenever an agent hangs up the phone, from any state in a call setup process. Following the Enforce statement, in the specification in Fig. 4, 14 transmissions in sequential order are representing the SIP call setup signaling sequence.

In the PROSA tool a static analysis can be performed as follows. The initial protocol specification is automatically, by the tool, augmented with pre- and postconditions expressing beliefs and trust at each stage in the specification. Some statistics taken from the static analysis of the SIP call setup protocol specification is presented in Table I. The length of the protocol indicates the number of statements in the original specification. In our case the Enforce statement is followed

$\mathsf{protocol}[\mathrm{SIPAttack}, 0,$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B) \wedge \mathsf{role}(I) \wedge \mathsf{role}(F),$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B) \wedge \mathsf{role}(I) \wedge \mathsf{role}(F),$
$\quad \mathsf{role}(A),$

$A \longrightarrow I(S) : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$I(A) \longrightarrow S : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge$
$\quad \mathsf{Text}(\text{Contact}, A) \wedge \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow I(A) : \mathsf{Text}(\text{Proxy Authentication Required}) \wedge \mathsf{Text}(\text{Username}, A) \wedge$
$\quad \mathsf{Text}(\text{Realm}) \wedge \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$I(S) \longrightarrow A : \mathsf{Text}(\text{Proxy Authentication Required}) \wedge \mathsf{Text}(\text{Username}, A) \wedge$
$\quad \mathsf{Text}(\text{Realm}) \wedge \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$A \longrightarrow I(S) : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$I(A) \longrightarrow S : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A)$
$\quad \wedge \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$A \longrightarrow I(S) : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A)) \wedge$
$\quad \mathsf{Hash}[\mathsf{Hash}[\mathsf{Text}(\text{Username}, A) \wedge \mathsf{Text}(\text{Realm}) \wedge \mathsf{isKey}(\mathsf{key}(\mathsf{s}, A, S))] \wedge$
$\quad \mathsf{isNonce}(\mathsf{n}(DIGESTRESPONSE, A)) \wedge$
$\quad \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad \mathsf{Hash}[\mathsf{Text}(\text{INVITE}) \wedge \mathsf{Text}(\text{URI}, B)]]$

$I(A) \longrightarrow S : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A)) \wedge$
$\quad \mathsf{Hash}[\mathsf{Hash}[\mathsf{Text}(\text{Username}, A) \wedge \mathsf{Text}(\text{Realm}) \wedge \mathsf{isKey}(\mathsf{key}(\mathsf{s}, A, S))] \wedge$
$\quad \mathsf{isNonce}(\mathsf{n}(DIGESTRESPONSE, A)) \wedge$
$\quad \mathsf{isNonce}(\mathsf{n}(DIGESTCHALLENGE, S)) \wedge$
$\quad \mathsf{Hash}[\mathsf{Text}(\text{INVITE}) \wedge \mathsf{Text}(\text{URI}, B)]]$

$I(S) \longrightarrow A : \mathsf{Text}(\text{CANCEL}) \wedge \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$A \longrightarrow I(S) : \mathsf{Text}(\text{487 Request Terminated}) \wedge \mathsf{Text}(\text{URI}, A) \wedge$
$\quad \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$I(S) \longrightarrow A : \mathsf{Text}(\text{ACK}) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow I(A) : \mathsf{Text}(\text{100 TRYING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow I(B) : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$I(S) \longrightarrow B : \mathsf{Text}(\text{INVITE}) \wedge \mathsf{Agent}(A) \wedge \mathsf{Agent}(B) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$B \longrightarrow I(S) : \mathsf{Text}(\text{100 TRYING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$I(B) \longrightarrow S : \mathsf{Text}(\text{100 TRYING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$B \longrightarrow I(S) : \mathsf{Text}(\text{180 RINGING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$I(B) \longrightarrow S : \mathsf{Text}(\text{180 RINGING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$I(S) \longrightarrow B : \mathsf{Text}(\text{CANCEL}) \wedge \mathsf{Agent}(A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$B \longrightarrow I(S) : \mathsf{Text}(\text{487 Request Terminated}) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$I(S) \longrightarrow B : \mathsf{Text}(\text{ACK}) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$S \longrightarrow I(A) : \mathsf{Text}(\text{180 RINGING}) \wedge \mathsf{Text}(\text{Contact}, B) \wedge$
$\quad \mathsf{Text}(\text{URI}, B) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$I(B) \longrightarrow S : \mathsf{Text}(\text{200 OK})$

$S \longrightarrow I(A) : \mathsf{Text}(\text{200 OK})$

$I(A) \longrightarrow S : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLID, A))$

$S \longrightarrow I(B) : \mathsf{Text}(\text{ACK}) \wedge \mathsf{Text}(\text{Contact}, A) \wedge$
$\quad \mathsf{Text}(\text{URI}, A) \wedge \mathsf{isNonce}(\mathsf{n}(CALLIDB, S))$

$\mathsf{Enforce}_I(\mathsf{Bel}_I(\mathrm{startProtocol}(\mathrm{MediaTransAttack},$
$\quad \mathsf{playRole}(\mathrm{Malice}, I, \mathrm{MediaTransAttack}) \wedge$
$\quad \mathsf{playRole}(\mathrm{Frank}, F, \mathrm{MediaTransAttack}), \mathsf{Text}(\text{Reference to callid's}))))$

$\mathsf{Enforce}_F(\mathsf{Bel}_F(\mathrm{startProtocol}(\mathrm{MediaTransAttack},$
$\quad \mathsf{playRole}(\mathrm{Malice}, I, \mathrm{MediaTransAttack}) \wedge$
$\quad \mathsf{playRole}(\mathrm{Frank}, F, \mathrm{MediaTransAttack}), \mathsf{Text}(\text{Reference to callid's}))))$

Fig. 5. Call hijacking attack on the SIP call setup sub-protocol

TABLE I
STATISTICS ON THE SUB-PROTOCOLS.

| protocol | Length | Refined | Crypto | Validation |
|---|---|---|---|---|
| ... | ... | ... | ... | ... |
| Call Setup | 15 | 88 | 3 | 27812 |
| ... | ... | ... | ... | ... |

TABLE II
STATISTICS ON THE SIMULATIONS.

| Simulation scenario | PROSA rewrites | Time (milli seconds) |
|---|---|---|
| SIP without Digest | 18 239 | 41 |
| SIP Digest simulation | 82 987 | 164 |
| SIP with eavesdropper | 83 365 | 188 |
| Active call-hijacking attack | 364 969 | 472 |

by 14 transmissions, totalling 15 statements. The length of the automatically refined protocol quantifies the number of statements plus the additional pre-and post conditions. The number of cryptographic operations involved are 3 instances of a hash functions while the last column is a count of the number of rewrites in PROSA tool performed to validate the specification.

After finishing static analysis and validation, the next step is simulation. Our simulation scenario included three components, two calling parties *Alice*, *Bob*, and a proxy server. Each agent runs an instance of the SIP sub-protocols described above. Here, we assume that Alice initiates a phone call with Bob in three variations:

$(a)$ without Digest Access Authentication;
$(b)$ using Digest Access Authentication; and
$(c)$ using Digest Access Authentication, but with an attacker eavesdropping the messages.

A standard digest simulation without an attacker, (b), is augmented with an attacker on the line just forwarding the messages, (c). The number of computation steps required to perform an eavesdropping differs insignificantly from the "good" simulation. This augmenting is automatically done. The results of the PROSA simulations are reported in Table II. The first simulation is without Digest Access Authentication, while the latter three include Digest Access Authentication. The last one is manually derived from (c). This due to the need for the intruder- and adversary model for PROSA to be extended. What takes place in the latter simulation scenario is the following: An attack where an intruder Ivory (denoted $I$) hijacks a call-setup session and establishes a phone call with another agent Frank (denoted $F$), as described in Fig. 5.

Initial results of our work indicates potential vulnerabilities in SIP authentication [14] and call-setup [34] that can lead to attacks, based on analysis under the Dolev-Yao attacker/intruder model [35].

### B. Security modeling

In the following we show the characterization of VoIP scenarios. Six different scenario patterns were visualized graphically in a metaphor as islands. These depict different VoIP basic setups as shown in Fig. 3: *Island, Archipelagos, Nomadic Islanders, Nomadic Libertarians, Fortress, Maginot Line*. These have been verified in a pre-study with selected stakeholders in the project. These profiles are used as a basis for classification of VoIP setups, and will be the basis for the development of security models. The first round of stakeholder interviews was performed in 2008 and early 2009. Through our industry connections, we got access to one VoIP system vendor, and five VoIP system operators which include universities, public administrations, and service providers.

TABLE III
INTERVIEWEES AND THEIR ROLES

| Stakeholder | Role |
|---|---|
| 1 | VoIP service provider / system vendor |
| 2 | municipality |
| 3 | university |
| 4 | municipality |
| 5 | county administration |
| 6 | VoIP service provider / energy provider |

We observed that most of the stakeholders were acting in more than one role. The vendor offered both system-building and service operation. The service operators originated either from public administration, such as municipalities and counties, or power companies. Both forms of operators own rights to operate telecommunication cable.

The interviews focused on the business model, the customer and user profiles, and security needs and incidents. The interviews were performed as conversations with moderated discussion, where the topics were raised, discussed along the contributions of the interviewees, and terminated with a list of questions from the interviewers. The interviews aimed at classifying the interviewees into the island metaphors, at learning the security requirements and conceptions and the realities. The island metaphors were introduced early to enable an abstraction away from particular details of the telecommunications infrastructure or security technology, as the interviewees mostly had a background in telecommunication technology or network administration. The interviews were following an outline made for each stakeholder category. An example for the outline is shown in Fig. 6.

Concerning their business models, all interviewees shown in Table III provide VoIP-based telephony to their customers. While Stakeholder 1 operates on the open telecommunications market, Stakeholder 6 targets consumers along the power network they operate. Stakeholder 3 is a large university, where VoIP is currently built up to replace PSTN in the offices and laboratories. Generally, the municipal or county organizations seek to replace their own phone infrastructure with an Internet-based infrastructure motivated by cost of ownership. As a side effect, many organizations begin to include users outside the public administration offices, such as schools or medical service centers that are under their governance.

The major reason for choosing VoIP – and in particular Asterisk-based solutions – was the favorable costs of Asterisk-based telecommunications infrastructures. Many of the interviewees were operating old telephony switches, and were facing high maintenance cost and expensive offers for

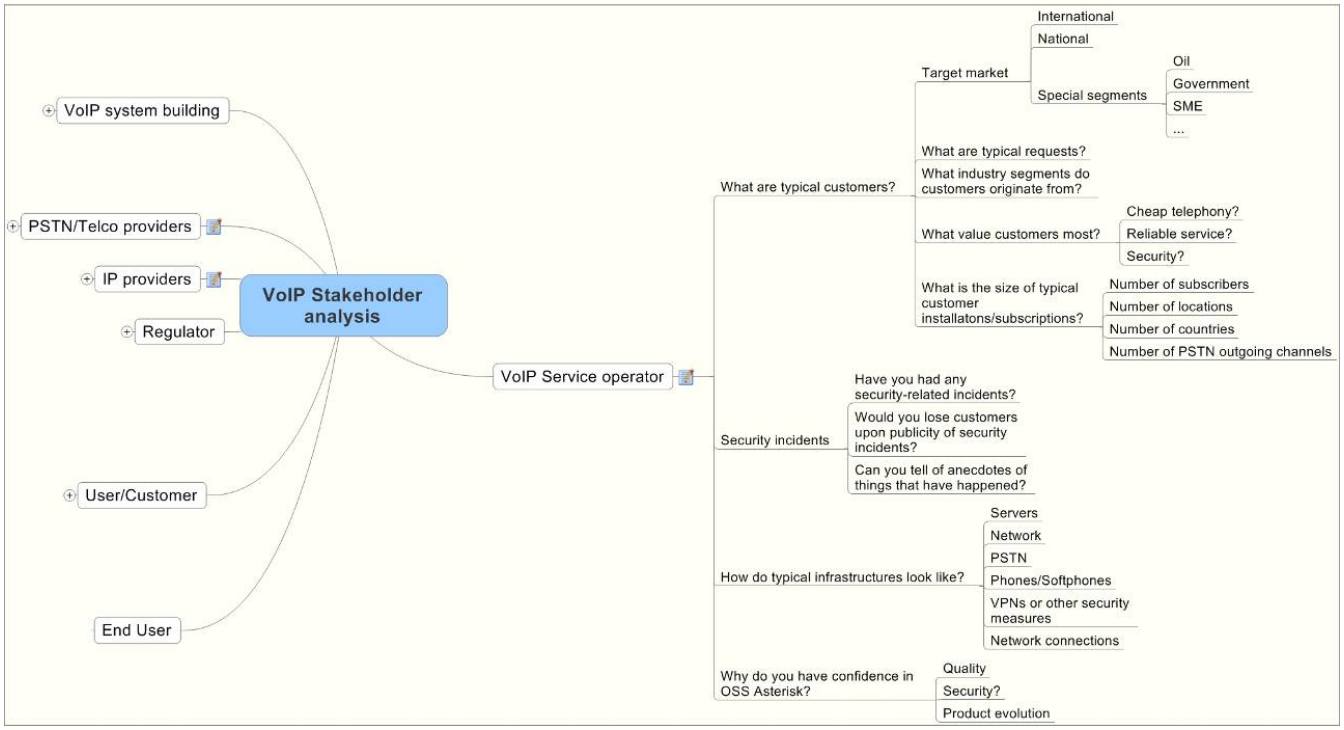



Fig. 6. Stakeholder interview outline for "VoIP system operator"

replacement of their PSTN switches. At the same time, they had already built up their own IP infrastructure. For most of the customers the seamless replacement of the ordinary desk or cordless phones with the same functionality was in focus. Only one of the stakeholders is actually deploying softphones on laptops for a particular user segment – school teachers who share offices that do not have personally assigned phones. In summary, most of the stakeholders' activities were targeted at migrating the switch-based phone functionality to VoIP.

The typical infrastructure is composed of one or more Asterisk servers, one or more PSTN trunks, and many pre-configured desktop VoIP phones for the end users.

Security concepts go along the lines of dedicated data connections, special routing or VPN tunneling. Probed for security measures and threat scenarios, the interviewees mainly responded that they were shielding their cable, or using dedicated IP addressing, MAC verification and on occasion VPN routers to "keep the VoIP traffic in its own network". This, in addition to the user-side need for the "old" telephony network, reinforces the insight that VoIP is built and used as if it was the PSTN. Asked for security incidents, the stakeholders reported a few billing fraud incidents, mainly based on successful ID theft based subscriber sign-ups. Some mentioned cost induced with 0900 service usage by their legitimate telephony users. The largest worries concerning security have been stated around the topic of identity fraud, fraudulent service usage, and losses due to fraudulent outgoing calls into a billed long-distance network – problems that pre-existed the times of VoIP. For some stakeholders availability of service, in particular of emergency calling, was an issue. None of the stakeholders mentioned IP-based attacks, session hijacking, break-ins into voice mail systems, SPIT calling or eavesdropping problems. There was a considerably low enthusiasm to discuss regulatory issues such as police wiretapping, data retention and crime investigation issues.

Some stakeholders, in particular the system builder, agreed that the complexity of configuration options in Asterisk and the related protocols and the options in the infrastructure is too high. Configuration errors are believed to provide greatly to potentials for unavailability of service or security problems.

Further interviewing and infrastructure inspection in EUX2010SEC will reveal whether some of the existing security threats on the Internet are known to the stakeholders, and help in the development of security concepts for VoIP infrastructures.

### C. Laboratory security testing

We work in close interaction with the industry partners participating in the project on how to set up, use, and test different VoIP configurations in the testbed. For this we install and configure different scenarios. For complex scenarios to be rolled out in real life, the industry partners install and configure the scenario in the testbed in order to get an implementation as close to reality as possible.

The routines for the VoIP testbed are as follows: After having installed and configured the lab to a given scenario, the setup is documented and the relevant configuration files are included into the configuration management. The testbed provides our partners with a VoIP infrastructure for experimentation, analysis, testing and prototyping of SIP/VoIP components in a controlled environment before deployment.

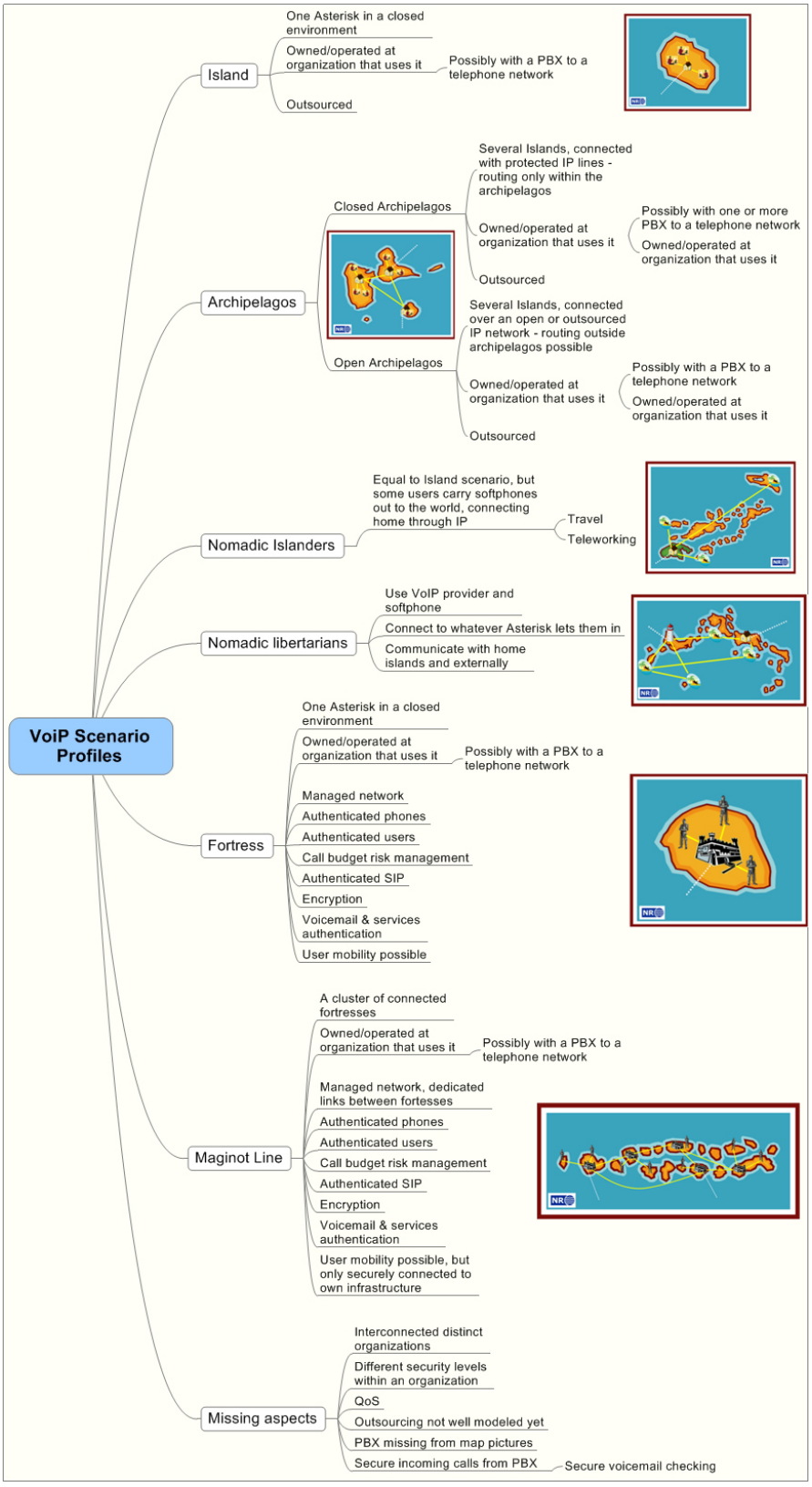



Fig. 7. Island metaphors for VoIP scenario profiles

The project partners responded entirely positive to having a testbed provided by the project. We observed a high interest to use the lab, and we conclude that there is a need for telecom testbeds available for research and experimentation.

A typical test-run proceeds as follows: The industry partner's request can range from a specific configuration they would like to test to a more broad "we want a more secure authentication". We then identify research questions that can be applied to this test. Examples for such research questions might be how to evaluate the performance difference between two authentication mechanisms, or to evaluate their vulnerability to remote attacks. After having configured the testbed, we execute a test, alongside which we have a range of different methods to measure on the testbed. For network performance tests, we use tools like *tcpdump, MRTG* and *Munin*, while for VoIP specific tests, we can use tools like *SIPP, SIPvicious, SIPSak, sip-kill, Scapy* or similar [36].

To implement the call-hijack attack [34] shown in Section III-A, we used three different tools: (1) the VoIP attack tool "sip-kill" to send the `SIP CANCEL` messages which block out Alice and Bob from the phone call, (2) the generic attack tool "Scapy" to send the remaining SIP messages between Frank and Ivory, and (3) the multimedia stream-server "VLC" to set up a RTP media stream between the attackers.

*Technical setup:* Our lab today consists of a wide variety of components with different hardware and software. The main platform for VoIP servers is Asterisk on Linux. See Table IV for a list of the equipment currently used in the lab.

We carefully document all different setups in an internal wiki, and keep all relevant configurations files under revision control. Using configuration management enables us to deploy repeatable, accurate test frameworks, to repeat a particular test under the same conditions for reproducibility, or to test a particular scenario with added functionality. In our lab we have set up and installed other standard services, such as internal DNS, email, LDAP, DHCP, and monitoring tools. These services are part of VoIP infrastructures, and therefore must be included in the testbed.

To capture raw network traffic from our testbed, we can use *tcpdump* on the participating hosts. However *tcpdump* can inflict a severe performance penalty at high network throughput, and thus (potentially) affects the measurement itself. To avoid this, we have enabled "port spanning", also called "port mirroring", on the network switch. This functionality duplicates network traffic from one network port to another. On the mirrored network port, we have a high end server running *tcpdump* that captures all network traffic.

We are aware that realistic VoIP experiments require a distributed testbed running over the Internet. Therefore, we have a permanent SIP trunk over the Internet to a public telephony provider in Norway. This enables us to make real-world phone calls. We have also performed VoIP tests to other project partners over the Internet using VoIP servers installed and configured at their locations.

Our current lab scenario setup is depicted in Fig. 9. The system layout is a replica of a large scale VoIP installation from one of our project partners. This configuration involves three SIP servers, 16 SIP phones as well as ordinary infrastructure

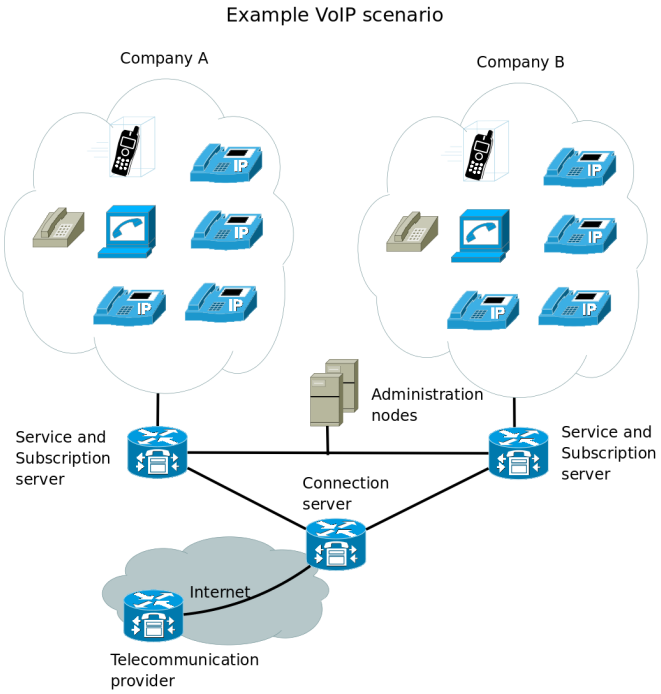



Fig. 9. A real-life VoIP scenario replicated in testbed

services like DNS, email and so forth. In this scenario, all the phones have real-world phone numbers (reachable from the outside). The two different network segments, labeled as "Company A" and "Company B" can also represent two different departments inside a larger company.

*Penetration testing:* An ongoing penetration test with external and internal attacks uses several security consultants as hired "evil hackers" trying to attack and compromise the installation. For this test we have set up an automatic phone conversation with a pre-recorded message setting up a new conversation every fifth minute, in which both participants play a pre-recorded message and then hang up. The conversation is between our testlab and a smaller lab located at one of our industry partners.

Each attacker gets an allocated time-slot (usually a day) where he can perform his attacks. The attackers are free to do whatever attack they can think of, but we instruct them to log every command (and output) with a timestamp, and we require that they write down a report of their method and findings. We will also debrief them after each attack attempt. At our side we carefully monitor the system for any changes, and we do a full network sniffing of all raw network traffic.

We plan several iterations using this scenario: We envisage first an external attack, and second an attack from the inside, impersonating a disgruntled employee of an organization. When the attackers perform an external attack, they are given two phone numbers and one external IP address of a VoIP server. Attackers on the inside also can log in and access the network infrastructure. As a usual action-pattern the attackers first gather information ("footprinting") about the victim, in terms of network infrastructure, VoIP platform, version num-

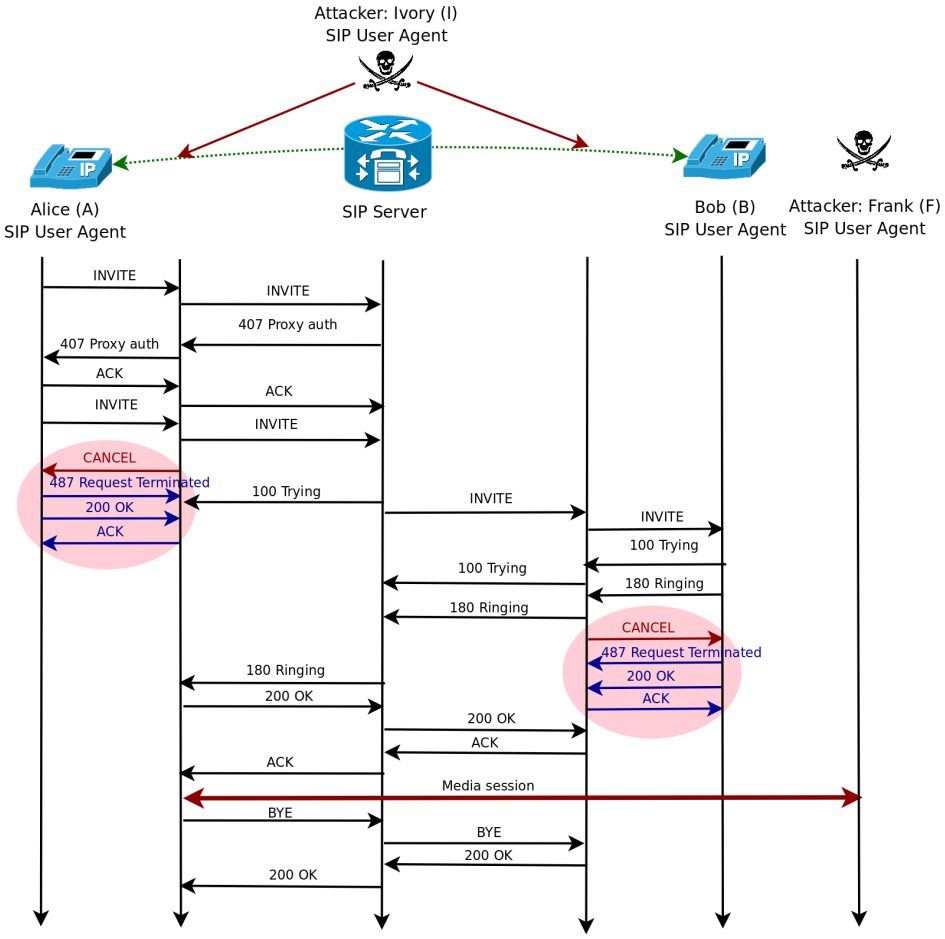



Fig. 8. Hijacking the initiator and the responder.

TABLE IV
LIST OF TESTBED EQUIPMENT AND FUNCTIONS

| Function | Equipment | Software | Comment |
|---|---|---|---|
| VoIP servers (UAS) | 3 high-end servers | Asterisk or OpenSIPS on Linux | Hardware typically used by several of our project partners |
| VoIP clients (UAC) | 16 SIP hardphones (8 different models), 2 different SIP softphones, 2 soft switchboards on laptop computers | Proprietary; softphones are free software. | Phone models typically used by our project partners. |
| Administrative functions | 1 high-end server, 1 desktop machine | DNS, LDAP, email, Subversion, Munin, Nagios, MRTG, Wiki | Relevant IT infrastructure services and monitoring |
| Network sniffing | 1 high-end server | tcpdump | Network sniffing to disk. |
| Attack nodes | 2 desktop machines | various | Various VoIP and network attack tools. |
| Connectivity | Internet, VPN | | Mobile users normally use VPN. Test of UAC over VPN. |

bers, and so on, before they perform any active attack.

To rank the attacks, we have set up a score board that is handed out to the attackers, with a prioritized list of security goals. The highest goal is modification of voice messages, i.e., to change one participants media stream (voice) in real-time undetected. We do not have any expectation that the attackers will be able to achieve this, but other more trivial attacks could be plausible, such as attacks on availability (DoS attack) or various SIP methods (registration, call-setup etc).

An external attack iteration is currently ongoing. During our experiment, one attacker was able to uncover a misconfigured service on the Asterisk VoIP server and log in. He did not manage to exploit this configuration error, but others might. Unless the attackers are able to compromise the VoIP server, we expect limited results from this iteration. Since the attackers do not have control over any relevant network infrastructure, it is hard or even impossible to intercept and modify the VoIP traffic.

## IV. CONCLUSION AND FUTURE WORK

As an outlook into the crystal ball for 2011, we see that EUX2010SEC will have developed security guidelines, best

practices and configurations for several VoIP scenarios that reflect business or user needs, and innovative options of VoIP technology. The configurations have been tested in the testbed, and aspects of them have been formally modeled and checked. The methodology of formal-methods based protocol analysis and implementation verification has been applied, improved and advanced. Thus we enable the practitioners to roll out better products and innovative services with high security levels.

From the interviews with stakeholders, we have had easy access to scenarios leading to only few of the profiles metaphors we have come up with. Is this due to our inability of covering all the different predefined profiles, or is this also reflecting the status, maturity, or majority of the (Norwegian) market? After having frequent contact with the VoIP market in Norway the last couple of years, it seems that replication of old telephony concepts onto VoIP infrastructure is where most organizations are today. The desire for enhanced functionality will sooner or later be pushing the limits in many organizations. The requirements elicitiation process therefore has to take into consideration both the requirements elicited from the interviews, but also near-future trends regarding the functionality. This makes it easier to help and guide organizations that are going to move from a conservative profile to a more challenging one.

The models described in this paper are based on security goals. Some of these goals might deduce sub-goals that are related to the selection of protocols and associated security mechanisms. Having the ability to use (deduced) security goals from the security models when performing formal protocol analysis, represents an added value when it comes to validation of systems against security models. Likewise, having a library of verfied protocols will also be valuable. Having a formal analysis of a protocol, the results are further taken into the testbed for validation. This to see if potential vulnerabilities identified in the formal analysis can be constructed at the system level, and under which conditions. Since the Asterisk systems are fully flexible the various configurations have to be validated against the security goals of the security models.

## V. Acknowledgements

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Research Council of Norway (Norges forskingsråd, project 180054). The authors would like to thank the anonymous reviewers for comments on earlier drafts of this paper.

## References

[1] Lothar Fritsch, Arne-Kristian Groven, and Lars Strand. A holistic approach to open-source VoIP security: Preliminary results from the EUX2010sec project. In *Proceedings of the Eight International Conference on Networking (ICN2009)*, pages 275–280. IEEE Computer Society, March 2009.

[2] Dorothy Leonard and Jeffrey Rayport. Spark innovation through emphatic design. *Harvard Business Review*, 75(6):102, 1997.

[3] Richard Lester. Universities, Innovation,and the competitiveness of local economies - MIT-IPC-05-010. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, December 2005.

[4] Thomas Porter. *Practical VoIP Security*. Syngress, March 2006.

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

[6] Jonathan Zar. VOIPSA VoIP Security and Privacy Threat Taxonomy - Public release 0.1. Technical report, October 2005.

[7] Richard Kuhn, Thomas Walsh, and Steffen Fries. Security Considerations for Voice over IP Systems - Recommendations of the National Institute of Standards and Technology. Technical report, US Nat'l Inst. Standards and Technology, Gaithersburg, MD, USA, 2005.

[8] M. Manulis, A. Adelsbach, A. Alkassar, K-H. Garbe, M. Luzaic, E. Scherer, J. Schwenk, and E. Siemens. VoIPSEC – Studie zur Sicherheit von Voice over Internet Protocol. Technical report, Godesberger Allee 185-189, 53175 BONN, 2005.

[9] Patrick Hung and Miguel Martin. Through the looking glass: Security issues in VoIP applications. In *IADIS International Conference on Applied Computing*, San Sebastian, Spain, 2006.

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

[11] Angelos D. Keromytis. Voice over ip: Risks, threats and vulnerabilities. In *Proceedings of the Cyber Infrastructure Protection (CIP) Conference*, New York, June 2009.

[12] 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.

[13] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. RFC 3261 - SIP: Session Initiation Protocol. Technical Report 3261, Internet Engineering Task Force, June 2002.

[14] Anders Moen Hagalisletto and Lars Strand. Formal modeling of authentication in SIP registration. In *Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08*, pages 16–21. IEEE Computer Society, August 2008.

[15] S. El Sawda and P. Urien. SIP Security Attacks and Solutions: A state-of-the-art review. In *Proc. 2nd conference on Information and Communication Technologies, ICTTA '06*, volume 2, pages 3187–3191. IEEE, 2006.

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

[17] Lothar Fritsch. Privacy-Respecting Location-Based Service Infrastructures: A Socio-Technical Approach to Requirements Engineering. *Journal of Theoretical and Applied E-Commerce research*, 2(3):1–17, December 2007.

[18] Lothar Fritsch and Tobias Scherner. A Multilaterally Secure, Privacy-Friendly Location-based Service for Disaster Management and Civil Protection. In Pascal Lorenz and Petre Dini, editors, *Networking - ICN 2005 - Proceedings of the 4th International Conference on Networking, Reunion Island (LNCS 3421), France, April 17-21, 2005*, volume 3421 of *Lecture Notes on Computer Science*, pages 1130–1137. Springer, Berlin, Heidelberg, New York, 2005.

[19] Benjamin L. Crosby. Stakeholder Analysis: A vital tool for strategic managers. *USAID IPC Technical Notes*, 2, March 1991.

[20] Günter Müller and Kai Rannenberg. *Multilateral Security in Communications - Technology, Infrastructure, Economy*. Addison-Wesley-Longman, München, 1999.

[21] Tom Sommerlatte. *Angewandte Systemforschung: ein interdisziplinärer Ansatz*. Gabler, Wiesbaden, first edition, 2002.

[22] J. Ramon Gil-Garcia, Theresa A. Pardo, and Andrea Baker. Understanding Context through a Comprehensive Prototyping Experience: A Testbed Research Strategy for Emerging Technologies. In *40th Hawaii International Conference on System Sciences (HICSS)*, page 104, Hawaii, 2007.

[23] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003.

[24] Anders-Moen Hagalisletto. *Automated support for the Design and Analysis of Security Protocols*. PhD thesis, University of Oslo, Oslo, June 2007.

[25] John C. Mitchell, Ajith Ramanathan, Andre Scedrov, and Vanessa Teague. A probabilistic polynomial-time process calculus for the analysis of cryptographic protocol. *Theoretical Computer Science*, 353(1-3):118–164, March 2006.

[26] Nancy A. Durgin, Patrick Lincoln, and John C. Mitchell. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004.

[27] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36, 1990.

[28] J. C. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using Murphi. In *IEEE Symposium on Security and Privacy 1997*, pages 141–151. IEEE Compuer Society, 1997.

[29] Gavin Lowe. Casper: a compiler for the analysis of security protocols. *Journal of Computer Security*, 6(1-2):53–84, 1998.

[30] David Basin, Sebastian Mödersheim, and Luca Viganò. OFMC: A symbolic model checker for security protocols. *International Journal of Information Security*, 4(3):181–208, June 2005.

[31] C.J.F. Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, USA, Proc.*, volume 5123/2008 of *Lecture Notes in Computer Science*, pages 414–418. Springer, 2008.

[32] Sebastian Mödersheim. Algebraic properties in alice and bob notation. *Availability, Reliability and Security, International Conference on*, 0:433–440, 2009.

[33] Jennifer G. Steiner, B. Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems. In *USENIX Winter*, pages 191–202, 1988.

[34] Anders Moen Hagalisletto, Lars Strand, Wolfgang Leister, and Arne-Kristian Groven. Analysing protocol implementations. In Feng Bao, Hui Li, and Guilin Wang, editors, *The 5th Information Security Practice and Experience Conference (ISPEC 2009)*, volume LNCS 5451, pages 171–182. Springer Berlin / Heidelberg, April 2009.

[35] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, Marc-1983 1983.

[36] Dorgham Sisalem, John Floroiu, Jiri Kuthan, Ulrich Abend, and Henning Schulzrinne. *SIP Security*. WileyBlackwell, March 2009.

# Paper B:
# Formal modeling of authentication in SIP registration

Anders Moen Hagalisletto, Lars Strand

In proceedings of the
$2^{nd}$ International Conference on Emerging Security Information, Systems and Technologies (SECURWARE) 2008
ISBN: 978-0-7695-3329-2

The Second International Conference on Emerging Security Information, Systems and Technologies

# Formal modeling of authentication in SIP registration

Anders Moen Hagalisletto
Norwegian Computing Center
and Department of Informatics,
University of Oslo, Norway
anders.moen@nr.no

Lars Strand
Norwegian Computing Center
and Department of Informatics,
University of Oslo, Norway
lars.strand@nr.no

## Abstract

*The Session Initiation Protocol (SIP) is increasingly used as a signaling protocol for administrating Voice over IP (VoIP) phone calls. SIP can be configured in several ways so that different functional and security requirements are met. Careless configuration of the SIP protocol is known to lead to a large set of attacks.*

*In this paper we show how 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 PROSA. In particular, we analyze the VoIP architecture of a medium size Norwegian company, and show that several of the well known threats can be found.*

## 1 Introduction

It was not until the late 1990s that phone calls over the Internet reached a significant number of users. The convergence of voice, video and data over the same IP infrastructure, reduces installation and maintainance cost since there is less need for separate networks. However, securing a VoIP system is challenging: Since VoIP shares the same infrastructure as traditional data networks, it also inherits the security problems of data communication. VoIP does not have a dominant standard that has been scrutinized over the years by security researchers. VoIP also has high requirements to the network with respect to Quality of Service 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 circuit-switched telephone networks. People have become accustomed to 99,999% availability on PSTN [10].

The Session Initiation Protocol (SIP) [12] is an application layer protocol for handling multimedia sessions. SIP is used to negotiate and establish a *context* for the media flow, where other protocols are used for the media transport. The media protocol Real Time Protocol (RTP) is often used in combination with SIP. SIP is a text based protocol, similar to HTTP.

Originally the focus on the SIP specification has been on functionality for providing additional services rather than security features [13]. Security was soon recognized to be an area of further investigation and improvement [2, 4, 6].

The rest of 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 of the case study show how the formal specifications can obtained. Section 4, the formalization of the protocol is presented, including a formalization of Digest Access Authentication, and the registration protocol. In Section 5, results from the analysis is presented, including an example of a call-hijacking attack on the registration sub-protocol and a description of the application of the tool PROSA.

## 2 Authentication in SIP

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

According to the RFC3261 [12], there are three ways to configure SIP authentication: plaintext authentication, weak authentication, and strong authentication. Plaintext authentication sends the authentication credentials unprotected. Weak authentication is an adaptation of the HTTP Digest Access Authentication [7] that requires a shared secret between the two participants. Strong authentication uses certificates in the same way as web browsers and

978-0-7695-3329-2/08 $25.00 © 2008 IEEE
DOI 10.1109/SECURWARE.2008.61

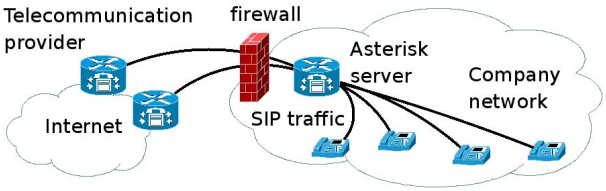



**Figure 1. Scenario for VoIP architecture.**

servers use them. Since it is currently common to use weak authentication, our paper discusses only Digest Access Authentication.

Digest Access Authentication method works like this: When receiving a request, the server may challenge the client with a random *nonce*. The client then hashes the nonce, secret password, username and other parameters. The server, upon receiving the hash from the client, does the same computation and compares 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 the Kerberos protocol or other public key based mechanism, see for instance [11].

## 3 Analysis

Based on the given voice over IP architecture, the documentation of SIP [12, 14], and Digest Access Authentication [7], we specified the particular instance of of SIP formally. Unfortunately, it was not possible to get a reasonable interpretation of the scenario, since the SIP RFC lacked several details of the message interactions and message content. Therefore we monitored the implementation by logging and analyzing network traffic using the network monitoring tool *Wireshark*[1].

### 3.1 VoIP in a medium size company

Our case study is taken from a medium sized Norwegian company with 100 employees. Most of them have a VoIP "hard phone" from Cisco. Some also have a "soft phone" installed on their computer, an application that supports VoIP phone calls. The configuration files counts 127 SIP phones, including both soft and hard phones. One VoIP server running the open source software *Asterisk*[2] on Linux provides the functionality of a Private Branch eXchange (PBX).

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

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

Phone calls destined to the outside, or incoming calls, are handled by an external telecommunication provider. The external connections use the Inter-Asterisk eXchange v2 protocol (IAX) [1]. An overview of the architecture is given in Figure 1.

The hard phones use DHCP to obtain and configure network settings. This simplifies network management. But since dynamic IP addresses are used, the phones need to register to the Asterisk server at startup time. During the registration they perform Digest authentication.

Two 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 *Wireshark*.

### 3.2 Method

It is well known that security protocols can be hard to specify formally [8, 3]. Based on information obtained from monitoring network traffic, a precise specification of SIP registration with Digest authentication was generated. This specification was used as input to the PROSA protocol analyzer in order to validate the specification and simulate uncompromized as well as compromised protocol instances. A severe call-hijacking attack was found eventually, which shows that formalization, simulation and automated analysis can reveal potential and real weaknesses with the implementation of VoIP systems.

We learned that the informal specifications in RFC 3261 were incomplete: Message interaction was not specified explicitly, neither was the the ordering of elements, and concrete adaptation of Digest Access Authentication.

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

The security analysis was performed by first obtaining a formal specification that could be studied in itself and by automated tool support. We used the protocol analyzer PROSA to specify a register session of SIP. The tool 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. By using PROSA typical errors like misprints of sender/receiver names or incomplete and incorrect message contents were easily discovered. 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 intercepts phone calls addressed to the client, but neither the client itself, nor the responder might know that the SIP channel is redirected and corrupted.

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

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

Written out explicitly the response yields:

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

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

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

Both parties, share a common secret, a password $K_{CR}^{\text{pwd}}$, 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 all the basic entities in the response in plaintext, except the password, and 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:

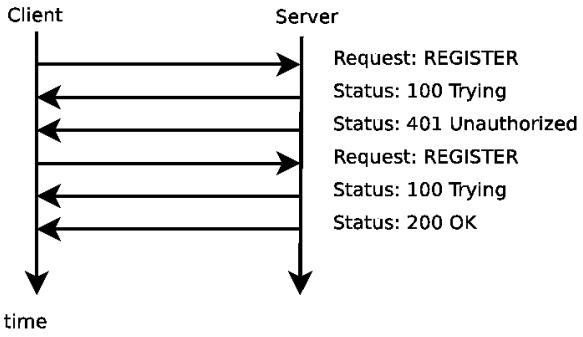



**Figure 2. Digest authentication in SIP.**

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

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

Initially in message $(\mathrm{P}_1)$, agent $C$ sends a registration request to the registrar server $R$. Agent $R$ gives a receipt that the registration has been received in $(\mathrm{P}_2)$, and then finally if $C$'s request is accepted by $R$, then a notification message is replied in $(\mathrm{P}_3)$. The message entities involved in the protocol scheme are:

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

The $N_C^{\mathrm{callid}}$ is the session identifier for the current instance of the protocol. In the realization $N_C^{\mathrm{callid}}$ 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 $N_C^{\mathrm{callid}}$, in order to prevent a delayed REGISTER request to arrive out of order [12, p. 58]. The available contact hosts can be more than one (e.g. several phones or email addresses), hence several potential bindings for the client can be specified by replacing $W_C^{\mathrm{Contact}}$, with a sequence of potential hosts $W^{\mathrm{Contact}} = W_C^{\mathrm{Contact}_1}, \ldots, W_C^{\mathrm{Contact}_n}$.

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

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

In this protocol 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 $(\mathrm{P}_1)$ and $(\mathrm{D}_2)$ 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 $K_{CR}^{\mathrm{pwd}}$ 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 interpreted stronger than $(\mathrm{P}_3)$, 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:

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

## 5 Attacks on registration

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 vulnerabilities of the company's VoIP system. By plugging in his laptop with VoIP attacker tool instead of his phone, he could easily launch attacks[3]. Once the attacker has access to the infrastructure, eavesdropping on phone calls are easy since VoIP-related communication in the company are transmitted unencrypted.

SIP calls routed externally to remote hosts, through several Internet domains might be subject to attacks as powerful as the Dolev Yao attacker [5]. 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. Hence $I$ cannot extract secret entities or decrypt encrypted messages if $I$ does not possess the required secret entities. However, the Dolev Yao attacker is assumed

[3] In our case, the attacker would have an easy target. All the phones in the company used the last three digits of the phones phone number as the shared secret.

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 implies that the attacker acts like a router: all messages pass through $I$. The attacker has the capability 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 $A \longrightarrow B \quad : \quad M$, 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).

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 $N_C^{\text{callid}}$, 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:

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

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 $W_C^{\text{Contact}}$, while the registration server is fooled to believe that $C$ should be contacted using the corrupt address $W_I^{\text{Contact}}$, 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 $W_I^{\text{Contact}}$.

The attacker $I$ is passive in the attack clauses $R_{1.2.a}^{\text{D}}$ to $R_{1.4.b}^{\text{D}}$, corresponding to the protocol clauses $\text{P}_2^{\text{D}}$ and $\text{P}_4^{\text{D}}$, the part of the protocol where authentication occurs, while $I$ is active and injecting the corrupt contact address in the protocol clauses $(\text{P}_1^{\text{D}}, \text{P}_5^{\text{D}}, \text{P}_6^{\text{D}})$. 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 compromise the contact address.The attack can be prevented by changing the Digest response to include the contact address(es):

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

When the registrar receives the response, the integrity of the contact address is preserved and fake contact addresses might be discovered. An alternative solution is to keep the specification unchanged but let $W_C^{\text{URI}} = W_C^{\text{Contact}}$ in order ascertain the integrity of the contact address.

### 5.1 Analysis of SIP in PROSA

The translation from the standard protocol specifications in Section 4 is straightforward as shown in Figure 3. A concise specification of the registration sub-protocol was obtained quickly by using *Wireshark*. It took approximately six man hours of work to have a executable specification of the concrete setup of SIP, thanks to the advanced validation mechanism of PROSA [9]. A test-scenario with the user and the Asterisk server was configured: Each agent possesses the protocol and capability of constructing appropriate nonce. Three configurations of the scenario were applied, that is, simulations with: (a) perfect network and no attacker, (b) an eavesdropper sniffing all messages, (c) an active call-hijacking attack. In each case Digest authentication was used, with assumption (DY-1). The integrity of the contact address was not preserved in the active call-hijacking attack: 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 for 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 protocols from IETF, like SIP, 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, an instance of the general call-hijacking attacks. Both

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

**Figure 3. SIP registration specified in PROSA.**

the discovery as well as the repair relied on the formal specification of the case study that was investigated. A practical problem discussed in this paper is that when mechanisms, like Digest Access Authentication, is combined with a given protocol, like registration, the security of the combined protocol is only clearly understood when it is formally analyzed. The approach taken in the paper can be used to rapidly specify and analyse different aspects and scenarios of SIP, closely related to implementations. Various configuration Call setup, Secure SIP, routing over several proxies can be explored with the same techniques as proposed in the paper.

## Acknowledgments

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council. The authors would like to thank Wolfgang Leister, Arne-Kristian Groven, Lothar Fritsch, Bjarte M. Østvold and the anonymous reviewers for comments on earlier drafts of this paper.

## References

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

[2] 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.

[3] 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.

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

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

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

[7] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication. RFC 2617 (Draft Standard), June 1999.

[8] 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.

[9] 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*.

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

[11] Wenbo Mao. *Modern Cryptography: Theory and Practice*. Prentice Hall, 2004.

[12] 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.

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

[14] 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, Aug 2006.

# Paper C:
# Designing Attacks on SIP Call Setup

Anders Moen Hagalisletto, Lars Strand

In
International Journal of Applied Cryptography, vol 2, 2010
ISSN: 1753-0563

# Designing attacks on SIP call set-up

Anders Moen Hagalisletto and Lars Strand*

Department of Applied Research in Information Technology (DART),
Norwegian Computing Center, Norway
Email: anders.moen@nr.no
Email: lars.strand@nr.no
*Corresponding author

**Abstract:** Many protocols running over the internet are neither formalised, nor formally analysed. The amount of documentation for telecommunication protocols used in real-life applications is huge, while the available analysis methods and tools require precise and clear-cut protocol clauses. A manual formalisation of the Session Initiation Protocol (SIP) used in Voice over IP (VoIP) applications is not feasible. Therefore, by combining the information retrieved from the specification documents published by the IETF and traces of real-world SIP traffic, we craft a formal specification of the protocol in addition to an implementation of the protocol. In the course of our work we detected several weaknesses, both of SIP call set-up and in the Asterisk implementation of the protocol. These weaknesses could be exploited and pose as a threat for authentication and non-repudiation of VoIP calls.

**Keywords:** formal methods; VoIP; voice over IP; SIP; session initiation protocol; authentication; call hijack attack; PROSA.

**Reference** to this paper should be made as follows: Hagalisletto, A.M. and Strand, L. (2010) 'Designing attacks on SIP call set-up', *Int. J. Applied Cryptography*, Vol. 2, No. 1, pp.13–22.

**Biographical notes:** Anders Moen Hagalisletto received his Bachelor's degree in Mathematics and Philosophy in 1993, Master's degree in Philosophy in 1997, and Master's degree in Computer Science in 1999. He received his PhD in Computer Security in 2007. His current research interests include industrial applications of formal methods, in particular automated software engineering; petri nets models of railway systems and tools for testing and analysing security protocols. He is currently developing the PROSA tool for analysing security protocols, in addition to consultancy for the industry.

Lars Strand is a PhD student at Norwegian Computing Center. He recevied his Bachelor's degree in Philosophy and Computer Science in 2002, and Master's degree in Computer Science in 2004 at the University of Oslo and Norwegian Defence Research Establishment. He is also employed as a System Consultant at the open source company Redpill Linpro where he works on Linux security and network-related topics. He has also created and held several security and Linux-specific courses. His current research interest includes voice-over-IP protocols with focus on security and free and open source software.

## 1 Introduction

Voice over IP (VoIP) is widely used, and is about to replace the traditional, Public Switched Telephone Networks (PSTN) for two main reasons: (1) service providers and customers experience cost savings, especially for long distance calls; since VoIP uses the internet as carrier, the cost of setting up a phone call needs no more effort than sending an email and (2) added functionality and flexibility; the VoIP protocols are capable of providing a number of additional services like instant messaging, presence, conferencing, events notification, video calls and other multimedia transmissions and location independence (mobility).

VoIP services cannot rely their security on the telecommunication infrastructure, dedicated lines, physically protected switches and certified telephony equipment. A number of VoIP security threats have been identified (VoIPSA, 2005) and discussed (Keromytis, 2009). Even if VoIP services have to be secured by cryptographic techniques, the employed protocols and their implementations must undergo a thorough formal crypto-analysis.

A common combination in VoIP is to use the Session Initiation Protocol (SIP) (Rosenberg et al., 2002) for signalling, e.g. setting up and tearing down calls, and specific protocols for the actual media transfer. The designers of SIP focused on functionality for providing specific services rather than security features (Salsano et al., 2002). However, security issues have been recognised to be an area for further investigation and improvement (Arkko et al., 2003; Geneiatakis et al., 2005; Kuhn et al., 2005; Endler and Collier, 2006). Discussions about potential weaknesses and attacks on SIP have, in most cases, been kept on an informal level (Sinnreich and Johnston, 2006; Persky, 2007; Porter, 2006; Xin, 2007; Zhang et al., 2007).

Our goal has been to use formal modelling of SIP in order to (1) verify whether the Asterisk implementation of SIP follow the specifications and (2) perform attacks exploiting weaknesses in the protocol definition, and its implementations. This work is based on Hagalisletto and Strand (2008), where the digest access authentication in SIP registration is analysed, and Hagalisletto et al. (2009) where an SIP call hijack is found and formalised. We use the same authentication mechanism in the call set-up explained in this paper.

There have been other works that analyse SIP and its security configurations formally (Gupta and Shmatikov, 2007; Diab et al., 2008), and the work of the AVISPA project[1]. However, they consider the authentication and registration sub-protocol in combination with the Diametre protocol, rather than the call set-up protocol as presented here.

The rest of this paper is organised as follows: In Section 2, we give a high level overview of SIP and the tool PROSA that we have used to analyse the call set-up. In Section 3, the formal specification of digest access authentication and call set-up is described. After discussing whether Asterisk implements the SIP correctly (Section 3.4), in Section 4 we show that a vicious attack on the call set-up specification can be performed using the specification obtained previously. In Section 5, we implement the attack in a testbed. Finally, in Section 6, we discuss and evaluate the approach.

## 2 Background

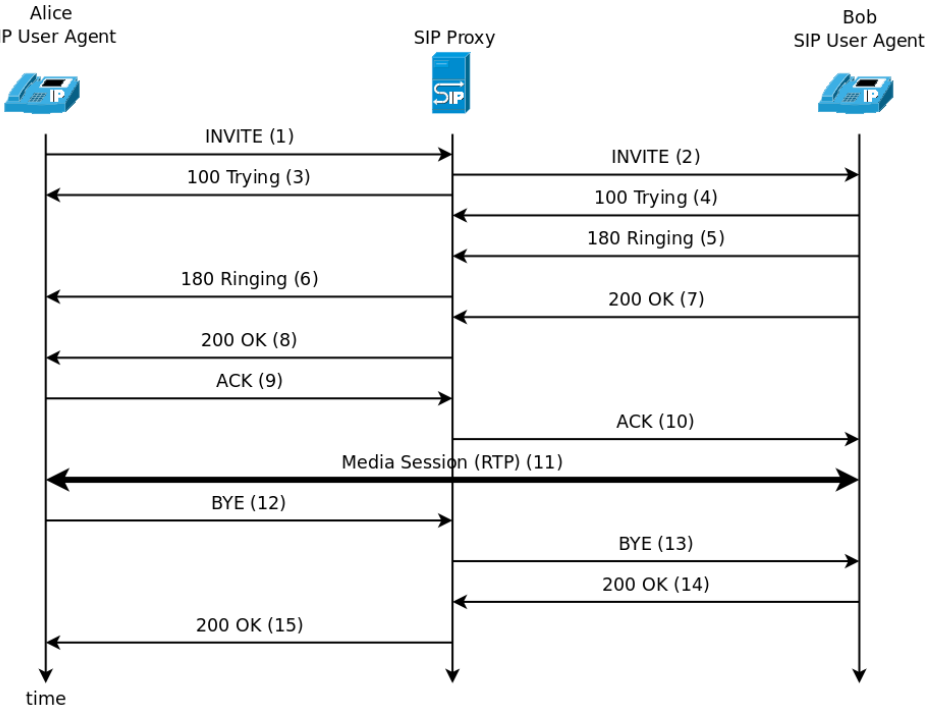
SIP is an application layer signalling protocol developed by the IETF. The core functionality of SIP is specified in RFC3261 (Rosenberg et al., 2002), additional functionality is specified in over 40 RFCs, and nearly 30 pending SIP-related drafts.[2] SIP is used to establish, maintain and tear down multimedia sessions between two or more participants. A session can be an ordinary call between two participants or an advanced multimedia conference session with several participants. More specific, SIP sets up the session context, but does not carry multimedia content.

We illustrate the operation of SIP with a scenario where Alice calls Bob. A session including call set-up, media transfer and tear down is shown in Figure 1.

While a session generally may traverse several SIP proxies, we restrict our scenario to only one SIP proxy. The SIP proxy has three roles: (a) it acts as registration server, (b) it handles call set-up and (c) it routes the SIP messages and in some cases the media stream. In the call set-up Alice calls Bob sending an INVITE message (1, 2). A receipt for each such hop is returned by the 'Trying' message (3, 4). If Bob's phone is connected then it starts to ring and propagates the 'Ringing' message back to Alice (5, 6). When Bob answers the call, he sends an OK message routed back to Alice through proxy $S$. Bob answers the phone in message (7), and the call content (voice) is transferred using the Real-time Transport Protocol (RTP) (Perkins, 2003) (11). Alice terminates the call (12) and a BYE message is sent. Before the INVITE message (1) is sent, an SIP proxy may challenge Alice to authenticate, as formalised in Section 3.1.

A dialogue between two user agents Alice and Bob is confirmed when Bob gets an acknowledgement (message 11) that Alice has accepted to communicate. A dialogue that is not confirmed is called an early dialogue. A dialogue can be terminated before it is confirmed. In case the caller is terminating the dialogue prematurely, the BYE method may be used. In case the callee is terminating the call, the method CANCEL must be used instead of BYE according to the standard. Thus this unclear specification of session termination is also reflected in section 15: 'The notion of "hanging up" is not well defined within SIP' (Rosenberg et al., 2002, p.90). The caller can also use CANCEL in an early dialogue.

**Figure 1** Flow graph showing successful session establishment and termination using SIP (see online version for colours)



For the purpose of this paper we restrict our work to how the widely used open source telephony platform Asterisk[3] (Meggelen et al., 2005) implements SIP. Asterisk is a private branch exchanges (PBX) whose functionality is to connect phone calls. Asterisk also supports a range of other common telephony services like voice mail, conference calls and telephone menus.

### *2.1 The method*

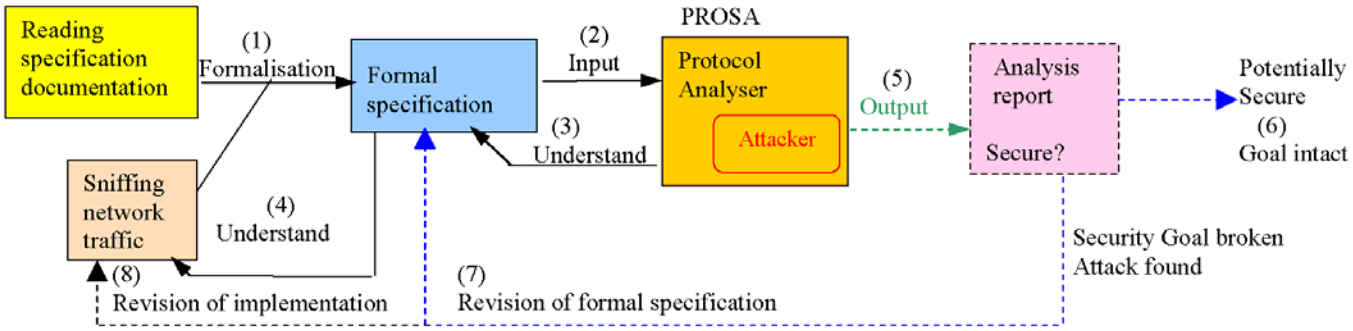
In order to gain initial knowledge of the behaviour of the SIP implementation in Asterisk, we record traces from real phone traffic going through the Asterisk server on a real-world Asterisk configuration. This is done by using the network monitoring tool Wireshark.[4] Traces retrieved from Wireshark can be presented both textually and as interaction diagrams.

The process of obtaining a formal specification of SIP involves the following steps. From the core IETF standards we derived an accurate description of SIP as possible. These resulting specifications were typically incomplete, interaction diagrams showing the transmissions and message content were lacking. Traces of the call set-up with real softphones were then used to supplement the incomplete specifications, with details of message credentials. Hence, based on the Asterisk traces and SIP standard we constructed the formal models manually, and analysed the specifications in the protocol analyser PROSA (Hagalisletto, 2007).

The analysis process is depicted in Figure 2. In this process, both IETF documentation and the traces are used to obtain a formal description (1) which typically enhances the understanding of the implementation (4). The formal specification is then validated by PROSA (2). If there are any errors or unreasonable elements found at this stage, the formal specification is revised (3).

A correct protocol specification will then be subject to hand-crafted or automatically generated attacks. The correctness of manually constructed attacks will then be checked by validation in a similar way as for non-compromised protocol specifications, thereafter simulated in PROSA.

**Figure 2** Workflow for analysis of implementations (see online version for colours)




Finally, the output of the analysis is a report that either confirms the initial security requirements or points at weaknesses that break some security goals (5). If no attacks are found then the protocol is considered preliminary secure (6). If an attack is found then the protocol is not secure and a revision is made (7). Since the formal specifications are derived from a concrete implementation, the report gives feedback onto the implementation (8).

### *2.2 The protocol analyser PROSA*

PROSA is a tool developed for the specification, static analysis and simulation of security protocols. PROSA consists of three main modules: (a) a specification language based on temporal epistemic logic; (b) a static analysis module and (c) a simulator for executing intended protocols and attacks on protocols. The static analysis module consists of algorithms for *automated refinement* of both protocol specifications and attack descriptions. The automated refinement results in an explicit specification that contains local assumptions, i.e. pre- and post-conditions, for each transmission clause. Refined specifications can then be *validated*.

The validation process of a trace specification is performed in two steps in PROSA: First, a tool-supported refinement of the specification is generated. This will give a specification that contains information about the agents beliefs and construction of credentials, like the generation of nonces, timestamps, assumptions about keys and cryptographic operation like encryption, decryption and hashing. Secondly, the refined specification is validated to check whether a participant in the protocol setting possesses any beliefs that have not been legally obtained through communication or cryptography.

## 3 Formal specification of SIP call set-up

SIP defines distinct functionality for registration, call set-up and modification, call control and mid-call signalling. We refer to these parts as 'sub-protocols' since each of these transactions requires its own sequence of message exchanges.

We take a closer look at the sub-protocols *digest access authentication*, *call set-up* and *call teardown*. *Call teardown* denotes the explicit event of terminating a call, specified by message 12–15 in Figure 1. These are specified in a form commonly used in the literature.

A protocol clause where 'agent $A$ sends a message $M$ to the agent $B$' is of the form:

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

Messages in the protocols consist of basic entities as follows:

| | |
|---|---|
| $A, B, C, D, T, S, I, I(A)$ | Agent terms |
| $K_{AB}$ | Symmetric key shared by A and $B$ |
| $N_A$ | Nonce generated by $A$ |
| $W_A^Y$ | String containing the text $Y$ related to $A$ |
| $X_A$ | Miscellaneous entities related to $A$ |

We use three composition operators in the notation: concatenation of message content denoted by ',' (comma), hashing H[$M$] and encryption denoted by $KM$, where $K$ denotes a key and $M$ a message content. The particular agent term $I(A)$ reads that 'the intruder $I$ impersonates as $A$'.

### *3.1 Authentication*

According to RFC3261 (Rosenberg et al., 2002), there are three ways to configure SIP authentication: plain-text authentication, weak authentication and strong authentication. Plain-text authentication sends the authentication credentials unprotected. Weak authentication is an adaptation of the HTTP digest access authentication (Franks et al., 1999) that requires a shared secret between the two participants. Strong authentication uses S/MIME (Ramsdell, 2004) and requires the participants to own personal certificates. Strong authentication has failed to gain widespread adoption due to increased complexity and cost (Sisalem et al., 2009).

A typical application of digest access authentication is given by a challenger $S$ requesting a client $A$ to authenticate as described in the following protocol skeleton:

$$
\begin{aligned}
&(\mathrm{D}_1) \quad S \to A \;:\; N_S \\
&(\mathrm{D}_2) \quad A \to S \;:\; W_A^{\text{uname}}, W^{\text{realm}}, N_S, W_A^{\text{URI}}, X_{\text{nc}}, N_A, \\
&\qquad\qquad \mathrm{H}\Big[\mathrm{H}\Big[W_A^{\text{uname}}, W^{\text{realm}}, K_{AS}^{\text{pwd}}\Big], \\
&\qquad\qquad\quad N_S, N_A, \mathrm{H}\Big[W^{\text{meth}}, W_A^{\text{URI}}\Big]\Big]
\end{aligned}
$$

Agents $A$ and $S$ share the symmetric key $K_{AS}^{\text{pwd}}$. Initially, the challenger $S$ sends a nonce $N_S$ to the client $A$. The client responds by sending the basic entities in plain text, except the password, and then the response itself, the payload of $D_2$.

The entities involved in digest access authentication are as follows:

| | |
|---|---|
| $W_A^{\text{uname}}$ | username of $A$ |
| $W^{\text{realm}}$ | a protective domain |
| $K_{AS}^{\text{pwd}}$ | shared password between $A$ and $S$ |
| $W^{\text{meth}}$ | main method of message (like HTTP) |
| $W_A^{\text{URI}}$ | digest URI for client $A$ |
| $N_S$ | nonce of the challenger $S$ |
| $N_A$ | $A$'s nonce |

The authentication is one way: $A$ is authenticated to $S$, guaranteed by the secrecy of the shared key $K_{AS}^{\text{pwd}}$. Agent $S$ can be certain that the message comes from $A$, since $A$ is the only agent except $S$ who possesses the key. Integrity of the message entities involved is provided by the fact that the hash could only be generated by $A$ and freshness of the message is provided by the challenger nonce $N_S$.

### *3.2 Teardown sub-protocols*

The trace described in Figure 1 is one out of many possible traces. *Teardown* of sessions can be performed at any stage in the session. Therefore, the final four messages 12–15 in Figure 1 and ($T_{14}$–$T_{17}$) in Figure 4 can be considered as a teardownBYE sub-protocol run by three agents.

Instances of the teardown protocol is running in parallel with the call set-up protocol, which implies that a BYE message received to any participant causes the host session of the call set-up to be terminated. An SIP-compliant specification where SIP methods are propagated correctly results in the following specification of teardown, extracted from the Wireshark protocol dump:

$$
\begin{aligned}
&(\mathrm{TB}_1) \quad C \to T \;:\; W^{\text{BYE}}, D, W_D^{\text{URI}}, W_C^{\text{Contact}}, W_C^{\text{URI}}, \\
&\qquad\qquad\qquad N_C^{\text{callid}}, N_D^{\text{callid}} \\
&(\mathrm{TB}_2) \quad T \to D \;:\; W^{\text{BYE}}, C, W_C^{\text{URI}}, N_D^{\text{callid}} \\
&(\mathrm{TB}_3) \quad D \to T \;:\; W^{\text{OK}}, W_D^{\text{Contact}}, W_D^{\text{URI}}, N_D^{\text{callid}} \\
&(\mathrm{TB}_4) \quad T \to C \;:\; W^{\text{OK}}, W_D^{\text{Contact}}, W_D^{\text{URI}}, N_C^{\text{callid}}
\end{aligned}
$$

where $C$ denotes the role of the agent initiating the teardown, $D$ denotes the responder agent, while $T$ denotes the proxy server. Instances of the teardown protocol are started by the call set-up protocol, while the call set-up is terminated by the teardown protocol.[5]

Since the callee might terminate a session before the dialogue is confirmed, using the CANCEL method, we have a sub-protocol teardownCANCEL, which is analogous to teardownBYE, except that the SIP method BYE is replaced with CANCEL , throughout in the previous specification, in addition to two chains of receipts.

$$
\begin{aligned}
&(\mathrm{TC}_1) \quad C \to T \;:\; W^{\text{CANCEL}}, D, W_D^{\text{URI}}, W_C^{\text{Contact}}, W_C^{\text{URI}}, \\
&\qquad\qquad\qquad N_C^{\text{callid}}, N_D^{\text{callid}} \\
&(\mathrm{TC}_2) \quad T \to D \;:\; W^{\text{CANCEL}}, C, W_C^{\text{URI}}, N_D^{\text{callid}} \\
&(\mathrm{TC}_3) \quad D \to T \;:\; W^{\text{OK}}, W_D^{\text{Contact}}, W_D^{\text{URI}}, N_D^{\text{callid}} \\
&(\mathrm{TC}_4) \quad D \to T \;:\; W^{\text{RequestTerminated}}, N_D^{\text{callid}} \\
&(\mathrm{TC}_5) \quad T \to C \;:\; W^{\text{OK}}, W_D^{\text{Contact}}, W_D^{\text{URI}}, N_C^{\text{callid}} \\
&(\mathrm{TC}_6) \quad T \to C \;:\; W^{\text{RequestTerminated}}, N_D^{\text{callid}} \\
&(\mathrm{TC}_7) \quad C \to T \;:\; W^{\text{ACK}}, W_C^{\text{Contact}}, W_C^{\text{URI}}, N_C^{\text{callid}} \\
&(\mathrm{TC}_8) \quad T \to D \;:\; W^{\text{ACK}}, W_D^{\text{Contact}}, W_D^{\text{URI}}, N_D^{\text{callid}}
\end{aligned}
$$

The teardownCANCEL protocol contains additional notification messages, first the '487 Request Terminated' method succeeding each OK message, and then finally the propagation of an acknowlegment for the agent cancelling the phone call, that the other party has teared down its session ($TC_7$ and $TC_8$).

### *3.3 Formalising call set-up permitting arbitrary teardown*

Since we do not know which of the agents actively closes a session, we model that each agent starts a passive session of the teardown sub-protocol. In the example shown in

Figure 1 Alice actively closes the session, where she plays the role of *B* of the teardown sub-protocol. Consequently, the Asterisk server acts in the role of *S*. Generally, both Alice and Bob might actively tear down a session by starting an instance of teardown in the role of *A*.

Combining the Wireshark protocol dump with the SIP specification we get Figure 3. In order to model this we go beyond standard protocol notation and introduce the clauses ($Q_1$–$Q_6$). The call set-up involves the additional SIP methods and primitives:

| | |
|---|---|
| $W^{\text{INVITE}}$ | The INVITE method that indicate a request for phone call |
| $W^{\text{ACK}}$ | An acknowledgement method |
| $W^{\text{PAR}}$ | Proxy Authentication Required |
| $W^{\text{Trying}}$ | 100 Trying, a receipt to a previous SIP message |
| $W^{\text{OK}}$ | 200 OK method gives a notification of a successful registration |
| $W^{\text{Ringing}}$ | The responder's phone is ringing |
| $W^{\text{BYE}}$ | Hang up phone |
| $W^{\text{CANCEL}}$ | Tearing down session prematurely |
| $W^{\text{RequestTerminated}}$ | Receipt of CANCEL termination |
| $N_A^{\text{callid}}, N_B^{\text{callid}}$ | Call identifiers for the sessions $A-S$ and $B-S$, respectively |

Clause ($Q_1$) reads 'agent *A* locally starts a session of the teardownCANCEL protocol, such that agent *A* plays the responder role *D*, agent *B* plays the initiator role *C* and the server *S* plays the server role'. The notation *AC* means that the agent *A* plays the *C* role in the given protocol, similar to procedure calls with parameter substitution in ordinary programming languages. The first local clause ($Q_1$) states that *A* initially starts listening for a possible CANCEL message from *B* within the early dialogue. The server propagates the CANCEL and OK methods involved in the teardown sub-protocol. In an early dialogue, the caller Alice is the only UAC that can send terminate an early dialogue using the BYE method. Hence, Bob might receive a BYE from the caller Alice, specified by ($Q_3$). In case of the server there are two cases: clause ($Q_2$) starts a session where the server *S* is waiting for a BYE from *A*, while in ($Q_4$) the server starts a similar session waiting for *B* sending a CANCEL message. When the dialogue is confirmed, the callee Bob might terminate the call seup by using BYE; hence, both Alice and server *S* initiates instances of the teardownBYE protocol in clause ($Q_5$) and ($Q_6$), respectively.

### *3.4 Deviations from the SIP specification*

The trace in Figure 4 shows that the Asterisk implementation of SIP diverges from the specification described in Figure 3 in three ways:

1 Alice's phone starts to ring (message $T_7$) *before* Bob is authenticated to the server. The meaning of an incoming 'Ringing' message received by Alice is that Bob has received an INVITE message, and she is ready to start a call if Bob answers the call. Hence, in order to follow the SIP RFCs message ($T_9$) should come *before* message ($T_7$). Hence, Alice is fooled to believe that Bob's phone is ringing, which is not the case. Therefore, we simulated scenarios where the responder Bob was disconnected from the network just after receiving the INVITE message. Alice still received a Ringing message. This behaviour is also confirmed in experiments using soft phones.

2 The acknowledgement received by Bob in message ($T_{11}$) arrives *before* Alice sends the message in clause ($T_{13}$). However, at this point Bob is mislead to believe that Alice has acknowledged the Ringing request from Bob.

3 After teardown initiated by Alice ($T_{14}$), the OK message from $S$ to Alice ($T_{15}$) is sent *before* the related OK message is sent from Bob. This breaks the specification, since ($T_{15}$) would implicate that Bob has received the BYE message, which is not the case in the implementation.

**Figure 3** Call set-up with flexible teardown

$(Q_1)$ $A$ : start(teardownCANCEL, $\text{Role}(B,C), \text{Role}(A,D), \text{Role}(S,T)$)

$(P_1)$ $A \to S$ : $W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}$

$(Q_2)$ $S$ : start(teardownBYE, $\text{Role}(A,C), \text{Role}(B,D), \text{Role}(S,T)$)

$(P_2)$ $S \to A$ : $W^{\text{PAR}}, W_A^{\text{uname}}, W^{\text{realm}}, N_S, A, B, N_A^{\text{callid}}$

$(P_3)$ $A \to S$ : $W^{\text{ACK}}, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}$

$(P_4)$ $A \to S$ : $W^{\text{INVITE}}, A, B, N_S, W_A^{\text{Contact}}, W_A^{\text{URI}}, N'_A, N_A^{\text{callid}}$ $\text{H}[\text{H}[W_A^{\text{uname}}, W^{\text{realm}}, K_{AS}^{\text{pwd}}], N'_A, N_S, \text{H}[W^{\text{INVITE}}, W_B^{\text{URI}}]]$

$(P_5)$ $S \to A$ : $W^{\text{Trying}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}$

$(P_6)$ $S \to B$ : $W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_B^{\text{callid}}$

$(Q_3)$ $B$ : start(teardownBYE, $\text{Role}(B,D), \text{Role}(A,C), \text{Role}(S,T)$)

$(P_7)$ $B \to S$ : $W^{\text{Trying}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}$

$(Q_4)$ $S$ : start(teardownCANCEL, $\text{Role}(B,C)\text{Role}(A,D), \text{Role}(S,T)$)

$(P_8)$ $B \to S$ : $W^{\text{Ringing}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}$

$(P_9)$ $S \to A$ : $W^{\text{Ringing}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}$

$(P_{10})$ $B \to S$ : $W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}$

$(P_{11})$ $S \to A$ : $W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}$

$(P_{12})$ $A \to S$ : $W^{\text{ACK}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}$

$(Q_5)$ $A$ : start(teardownBYE, $\text{Role}(B,C), \text{Role}(A,D), \text{Role}(S,T)$)

$(P_{13})$ $S \to B$ : $W^{\text{ACK}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_B^{\text{callid}}$

$(Q_6)$ $S$ : start(teardownBYE, $\text{Role}(B,C), \text{Role}(A,D), \text{Role}(S,T)$)

$A \leftrightarrow B$ : Media session

**Figure 4** A trace of call set-up and teardown using Asterisk as server $S$

$$
\begin{aligned}
(\mathrm{T}_1)\ & A \to S : W^{\mathrm{INVITE}}, A, B, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_2)\ & S \to A : W^{\mathrm{PAR}}, W_A^{\mathrm{uname}}, W^{\mathrm{realm}}, N_S, A, B, N_A^{\mathrm{callid}} \\
(\mathrm{T}_3)\ & A \to S : W^{\mathrm{ACK}}, B, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_4)\ & A \to S : W^{\mathrm{INVITE}}, A, B, N_S, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, \\
& \quad N'_A, N_A^{\mathrm{callid}} \\
& \quad \mathrm{H}[\mathrm{H}[W_A^{\mathrm{uname}}, W^{\mathrm{realm}}, K_{AS}^{\mathrm{pwd}}], N'_A, \\
& \quad N_S, \mathrm{H}[W^{\mathrm{INVITE}}, W_B^{\mathrm{URI}}]] \\
(\mathrm{T}_5)\ & S \to A : W^{\mathrm{Trying}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_6)\ & S \to B : W^{\mathrm{INVITE}}, A, B, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_7)\ & S \to A : W^{\mathrm{Ringing}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_8)\ & B \to S : W^{\mathrm{Trying}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_9)\ & B \to S : W^{\mathrm{Ringing}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_{10})\ & B \to S : W^{\mathrm{OK}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_{11})\ & S \to B : W^{\mathrm{ACK}}, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_{12})\ & S \to A : W^{\mathrm{OK}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_{13})\ & A \to S : W^{\mathrm{ACK}}, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
& A \leftrightarrow B \quad \text{Media session (RTP or SRTP)} \\
(\mathrm{T}_{14})\ & A \to S : W^{\mathrm{BYE}}, B, W_B^{\mathrm{URI}}, W_A^{\mathrm{Contact}}, W_A^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_{15})\ & S \to A : W^{\mathrm{OK}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_A^{\mathrm{callid}} \\
(\mathrm{T}_{16})\ & S \to B : W^{\mathrm{BYE}}, A, W_A^{\mathrm{URI}}, N_B^{\mathrm{callid}} \\
(\mathrm{T}_{17})\ & B \to S : W^{\mathrm{OK}}, W_B^{\mathrm{Contact}}, W_B^{\mathrm{URI}}, N_B^{\mathrm{callid}}
\end{aligned}
$$

This implementation can lead to unexpected results:

An obvious attack targets the Ringing method: an intruder $I$ could act as an eavesdropper until clause ($\mathrm{T}_7$), then take over Bob's session entirely, kick Bob out of the call and for the rest of the trace masquerade as Bob. Persons that are used to the particularly quick response (immediate ringing) from Asterisk-based VoIP would not be alerted when the intruder $I$ impersonates as Bob later in the session.

## 4 Attack on the call set-up

In the following, we consider attacks that do not rely on successful registration attacks. We assume that the attacker $I$ is as powerful as the Dolev Yao attacker (Dolev and Yao, 1983) who controls the entire network, can intercept any message, impersonate as any other agent and inject whatever entity it knows into SIP messages. Cryptography is assumed to be perfect, no brute force attacks on the underlying algorithms are considered in this paper.cryptographic

In the following, we describe how it is possible for an attacker to hijack both the initiator and responder roles. In the initial part of the attack, described in Figure 6, the intruder only passively listens in the authentication sub-protocol. In the protocol clauses ($P_{1.1.a}$) through ($P_{1.4.b}$) the intruder acts as a passive man-in-the-middle, obtaining information from plain-text entities. From the knowledge gained during the initial eavesdropping, an attacker can perform a combined attack on both the caller and the callee, as shown in Figure 5.

**Figure 5** Hijacking the initiator and the responder (see online version for colours)

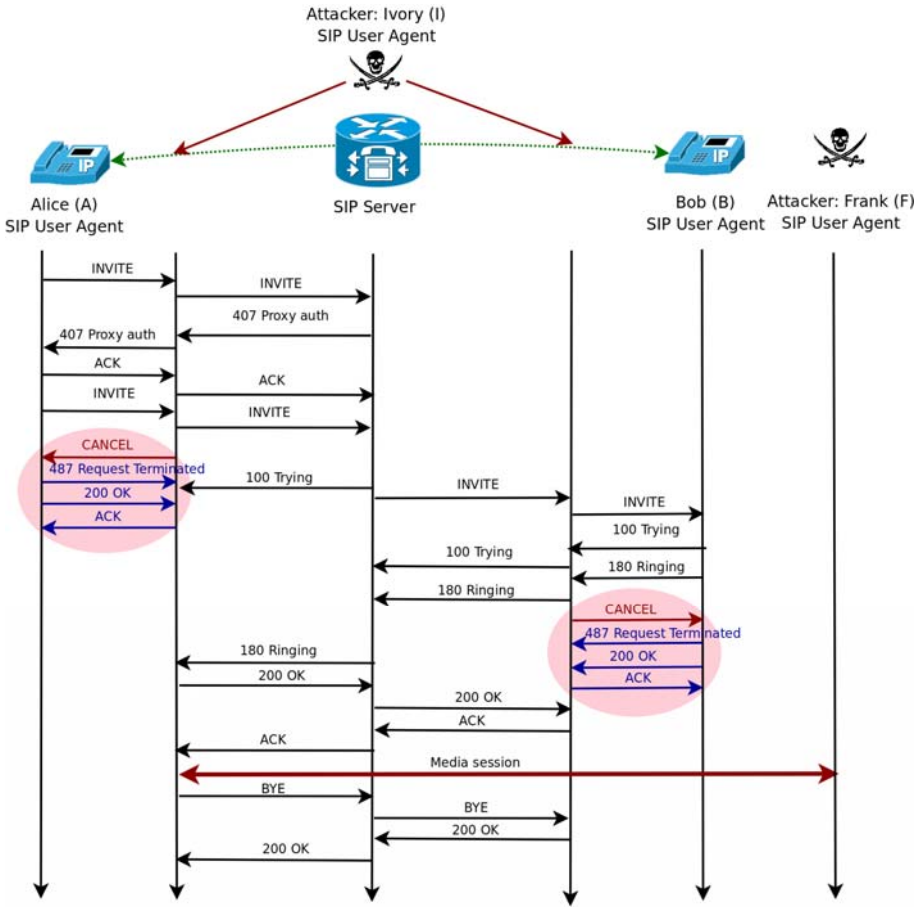


**Figure 6** Formal attack when hijacking the initiator and the responder (see online version for colours)

$$
\begin{aligned}
&(P_{1.1.a}) \quad A \to I(S) : W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{1.1.b}) \quad I(A) \to S : W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{1.2.a}) \quad S \to I(A) : W^{\text{PAR}}, W_A^{\text{uname}}, W^{\text{realm}}, N_S, A, B, N_A^{\text{callid}}\\
&(P_{1.2.b}) \quad I(S) \to A : W^{\text{PAR}}, W_A^{\text{uname}}, W^{\text{realm}}, N_S, A, B, N_A^{\text{callid}}\\
&(P_{1.3.a}) \quad A \to I(S) : W^{\text{ACK}}, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{1.3.b}) \quad I(A) \to S : W^{\text{ACK}}, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{1.4.a}) \quad A \to I(S) : W^{\text{INVITE}}, A, B, N_S, W_A^{\text{Contact}}, W_A^{\text{URI}}, N'_A, N_A^{\text{callid}}\\
&\qquad\qquad \text{H}[\text{H}[W_A^{\text{uname}}, W^{\text{realm}}, K_{AS}^{\text{pwd}}], N'_A,\\
&\qquad\qquad N_S, \text{H}[W^{\text{INVITE}}, W_B^{\text{URI}}]]\\
&(P_{1.4.b}) \quad I(A) \to S : W^{\text{INVITE}}, A, B, N_S, W_A^{\text{Contact}}, W_A^{\text{URI}}, N'_A, N_A^{\text{callid}}\\
&\qquad\qquad \text{H}[\text{H}[W_A^{\text{uname}}, W^{\text{realm}}, K_{AS}^{\text{pwd}}], N'_A,\\
&\qquad\qquad N_S, \text{H}[W^{\text{INVITE}}, W_B^{\text{URI}}]]\\
&(T_{2.3.2}) \quad I(S) \to A : W^{\text{CANCEL}}, B, W_B^{\text{URI}}, N_A^{\text{callid}}\\
&(T_{2.3.3}) \quad A \to I(S) : W^{\text{OK}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(T_{2.3.4}) \quad A \to I(S) : W^{\text{RequestTerminated}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(T_{2.3.5}) \quad I(S) \to A : W^{\text{ACK}}, N_A^{\text{callid}}\\
&(P_{2.5}) \quad S \to I(A) : W^{\text{Trying}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{2.6.a}) \quad S \to I(B) : W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.6.b}) \quad I(S) \to B : W^{\text{INVITE}}, A, B, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.7.a}) \quad B \to I(S) : W^{\text{Trying}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.7.b}) \quad I(B) \to S : W^{\text{Trying}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.8.a}) \quad B \to I(S) : W^{\text{Ringing}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.8.b}) \quad I(B) \to S : W^{\text{Ringing}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(T_{2.4.2}) \quad I(S) \to B : W^{\text{CANCEL}}, A, W_A^{\text{URI}}, N_B^{\text{callid}}\\
&(T_{2.4.3}) \quad B \to I(S) : W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(T_{2.4.4}) \quad B \to I(S) : W^{\text{RequestTerminated}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(T_{2.4.5}) \quad I(S) \to B : W^{\text{ACK}}, N_B^{\text{callid}}\\
&(P_{2.9}) \quad S \to I(A) : W^{\text{Ringing}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{2.10}) \quad I(B) \to S : W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_B^{\text{callid}}\\
&(P_{2.11}) \quad S \to I(A) : W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{2.12}) \quad I(A) \to S : W^{\text{ACK}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(P_{2.13}) \quad S \to I(B) : W^{\text{ACK}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_B^{\text{callid}}\\
&\mathbf{I(A) \leftrightarrow F(B)} \quad : \mathbf{Media\ session}\\
&(T_{2.2.1}) \quad I(A) \to S : W^{\text{BYE}}, B, W_B^{\text{URI}}, W_A^{\text{Contact}}, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(T_{2.5.2}) \quad S \to I(B) : W^{\text{BYE}}, A, W_A^{\text{URI}}, N_A^{\text{callid}}\\
&(T_{2.2.3}) \quad I(B) \to S : W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_D^{\text{callid}}\\
&(T_{2.2.4}) \quad S \to I(A) : W^{\text{OK}}, W_B^{\text{Contact}}, W_B^{\text{URI}}, N_A^{\text{callid}}
\end{aligned}
$$

In the attack, as shown, the attacker Ivory (denoted *I* in Figure 6) begins by eavesdropping the initial four messages concerned with establishing the call and authenticating the caller Alice to the server. Then Ivory tears down Alice's session prematurely by using a BYE message, and thereafter terminates Bob's session before he has entered the media session. Before the media session has started, the attacker has taken over the call, and can start a conversation with agent Frank (denoted *F* in Figure 6).

The attacker tears down the session after pretending to be Alice. The server *S* cannot discover that the two local sessions at each calling party are teared down. The attack effectively breaks the authenticity of the participants, since we no longer can trust the identity of the users involved in the phone call. As a consequence the intruder *I* could set up an arbitrary call, that Alice is billed for and that the logs, that telephony providers are obliged to carry out by legislation, are incorrect. Hence, the attack shows that non-repudiation is broken as well.

## 5 Real-world attack

A large number of VoIP attack tool exists (Endler and Collier, 2006; Porter, 2006; Park, 2008; Sisalem et al., 2009), and many of these are SIP specific. To implement the attack, SIP messages must be intercepted, crafted and then injected into the ongoing SIP transaction. Some SIP attack tools are specific, like SIPp,[6] and some are generic which can be used to craft arbitrary SIP messages, like SipSak.[7] SipSak takes a text file, with handcrafted SIP messages, as input.

The attacker must intercept the SIP header Call-ID and the 'tag' parameter used in the To: and From: header fields to masquerade as a legitimate user. These three random values are used to identify the SIP dialogue. The SIP messages can be captured using the network sniffer tcpdump or Wireshark manually, but this operation must be done automatically in real time, since the ongoing SIP dialogue must be intercepted and modified.

To implement, we split the attack into three main tasks. The first task is to block out Alice and Bob from the phone call using SIP CANCEL . Then we change the RTP media (voice) metadata in the SIP stream to use Frank's and Ivory's IP address instead of Alice's and Bob's. Third and finally, a RTP stream between Frank and Ivory is set up.

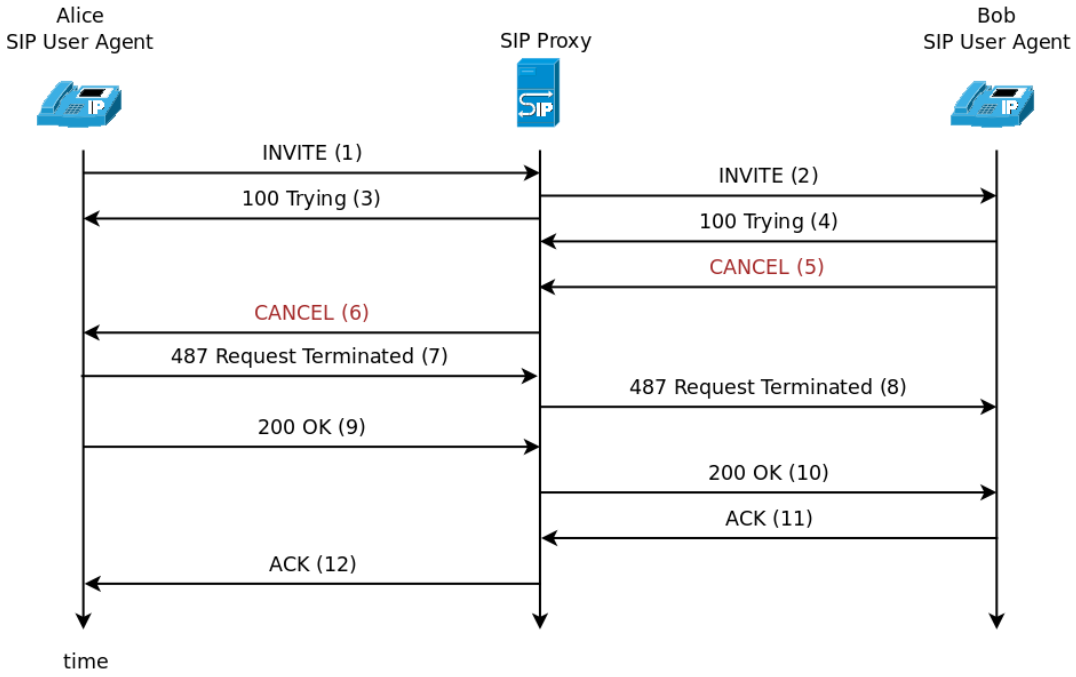
### *5.1 First part: block out Alice and Bob*

We can block out Bob and Alice from the phone call by either sending an SIP BYE or SIP CANCEL message. A BYE message is normally sent when one of the calling parties hangs up the phone. According to the SIP specification (Rosenberg et al., 2002, p.89) the caller can send a BYE message before the INVITE processing is completed, but the callee cannot. The callee can only send a BYE *after* the INVITE processing is completed and the callee has taken the phone (see the first ten messages in Figure 1). If the callee, in our case Bob, wants to abort an ongoing INVITE processing he must send a CANCEL. The CANCEL message can be sent by both the caller and the callee before the INVITE processing is completed. The result of a CANCEL message is an immediate teardown of the ongoing INVITE session.

Sending an SIP CANCEL message initiates a 487 Request Terminated response from the receiver and effectively tears down the ongoing INVITE session *before* the callee answers the phone (see Figure 7). The CANCEL message includes a Call-ID value that identifies this particular SIP transaction.

We use a slightly modified version of the VoIP attack tool sip-kill[8] which implements this part of the attack. The program listens to the network stream, identifies a SIP INVITE transaction and store its Call-ID value. As soon as this value is fetched, two SIP CANCEL messages with this Call-ID is injected. One CANCEL message to Bob masqueraded as Alice, and another to Alice masqueraded as Bob. The result is an immediate teardown of Alice's and Bob's INVITE transaction (phone call). In our testbed Alice has number 1004 and Bob number is 1005.

**Figure 7** Using SIP CANCEL to tear down an ongoing INVITE dialogue (see online version for colours)




A screen dump from the attack tool in action can be seen below. The first line is the command itself. The option to sip-kill specifies which network interface to listen to and from who should the CANCEL be masqueraded as:

```
1  sip-kill -ieth0 -dfrom
2  CANCEL INVITE from 1004@euxss1 to
   1005@euxss1
3  CANCEL INVITE from 1005@euxss1 to
   1004@euxss1
```

This part of the attack was implemented on a Linksys hardphone and a Linux softphone (see Figure 8) and both teared down their ongoing SIP INVITE transaction.

## 5.2 *Second part: modify RTP metadata*

An SIP server can be configured to route the media (voice) *through* the SIP server. Routing the media through the SIP server must be done if (1) the SIP server needs to transcode the media stream, or (2) the SIP server needs to perform Interactive Voice Response (IVR) to detect keypad inputs from the callers, or (3) the SIP server also act as a network router. The network administrator can also force the media stream to go through the SIP server for additional control of the network traffic to apply QoS, VLAN-trunking or similar. In our testbed, the media is routed through the SIP server for either of these reasons.

The media (voice) itself is carried using the RTP (Schulzrinne et al., 2003). To set up a duplex RTP stream, variables must be agreed upon before communicating. Variables like IP address and port number to use, codec to use and a number of other variables. This metadata is carried in the SIP INVITE message body using the Session Description Protocol (SDP) (Handley et al., 2006), as shown in Figure 9.

**Figure 8** Some of testphones. A screenshot of the Twinkle softphone at the top and Linksys WIP330 hardphone at the bottom (see online version for colours)

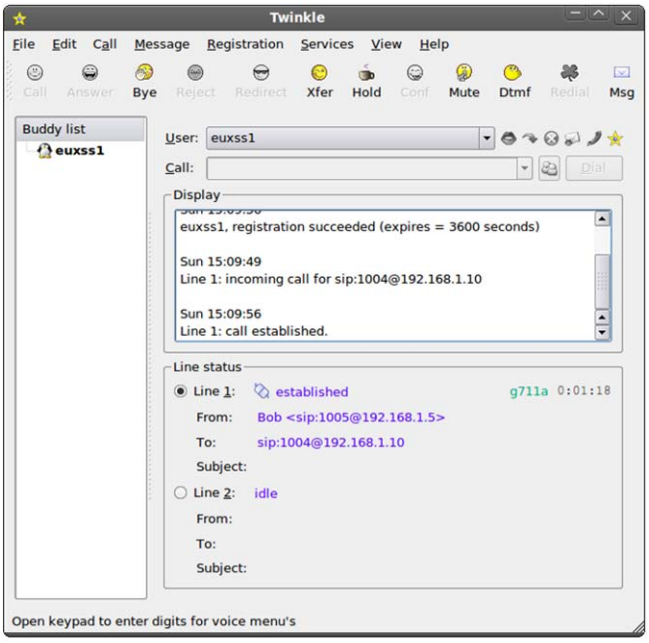
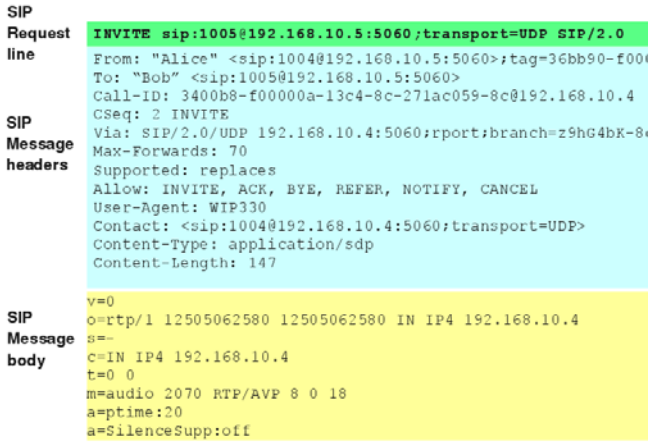



**Figure 9** SIP INVITE message showing SIP headers and body with SDP message content containing RTP metadata (see online version for colours)




The SIP INVITE that goes from the attacker Ivory to the SIP server must be modified to include Frank's and Ivory's IP addresses. To implement this, we use Scapy.[9]

### *5.3 Third part: set up a hostile RTP session*

The third part of the attack is to set up the actual media (voice) stream between the attacker Ivory and Frank. The media stream is carried using RTP.

Both Ivory and Frank must know each other's IP address (or hostname) and must have agreed upon which port numbers to use for the RTP stream before the attack is executed.

To set up a duplex RTP stream between the attackers, we use the VLC media player.[10] VLC supports both reading (playing) and sending (streaming) of RTP.

Frank set up an RTP server, and waits for Ivory to connect. When Ivory has finished her attack, she connects to Frank and starts streaming RTP (voice) to Ivory. Frank then connects to Ivory the same way.

Frank must execute the below command to allow Ivory to connect:

```
cvlc alsa:// --sout \\
'#transcode{acodec=alaw,ab=64,scale=1,\\
channels=1,ar=8000}:rtp{dst=192.168.10.50
,\\
name=Frank,\\
sdp=rtsp://192.168.10.50:16376/frank.sdp}
'
```

Here Frank captures voice input from his microphone, encode it to the sound codec 'alaw' and start streaming the sound using RTP on port 16377. Ivory then connects to Frank, by issuing the command given below.

```
cvlc rtsp://192.168.10.50:16376/frank.sdp
```

The above commands set up a one-way communication channel from Frank to Ivory. To enable Frank to hear Ivory, she needs to set up a streaming on her end as well. The command is identical, except for a replace of the IP address:

```
cvlc alsa:// --sout \\
'#transcode{acodec=alaw,ab=64,scale=1,
\\channels=1,ar=8000}:rtp{dst=192.168.10.4
0,\\
name=Ivory,\\
sdp=rtsp://192.168.10.40:16378/ivory.sdp}'
```

And Frank can listen to Ivory by issuing the command given below.

```
cvlc rtsp://192.168.10.40:16378/ivory.sdp
```

When both Frank and Ivory have set up a two-way RTP stream between each other, they can both send and receive media (voice) from the other and can thus effectively communicate.

## 6 Discussion and conclusions

Our method reveals concrete attacks that indicate where improvements in the protocol are necessary. We discovered that the well-known SIP implementation Asterisk deviates from the SIP specification, and found a severe call hijack attack, where intruders can completely take over a phone call.

Our work provides the designers of VoIP protocols with a set of tools for protocol analysis. The PROSA language and framework can be used to formally specify and analyse protocols and their specific implementations in a rigorous way. The use of traces, and the analysis with PROSA, can help to detect differences between protocol specification and implementation. The behaviour of implementations is therefore analysed and treated as specification for a variant of the employed protocols. Attacks could be designed and tested rapidly compared to the traditional approaches as described in Porter (2006) and Endler and Collier (2006).

## Acknowledgements

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council. The authors would like to thank Wolfgang Leister, Arne-Kristian Groven Jørn Inge Vestgaarden, Lothar Fritsch, Einar W. Høst, Svetlana Boudko and Truls Fretland for comments on earlier drafts of this paper.

## References

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

Diab, W.B., Tohme, S. and Bassil, C. (2008) 'VPN analysis and new perspective for securing voice over VPN networks', *ICNS*, pp.73–78.

Dolev, D. and Yao, A.C-C. (1983) 'On the security of public key protocols', *IEEE Transactions on Information Theory*, Vol. 29, No. 2, pp.198–207.

Endler, D. and Collier, M. (2006) *Hacking Exposed VoIP: Voice over IP Security Secrets and Solutions*, McGraw-Hill Osborne Media.

Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and Stewart, L. (1999) *HTTP Authentication: Basic and Digest Access Authentication*, RFC 2617 (Draft Standard).

Geneiatakis, D., Kambourakis, G., Dagiuklas, T., Lambrinoudakis, C. and Gritzalis, S. (2005) 'SIP security mechanisms: a state-of-the-art review', *Proceedings of the Fifth International Network Conference (INC 2005)*, pp.147–155.

Gupta, P. and Shmatikov, V. (2007) Security Analysis of Voice-over-IP Protocols, in *Computer Security Foundations Symposium*, CSF '07, 20th IEEE, pp.49–63.

Hagalisletto, A.M. (2007) *Automated Support for the Design and Analysis of Security Protocols*, PhD thesis, University of Oslo, Oslo, Norway.

Hagalisletto, A.M. and Strand, L. (2008) 'Formal modeling of authentication in SIP registration', *Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08*, IEEE Computer Society, pp.16–21.

Hagalisletto, A.M., Strand, L., Leister, W. and Groven, A-K. (2009) 'Analysing protocol implementations', in Bao, F., Li, H. and Wang, G. (Eds): *The 5th Information Security Practice and Experience Conference (ISPEC 2009)*, volume LNCS 5451, pp.171–182, Springer Berlin/Heidelberg.

Handley, M., Jacobson, V. and Perkins, C. (2006) *SDP: Session Description Protocol*, RFC 4566 (Proposed Standard).

Keromytis, A.D. (2009) 'Voice over ip: risks, threats and vulnerabilities', *Proceedings of the Cyber Infrastructure Protection (CIP) Conference*, New York.

Kuhn, D.R., Walsh, T.J. and Fries, S. (2005) 'Security consideration for voice over IP systems', *National Institute of Standards and Technology (NIST)*, special publication, pp.800–858,.

Meggelen, J., Smith, J. and Madsen, L. (2005) *Asterisk: The Future of Telephony*, O'Reilly Media.

Park, P. (2008) *Voice over IP Security*, 1st ed., Cisco Press.

Perkins, C. (2003) *RTP: Audio and Video for the Internet*, Addison-Wesley.

Persky, D. (2007) *VoIP Security Vulnerabilities*, Technical report, SANS Institute.

Porter, T. (2006) *Practical VoIP Security*, Syngress.

Ramsdell, B. (2004) *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification*, RFC 3851 (Proposed Standard).

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and Schooler, E. (2002) *SIP: Session Initiation Protocol*, RFC 3261 (Proposed Standard), Updated by RFCs 3265, 3853, 4320, 4916, 5393.

Salsano, S., Veltri, L. and Papalilo, D. (2002) 'SIP security issues: the SIP authentication procedure and its processing load', *Network, IEEE*, Vol. 16, pp.38–44.

Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V. (2003) *RTP: A Transport Protocol for Real-Time Applications*, RFC 3550 (Standard).

Sinnreich, H. and Johnston, A.B. (2006) *Internet Communications Using SIP: Delivering VoIP and Multimedia Services with Session Initiation Protocol*, 2nd ed., John Wiley & Sons, Inc., New York, NY, USA.

Sisalem, D., Floroiu, J., Kuthan, J., Abend, U. and Schulzrinne, H. (2009) *SIP Security*, Wiley Blackwell.

VoIPSA (2005) *VoIP security and privacy threat taxonomy. Public Realease 1.0*. Available online at: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf

Xin, J. (2007) *Security Issues and Countermeasure for VoIP*, Technical report, SANS Institute.

Zhang, R., Wang, X., Yang, X. and Jiang, X. (2007) 'Billing attacks on SIP-based VoIP systems, *First USENIX Workshop on Offensive Technologies (WOOT '07)*, USENIX.

## Notes

1 Automated Validation of Internet Security Protocols and Applications (AVISPA) project: http://avispa-project.org/library/sip.html

2 IETF Session Initiation Protocol Charter: http://www.ietf.org/html.charters/sip-charter.html

3 Asterisk homepage: http://www.asterisk.org/

4 Wireshark homepage: http://www.wireshark.org

5 The sub-protocol deviates from SIP in the sense that the reference to call-ID $N_D^{\text{callid}}$ is captured in message ($TB_1$)

6 SIPp homepage: http://sipp.sourceforge.net/

7 SipSak – the 'SIP swiss army knife'. Homepage: http://sipsak.org/

8 sip-kill can be downloaded at: http://skora.net/images/voip-security/sip-kill

9 Scapy network manipulation program: http://www.secdev.org/projects/scapy/

10 VLC media player homepage: http://www.videolan.org/vlc/

# Paper D:
# A Survey of SIP Peering

Lars Strand, Wolfgang Leister

In proceedings of
NATO ASI - Architects of Secure Networks (ASIGE) 2010

# A Survey of SIP Peering

Lars Strand
lars.strand@nr.no

*Norwegian Computing Center, P.O. Box 114 Blindern, NO-0314 Oslo, Norway*

Wolfgang Leister
wolfgang.leister@nr.no

*Norwegian Computing Center, P.O. Box 114 Blindern, NO-0314 Oslo, Norway*

**Abstract**

When placing a call from one SIP Service Provider to another, the call is traditionally routed over PSTN, instead of IP. This can lead to higher costs, reduced quality, and lack of functionality. These issues can be addressed by setting up a SIP peer between providers. A SIP peer is a layer 5 interconnection between two SIP Service Providers for the purpose of routing real-time and quasi-real-time call signalling between their customers. We survey the SIP peering architecture and show security implications.

*Keywords:* VoIP, SIP, security, peering, PSTN

## 1. Introduction

Conventional telephony, also called Plain Old Telephone Service (POTS), still accounts for the majority of telephony calls. POTS uses a circuit switched network, where each call establishes a dedicated circuit between two nodes with fixed bandwidth before communicating. These circuit switching networks form what is called the Public Switched Telephone Network (PSTN).

In contrast to PSTN, Voice over IP (VoIP) uses packet switching for sending data. A packet switching network divides the traffic into a sequence of packets that are sent over a shared network. VoIP is considered the emerging technology that will eventually take over from PSTN. In the 1970s, experiments with transmitting voice over IP networks was conducted [1]. However, it was not until the mid 1990s that the H.323 protocol [2] and the Session Initiation Protocol (SIP) [3] were standardised and widely deployed. Today, SIP is the preferred signalling protocol in the VoIP industry.

SIP has evolved into a mature and stable standard, and is rapidly being adopted for VoIP applications by the industry. However, in contrast to the original plans of the designers of SIP, the VoIP providers do not use the so-called open *email model* for communication in-between them due to security concerns. As a result, providers have started to set up ad hoc SIP peering. However, such individual solutions are not scalable. Therefore, we review initiatives to organise SIP peering.

The rest of the paper is organised as follows: We give an introduction to SIP and the *email model* in Section 2. We explain SIP peering, its architecture and logical functions in Section 3. In Section 4, we discuss security concerns and investigate whether SIP peering solves any of these. We then summarise and try to identify which areas require more work in Section 5 and give an outlook in Section 6.

## 2. SIP and the email model

SIP is an application layer protocol that handles multimedia sessions. It is used to negotiate, establish, change and tear-down the *context* of a multimedia flow; other protocols, such as the Real-time Transport Protocol (RTP), are used for the media (voice) transport [4].

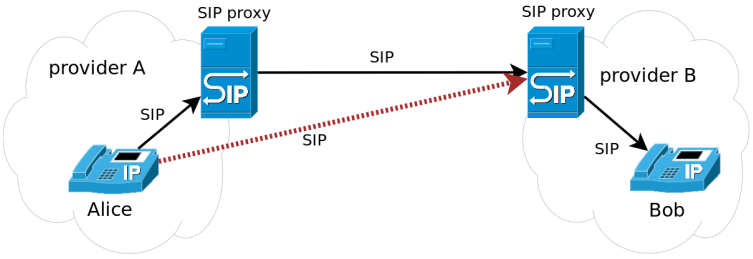



Figure 1: When Alice calls Bob, the message path form a trapezoid going from Alice's UA through provider A's SIP proxy which forwards the call to provider B's proxy and then to Bob.

Illustrated in a simple example, SIP works as follows: When Alice calls Bob, as depicted in Figure 1, a SIP `INVITE` message is sent from Alice's User Agent (UA), via one or more SIP proxies, to Bob's UA. Before allowing Alice to send an `INVITE` request to Bob, Alice's SIP proxy may request Alice to authenticate and do rudimentary SIP header checks before forwarding to Bob's SIP proxy. The media (RTP) might take a direct path between Alice and Bob, thus forming a "SIP trapezoid" for the message path. However, Alice may send an SIP `INVITE` directly to Bob's SIP proxy. Provider B's SIP proxy cannot distinguish whether a request arrived from a SIP proxy or a UA directly. This is depicted in red in Figure 1.

To address Bob, Alice uses Bob's SIP address (URI). A SIP URI is structured in the same way as email addresses with *username@domain*. While a call is routed to the destination, only the domain part is relevant. The username is only interpreted by the SIP server in the receiver's domain [5]. The global public DNS is used to map the domain part to one or more SIP ingress servers that handle incoming SIP requests. We denote this mode of routing and addressing SIP as the *email model*.

SIP has not seen global reachability as outlined in the SIP standard [6]. There are three main reasons why the *email model* for SIP has failed. *1*) The telephony providers have traditionally collected termination fees between communication partners (other providers). If everyone directly is able to connect to everyone, no business relationships between providers are necessary. Therefore, the carriers have no economic incentive to switch to a global reachable SIP addressing scheme. *2*) Operators of public telephony services need to comply to a range of legal regulatory requirements. These requirements are applicable for the PSTN with clear boundaries between telephony operators and telephony users. *3*) There are a range of security concerns to which no simple solution exists:

(a) Unwanted calls, also known as "Spam over Internet Telephony" (SPIT), are a threat to the VoIP infrastructure. Since there are currently few open SIP servers, SPIT has not yet grown to be a widespread problem compared to *email spam*. Note that problems related to spam emerged after the number of open SMTP[1] servers increased. SPIT is harder to prevent than email spam, since VoIP calls are interactive — the content or the intentions of a call are not identifiable in advance, before the receiver picks up the phone. Therefore, filters for SPIT cannot be applied as easily as spam filters for email. Also social factors are important, since one usually picks up the phone when it rings, instead of being able to choose when and how often to check email. Providers fear that SPIT could become common if they open up their SIP services to the Internet [7].

(b) Assuring the *identity* of the caller. Signalling in PSTN has traditionally been trusted between carriers, and by end users (caller-id). This trust is not applicable to open SIP servers, since the INVITE message can come from any user on the Internet and the caller-id can easily be spoofed. Different authentication mechanisms for SIP exist like S/MIME [8], the "Asserted Identity" extension [9] and the "Identity" header extension [10]. These will be discussed in Section 4.2. Unfortunately, these identity mechanisms have not been deployed nor supported to a great extent worldwide [11].

[1] Simple Mail Transfer Protocol (SMTP) is the Internet standard for email.

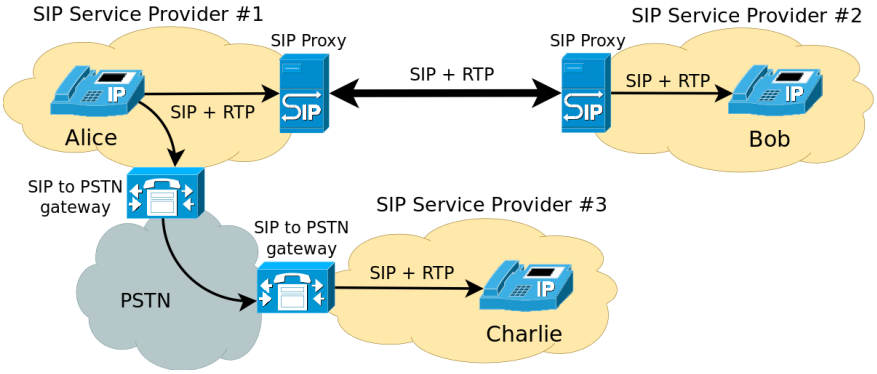



Figure 2: The two scenarios: When Alice calls Bob, SIP peering is used. Without SIP peering between two SSPs, calls are routed through the PSTN, as illustrated where Alice calls Charlie.

(c) *Denial of Service* attacks are threats to availability. The *email model* is particularly vulnerable to DoS attacks, since SIP servers need to accept request from anyone on the Internet. This makes it hard to guarantee a stringent Quality of Service (QoS) agreement to customers and other SIP providers.

As a result of these concerns, SIP Service Providers (SSP) have not deployed open SIP servers (the *email model*). Therefore, when a call is destined for outside the SSPs domain, it is routed over PSTN to the provider network of the receiver, as outlined in Figure 2. This makes it necessary to transcode both content and signalling of the call. Routing calls over PSTN has disadvantages:

1. Managing VoIP-to-PSTN gateways adds administrative overhead, extra hardware costs, and extra resources to configure, deploy and manage the gateways on a daily basis.

2. Sending calls over PSTN is more expensive for the VoIP provider, since the provider must pay termination fees to the PSTN provider.

3. A call traversing the VoIP-to-PSTN gateway needs to be transcoded from VoIP before going into the PSTN. If the receiving end, the callee, also uses VoIP, the call must be transcoded from PSTN to VoIP. This is illustrated in Figure 2 in the scenario where Alice calls Charlie. Even if both VoIP and PSTN use the G.711 voice codec [12], delays are possible, and information can get lost during transcoding. The use of "wideband" speech codecs like G.722 [13] does not give any advantages. These wideband codecs provide superior voice quality and have the potential to become a differentiator for VoIP [14].

4. PSTN does not carry services that are offered by SIP, such as video, IM and presence. Therefore, these services cannot be offered to the customers as a service between providers.

To overcome these disadvantages without deploying open SIP servers, the SSPs intend to set up SIP peering between each other. This eliminates the need for transcoding to/from PSTN. Unfortunately, there is no standard nor suitable best practices for how to set up these peers between the SSPs.

There have been some industry attempts to create SIP peering recommendations [15], but no defining standards. The Internet Engineering Task Force (IETF) has acknowledged this, and created the SPEERMINT[2] Working Group (WG) with the goal to identify architecture requirements, discuss security considerations, and define best practises for SIP peering.

[2]The charter of the Session PEERing for Multimedia INTerconnect (SPEERMINT) WG is available at http://www.ietf.org/dyn/wg/charter/speermint-charter.html.

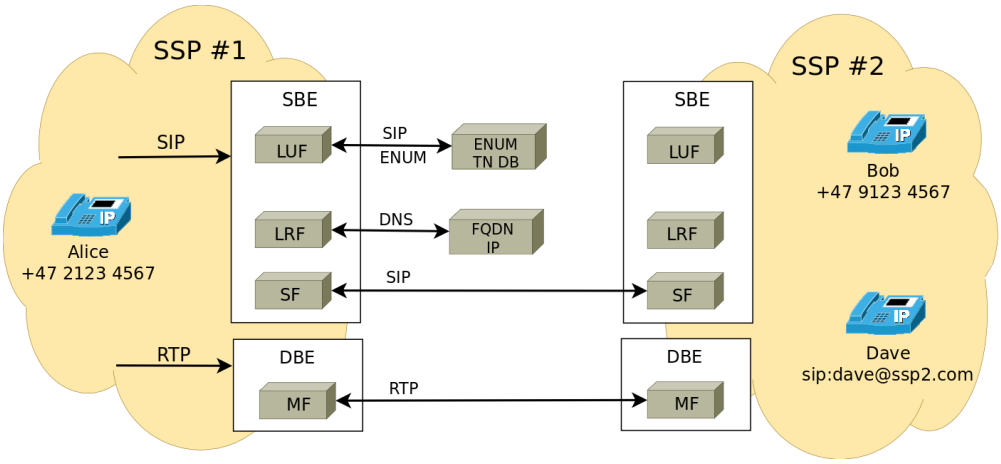



Figure 3: Logical functions defined for the SIP peering architecture.

## 3. SIP peering

The term "Voice over IP peering" or just "Voice peering" refers to a wide range of practices pertaining to the interconnections of VoIP service providers. "SIP peering" refers to Voice peering, taking into account how SIP can be used in a secure and standardised manner for interconnections between VoIP service providers.

An important distinction must be made between the traditional peering on the IP layer (Layer 3 in the OSI protocol stack [16]), and peering on the application layer (OSI layers 5–7). SIP is an application layer protocol designed to run independently of the transport layer (TCP/UDP/SCTP). Therefore, SIP peering interconnection operates on the application layer, and assumes that lower layer functionality, like IP routing, are handled by other network processes.

Rather than introducing new SIP extensions, SIP peering uses existing protocol standards (SIP, RTP, ENUM, DNS) as building blocks, to create a set of best practices and operational procedures. A number of logical functions have been defined as part of the SIP peering architecture, which we briefly describe.

### *3.1. The SPEERMINT Architecture*

The SPEERMINT architecture, shown in Figure 3, consists of two logical functions, the Signalling path Border Element (SBE) and the Data path Border Element (DBE). The SBE provides the signalling functionality to and from peering partners, i.e., the SSPs. The DBE provides media-related functions, such as firewall-traversal support and transcoding.

The *SBE* consists of a number of logical functions, but does not redefine how SIP uses input and output variables to create Session Establishment Data (SED). *SED* denotes a set of parameters that the outgoing SBE need to complete a call. These parameters may include SIP addresses (SIP URIs), the address of the ingress SIP proxy including the fully qualified domain name (FQDN), port and transportation method (TCP/UDP/TLS), and security parameters (TLS certificate data).

For the SBE to acquire the parameters needed to complete a call, it relies on two functions: the *Look-Up Function* (LUF) and the *Location Routing Function* (LRF). The LUF is used if the destination is a telephone number, and we need to translate the number to a routable Internet SIP address. This is done by using *ENUM* [17, 18] which translates public telephone numbers [19] to SIP addresses (SIP AOR) by DNS.

After the LUF has provided a SIP address, the SBE needs to determine the target domain to which the request should be routed. This lookup function is performed by the LRF, and consists of an ordinary DNS lookup of the target domain. The result of the DNS lookup will provide to the *Signalling Function* (SF) the necessary SED parameters needed to address and find the target domain (ingress point). The SF, usually a SIP proxy, can then perform routing of the request to the correct destination.

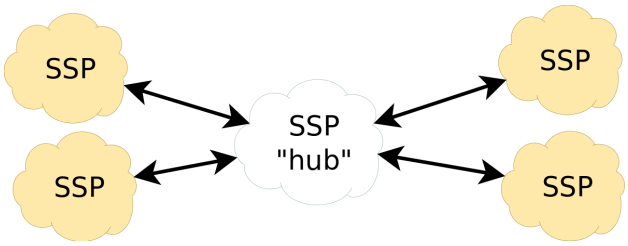



Figure 4: A "hub-based federation" where SSPs form a star peering topology.

After the signalling phase (SIP handshake) has been completed, the actual content (voice/video) session is established using the DBE. The DBE consists of one logical function, the *Media Function* (MF), which is responsible for the actual delivery of multimedia communication between users and providers.

### *3.2. Peering arrangements*

SIP peering is classified into *static* and *on-demand*, and each of these into *direct* and *indirect* peering. The adoption of on-demand peering has seen low industrial penetration [20].

For *static direct peering* two SSPs have agreed on a peering relationship before the exchange of phone calls, and the sending SSP reaches the receiving SSP in one hop. For *static indirect peering* signalling and media path must be established via one or more SSPs before reaching the destination SSP. A group of SSPs that peer with each other is called a *federation*.

There is an administrative overhead to exchange and keep the various lookup-function data up to date between the SSPs. Therefore, if each SSP would peer with every other SSP, the number of peering agreements would be infeasibly high. An emerging solution consists of using one single SSP as a *hub* for the other SSPs. This hub SSP provides an assisted LUF/LRF for the other SSPs as well as a central peering point, thus forming a star topology as shown in Figure 4. This hub-based federation model is currently being adopted by the industry, delegating LUF/LRF and administrative tasks, like abuse handling or technical requirements for the peers, to the central hub.

## 4. Security considerations

The SIP core specification states that *"SIP is not an easy protocol to secure"* [3, page 232]. This is because SIP has been extended extensively with additional functionality for different targets [4], and the number of components involved in a typical SIP based VoIP setup ranges from user devices (UAs) to SIP servers (proxies and registration), firewalls, gateways, and supporting servers like LUF/LRF lookup. With this diversification, different security requirements and mechanisms apply.

A clear and concise VoIP threat taxonomy is given by VOIPSA [21]. VoIP threats are discussed by Keromytis [22] and the wide range of VoIP security threats have stimulated research in this area [23]. Several security mechanisms for countermeasures have been proposed, but no single security mechanism is suited to address all these security threats concerning SIP [24, 25, 26]. While there are many threats for VoIP, we consider *Spam over Internet Telephony* (SPIT), weaknesses of *authentication*, and *Denial of Service* (DoS) attacks as the most relevant from the perspective of SIP peering.

### *4.1. SPIT, SPIM and SPPP*

As spam is a problem for the email infrastructure, it is expected that the VoIP counterparts of spam will emerge for the VoIP infrastructure. The amount of email traffic that is classified as spam compared to ordinary email is estimated to be around 90% according to the Spamhaus project[3]. For an end-user, it

[3]The Spamhaus Project: http://www.spamhaus.org

usually takes a couple of seconds to classify an email as spam, and delete it. However, the costs on a global scale can be significant, when adding up the amount of time each user uses on handling spam. Additionally, email providers must invest in anti-spam measures and develop routines to handle spam.

For VoIP, the spam problem is related to vulnerabilities of SIP, and to the fact that VoIP, in contrast to email, is used interactively. Spam over SIP can be classified in three groups [27]: *1*) Spam over Internet telephony (SPIT) is defined as unwanted calls. Here, the *spitter* either plays back a pre-recorded message, or does telemarketing. *2*) Spam over instant message (SPIM) is defined as sending unsolicited instant messages. SPIM has similar properties as email spam. *3*) Spam over presence protocol (SPPP) is defined as sending bulks of presence requests (SIP `SUBSCRIBE`). If a user accepts such a request, the spammer is usually is put on the user's "buddy list", and is consequently allowed to send SPIM or SPIT.

Of these, SPIT is considered the most attractive and effective for spammers [24]. SPIT differs from email spam in two distinct ways: First, there are social norms and behaviour. A user can choose when to check email, but when the phone rings, the call usually is answered. It will take us a couple of seconds to classify the call as SPIT, but then we have already been disrupted. If this happens often it is considered very disruptive. Second, preventing SPIT is harder than email spam, since the content (media) is not available until after the user picks up the phone. This differs from email spam, where the content can be filtered and classified in advance.

There is a low amount of SPIT today, since the number of open SIP servers is limited. However, once the number of VoIP users increases above a certain limit, this could change [6]. Several anti-SPIT solutions have been proposed. The SPIDER research project provided an extensive analysis of SPIT, and presents different anti-SPIT measures[4]. The RFC 5039 [27] analyses spam over SIP, and discusses whether anti-spam solutions for email are applicable for SPIT.

Both the SPIDER project and RFC 5039 conclude that there are no simple and clear solutions to avoid SPIT completely. The SPIT problem would have been less significant if these threats had been taken into account already during the specification phase of SIP. Improved SIP authentication methods, and assuring a caller's identity across SSPs, are necessary to prevent SPIT. However, it might be problematic to improve authentication methods, after the industry adoption of SIP.

### *4.2. Authentication*

A definition of authentication is *"the binding of one identity to a subject"* [28]. The user has several expectations that are related to authentication. For example, the caller will expect a call to be established with the intended callee. The callee will expect to talk to the person that the caller claims to be. There are several authentication mechanisms in SIP, some are mandatory and others are SIP extensions that have not seen widespread adaption.

Within one administrative domain (usually within the domain of an SSP), the SIP proxies can demand the clients (UAs) to authenticate themselves before use. But there are no obligations do to so. Some providers only authenticate calls that are destined to PSTN, since the providers pay interconnection fees when routing through PSTN [24]. A UA can also avoid authentication by contacting the callee's SIP proxy directly, as shown in Figure 1, where Alice contacts provider B's SIP proxy without going through provider A's SIP proxy. Both the caller and the callee should be mutually authenticated.

SIP provides several authentication mechanisms, but only the Digest Access Authentication method is mandatory:

- The Digest Access Authentication (DAA) [3, section 22] is the most common authentication method, since its support is mandatory. Unfortunately, DAA provides only weak authentication, and is vulnerable to a series of attacks, including off-line dictionary attacks and registration attacks [11].

- A more secure authentication method is achieved by encapsulating the SIP message into a Secure MIME (S/MIME) format [8]. The encapsulated SIP message can be signed or encrypted, or both. The S/MIME content is carried in the payload of a new outer SIP message. Since S/MIME relies on

[4]Spam over Internet telephony detection service (SPIDER): http://www.projectspider.org/

certificates, each UA must obtain and install an individual certificate from a Certification Authority (CA) before use. Since no single CA is trusted by all UAs across the SSPs, a UA must have support for multiple root certificates. This, and other certificate handling issues like revoking and renewing, complicates the usage of certificates. The industry support for S/MIME has been limited so far.

- SSPs require that the caller's identity can be assured, to comply to regulatory requirements such as the ability to trace back a call. Since the "`From`" SIP header field easily can be manipulated, there is a need to assert the caller's identity between SSPs. This can be achieved by including an asserted-identity SIP header [9]. After a UA has been authenticated against the local proxy using DAA, the proxy adds a "`P-Asserted-Identity`" SIP header. This identity header is sent in clear, and is not protected by cryptography in any way. Since an attacker could use reply- and modification attacks, or remove the header altogether, this method is limited to trusted domains where SIP proxies communicate over trusted links.

- Another approach introduces two new SIP headers and a SIP "authentication service" [10]. After the UA has been authenticated to its SIP proxy, the originating SIP proxy signs a hash over some particular SIP headers, and includes the signature as an "`Identity`" header. Also included is an "`Identity-Info`" header, which contain an URI for the caller's certificate. The callee computes the same hash, and compares the result from the originating proxy. This authenticates the caller, but not the callee. An attacker might remove these headers in transit without an implication for the callee.

Within one SSP, the use of DAA in combination with TLS can authenticate, with confidence, the identity of the UA. But within a federation of SIP peers, no direct mutual authentication exists between UAs of different SSPs. We can only achieve transitive trust between SSPs, and must hope that there are no weak links in this chain.

### *4.3. Denial of Service*

A Denial of Service (DoS) attack affects the availability of a service. For VoIP, this means that legitimate voice communication could be prevented from an attacked service. A *Distributed DoS* (DDoS) attack is when several nodes are involved in the attack, and are one of the most common network attacks on the Internet. A DDoS attack is rather simple to achieve, and effective, while it is hard to protect a service against it. There are three categories of DoS attacks: *1*) The most common DoS attack involves flooding the victims node/service with network traffic, thereby exhausting its resources such as memory, CPU, bandwidth, or otherwise. *2*) A misuse attack intentionally misuses or malforms SIP messages to interrupt or terminate a VoIP call or service resulting in a denied availability for legitimate users. An example is SIP call hijacking [29]. *3*) An indirect attack, is an attack on supporting infrastructure services that VoIP rely on. For example, an attack on the LUF/LRF services would reduce the availability of the VoIP service, since SIP address resolution would not be available.

All components in a VoIP installation are vulnerable to DoS attacks. Attacking an SBE has thus more far-reaching consequences than an attack on a single UA. With the adoption of "federated hub peering", a successful DoS attack against the hub SSP will seriously degrade the availability of inter-SSP VoIP calls, since it can be considered as a single point of failure.

## 5. Challenges for SIP Peering

The rationale behind SIP peering is to achieve global and universal VoIP connectivity, without using the *email model* in inter-SSP traffic. The peering architecture needs to build a global federation that handles trust. In addition to trust between the SSPs, the trust relationship between the SSPs and their customers is important.

While trust and security concerns can be solved on a small scale, the global VoIP infrastructure needs to be scalable. It is a challenge to scale SIP peering to a global level without exposing LUF and LRF data to the world. It is also important to be able to route VoIP calls effectively through federations of federations.

In the case of technical problems with a VoIP service, it is necessary to access diagnostics, and be able to trace calls, in order to figure out which network is causing problems. The challenge of receiving suitable diagnostics is not specific to SIP peering, but SIP peering amplifies this need.

Security considerations related to SIP are also applicable in a peering relationship. But to demand that the SSPs use and enforce network security mechanisms like TLS/IPSec and deploy secure and scalable network design everywhere is unrealistic. Focus should be on enhancing the existing security mechanisms already defined in SIP.

Also, more detailed work is needed on authentication. The only mandatory authentication method is DAA, which is vulnerable to several attacks. Improving this authentication method would counter the most rudimentary attacks. When using DAA, a UA is authenticated locally within a SSP, but this authentication is not assured across multiple SSPs. SPEERMINT mandates a trust relationship between peering SIP proxies, but the available authentication methods do no provide mutual authentication between UAs. Other approaches should be investigated, like reputation based systems or multi-factor authentication systems.

## 6. Outlook

While SIP is an established standard within VoIP, we have seen that the *email model* is not a good solution for signalling between SSPs due to security concerns. SIP peering has emerged as one solution. However, SIP peering will only provide a solution if the SSPs adhere to the recommendations of the SPEERMINT WG, instead of developing their own non-standard or ad-hoc peering solutions. Besides security, issues of scalability and service quality need to be addressed.

Interconnecting all SSPs in one global SIP peering mesh (federation) is unfeasible. Even one global federation of federation is impractical and exposed to huge challenges, such as SIP routing. Also, demanding support for roaming between SSPs, adds to the complexity of any solution.

## Acknowledgement

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054). The authors want to thank Josef Noll who inspired us for this survey. We also thank Trenton Schulz for discussions while preparing this paper and Arne-Kristian Groven for comments on the earlier drafts of this paper.

## References

[1] D. Cohen, Specifications for the Network Voice Protocol (NVP), RFC 741, 1977.
[2] International Telecommunication Union, H.323: Packet based multimedia systems, ITU-T Recommendation H.323, 2006.
[3] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, SIP: Session Initiation Protocol, RFC 3261 (Proposed Standard), 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630.
[4] H. Sinnreich, A. 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, 2006.
[5] J. Rosenberg, H. Schulzrinne, Session Initiation Protocol (SIP): Locating SIP Servers, RFC 3263 (Proposed Standard), 2002.
[6] O. Lendl, VoIP Peering: Background and Assumptions, Technical Report, IETF, 2008.
[7] J. Rosenberg, SIP Peering: What It Does and Doesn't Mean, SIP Magazine Speaking SIP (2006).
[8] J. Peterson, S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP), RFC 3853 (Proposed Standard), 2004.
[9] C. Jennings, J. Peterson, M. Watson, Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks, RFC 3325 (Informational), 2002.
[10] J. Peterson, C. Jennings, Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP), RFC 4474 (Proposed Standard), 2006.
[11] A. M. Hagalisletto, L. Strand, Formal modeling of authentication in SIP registration, in: Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08, IEEE Computer Society, 2008, pp. 16–21.
[12] International Telecommunication Union, Pulse Code Modulation PCM of Voice Frequencies, ITU-T Recommendation G.711, 1993.
[13] International Telecommunication Union, 7 kHz Audio-Coding within 64 kbits/s, ITU-T Recommendation G.722, 1993.

[14] M. Miller, The VoIP Peering Puzzle, Enterprise VoIP Planet (2006).
[15] C. Sibley, C. Gatch, (eds), SIPconnect 1.0 Technical Recommendation, Technical Report, SIP Forum, 2008.
[16] H. Zimmermann, OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection, IEEE Transactions on Communications 28 (1980) 425–432.
[17] P. Faltstrom, M. Mealling, The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM), RFC 3761 (Proposed Standard), 2004.
[18] J. Peterson, enumservice registration for Session Initiation Protocol (SIP) Addresses-of-Record, RFC 3764 (Proposed Standard), 2004.
[19] International Telecommunication Union, The International Public Telecommunication Numbering Plan, ITU-T Recommendation E.164, 2005.
[20] A. Uzelac, Y. L. Lee, IETF DRAFT: VoIP SIP Peering Use Cases, Technical Report, IETF, 2010.
[21] VoIPSA, VoIP security and privacy threat taxonomy, Public Realease 1.0, 2005. http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf.
[22] A. D. Keromytis, Voice over IP: Risks, Threats and Vulnerabilities, in: Proceedings of the Cyber Infrastructure Protection (CIP) Conference, New York.
[23] A. D. Keromytis, A Survey of Voice Over IP Security Research, in: Proceeding of the 5th International Conference on Information Systems Security (ICISS), pp. 1 – 17.
[24] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, H. Schulzrinne, SIP Security, WileyBlackwell, 2009.
[25] P. Park, Voice over IP Security, Cisco Press, 2008.
[26] S. Niccolini, H. Scholz, E. Chen, J. Seedorf, IETF DRAFTv2: SPEERMINT Security Threats and Suggested Countermeasures, Technical Report, IETF, 2010.
[27] J. Rosenberg, C. Jennings, The Session Initiation Protocol (SIP) and Spam, RFC 5039 (Informational), 2008.
[28] S. Bishop, M. Fairbairn, M. Norrish, P. Sewell, M. Smith, K. Wansbrough, Rigorous specification and conformance testing techniques for network protocols, as applied to TCP, UDP, and sockets, SIGCOMM Comput. Commun. Rev. 35 (2005) 265–276.
[29] A. M. Hagalisletto, L. Strand, W. Leister, A.-K. Groven, Analysing protocol implementations, in: F. Bao, H. Li, G. Wang (Eds.), The 5th Information Security Practice and Experience Conference (ISPEC 2009), volume LNCS 5451, Springer Berlin / Heidelberg, 2009, pp. 171–182.

# Paper E:
# Improving SIP authentication

Lars Strand, Wolfgang Leister

In proceedings of the
$10^{th}$ International Conference on Networking (ICN) 2011
ISBN: 978-1-61208-002-4

# Improving SIP authentication

Lars Strand

Norwegian Computing Center / University of Oslo
Oslo, Norway
Email: lars.strand@nr.no

Wolfgang Leister

Norwegian Computing Center
Oslo, Norway
Email: wolfgang.leister@nr.no

***Abstract*—The digest access authentication method used in the voice over IP signaling protocol, SIP, is weak. This authentication method is the only method with mandatory support and widespread adoption in the industry. At the same time, this authentication method is vulnerable to a serious real-world attack. This poses a threat to VoIP industry installations and solutions. In this paper, we propose a solution that counters attacks on this wide-spread authentication method.**

***Index Terms*—SIP, authentication, Digest Access Authentication, security attack.**

## I. Introduction

The most common protocol pair used for sending Voice over IP (VoIP) is the Session Initiation Protocol (SIP) [1] and Real-time Transport Protocol (RTP) [2]. RTP transfers the media content, while SIP handles the signaling, i.e., set up, modification and termination of sessions between two or more participants. VoIP is the emerging technology that will eventually take over from the traditional Public Switched Telephone Network (PSTN) [3] due to VoIP's improved flexibility and functionality, such as improved sound quality ("HD sound") using wideband codecs like G.722 [4], instant messaging (IM), presence, mobility support, and secure calls. VoIP reduces maintenance and administration costs since it brings convergence to voice, video and data traffic over the IP infrastructure.

SIP is an application layer protocol developed by the IETF. Its core functionality is specified in RFC3261 [1]. Additional functionality is specified in additional RFCs [5]. SIP sessions range from ordinary calls between two participants to advanced conference sessions between multiple participants communicating over video, voice, and IM.

However, SIP and RTP-based VoIP installations are rather difficult to secure [6]. VoIP inherits many security threats and Quality of Service (QoS) properties from the Internet, in addition to threats that come from the VoIP-specific technologies [7]. A clear and concise VoIP threat taxonomy is given by VOIPSA [8]. There are many obstacles in securing SIP, due to its use of intermediaries and the fact that functionality was the primary focus for the SIP designers, not security [1, page 232].

SIP supports several security services, and the RFC recommends their use. These security services can provide protection for authentication, confidentiality, and more. Yet, only one such security service is mandatory: the SIP Digest Access Authentication (DAA) method [1, page 193]. In our experience the other security services are neither implemented nor used. The only security service used is the mandatory authentication method.

DAA is primarily based on the HTTP Digest Access Authentication [9], and is considered to be weak and vulnerable to serious real-world attacks [10].

The main contribution of this paper is to present and analyze the seriousness of a vulnerability we presented in our earlier work – the registration attack [10]. We propose a solution to secure DAA that will counter this vulnerability.

The rest of the paper is organized as follows: We show our approach in Section II. We explain SIP authentication in Section III, and show the registration attack previously discovered in Section IV. In Section V, we show how to improve the authentication method to counter this attack. Related work is given in Section VI, before concluding in Section VII.

## II. Method and case study

In Norway, both private companies and public authorities are migrating from PSTN to VoIP [11]. Our case study is taken from three companies in Norway; one medium sized company with 150 employees, and two larger companies with 3000 and 4700 employees. We have gathered several of these VoIP configurations and setups, and replicated the installations in our test lab [12]. In these companies, most of the employees have their own VoIP phone, called a User Agent (UA). All VoIP servers run the Linux operating system with the open source telephony platform *Asterisk* [13]. We found in these configurations that the digest authentication is the only authentication method for the UAs.

Our analysis follows the workflow shown in Fig. 1. In the following paragraphs, the numbers in parentheses refer to the numbers in Fig. 1.

In order to gain knowledge of the SIP protocol we use the specification documents (1), here the SIP standard. Then, we analyze VoIP network traffic going through the test lab (5). We have implemented two VoIP setups based on configurations from our industry partners ((2) and (3)). The network traffic is intercepted and saved to file using the network tool *tcpdump* (4). The network traffic is then analyzed off-line using the packet analyzer, *Wireshark* (5). An example of such an analysis is shown in Fig. 2.

 ISBN:978-1-61208-002-4 

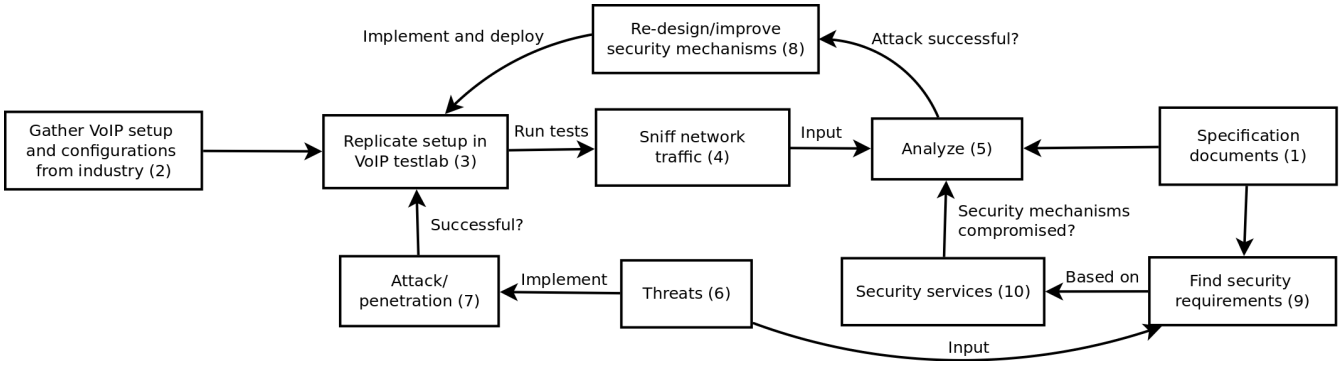



Fig. 1: Workflow for analysis of the SIP authentication method.

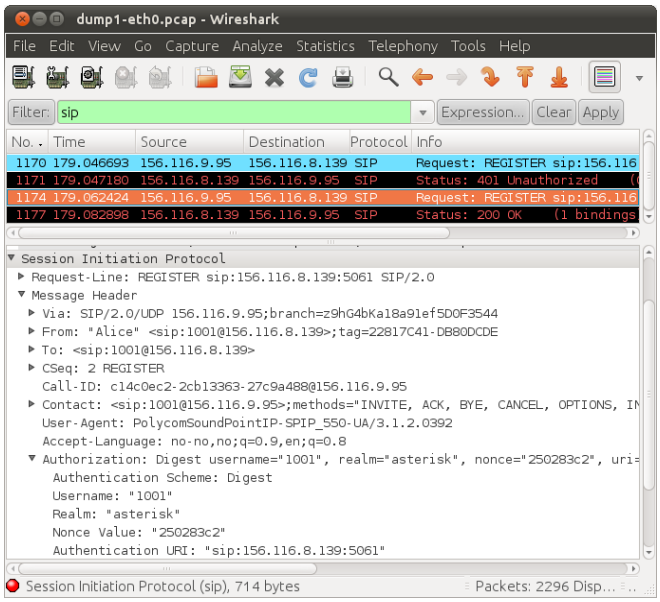



Fig. 2: Network analysis using the network tool Wireshark.

As an additional input we consider threats deducted from formal analysis of the protocol, such as a SIP attack analyzed by Hagalisletto and Strand [10], using the protocol analyzer PROSA (6). We explain the attack in more detail in Section IV, and implement and execute the attack using the network tool *NetSED* (7) as shown in Fig. 6. Based on the security requirements (9) obtained from the SIP specification, we then checked if the authentication method (10) was compromised by the real-world attack. After careful analysis of the SIP headers we found that the SIP registration attack could be countered by a modification of the SIP authentication method (8).

## III. Authentication in SIP

Authentication is the assurance that a communicating entity is the one that it claims to be [14]. Authentication consists of two basic steps: *a*) *Identification*, where an entity/client presents a value to the authentication system, and *b*) *Verification* where this value is validated against the authentication system [15]. When people that know each other are dialing or answering a phone call, they can often authenticate the other by just recognizing the other person's voice. However, when using new communications channels, such as instant messaging (IM), video, screencast and presence, determining the authenticity of the communicating partner is more difficult than for a voice call. To have established the identity of the caller is also important when, for instance, a physician need to communicate with a patient and discuss sensitive health information. For instance, someone else could masquerade as the patient and illegally obtain sensitive health information on the patient.

The SIP Digest Access Authentication (DAA) is currently the most common authentication scheme for SIP. Other authentication schemes have emerged, but DAA is the only mandatory authentication scheme [1, Section 22]. DAA uses a challenge-response pattern, and relies on a shared secret between client and server.

SIP is heavily influenced by the HTTP request-response model, where each transaction consists of a request that requires a particular response. The SIP messages are also similar in syntax and semantics to both HTTP and SMTP [16]. A SIP message consists of headers and a body. The SIP header fields are textual, always in the format `<header_name>: <header_value>`. The header value can contain one or more parameters. We show an example SIP header message in Fig. 4.

Any SIP request can be challenged for authentication. We show an example SIP DAA handshake in Fig. 3, and refer to the protocol clauses with a number in parentheses. The initial SIP `REGISTER` message (1) from Alice is not authorized and must be authenticated. The SIP server responds with a `401 Unauthorized` status message (3) which contains a `WWW-Authenticate` header with details of the challenge, including a *nonce* value. The client computes the required SIP digest that is embedded in (4) as an `Authorization` header. The SIP server, upon receiving the `Authorization` header, must perform the same digest operation, and compare the result. If the results are identical, the client is authenticated,

 ISBN:978-1-61208-002-4 

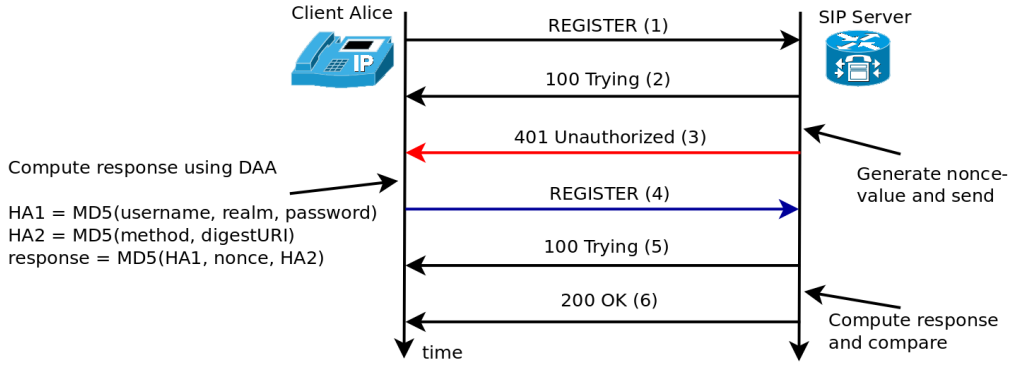



Fig. 3: The SIP Digest Access Authentication method during a SIP `REGISTER` transaction.

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    156.116.9.95;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@156.116.9.95:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response="
    ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",algorith
    m=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Fig. 4: The only attributes included in the digest response (blue) are depicted in green.

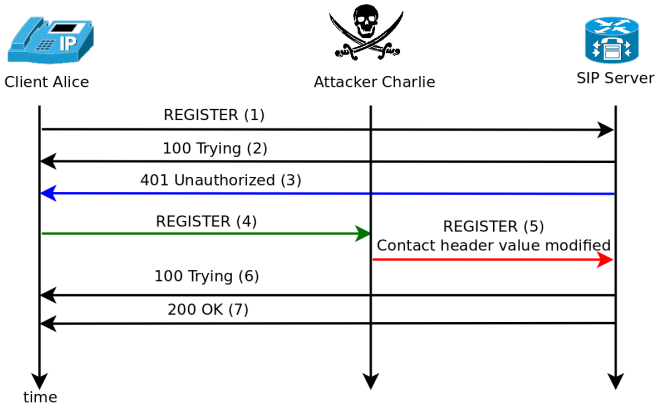



Fig. 5: The attacker Charlie can modify the `Contact` header value, and thereby have all Alice's calls redirected to him.

and a `200 OK` message (6) is sent.

The SIP DAA is almost identical to the HTTP digest access authentication [9]. As we will show later, too few attributes are included in the digest computation, thus leaving some values unprotected. Formally, the DAA is expressed as follows:

$$\begin{aligned} HA1 &= \mathrm{MD5}(A1) \\ &= \mathrm{MD5}(username : realm : password) \\ HA2 &= \mathrm{MD5}(A2) = \mathrm{MD5}(method : digestURI) \\ response &= \mathrm{MD5}(HA1 : nonce : HA2) \end{aligned}$$

In this context, $A1$ is the concatenated string of Alice's $username$, the $realm$ (usually a hostname or domain name) and the shared secret $password$ between Alice and the server. For $A2$, the $method$ is the SIP method used in the current transaction, in the above example that would be `REGISTER`. In a `REGISTER` transaction the $digestURI$ is set to the URI in the *To:*-field. The digest authentication $response$ is the hash of the concatenated values of $HA1$, the $nonce$ received from the server, and $HA2$. A SIP `REGISTER` message with a computed digest embedded in the `Authorization` header is shown in Fig. 4. DAA provides only reply protection due to the nonce value and one-way message authentication. There is no encryption of the content, nor confidentiality support, except the shared secret $password$ between client and server. All messages are sent in clear. DAA only works within a local domain so cross-domain authentication is not supported, which implies that end-to-end authentication is not supported. There is no provision in the DAA for the initial secure arrangement between a client and server to establish the shared secret. However, DAA has low computation overhead compared to other methods [17].

## IV. Attack on DAA

When a UA comes online it registers its contact point(s) to a *location service*. Contact points are the preferred methods a user can be contacted by, for example using SIP, mail, or IM. Usually, only a SIP URI contact method is present. The location service is responsible to redirect SIP requests (for VoIP calls) to the correct SIP end-point. For example, an incoming SIP call destined to `alice@CompanyA.org` does not contain information about which hostname or IP-address Alice's phone can be reached. Therefore, a SIP proxy will query the location service to receive Alice's phone's hostname or IP-address, and then redirect the call to this address.

The binding of Alice's phone to a hostname or IP-address is done during the `REGISTER` transaction, as depicted in Fig. 3. Before the binding, or registration, the SIP server should ask the client to authenticate itself, as explained in the

 ISBN:978-1-61208-002-4 

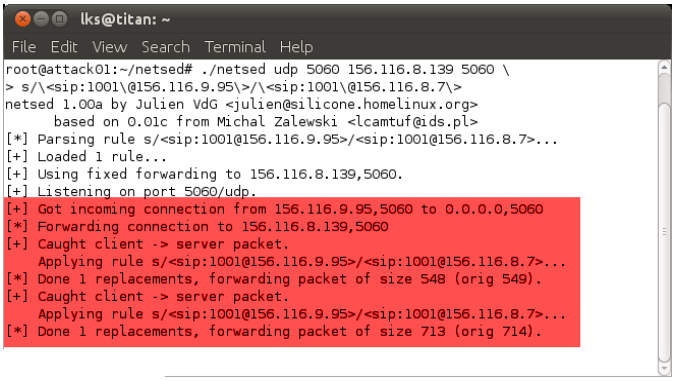



Fig. 6: The network packet stream editor NetSED modifies network packets in real time based on a regular expression (in red).

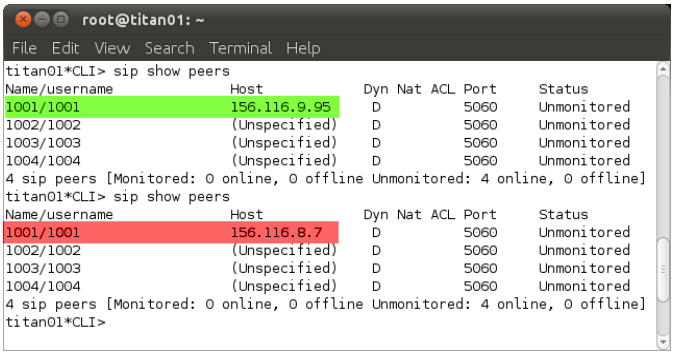



Fig. 7: Host name before (green) and after a successful attack (red), which makes Asterisk believe that Alice's phone (with number 1001) is reachable at an IP-address of the attacker's choice.

previous section. After a successful authentication, the client's hostname or IP-address is registered. A re-registration is normally done at regular intervals. This registration is repeated usually every 3-10 minutes, depending on the configuration. The client's preferred contact methods, including hostname or IP-address, is carried in the SIP header `Contact`, as depicted in Line 5 in Fig. 4. However, this SIP header value is sent in clear, and is not protected by DAA. Thus, the registration is vulnerable to a man-in-the-middle attack [10].

If an attacker modifies the hostname or IP-address in the *contactURI* header value during a `REGISTER` phrase, as depicted in Fig. 5, all requests, and hence calls, to the client will be diverted to a hostname or IP-address controlled by an attacker. Here, Alice cannot perceive that she is unreachable. An attacker can modify Alice's `REGISTER` session in real-time using NetSED [18] as depicted in Fig. 6. The SIP server (Asterisk), will not detect nor suspect that anything is wrong, and register Alice's phone number with the attackers IP address, as seen on Asterisk's terminal in Fig. 7. When Asterisk receives a call to Alice, the call will be forwarded to the attackers registered IP address.

## V. Improving DAA

The SIP digest authentication is weak, which is stated in both the SIP specification [1], and the digest specification [9]. Specifically, DAA only offers protection of the value in the `To` header called the `Request-URI` and the *method*, but no other SIP header values are protected. Other better and stronger authentication methods have been recommended [19]. Nonetheless, we suggest improving the DAA as well as possible, since DAA is the authentication method commonly used due to its simplicity and widespread support and adoption.

A minor modification of DAA can counter the registration hijack attack [10], which is caused by having too few SIP header parameters protected by the digest. Since an attacker can modify and redirect all requests, we protect the header by including the `Contact` header value in the digest. By including the `Contact` value, which we name *contactURIs* in the digest, we effectively counter the registration hijack attack.

We define $HA0$ with $contactURIs$. The new digest computation algorithm is as follows:

$$\begin{aligned} HA0 &= \text{MD5}(A0) = \text{MD5}(contactURIs) \\ HA1 &= \text{MD5}(A1) \\ &= MD5(username : realm : password) \\ HA2 &= \text{MD5}(A2) = \text{MD5}(method : digestURI) \\ response &= \text{MD5}(HA0 : HA1 : nonce : HA2) \end{aligned}$$

Weaknesses in the MD5 hash have been found. In particular we mention collision attacks where two different input values produce the same MD5 hash [20]. This weakness is not known to be exploitable to reveal a user's password [21]. Nonetheless, a stronger hash function, like SHA1 [22], is recommend.

We implemented and tested our modified DAA by using the Python Twisted [23] networking engine, using both MD5 and SHA1. According to our test, the computation overhead by including $HA0$ with the $ContactURIs$ is minimal, as shown in Fig. 8. The difference between the original DAA and our modified DAA with MD5 for 100.000 authentication requests on a 2.2Ghz Intel CPU, is only 0.44 seconds, a negligible amount.

A modified DAA means a modification of the SIP standard. Since the SIP standard has seen widespread industry adoption, it can be difficult to re-deploy a non-standardized SIP DAA. To prevent a modification of the SIP standard, we can use the DAA parameter `auth-param` to store our modified digest response. The parameter `auth-param` is reserved "for future use" [9, page 12], and can be a part of the `Authorization` header.

SIP devices that do not support the updated and more secure digest, can and will ignore this value, and use the original DAA for authentication. However, we cannot recommend this approach, since an attacker could remove this value and force the usage of the original standardized DAA. We would prefer to modify the DAA digest computation to force an upgrade to

 ISBN:978-1-61208-002-4 

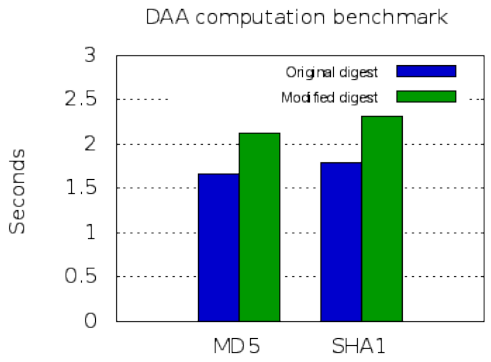



Fig. 8: The computation overhead for 100.000 iterations for original DAA and our modified DAA for both MD5 and SHA1.

the new improved DAA method, instead of compromising on security.

## VI. Related work

Based on the DAA, Undrey [24] proposed a more flexible use of variables protected by the digest. His paper addresses the shortcomings of DAA and suggests to allow the server to decide which headers it requires to be included and protected by the digest computation. Unfortunately, his approach does not require specific headers fields to be included. Therefore, transactions that do not include `Contact` fields are still vulnerable to the registration attack.

Palmieri et al. [25], [26], dismiss DAA as a usable authentication method, and instead craft a new authentication schema with digital signatures based on public-key encryption. They rely on public key infrastructure (PKI), but admit that PKI is difficult and costly to implement.

Yang et al. [27] also conclude that DAA is weak. They argue that, since DAA is vulnerable to an off-line password guessing attacks, a more secure authentication method is required. They propose an authentication method based on Diffie-Hellman. Unfortunate, they do not discuss nor add any additional SIP headers in their new authentication scheme. So their solution is also vulnerable to the registration attack.

The H.323 recommendation for the VoIP protocol from the International Telecommunication Union (ITU) has failed to see widespread adoption by industry players, and is considered abandoned in favor of SIP/RTP [16]. The authentication methods in H.323, specified in H.235 [28], [29] uses well established security mechanism, like certificates, and Diffie-Hellman key exchange, to enforce authentication. Further analysis is needed to see whether the H.235 standard protects the signaling better than SIP.

The Inter-Asterisk eXchange (IAX) [30], also published by the IETF, establishes a competing protocol to SIP/RTP. IAX has several security properties that are better than SIP. By multiplexing channels over the same link and transporting both signaling and media over the same port, enforcing security mechanisms is easier. IAX supports two authentication methods: *1*) MD5 Message Digest authentication [31] computed over a pre-shared secret and a challenge (nonce), or *2*) using RSA public-key encryption on the challenge. In both methods, the nonce value is the only protocol parameter that is integrity protected by the authentication. Future work needs to investigate whether the IAX authentication method is adequately secure.

Other, more secure, authentication methods for SIP have been standardized, such as the support for public key encryption with S/MIME [32], the "Asserted Identity" extension [33], and the "Identity" header extension [34]. None of these authentication methods have seen any widespread deployment yet [19].

## VII. Conclusion

We have seen that the widely deployed authentication method DAA in SIP is weak and vulnerable to attacks. Moreover, we have confirmed and verified that the attack analyzed earlier [10] can be performed on the SIP protocol in real-time. We have examined this authentication method, and proposed a solution to counter the serious registration attack. By including more SIP header parameters in the authentication digest this attack can be countered.

The original SIP designers focused on functionality and compliance at the cost of security. A more thorough investigation of the SIP DAA in the design phase would have revealed the vulnerability presented here, and the vulnerability could have been prevented early on.

Our remedy presented here solves an serious problem with the DAA. However, other weaknesses and shortcomings of DAA are too serious to be part of a strong and secure authentication scheme for SIP. Therefore, we intend to investigate other authentication methods for SIP, including support for Generic Security Service API (GSS-API) [35].

## Acknowledgment

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054). The authors would like to thank Trenton Schulz for discussions. We also thank Anders Moen Hagalisletto and the anonymous reviewers for comments on earlier drafts of this paper.

## References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261 (Proposed Standard), Internet Engineering Task Force, Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt [Accessed: 1. Nov 2011]

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (Standard), Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt [Accessed: 1. Nov 2011]

[3] L. Strand and W. Leister, "A Survey of SIP Peering," in NATO ASI - Architects of secure Networks (ASIGE10), May 2010.

[4] International Telecommunication Union, "7 kHz Audio-Coding within 64 kbits/s," ITU-T Recommendation G.722, 1993.

 ISBN:978-1-61208-002-4 

[5] "IETF Session Initiation Protocol Core Charter." [Online]. Available: http://datatracker.ietf.org/wg/sipcore/charter/ [Accessed: 1. Nov 2011]
[6] D. York, Seven Deadliest Unified Communications Attacks. Syngress, Apr. 2010.
[7] H. Dwivedi, Hacking VoIP: Protocols, Attacks, and Countermeasures, 1st ed. No Starch Press, Mar. 2009.
[8] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf [Accessed: 1. Nov 2011]
[9] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617 (Draft Standard), Internet Engineering Task Force, Jun. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt [Accessed: 1. Nov 2011]
[10] A. M. Hagalisletto and L. Strand, "Formal modeling of authentication in SIP registration," in Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08. IEEE Computer Society, August 2008, pp. 16–21.
[11] L. Fritsch, A.-K. Groven, L. Strand, W. Leister, and A. M. Hagalisletto, "A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project," International Journal on Advances in Security, no. 2&3, pp. 129–141, 2009.
[12] L. Strand, "VoIP lab as a research tool in the EUX2010SEC project," Norwegian Computing Center, Department of Applied Research in Information Technology, Tech. Rep. DART/08/10, April 2010.
[13] "Asterisk: The Open Source PBX & Telephony Platform." [Online]. Available: http://www.asterisk.org/ [Accessed: 1. Nov 2011]
[14] International Telecommunication Union (ITU), "Security Architecture For Open Systems Interconnection (OSI)," The International Telegraph and Telephone Consultative Comittee (CCITT), X.800 Standard, 1991.
[15] R. Shirey, "Internet Security Glossary, Version 2," RFC 4949 (Informational), Internet Engineering Task Force, Aug. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4949.txt [Accessed: 1. Nov 2011]
[16] H. Sinnreich and A. B. Johnston, Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006.
[17] S. Salsano, L. Veltri, and D. Papalilo, "SIP security issues: The SIP authentication procedure and its processing load," Network, IEEE, vol. 16, pp. 38–44, 2002.
[18] "NetSED: The network packet stream editor." [Online]. Available: http://silicone.homelinux.org/projects/netsed/ [Accessed: 1. Nov 2011]
[19] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, SIP Security. WileyBlackwell, Mar. 2009.
[20] X. Wang and H. Yu, "How to break MD5 and other hash functions," IN EUROCRYPT, vol. 3494, 2005.
[21] P. Hawkes, M. Paddon, and G. G. Rose, "Musings on the wang et al. md5 collision," Cryptology ePrint Archive, Report 2004/64, 2004.
[22] D. Eastlake 3rd and P. Jones, "US Secure Hash Algorithm 1 (SHA1)," RFC 3174 (Informational), Internet Engineering Task Force, Sep. 2001, updated by RFC 4634. [Online]. Available: http://www.ietf.org/rfc/rfc3174.txt [Accessed: 1. Nov 2011]
[23] "Twisted Matrix Labs." [Online]. Available: http://twistedmatrix.com [Accessed: 1. Nov 2011]
[24] J. Undery, "Ieft draft: SIP authentication: SIP digest access authentication," IETF, Tech. Rep., Jul. 2001.
[25] F. Palmieri, "Improving authentication in voice over IP infrastructures," in Advances in Computer, Information, and Systems Sciences, and Engineering, K. Elleithy, T. Sobh, A. Mahmood, M. Iskander, and M. Karim, Eds. Springer Netherlands, 2006, pp. 289 – 296. [Online]. Available: http://www.springerlink.com/content/pj11582775h177q0/ [Accessed: 1. Nov 2011]
[26] F. Palmieri and U. Fiore, "Providing true end-to-end security in converged voice over IP infrastructures," Computers & Security, vol. 28, no. 6, pp. 433–449, Sep. 2009.
[27] C. Yang, R. Wang, and W. Liu, "Secure authentication scheme for session initiation protocol," Computers & Security, vol. 24, no. 5, pp. 381–386, Aug. 2005.
[28] International Telecommunication Union, "H.323 security: Framework for security in H-series (H.323 and other H.245-based) multimedia systems," ITU-T Recommendation H.235.0, 2005.
[29] ——, "H.323 security: Framework for secure authentication in RAS using weak shared secrets," ITU-T Recommendation H.235.5, 2005.
[30] M. Spencer, B. Capouch, E. Guy, F. Miller, and K. Shumard, "IAX: Inter-Asterisk eXchange Version 2," RFC 5456 (Informational), Internet Engineering Task Force, Feb. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5456.txt [Accessed: 1. Nov 2011]
[31] R. Rivest, "The MD5 Message-Digest Algorithm," RFC 1321 (Informational), Internet Engineering Task Force, Apr. 1992. [Online]. Available: http://www.ietf.org/rfc/rfc1321.txt [Accessed: 1. Nov 2011]
[32] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC 3853 (Proposed Standard), Internet Engineering Task Force, Jul. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3853.txt [Accessed: 1. Nov 2011]
[33] C. Jennings, J. Peterson, and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks," RFC 3325 (Informational), Internet Engineering Task Force, Nov. 2002, updated by RFC 5876. [Online]. Available: http://www.ietf.org/rfc/rfc3325.txt [Accessed: 1. Nov 2011]
[34] J. Peterson and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)," RFC 4474 (Proposed Standard), Internet Engineering Task Force, Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4474.txt [Accessed: 1. Nov 2011]
[35] J. Linn, "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743 (Proposed Standard), Internet Engineering Task Force, Jan. 2000, updated by RFC 5554. [Online]. Available: http://www.ietf.org/rfc/rfc2743.txt [Accessed: 1. Nov 2011]

 ISBN:978-1-61208-002-4 

# Paper F:
# Generic Security Services API authentication support for the Session Initiation Protocol

Lars Strand, Josef Noll, Wolfgang Leister

In proceedings of the
$7^{th}$ Advanced International Conference on Telecommunications (AICT) 2011
ISBN: 978-1-61208-004-8.

# Generic Security Services API authentication support for the Session Initiation Protocol

Lars Strand
Norwegian Computing Center
Oslo, Norway
lars.strand@nr.no

Josef Noll
UniK-University Graduate Center
Kjeller, Norway
josef.noll@unik.no

Wolfgang Leister
Norwegian Computing Center
Oslo, Norway
wolfgang.leister@nr.no

***Abstract*—The mandatory and most deployed authentication method used in the Session Initiaton Protocol, the Digest Access Authentication method, is weak. Other, more secure authentication methods have emerged, but have seen little adoption yet. In this paper, support for using a generic authentication method, the Generic Security Services API, is added to the Session Initiaton Protocol. When using this method, the Session Initation Protocol does not need to support nor implement other authentication methods, only use the provided API library. This enables the Session Initiation Protocol to transparently support and use more secure authentication methods in a unified and generic way. As the suggested method includes a modification of the Session Initiation Protocol, an initial deployment strategy towards the Generic Security Services API authentication methods is added. To negotiate an authentication service, we use the pseudo security mechanism Simple and Protected GSS-API Negotiation Mechanism.**

***Keywords*—VoIP, SIP, authentication, GSS-API, SPNEGO.**

## I. Introduction

Voice over IP (VoIP) is taking over for the traditional Public Switched Telephony Network (PSTN). At the end of 2009, 29.1 % of the private market in Norway was using VoIP (not including mobile phones). There has been a steady increase in the number of VoIP users since 2002, as well as a decrease in PSTN [1]. With two billion users worldwide having access to the Internet by the end of 2010 [2], the VoIP growth potential is huge.

Several proprietary and non-proprietary VoIP protocols have been created, but the protocol pair Session Initiation Protocol (SIP) and Real-time Transport Protocol (RTP) is emerging as the industry standard. RTP transfers the media content (voice), while SIP handles the signaling, i.e., setup, modification and termination of sessions between two or more participants. SIP is an open standard developed by the Internet Engineering Task Force (IETF) and specified in RFC3261 [3]. Additional functionality is specified in numerous Request For Comments (RFC) standard documents, making the SIP standard large and complex [4].

PSTN is a mature and stable technology providing 99.999% uptime [5], and users will expect VoIP to perform at similar service level. But with an increasing number of VoIP users, VoIP will become a target for attackers looking for financial gain or mischief. A clear threat taxonomy is given by the "VoIP Security Alliance" [6] and is discussed by Keromytis [7]. Several vulnerabilities exist [8] and securing SIP based installation is far from trivial [9].

In the EUX2010sec research project [10], we revealed, in close collaboration with our project partners, that most VoIP installations only use the mandatory, but weak, Digest Access Authentication (DAA) method [11]. In two case studies, with a VoIP installation supporting 3000 and 4700 phones respectively, the public tender for those VoIP installations greatly emphasized security and authentication requirements; requirements that DAA does not cover adequately. We have replicated the project partner's VoIP installation in our VoIP lab [12]. An attack against authentication has been analyzed [13] and countered [14].

The main contribution of this paper is to present and add support for the Generic Security Services Application Program Interface (GSS-API) to SIP. Different security requirements may require different authentication mechanisms. Instead of adding support for many different authentication mechanisms in SIP, support for GSS-API will provide a generic interface that makes different authentication methods transparent to the SIP protocol. To negotiate the best available authentication service between two peers, the Simple and Protected GSS-API NEGOtiation (SPNEGO) mechanism is used on top of GSS-API.

The rest of the paper is organized as follows: a brief overview of other SIP authentication methods is given in Section II. The GSS-API and SPNEGO as a GSS-API mechanism are explained in Section III and how the GSS-API can be supported and implemented in SIP is shown in Section IV. The industry evolution uptake strategy is briefly discussed in Section V before future work and the conclusion are presented in Section VI.

## II. Authentication in SIP

When SIP was designed, functionality — and not security — was the primary goal. The results today are a number of uncovered vulnerabilites and attacks [15], [16]. SIP supports a wide range of functionalities that can be utilized, ranging from mobile handsets to high-end servers, each with different security requirements. Different security requirements may use different authentication methods depending on the

 ISBN: 978-1-61208-004-8

usage and threat scenario. For example, a mobile handset may have different requirements for authentication than that authentication between SIP servers. Additional requirements like power consumption and computational power must also be considered. Thus, adding new security services to SIP to improve the security design and meet different security requirements can be challenging.

The Digest Access Authentication (DAA) method is the mandatory and most used authentication method used with SIP [17]. DAA is heavily influenced by HTTP digest authentication. It relies on a cryptographic verification of a plaintext password shared between client and server. The client computes a MD5 hash value using the password, a nonce value received from the server, and a few SIP header values. The server computes the same digest, which then is compared against the one received from the client. If these digests are identical, the client has proven its identity and is authenticated. Unfortunately, the DAA is considered weak and is vulnerable to a series of attacks [8], including registration hijacking [14].

A more secure authentication method can be achieved by using Secure MIME (S/MIME) [18]. There, the entire SIP message is encapsulated in a MIME body that is signed and optionally encrypted. When the S/MIME header is received, the receiver checks whether the sender's certificate is signed by a trusted authority. A client must support multiple root certificates since there is no consolidated root authority that is trusted by all clients. This and other certificate handling issues like revoking and renewing complicates the use of certificates. Industry support for S/MIME has been limited [19].

Two other authentication methods have emerged within the Internet Engineering Task Force (IETF):

1) The *Asserted Identity* [20] is intended to work within a trusted environment. An additional, unprotected SIP header is sent in clear that informs that the identity of the client has been checked. Since the SIP header is sent in clear rather than protected by cryptography methods, it can easily be removed by an attacker without any of the communicating peers noticing this.
2) The *SIP Strong Identity* [21] introduces a new SIP service, the "authentication service", which signs a hash over selected SIP header values, and includes the signature as a SIP header along with a URI that points to the sender's certificate. The receiver computes the same hash and compares the results. Using this method, only the client is authenticated. As above, an attacker can remove these headers without implications.

Note that both of these authentication methods rely on a successful DAA authentication to be applicable. These are also applied by the SIP servers rather than the clients themselves, and are thus only providing indirect authentication of the client since the server is authenticating on behalf of the client.

## III. GSS-API WITH SPNEGO

The GSS-API [22] provides a generic interface for application layer protocols like SIP, with a layer of abstraction for different security services like authentication, integrity or confidentiality. With the GSS-API, an application does not need to support or implement every authentication method, but use the provided security API [23]. The GSS-API is developed by the IETF and has been scrutinized by security professionals over the years. It has been extensively tested, and is now classified as a mature standard by the IETF. Further extensions and improvements to GSS-API are done by IETF's "kitten" Working Group [24].

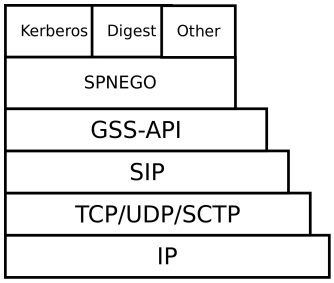



Figure 1: The GSS-API protocol stack with the SPNEGO negotiation mechanism and underlying security mechanisms.

The GSS-API is not a communication protocol in itself, but relies on the application to encapsulate, send, and extract data messages called "tokens" between the client and server. The tokens' content are opaque from the viewpoint of the calling application, and contain authentication data, or, once the authentication is complete, portion of data that the client and server want to sign or encrypt. The tokens are passed through the GSS-API to a range of underlying security mechanisms, ranging from secret-key cryptography, like Kerberos [25], to public-key cryptography, like the Simple Public-Key GSS-API Mechanism (SPKM) [26]. For an application, the use of the GSS-API becomes a standard interface to request authentication, integrity, and confidentiality services in a uniform way. However, GSS-API does not provide credentials needed by the underlying security mechanisms. Both server and client must aquire their respective credentials before GSS-API functions are called.

To establish peer entity authentication, a security context is initialized and established. After the security context has been established, additional messages can be exchanged, that are integrity and, optionally, confidentially protected. To initiate and manage a security context, the peers use the *context-level* GSS-API calls. The client calls `GSS_Init_sec_context()` that produces a "output_token" that is passed to the server. The server then calls `GSS_Accept_sec_context()` with the received token as input. Depending on the underlying security mechanism, additional token exchanges may be required in the course of context establishment. If so, `GSS_S_CONTINUE_NEEDED` status is set and additional tokens are passed between the client and server until a security context is established, as depicted in Figure 4.

After a security context has been established, *per-message* GSS-API calls can be used to protect a message by adding

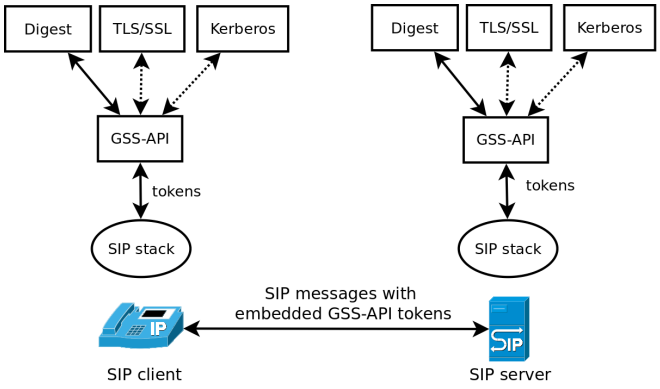



Figure 2: The GSS-API interface in SIP.

a Message Integrity Code (MIC) with `GSS_GetMIC()` and verifying the message with `GSS_VerifyMIC()`. To encrypt and decrypt messages, the peers can use `GSS_Wrap()` and `GSS_Unwrap()`. Thus, two different token types exist:

1) *Context-level tokens* are used when a context is established.
2) *Per-message tokens* are used after a context has been established, and are used to integrity or confidentiality protect data.

In addition to send and receive tokens, the application is responsible to distinguish between token types. This is necessary because different tokens types are sent by the application to different GSS-API functions. But since the tokens are opaque to the application, the application must use a method to distinguish between the token types. In our solution, we use explicit tagging of the token type that accompanies the token message.

SPNEGO [27] is a pseudo security mechanism that enables peers to negotiate a common set of one or more GSS-API security mechanisms. The GSS-API stack with SPNEGO is shown in Figure 1. The client sends a prioritized list of supported authentication mechanisms to the server. The server then chooses the preferred authentication method based on the received list from the client. The client initiates `GSS_Init_sec_context()` as with an ordinary GSS-API security mechanism, but requests that SPNEGO is used as the underlying GSS-API mechanism ("mech_type"). The SPNEGO handshake between client and server is communicated by sending and receiving tokens. After the handshake, the client and server initiate and set up a security context (authentication) using the agreed GSS-API security mechanism.

## IV. GSS-API SUPPORT FOR SIP

Instead of adding numerous different authentication methods to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. The industry might also be reluctant to adopt immature and non-standardized security services, like different authentication methods. Adding support for the GSS-API requires only one small change to the SIP standard, and will open up for a wide range of different authentication methods. In the following subsections, we outline how to include support for the GSS-API into the SIP authentication to replace the original weak DAA.

### A. SIP authentication using DAA

When a SIP client is authenticated to a server using DAA, the authentication handshake data is encapsulated in the `WWW-Authenticate` header from server to client, and the `Authorization` header from client to server. We reuse these headers for GSS-API support, and instead of encapsulate DAA data, we send the GSS-API tokens. An example of both DAA `Authorization` header and the new `Authorization` header with GSS-API data is depicted in Figure 3.

During the initialization of a security context it is necessary to identify the underlying security mechanism to be used. The caller initiating the context indicates at the start of the token the security (authentication) mechanism to be used. The security mechanism is denoted by a unique Object Identifier (OID). For example, the OID for the Kerberos V5 mechanism is `1.2.840.113554.1.2.2`. However, there is no way for the initiating peer to know which security mechanism the receiving peer supports. If an unsupported "mech_type" is requested, the authentication fails. The GSS-API standard resolves this by recommending to manually standardizing on a fixed "mech_type" within a domain. Since SIP addresses are designed to be global [28], and not confined to a local domain, a GSS-API *negotiation* mechanism is required. The SPNEGO is such a GSS-API negotiation mechanism.

### B. SIP authentication using GSS-API and SPNEGO

When using GSS-API with the SPNEGO mechanism, the number of SIP messages between client and server during authentication needs to be increased. During a DAA authentication, the client sends a `REGISTER` message to the server. The server, upon receiving a `REGISTER`, challenges the client with a nonce. The client then generates a digest response, a hash value computed over several SIP header values, the nonce, and a shared secret. The client then re-sends the `REGISTER` message with the digest response embedded. The message flow of a SIP DAA handshake is shown in the first four messages depicted in Figure 4.

In the following paragraphs, the numbers in parentheses refer to the numbers in Figure 4. When a client comes online and registers itself to a "location service" (SIP server), it does so by sending a SIP `REGISTER` message (1). We define the token type in the variable `ttype`. In the following messages, the `ttype` is set to "context" indicating that these tokens are *context-level tokens*. The first message (1) does not contain any `Authorization` header. The server responds with an empty `WWW-Authenticate` header (3):

```
REGISTER SIP/2.0
WWW-Authenticate: GSSAPI ttype="context"
  token=""
```

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response="ccbde1c3c129b3dcaa14a4d5e35
    519d7",uri="sip:CompanyA",algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: GSSAPI ttype="context"
    token="0401000B06092A864886F712010202DACD139402AAF44350CDE32"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Figure 3: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying GSS-API data to the right.

The client then calls `GSS_Init_sec_context()` with SPNEGO as underlying GSS-API mechanism to negotiate a common authentication mechanism (4). The GSS-API "mech_type" is set to SPNEGOs OID `1.3.6.1.5.5.2`. The token data might be in binary format, depending on the security mechanism used. Since the SIP headers are in ASCII string format, the token data is base64 encoded:

```
SIP/2.0 401 Unauthorized
Authorization: GSSAPI ttype="context"
  token="0401000B06092A864886F7120..."
```

The server retrieves the GSS-API data, the token, and passes this to the SPNEGO GSS-API mechanism. In this first initial token, the client embeds authentication data for its first preferred authentication mechanism. This way, should the server accept the clients preferred mechanism, we avoid an extra SIP message round trip. If the client's preferred method was accepted by the server, the server passes the relevant authentication data to the selected authentication mechanism in a `401` SIP message (5). The selected authentication method continues to pass tokens between client and server as many times as necessary to complete the authentication (6-7-N) and establish a security context. Once the security context is established, it sends a `200 OK` SIP message (N+2). Should the server have some last GSS-API data to be communicated to the client to complete the security context, it can be carried in a `WWW-Authenticate` header embedded in the `200 OK` message:

```
SIP/2.0 200 OK
WWW-Authenticate: GSSAPI ttype="context"
  token="dd02c7c2232759874e1c20558701..."
```

If the client's preferred mechanism is not the server's most preferred mechanism, the server outputs a negotiation token and sends it to the client embedded in a new `401` SIP message (5). The client processes the received SIP message and passes the authentication data to the correct authentication mechanism. The GSS-API then continues as described in the previous paragraph.

## V. Evolution strategy

Industrial uptake of new ideas and protocols requires an evolution strategy, especially in the telecom world where standards are used to serve more than four billion people. TCP/IP, IPv6, and UMTS are all examples of technologies where the evolutionary path was not clearly identified, and the technological uptake was significantly delayed.

Authentication for SIP-based services will not only be limited to mobile handsets and SIP authentication servers, it will be used for sensors and devices in the future Internet of Things (IoT). Thus the basic requirement of an advanced authentication scheme is modularity and flexibility: both of these are provided by the suggested approach. Support for the GSS-API will extend the SIP protocol with an improved security mechanism that offers more flexibility in different scenarios.

An industrial uptake will first be envisaged for mobile devices such as smartphones or tabs, where SIP clients can be provided with the new functionality. Discussions with industrial actors are ongoing to ensure the compatibility on the server side. After this initial phase, an extension of the approach is envisaged including access to services in a sensor network.

## VI. Conclusion and future work

Since the only mandatory and widely deployed Digest Access Authentication method in SIP is weak, other more secure authentication methods are desired. In this paper, we have added support for GSS-API in SIP, as well as for the SPNEGO mechanism that is used to negotiate the preferred GSS-API security mechanism supported by both client and server. The required change to the SIP protocol has been kept to a minimum, and the authentication header from DAA has been reused to prevent adding additional SIP headers to the standard.

Different VoIP installations have different security requirements that may require different security services. We have shown that the use of the GSS-API provides SIP with a wide range of different authentication methods in a uniform and standardized way. Different authentication methods can be used depending on the different security requirements for each SIP installation. This adds to the flexibility of SIP, like adding a new authentication method, without requiring further changes to the SIP standard, once the GSS-API is supported.

In our earlier work, we have shown that the DAA is weak [14]. Therefore, we want to replace DAA with support for a better, more robust authentication scheme that authentication methods like GSS-API supports. This implies that we *replace*

 ISBN: 978-1-61208-004-8 

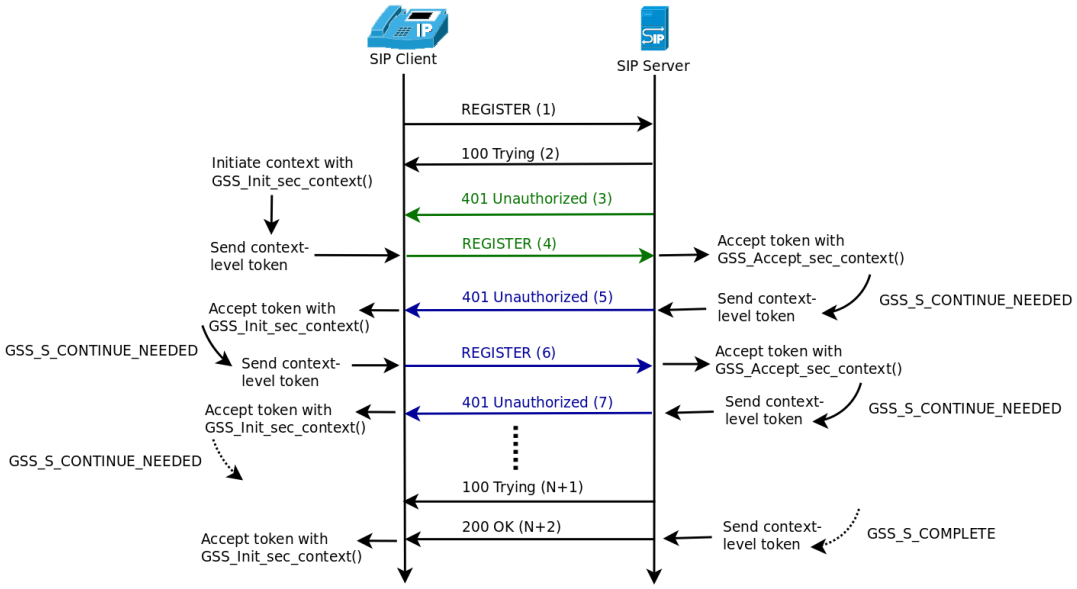



Figure 4: SIP `REGISTER` message flow with GSS-API security context establishment (authentication).

the original DAA header content with GSS-API data content. However, since the GSS-API only is an interface to underlying security mechanisms, the use of the GSS-API does not in itself provide any security service. Thus, the security of the GSS-API is no stronger than the weakest security mechanism acceptable to the client and server using the GSS-API. So, if the underlying GSS-API authentication mechanism does not protect relevant SIP headers, it might be as vulnerable to the attack shown previously with the DAA. We still need to examine what kind of SIP message integrity protection is offered by the different GSS-API authentication mechanisms.

The credential acquisition between peers must also be completed before initiating the GSS-API. Also, if the SP-NEGO negotiation is not integrity-protected, the negotiation is vulnerable to a man-in-the-middle "down-grade" attack. An attacker can intercept and modify the negotiation messages so that the least favorable authentication method is used.

Future work will look into different GSS-API security mechanisms and their implications for SIP, including overhead evaluation benchmark. Implementing a proof of concept with GSS-API support for SIP is also desired. We plan to co-operate with the IETF and the "kitten" WG to further elaborate the GSS-API for SIP. Further challenges are the Simple Authentication and Security Layer (SASL) [29] for SIP, and a comparison of SASL with GSS-API, as well as the support for using GSS-API mechanisms within SASL [30].

## Acknowledgment

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054). The authors would like to thank Trenton Schulz and the anonymous reviewers for valuable comments on earlier drafts of this paper.

## References

[1] "Det norske markedet for elektroniske kommunikasjonstjenester 2009 (The Norwegian market for electronic communication services 2009)," Post- og teletilsynet (The Norwegian Post and Telecommunications Authority), 2010. [Online]. Available: http://www.npt.no/ikbViewer/Content/119027/Ekomrapport_2009_.pdf 10. Jan 2011

[2] Telecommunication Development Sector (ITU-D), "The world in 2010," ITU-T ICT facts and figures, 2010.

[3] 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), Internet Engineering Task Force, Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt 10. Jan 2011

[4] H. Sinnreich and A. B. Johnston, Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006.

[5] D. Kuhn, "Sources of failure in the public switched telephone network," Computer, vol. 30, pp. 31–36, 1997.

[6] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf 1. Nov 2011

[7] A. D. Keromytis, "Voice over IP: Risks, Threats and Vulnerabilities," in Proceedings of the Cyber Infrastructure Protection (CIP) Conference, New York, June 2009.

[8] H. Dwivedi, Hacking VoIP: Protocols, Attacks, and Countermeasures, 1st ed. No Starch Press, Mar. 2009.

[9] A. D. Keromytis, "Voice-over-IP security: Research and practice," IEEE Security & Privacy Magazine, vol. 8, no. 2, pp. 76–78, 2010.

[10] "Research project: EUX2010SEC – Enterprise Unified Exchange Security"." [Online]. Available: http://www.nr.no/pages/dart/project_flyer_eux2010sec 1. Nov 2011

[11] L. Fritsch, A.-K. Groven, L. Strand, W. Leister, and A. M. Hagalisletto, "A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project," International Journal on Advances in Security, no. 2&3, pp. 129–141, 2009.

[12] L. Strand, "VoIP lab as a research tool in the EUX2010SEC project," Norwegian Computing Center, Department of Applied Research in Information Technology, Tech. Rep. DART/08/10, April 2010.

[13] A. M. Hagalisletto and L. Strand, "Formal modeling of authentication in SIP registration," in Second International Conference on Emerging

 ISBN: 978-1-61208-004-8 

Security Information, Systems and Technologies SECURWARE '08. IEEE Computer Society, August 2008, pp. 16–21.

[14] L. Strand and W. Leister, "Improving SIP authentication," in Accepted for publication in The Tenth International Conference on Networks (ICN 2011), Jan 2011.

[15] A. M. Hagalisletto and L. Strand, "Designing attacks on sip call set-up," International Journal of Applied Cryptography, vol. 2, no. 1, pp. 13–22(10), July 2010. [Online]. Available: http://inderscience.metapress.com/link.asp?id=jh437k6747064307 10. Jan 2011

[16] A. D. Keromytis, "A Survey of Voice Over IP Security Research," in Proceeding of the 5th International Conference on Information Systems Security (ICISS), December 2009, pp. 1 – 17.

[17] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617 (Draft Standard), Internet Engineering Task Force, Jun. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt 10. Jan 2011

[18] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC 3853 (Proposed Standard), Internet Engineering Task Force, Jul. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3853.txt 10. Jan 2011

[19] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, SIP Security. WileyBlackwell, Mar. 2009.

[20] C. Jennings, J. Peterson, and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks," RFC 3325 (Informational), Internet Engineering Task Force, Nov. 2002, updated by RFC 5876. [Online]. Available: http://www.ietf.org/rfc/rfc3325.txt 10. Jan 2011

[21] J. Peterson and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)," RFC 4474 (Proposed Standard), Internet Engineering Task Force, Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4474.txt 10. Jan 2011

[22] J. Linn, "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743 (Proposed Standard), Internet Engineering Task Force, Jan. 2000, updated by RFC 5554. [Online]. Available: http://www.ietf.org/rfc/rfc2743.txt 10. Jan 2011

[23] D. Todorov, Mechanics of User Identification and Authentication: Fundamentals of Identity Management, 1st ed. Auerbach Publication, Jun. 2007.

[24] "IETF Common Authentication Technology Next Generation (kitten)." [Online]. Available: http://datatracker.ietf.org/wg/kitten/charter/ 1. Nov 2011

[25] L. Zhu, K. Jaganathan, and S. Hartman, "The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2," RFC 4121 (Proposed Standard), Internet Engineering Task Force, Jul. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4121.txt 10. Jan 2011

[26] C. Adams, "The Simple Public-Key GSS-API Mechanism (SPKM)," RFC 2025 (Proposed Standard), Internet Engineering Task Force, Oct. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc2025.txt 10. Jan 2011

[27] L. Zhu, P. Leach, K. Jaganathan, and W. Ingersoll, "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism," RFC 4178 (Proposed Standard), Internet Engineering Task Force, Oct. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4178.txt 10. Jan 2011

[28] L. Strand and W. Leister, "A Survey of SIP Peering," in NATO ASI - Architects of secure Networks (ASIGE10), May 2010.

[29] A. Melnikov and K. Zeilenga, "Simple Authentication and Security Layer (SASL)," RFC 4422 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4422.txt 10. Jan 2011

[30] S. Josefsson and N. Williams, "Using Generic Security Service Application Program Interface (GSS-API) Mechanisms in Simple Authentication and Security Layer (SASL): The GS2 Mechanism Family," RFC 5801 (Proposed Standard), Internet Engineering Task Force, Jul. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5801.txt 10. Jan 2011

 ISBN: 978-1-61208-004-8 

# Paper G:
# Migration towards a more secure authentication in the Session Initiation Protocol

Lars Strand, Wolfgang Leister, Alan Duric

In proceedings of the
$5^{nd}$ International Conference on Emerging Security Information, Systems and Technologies (SECURWARE) 2011
ISBN: 978-1-61208-010-9

# Migration towards a more secure authentication in the Session Initiation Protocol

Lars Strand
Norwegian Computing Center / University of Oslo
Oslo, Norway
lars.strand@nr.no

Wolfgang Leister
Norwegian Computing Center
Oslo, Norway
wolfgang.leister@nr.no

Alan Duric
Telio Telecom AS
Oslo, Norway
alan.duric@telio.no

***Abstract*—This paper specifies a two-step migration towards a stronger authentication in the Session Initiation Protocol. First, we add support for a Password Authenticated Key Exchange algorithm that can function as a drop-in replacement for the widely adopted Digest Access Authentication mechanism. This new authentication mechanism adds support for mutual authentication, is considered stronger and can rely on the same shared password used by the digest authentication. A more long-term solution is to replace the authentication scheme with the Simple Authentication and Security Layer. The Simple Authentication and Security Layer separates the authentication mechanisms from the Session Initiation Protocol, and adds support for a range of more secure authentication mechanisms in a generic and unified way. Both methods are presented, discussed, and shown how to integrate into the Session Initiation Protocol.**

***Keywords*—VoIP, SIP, authentication, PAKE, SASL.**

## I. INTRODUCTION

Voice over IP (VoIP) is rapidly taking over for the traditional, public switched telephone networks (PSTN). Although there exist several competing network protocols that are capable of delivering VoIP, the Session Initiation Protocol (SIP) [1] and the Real-time Transport Protocol (RTP) [2] developed by the IETF have become the *de facto* industry standard. These two protocols fulfill two different functions – SIP is used for signaling, e.g., responsible for setting up, modifying and tearing down multimedia sessions, while RTP transports the actual media stream (voice). Although the SIP protocol is flexible and rich in functionality [3], several vulnerabilities and security attacks have been found [4]–[6].

Securing a SIP-based VoIP system has proven challenging and the reasons are multi-faceted:

- The scale and complexity of the SIP protocol specification, with primary focus on functionality rather than a sound security design [7].
- SIP usage of intermediaries, expected communication between nodes with no trust at all, and its user-to-user operation make security far from trivial [1, page 232].
- A large number of threats against VoIP systems have been identified [8]. Several security mechanisms for countermeasures have been proposed, but no single security mechanism is suited to address all these security threats concerning VoIP and SIP [9], [10].
- Since the SIP and RTP protocols share the same infrastructure as traditional data networks, they also inherit the security problems of data communication.
- VoIP services have strict requirements to the network performance with respect to Quality of Service since it is a duplex communication with low tolerance for latency, packet loss and saturation. Introducing strong security mechanisms might affect network performance [11].

Signaling in PSTN has traditionally involved trust between carriers, and by end users (caller-id). To achieve the same trust level using SIP, we need to employ secure authentication.

In VoIP, authentication tries to validate the identity of the communication peers and to bind that identity to a subject (peer). It must be stressed that the user is not authenticated, but the user's phone. In VoIP terminology, a subject could be a User Agent (UA), such as a phone, identified by a phone-number/username and IP-address/hostname pair, denoted as an Address-of-Record (AoR). The authentication in VoIP is therefore the assurance that a communicating entity, the UA, is the one that it claims to be [12]. However, the authentication in SIP has proven weak [13] and vulnerable to a real-world security attack [14].

Equally important for the UA is to establish the identity of the communicating peer, i.e., the SIP server. If the client does not authenticate the SIP server, it might risk to communicate and send content to a hostile SIP server.

The main contribution of this paper is to propose a migration towards a more secure SIP authentication. First, we introduce an authentication method based on the Password Authenticated Key Exchange (PAKE) [15], which provides mutual authentication based on a shared secret, and can function as a drop-in replacement of the digest authentication currently used. However, a more flexible authentication method is desired. As a second authentication method, we propose the Simple Authentication and Security Layer (SASL) [16], which enables SIP to transparently support and use more secure authentication methods in a unified and generic way.

The rest of the paper is organized as follows: In Section II we give an introduction to authentication in SIP and discuss related work. In Section III we show how a modified PAKE can be used to add mutual authentication in SIP. The second authentication mechanism added to SIP, SASL, is explained

 ISBN: 978-1-61208-010-9 

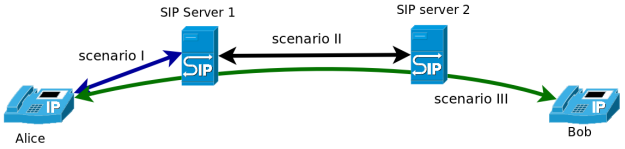



Figure 1: Three different usage scenarios where authentication in SIP is desired.

and discussed in Section IV. Conclusion and future work is presented in Section V.

## II. Identity in the Session Initiation Protocol

We identify three scenarios where identity in SIP needs to be handled, as depicted in Figure 1: Scenario *I* between the UA and the local SIP server; Scenario *II* between SIP servers; and Scenario *III* end-to-end.

Scenario I between the UA and the local SIP server is relevant when the UA comes online and before any outgoing calls can be placed. Then, the UA must register itself to a local SIP server. During the SIP register handshake, the server usually challenges the UA to authenticate. Before placing a call (sending a SIP `INVITE`), the UA might be challenged again by the server to authenticate. The most common authentication method used between UA and server today is the Digest Access Authentication (DAA) [17].

Scenario II handles the authentication between SIP servers to achieve trust between SIP servers. It is not desirable to have SIP traffic handled by an unknown or untrusted SIP server that might have malicious intent. However, since most SIP servers today use some kind of SIP peering [18], the relationships between servers are often static and pre-defined. Therefore the identities between SIP servers are often predetermined by other security mechanisms than what are offered by SIP (like IPSec, TLS etc.).

Scenario III is about end-to-end authentication, which determines the identity of both the caller and the callee across different SIP domains. This is of particular importance and not easily attained in SIP. There is an increased threat and fear for both VoIP phishing and SPIT (Spam over Internet Telephony), that might seriously affect SIP-based VoIP services. By enforcing end-to-end authentication in SIP, these threats might be mitigated or prevented.

We list authentication mechanisms in SIP and their support in these three SIP scenarios in Table I. We shall discuss these authentication mechanisms in the following.

The DAA is currently the most common authentication mechanism for SIP. DAA is simple but rather insecure. It is the only authentication mechanism which support in SIP is mandatory [1, Section 22]. DAA uses the MD5 hash function and a challenge-response pattern, and relies on a shared secret between client and server within a SIP domain [17]. DAA is performed during the SIP `REGISTER` handshake between the UA and the SIP server, as depicted in messages 1-3 and 6 in Figure 2. The UA receives a nonce value from the SIP server, computes a digest hash value over the nonce, the shared secret and some other SIP header values, and send it to the SIP server. The SIP server computes the same digest hash. If both digests are identical, the UA is authenticated. The DAA is weak and vulnerable to a serious real-world attack [14]. Since the DAA relies on a shared secret and is only meaningful for a specific realm, its usage is limited to Scenario I.

Secure MIME (S/MIME) [19] is an authentication mechanism presented in the SIP core specification document RFC3261 [1]. S/MIME intends to achieve end-to-end authentication between UAs. The entire SIP message is encapsulated in a specific SIP message using MIME, which is signed and optionally encrypted. The receiving UA checks whether the sending UA's certificate is signed by a trusted authority. Since S/MIME depend on end-user certificates, the UAs must support multiple root certificates since no consolidated certificate authority exists. Additionally, certificate handling issues, such as revocation and renewal, complicate the use of certificates. There has been rather limited industry support for S/MIME.

Palmieri et al. [20] introduce a new authentication mechanism using digital signatures. But since they rely on certificates, their solution suffers under similar certification handling issues as S/MIME. They also admit that relying on public key infrastructure (PKI) is both difficult and costly to implement. Liao et at. [21], propose an improved authentication in SIP with self-signed public keys on elliptic curves. However, Liao's proposal uses smart-cards to store authentication data and rely on a trusted third party [22].

The SIP protocol needs an authentication mechanism that avoids the security vulnerabilities the currently used DAA has. A replacement authentication mechanism should preferably not rely on PKI, have support for strong mutual authentication, and support all three scenarios listed previously in this section.

## III. Password Authenticated Key Exchange

We propose to add support for a variant of PAKE denoted as "Key Agreement Method 3" (KAM3) as a cryptographic protocol [15, page 17]. PAKE has the following attractive features: *1*) PAKE provides mutual authentication between UA and the SIP server, and thus a rogue SIP server can not claim that the authentication succeed without knowing the shared password. PAKE assures the UA that the SIP server knows the UA's encrypted password. *2*) Reuse of the shared password used by DAA as the UA's credential, which enables our approach to easily replace DAA used within a local SIP domain (scenario I). *3*) PAKE offers strong protection of the shared secret if the communication is eavesdropped, that prevents brute-force attacks, including dictionary-based off-line attacks, to which the DAA is vulnerable to.

Our approach follows the work of Oiwa et al. [23]. They use KAM3 to introduce a stronger authentication in HTTP and their initial design and specification is submitted to the IETF as an Internet Draft [24]. We have adapted their approach to SIP, since SIP closely resembles HTTP in both message structure and flow, and we need to prevent the `REGISTER` hijack attack presented earlier [14].

 ISBN: 978-1-61208-010-9 

| Authentication mechanisms | Supported authentication scenarios | | | Supported SIP methods | |
|---|---|---|---|---|---|
| | scenario I | scenario II | scenario III | REGISTER | INVITE |
| Digest authentication | yes | no | no | yes | yes |
| PAKE | yes | no | no | yes | yes |
| SASL | yes | yes | yes | yes | yes |
| GSS-API | yes | yes | yes | yes | yes |
| S/MIME | no | no | yes | no | yes |

Table I: List of SIP authentication mechanisms and their support.

In KAM3, the UA and the SIP server compute cryptographic keys based on the shared password. These keys are exchanged, and a shared session secret is computed based on these keys. Each peer then computes a hash value of the session secret and some other values, which is sent to the requesting peer. The receiving peer computes the same hash value, and compares it with the received hash value. If these are identical, the sending peer is authenticated.

PAKE supports several authentication algorithms, which differ in their underlying mathematical groups and security parameters [24]. The only mandatory supported authentication algorithm, the *iso-kam3-dl-2048-sha256*, uses the 2048-bit discrete-logarithm defined in RFC3526 [25] and the SHA-256 hash function.

### A. Initial requirements

In the following section, we let $q$ an odd prime integer defining the number of elements in $F(q)$ which is a representation of a finite group. We let $g$ the generator of a subgroup of $r$ elements in $F(q)$. The one-way hash function is denoted as $H$.

Before the authentication starts, username and password must be set and configured. We compute a weak secret $\pi$ used by the client as a one-way hash of the values $realm$, $username$ and $password$:

$$\pi = H(realm, username, password)$$

Here, $realm$ is the protection domain where SIP authentication is meaningful for a set of $username$ and $password$. The server does not need to store the shared password directly, only a specially encrypted version $J(\pi)$, where $J$ is the password verification element derivation function defined as:

$$J(\pi) = g^{\pi} \bmod q$$

### B. Message exchange

We need to extend the current SIP `REGISTER` handshake by one extra round-trip of SIP messages between the UA and the SIP server. These two extra messages are depicted in blue and numbered (4) and (5) in Figure 2. A more detailed specification is given in the following paragraphs, where the numbers refer to the protocol clauses depicted in Figure 2.

The UA registers to a SIP *location service* (SIP server). The initial SIP `REGISTER` message (1) from the UA is not authorized, and must be authenticated. The SIP server responds with a `401 Unauthorized` status message (2),

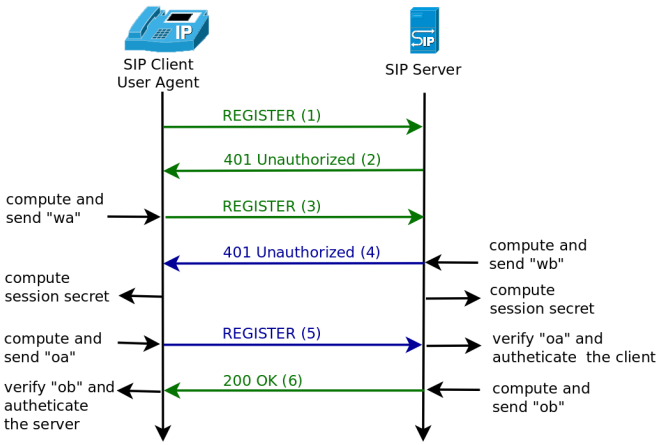



Figure 2: SIP `REGISTER` message flow with mutual authentication security using PAKE.

which contains a `WWW-Authenticate` header with details of the challenge, including *realm* and *algorithm*. The UA constructs a cryptographic value $w_a$ generated from a random integer $s_a$:

$$w_a = g^{s_a} \bmod q$$

This value is sent in a new SIP `REGISTER` message (3) to the SIP server. The SIP server proceeds to generate and send another cryptographic value $w_b$, which is generated from $J(\pi)$, the received value $w_a$ and a random integer $s_b$:

$$w_b = (J(\pi) \times w_a^{H(1,w_a)})^{s_b} \bmod q$$

At the next step, each peer computes a session secret $z$. The UA derives $z$ based on $\pi$, $s_a$, $w_a$ and $w_b$:

$$z = w_b^{(s_a + H(2,w_a,w_b))/(s_a * H(1,w_a) + \pi) \bmod r} \bmod q$$

Likewise, the SIP server derives $z$ based on $s_b$, $w_a$ and $w_b$ using the following function:

$$z = (w_a \times g^{H(2,w_a,w_b)})^{s_b} \bmod q$$

The session secret $z$ matches only if both peers have used the secret credentials generated from the same shared secret. The above equations are directly derived from the PAKE HTTP authentication specifications [24]. The next step is to validate the value of $z$ at the communicating peer.

 ISBN: 978-1-61208-010-9

The UA sends a third SIP `REGISTER` message (5) and includes the value $o_a$ which is a hash value computed as:

$$o_a = H(4, w_a, w_b, z, contactURIs)$$

Here, $contactURIs$ is the value of the UA's `Contact` SIP header value. This value is integrity-protected to prevent register hijacking attacks as presented in [14]. The SIP server, upon receipt of $o_a$, performs the same hash operation, and compares the results. If these results are identical, the UA is authenticated. The SIP server then sends a final message (6), with the value $o_b$ computed as:

$$o_b = H(3, w_a, w_b, z, contactURIs)$$

When the UA receives $o_b$, it verifies this value by computing its hash value. If the results are identical, the SIP server is authenticated to the UA. After a complete message exchange, the UA is authenticated to the SIP server, and the SIP server has been authenticated to the UA.

### C. SIP message syntax

A SIP message consists of several headers and a body. The SIP header fields are textual, always in the format `<header_name>: <header_value>`. The header value can contain one or more parameters. We embed the cryptographic values derived in the previous section as base64-encoded [26] SIP header values. We re-use the SIP DAA headers to carry PAKE authentication data, so that PAKE can be used as a drop-in replacement for DAA. A SIP `REGISTER` message with a DAA `Authorization` header is depicted in Figure 4. Again, we refer to the protocol clauses with a number in parentheses as depicted in Figure 2.

The UA first sends a SIP `REGISTER` without any authentication credentials (1). The SIP server responds with a `401 Unauthorized` status message (2), which contains a `WWW-Authenticate` header with header values *realm* and *algorithm*:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: Mutual realm="asterisk",
  algorithm="iso-kam3-dl-2048-sha256"
```

The UA then computes $w_a$ and sends it to the SIP server using a new SIP `REGISTER` message (3), with the required values embedded in the `Authorization` header:

```
SIP/2.0 REGISTER
Authorization: Mutual user="alice",
  algorithm="iso-kam3-dl-2048-sha256",
  wa="Q29tcHV0ZWQgd2E...ljaCBcyBsb25nCg=="
```

The next required values in the authentication mechanism $w_b$, $o_a$ and $o_b$ are embedded and sent using these two SIP headers.

## IV. Simple Authentication and Security Layer

The Simple Authentication and Security Layer (SASL), defined in RFC4422 [16], provides an interface for authentication and an authentication negotiation mechanism. The SASL specification is developed and maintained within the IETF, and have been scrutinized by security professionals over the years. It has been extensively tested, and is now classified as a mature standard by the IETF. SASL is also implemented and used in several popular communications protocols applications like IMAP, SMTP and LDAP[1].

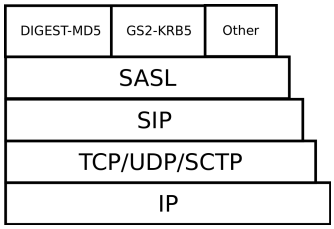



Figure 3: The SIP protocol stack with SASL and underlying security mechanisms.

The SASL framework does not provide authentication mechanisms in itself, but supports different underlying authentication mechanisms through a standardized interface[2]. SASL does not provide a transport layer and thus relies on the application to encapsulate, send and extract SASL messages between client and server. The SASL messages sent between client and server contain authentication data, and are opaque from the viewpoint of the calling application. The application only needs to add support to a SASL software library implementation, and thus have support to a range of underlying authentication mechanisms the library supports.

Adding support for the security abstraction layer framework Generic Security Services API (GSS-API) has been done earlier [27]. While the GSS-API is intended for use with applications, SASL is used in, and intended for, communication protocols. The functionalities offered by the GSS-API and SASL are alike, but the SASL specification is more high-level, and allows more freedom in implementing the SASL requirements. SASL also supports more underlying security mechanisms than the GSS-API. By using the "GS2" mechanism family, the GSS-API can be used as an underlying security mechanism in SASL. However, the GSS-API negotiation mechanism cannot be used due to security concerns [28, Section 14].

### A. SASL profile for SIP

A modified PAKE authentication can more easily replace the current digest (DAA) authentication used in SIP, since they both rely on a shared secret and use the same SIP headers. PAKE also introduces a stronger authentication than DAA. However, a more flexible authentication mechanism is desired. Different VoIP scenarios require different security requirements, and the communicating peers should be able

[1]The Carnegie Mellon University's implementation: http://asg.web.cmu.edu/sasl/ and the GNU SASL library: http://www.gnu.org/software/gsasl/ are two popular and freely available SASL libraries.

[2]A list of registered SASL mechanisms is maintained by IANA: http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml

 ISBN: 978-1-61208-010-9 

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response=
    "ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",
    algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: SASL mechanism="DIGEST-MD5"
    data="YAzgusSGGeRFGw9nfUvOAxcEDzZCBmKY1H2ElnegaccBcx3DUSkGNW
    Y4qfiSwcXwjLtoqW0eBNog7ixHN"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Figure 4: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying SASL data to the right.

to negotiate the best possible authentication mechanism supported.

Instead of adding numerous different authentication mechanisms to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. The industry might also be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Adding support for SASL requires only small changes to the SIP standard, and can utilize several underlying authentication mechanisms. The SIP protocol stack with SASL is shown in Figure 3.

In SASL terminology, the description on how to encapsulate SASL negotiation and SASL messages for a given protocol, is called a "SASL profile". We create a SASL profile for SIP by reusing the `WWW-Authenticate` and `Authorization` SIP headers used by the digest authentication, shown earlier. Instead of encapsulating DAA data, we embed SASL messages, as depicted in Figure 4.

When discussing PAKE authentication earlier, we added one round-trip of SIP messages between the UA and the SIP server. When using SASL, the number of messages going back and forth depends on the underlying authentication mechanism. We therefore extend the SIP `REGISTER` handshake with an arbitrary number of round-trips, until the underlying authentication mechanism has completed communication.

In the following paragraphs, the numbers in parentheses refer to the SIP message numbers in Figure 2. The SASL specification only outlines a very high-level method of how the server should advertise its supported mechanisms to the client. We implement the mechanism negotiation in the first three messages in the SIP `REGISTER` handshake (1-3). The UA starts by requesting authentication from the SIP server, with no `Authorization` header (1). The SIP server responds with a `401 Unauthorized` SIP message (2), with the supported and available mechanisms embedded in the `WWW-Authenticate` header:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: SASL
  negotiate="DIGEST-MD5 NTLM GS2-KRB5"
```

The client selects the best mechanism from the received list that it supports and sends a new SIP `REGISTER` message (3). This message includes an `Authorization` header requesting authentication with "GS2-KRB5" as the preferred mechanism. The initial authentication data is embedded base64 encoded to the *data* parameter:

```
SIP/2.0 REGISTER
Authorization: SASL mechanism="GS2-KRB5",
  data="SUZZT1VDQU5SR...JUPVVQU5FUkQK="
```

The server retrieves the SASL data, and passes the message to the SASL library which handles the authentication. The selected authentication method continues to pass SASL messages between client and server as many times as necessary to complete the authentication (messages 4-5 are repeated). Once the authentication is complete, the SIP server sends a `200 OK` SIP message. Should the server have some last SASL data to be communicated to the client to complete the authentication, it can be carried in a `WWW-Authenticate` header embedded in the `200 OK` message:

```
SIP/2.0 200 OK
WWW-Authenticate: SASL mechanism="GS2-KRB5",
  data="TFoG9rP56zrVH...YaAOndwPew6NdxKr"
```

As soon as the `200 OK` message is received and processed, the client is authenticated to the SIP server. Since the mechanism negotiation is not integrity-protected, the UA is vulnerable to a "down-grade" attack. An attacker can intercept and modify the negotiation messages so that the least favorable authentication method is used.

## V. Conclusion and Future Work

In our earlier work, we have shown and implemented a real-world attack to the widely deployed DAA method [27]. In this paper we have added support to a new improved authentication mechanism that can easily replace DAA based on a modified PAKE algorithm. This new authentication mechanism adds support for mutual authentication and is more secure than DAA. We have also shown that the modified PAKE authentication can easily function as a drop-in replacement for DAA. However, a more flexible authentication mechanism is desired in the long-term.

 ISBN: 978-1-61208-010-9

Our second authentication mechanism supported in SIP is SASL, which is not an authentication mechanism *per se*, but introduces a security abstraction layer. This abstraction layer adds support to a range of underlying authentication mechanism in a unified way. As long as SIP supports SASL, new authentication mechanisms can be added later to the SASL library, without requiring any change to the SIP protocol. We have also introduced a SASL mechanism negotiation that enables the communicating peers to agree upon the "best" available authentication mechanism.

We envisage a two-step migration towards a stronger authentication scheme in SIP. First, the modified PAKE authentication is implemented and deployed. Second, the long-term solution is to deploy SASL with support for a range of underlying authentication mechanisms.

Future work will look into implementing a proof of concept for PAKE-enabled UA and SIP server, including overhead evaluation benchmarks for the new authentication algorithm. We also plan to evaluate different SASL security mechanism and their implications for SIP, and decide which authentication mechanisms should be mandatorily supported through SASL. A co-operation with the IETF and the kitten Working Group to further elaborate a SASL profile for SIP is also planned. We hope our work will gain acceptance and industrial deployment, so that the previously mentioned security attacks can be countered.

## ACKNOWLEDGMENT

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054). The authors would like to thank the anonymous reviewers for their valuable comments on earlier drafts of this paper.

## REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261 (Proposed Standard), Internet Engineering Task Force, Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt 11. Apr 2011

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (Standard), Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt 11. Apr 2011

[3] H. Sinnreich and A. B. Johnston, Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006.

[4] H. Dwivedi, Hacking VoIP: Protocols, Attacks, and Countermeasures, 1st ed. No Starch Press, Mar. 2009.

[5] D. Endler and M. Collier, Hacking Exposed VoIP: Voice over IP Security Secrets and Solutions. McGraw-Hill Osborne Media, November 2006.

[6] A. M. Hagalisletto and L. Strand, "Designing attacks on SIP call set-up," International Journal of Applied Cryptography, vol. 2, no. 1, pp. 13–22(10), July 2010.

[7] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, SIP Security. WileyBlackwell, Mar. 2009.

[8] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf 1. Nov 2011

[9] D. York, Seven Deadliest Unified Communications Attacks. Syngress, Apr. 2010.

[10] P. Park, Voice over IP Security. Cisco Press, Sep. 2008.

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

[12] International Telecommunication Union (ITU), "Security Architecture For Open Systems Interconnection (OSI)," The International Telegraph and Telephone Consultative Comittee (CCITT), X.800 Standard X.800, 1991.

[13] A. M. Hagalisletto and L. Strand, "Formal modeling of authentication in SIP registration," in Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08. IEEE Computer Society, August 2008, pp. 16–21.

[14] L. Strand and W. Leister, "Improving SIP authentication," in Proceedings of the Tenth International Conference on Networking (ICN2011). Xpert Publishing Services, Jan 2011, pp. 164 – 169.

[15] International Organization for Standardization and ISO, "ISO/IEC 11770-4:2006: Information technology – Security techniques – Key management – Part 4: Mechanisms based on weak secrets," 2006.

[16] A. Melnikov and K. Zeilenga, "Simple Authentication and Security Layer (SASL)," RFC 4422 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4422.txt 11. Apr 2011

[17] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617 (Draft Standard), Internet Engineering Task Force, Jun. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt 11. Apr 2011

[18] L. Strand and W. Leister, "A Survey of SIP Peering," in NATO ASI - Architects of secure Networks (ASIGE10), May 2010.

[19] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC 3853 (Proposed Standard), Internet Engineering Task Force, Jul. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3853.txt 11. Apr 2011

[20] F. Palmieri and U. Fiore, "Providing true end-to-end security in converged voice over IP infrastructures," Computers & Security, vol. 28, no. 6, pp. 433–449, Sep. 2009.

[21] Y. Liao and S. Wang, "A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves," Computer Communications, vol. 33, no. 3, pp. 372–380, Feb. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366409002631 11. Apr 2011

[22] Y. Liao, "Secure password authenticated key exchange protocols for various enviroments," Ph.D. dissertation, Tatung University, Dec. 2009.

[23] Y. Oiwa, H. Watanabe, and H. Takagi, "Pake-based mutual http authentication for preventing phishing attacks," CoRR, vol. abs/0911.5230, 2009. [Online]. Available: http://arxiv.org/abs/0911.5230 11. Apr 2011

[24] Y. Oiwa, H. Watanabe, H. Takagi, Y. Ioku, and T. Hayashi, "Mutual Authentication Protocol for HTTP," Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://tools.ietf.org/html/draft-oiwa-http-mutualauth-08 11. Apr 2011

[25] T. Kivinen and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)," RFC 3526 (Proposed Standard), Internet Engineering Task Force, May 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3526.txt 11. Apr 2011

[26] S. Josefsson, "The Base16, Base32, and Base64 Data Encodings," RFC 4648 (Proposed Standard), Internet Engineering Task Force, Oct. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4648.txt 11. Apr 2011

[27] L. Strand, J. Noll, and W. Leister, "Generic security services API authentication support for the session initiation protocol," in Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT2011). Xpert Publishing Services, Mar 2011, pp. 117 – 122.

[28] S. Josefsson and N. Williams, "Using Generic Security Service Application Program Interface (GSS-API) Mechanisms in Simple Authentication and Security Layer (SASL): The GS2 Mechanism Family," RFC 5801 (Proposed Standard), Internet Engineering Task Force, Jul. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5801.txt 11. Apr 2011

 ISBN: 978-1-61208-010-9

# Paper H:
# Advancement towards secure authentication in the Session Initiation Protocol

Lars Strand, Wolfgang Leister

Submitted to
International Journal on Advances in Security
ISSN: 1942–2636

# Advancement towards secure authentication in the Session Initiation Protocol

Lars Strand

Norwegian Computing Center / University of Oslo
Oslo, Norway
Email: lars.strand@nr.no

Wolfgang Leister

Norwegian Computing Center
Oslo, Norway
Email: wolfgang.leister@nr.no

***Abstract*—The Digest Access Authentication method used in the voice over IP signaling protocol, SIP, is weak. This authentication method is the only method with mandatory support and widespread adoption in the industry. At the same time, this authentication method is vulnerable to a serious real-world attack. This poses a threat to VoIP industry installations and solutions. In this paper, we propose a solution that counters attacks on this wide-spread authentication method. We also propose a two-step migration towards a stronger authentication in SIP. We add support for a Password Authenticated Key Exchange algorithm that can function as a drop-in replacement for the widely adopted Digest Access Authentication mechanism. This new authentication mechanism adds support for mutual authentication, is considered stronger and can rely on the same shared password used by the digest authentication. A long-term solution is to replace the authentication scheme in SIP with a security abstraction layer. Two such security frameworks are introduced, discussed and evaluated: the Generic Security Services Application Program Interface and the Simple Authentication and Security Layer, which both enable SIP to transparently support and use more secure authentication methods in a unified and generic way.**

***Index Terms*—SIP, authentication, Digest Access Authentication, PAKE, SASL.**

## I. Introduction

Considering the growing market share for Voice over IP (VoIP) technologies, VoIP services need to be stable and secure, for the benefit of both users and service providers. Authentication methods are an important part of this, and need to be thoroughly examined. We base our work on a conference article [1], where we analyzed and implemented an attack on the Digest Access Authentication used in the Session Initiation Protocol (SIP), and proposed a correction to mitigate this attack. Since there is a need for better authentication methods in SIP, we add support for a security abstraction layer in SIP [2], and propose a migration strategy towards a secure authentication in SIP [3].

The importance of analyzing and improving the SIP authentication methods comes from the fact that there has been a steady increase in the number of VoIP users since 2002, as well as a decrease in the number of PSTN installations [4]. With two billion users worldwide having access to the Internet by the end of 2010 [5], the VoIP growth potential is huge. For example, at the end of 2009, 29.1 % of the private land-line phone market in Norway used VoIP.

VoIP is the emerging technology that will eventually take over from the traditional Public Switched Telephone Network (PSTN) [6] due to VoIP's improved flexibility and functionality, such as improved sound quality ("HD sound") using wideband codecs like G.722 [7], instant messaging (IM), presence, mobility support, and secure calls. VoIP also reduces maintenance and administration costs since it brings convergence to voice, video and data traffic over the IP infrastructure.

Although there exist several competing network protocols that are capable of delivering VoIP, the Session Initiation Protocol (SIP) [8] and the Real-time Transport Protocol (RTP) [9] developed by the IETF have become the *de facto* industry standard. These two protocols fulfill two different functions – SIP is used for signaling, e.g., responsible for setting up, modifying and tearing down multimedia sessions, while RTP transports the actual media stream (voice). Although the SIP protocol is flexible and rich in functionality [10], several vulnerabilities and security attacks have been found [11]–[13].

Securing a SIP-based VoIP system has proven challenging and the reasons are multi-faceted:

- The scale and complexity of the SIP protocol specification, with primary focus on functionality rather than a sound security design [14].
- SIP usage of intermediaries, expected communication between nodes with no trust at all, and its user-to-user operation make security far from trivial [8, page 232].
- A large number of threats against VoIP systems have been identified [15]. Several security mechanisms for countermeasures have been proposed, but no single security mechanism is suited to address all these security threats concerning VoIP and SIP [16], [17].
- Since the SIP and RTP protocols share the same infrastructure as traditional data networks, they also inherit the security problems of data communication.
- VoIP services have strict requirements to the network performance with respect to Quality of Service since it is a duplex communication with low tolerance for latency, packet loss and saturation. Introducing strong security mechanisms might affect network performance [18].

PSTN is a mature and stable technology providing 99.999% uptime [19], and users will expect VoIP to perform at similar

service level. But with an increasing number of VoIP users, VoIP will become a target for attackers looking for financial gain or mischief. A clear threat taxonomy is given by the "VoIP Security Alliance" [15] and is discussed by Keromytis [20].

In VoIP, authentication tries to validate the identity of the communication peers and to bind that identity to a subject (peer). It must be stressed that the user's phone is authenticated rather than the user herself. In VoIP terminology, a subject could be a User Agent (UA), such as a phone, identified by a phone-number/username and IP-address/hostname pair, denoted as an Address-of-Record (AoR). The authentication in VoIP is therefore the assurance that a communicating entity, the UA, is the one that it claims to be [21]. Equally important for the UA is to establish the identity of the communicating peer, i.e., the SIP server. If the client does not authenticate the SIP server, it might risk to communicate and send content to a hostile SIP server.

SIP supports several security services, and the RFC specification documents recommends their use. These security services can provide protection for authentication, confidentiality, and more. Yet, only one such security service is mandatory: the SIP Digest Access Authentication (DAA) method [8, page 193]. In the EUX2010sec research project [22], we revealed, in close collaboration with our project partners, that most VoIP installations only use the mandatory, Digest Access Authentication (DAA) method [23]. DAA is primarily based on the HTTP Digest Access Authentication [24], and is considered to be weak and vulnerable to serious real-world attacks [25].

One contribution of this paper is to present and analyze the seriousness of a vulnerability we presented in our earlier work – the registration attack [25]. We implement a real-world attack, and propose a solution to the DAA that will counter this vulnerability. Further, we introduce an authentication method based on the Password Authenticated Key Exchange (PAKE) [26], which provides mutual authentication based on a shared secret, and can function as a drop-in replacement of the digest authentication currently used. However, a more flexible authentication method is desired. Different security requirements may require different authentication mechanisms. Instead of adding support for many different authentication mechanisms in SIP, we introduce support for a security abstraction layer. Two such security frameworks are introduced, discussed and evaluated. The Generic Security Services Application Program Interface (GSS-API) [27] and Simple Authentication and Security Layer (SASL) [28], which both enables SIP to transparently support and use more secure authentication methods in a unified and generic way.

The rest of the paper is organized as follows: Related work and the current state of authentication in SIP is given in Section II, and show our method in Section III. We explain and implement the registration attack, and propose a solution on how to counter the attack in Section IV. In Section V we show how a modified PAKE can be used to add mutual authentication in SIP. Support for the security abstraction layers GSS-API and SASL is added, discussed and evaluated in Section VI. We present the conclusion and future work in Section VIII.

## II. State of knowledge

The DAA is currently the most common authentication mechanism for SIP. DAA is simple, but rather insecure. It is the only authentication mechanism which support in SIP is mandatory [8, Section 22]. DAA uses the MD5 hash function and a challenge-response pattern, and relies on a shared secret between client and server within a SIP domain [24]. DAA is performed during the SIP `REGISTER` handshake between the UA and the SIP server, as depicted in messages 1-3 and 6 in Fig. 10. The UA receives a nonce value from the SIP server, computes a digest hash value over the nonce, the shared secret and some other SIP header values, and send it to the SIP server. The SIP server computes the same digest hash. If both digests are identical, the UA is authenticated. The DAA is weak and vulnerable to a serious real-world attack, as described in Section IV-A.

Based on the DAA, Undrey [29] proposed a more flexible use of variables protected by the digest. His paper addresses the shortcomings of DAA and suggests to allow the server to decide which headers it requires to be included and protected by the digest computation. Unfortunately, his approach does not require specific headers fields to be included. His approach is therefore vulnerable to the same vulnerability presented and implemented in this paper.

Yang et al. [30] also conclude that DAA is weak. They argue that, since DAA is vulnerable to an off-line password guessing attacks, a more secure authentication method is required. They propose an authentication method based on Diffie-Hellman. Unfortunate, they do not discuss nor add any additional SIP headers in their new authentication scheme. Therefore, their solution is also vulnerable to the registration attack implemented in this paper.

Secure MIME (S/MIME) [31] is an authentication mechanism presented in the SIP core specification document RFC3261 [8]. S/MIME intends to achieve end-to-end authentication between UAs. The entire SIP message is encapsulated in a specific SIP message using MIME, which is signed and optionally encrypted. The receiving UA checks whether the sending UA's certificate is signed by a trusted authority. Since S/MIME depend on end-user certificates, the UAs must support multiple root certificates since no consolidated certificate authority exists. Additionally, certificate handling issues, such as revocation and renewal, complicate the use of certificates. There has been rather limited industry support for S/MIME.

Transport Layer Security (TLS) [32] support for SIP, called "Secure SIP" and denoted "SIPS", has gained some industry momentum. TLS is designed to make use of TCP to provide a protected end-to-end communication between two endpoints. The application data, here SIP, are encrypted and integrity-protected. The communicating endpoints authenticate using digital certificate, usually X.509 certificates, and thus require a PKI. TLS does not offer end-to-end confidentiality and

integrity protection of SIP messages, since the TLS connection must be terminated and initiated for each hop between intermediate SIP servers. The use of TLS also restricts SIP to use TCP as transport protocol. By using TLS, SIP relies on a lower communication layer protocol to enforce security mechanisms.

Two other authentication methods have emerged within the Internet Engineering Task Force (IETF):

1) The *P-Asserted Identity* [33] is intended to work within a trusted environment. An unprotected SIP header is appended by the UAs SIP server that informs the receiving SIP server that the identity of the UA has been checked and thus can be trusted. However, since the SIP header is sent in clear rather than protected by cryptography methods, it can easily be removed by an attacker without any of the communicating peers noticing this.
2) The *SIP Strong Identity* [34] introduces a new SIP service, the "authentication service", which signs a hash over selected SIP header values, and includes the signature as a SIP header along with a URI that points to the sender's certificate. The receiver computes the same hash and compares the results. However, using this method, only the client is authenticated and an attacker can remove these headers without implications.

Note that both "P-Asserted Identity" and "SIP Strong Identity" rely on a successful DAA authentication to be applicable. These are also applied by the SIP servers rather than the clients themselves, and are thus only providing indirect authentication of the client since the server is authenticating on behalf of the client. None of these authentication methods have seen any widespread deployment yet [14].

Palmieri et al. [35], [36], dismiss DAA as a usable authentication method, and instead craft a new authentication schema with digital signatures based on public-key encryption. But since they rely on certificates, their solution suffers under similar certification handling issues as S/MIME and TLS. They also admit that relying on public key infrastructure (PKI) is both difficult and costly to implement. Liao et at. [37], propose an improved authentication in SIP with self-signed public keys on elliptic curves. However, Liao's proposal uses smart-cards to store authentication data and rely on a trusted third party [38].

The H.323 recommendation for the VoIP protocol from the International Telecommunication Union (ITU) has failed to see widespread adoption by industry players, and is considered abandoned in favor of SIP/RTP [10]. The authentication methods in H.323, specified in H.235 [39], [40] uses well established security mechanism, like certificates, and Diffie-Hellman key exchange, to enforce authentication. Further analysis is needed to see whether the H.235 standard protects the signaling better than SIP.

The Inter-Asterisk eXchange (IAX) [41], also published by the IETF, establishes a competing protocol to SIP/RTP. IAX has several security properties that are better than SIP. By multiplexing channels over the same link and transporting both signaling and media over the same port, enforcing security

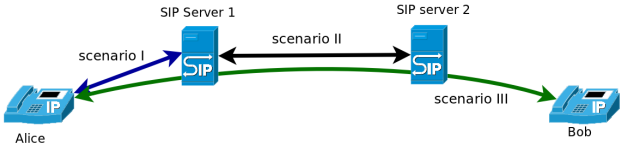



Fig. 1: Three different usage scenarios where authentication in SIP is desired.

mechanisms is easier. IAX supports two authentication methods: *1*) MD5 Message Digest authentication [42] computed over a pre-shared secret and a challenge (nonce), or *2*) using RSA public-key encryption on the challenge. In both methods, the nonce value is the only protocol parameter that is integrity protected by the authentication. Future work needs to investigate whether the IAX authentication method is adequately secure.

The SIP protocol needs an authentication mechanism that avoids the security vulnerabilities the currently used DAA has. A replacement authentication mechanism should preferably not rely on PKI, have support for strong mutual authentication, and support all three scenarios listed in the upcoming Section III.

## III. Method and Case Study

In Norway, both private companies and public authorities are migrating from PSTN to VoIP [23]. To create suitable scenarios we study the VoIP installation of three companies in Norway; one medium sized company with 150 employees, and two larger companies with 3000 and 4700 employees. We have gathered several of these VoIP configurations and setups, and replicated the installations in our test lab [43]. In these companies, most of the employees have their own VoIP phone, called a User Agent (UA). All VoIP servers run the Linux operating system with the open source telephony platform *Asterisk* [44]. We found in these configurations that the digest authentication is the only authentication method for the UAs.

In the following paragraphs, the numbers in parentheses refer to the numbers in Fig. 2, where the workflow in our method is shown.

In order to gain knowledge of the SIP protocol we use the specification documents (1), here the SIP standard. Then, we analyze VoIP network traffic going through the test lab (5). We have implemented two VoIP setups based on configurations from our industry partners ((2) and (3)). The network traffic is intercepted and saved to file using the network tool *tcpdump* (4). The network traffic is then analyzed off-line using the packet analyzer, *Wireshark* (5). An example of such an analysis is shown in Fig. 3.

As an additional input we consider threats given by [15] and given in earlier work, such as a SIP attack analyzed by Hagalisletto and Strand [25], using the protocol analyzer PROSA (6). We explain this attack in more detail in Section IV-A, and implement and execute the attack using the network tool *NetSED* (7) as shown in Fig. 7. Based on the security

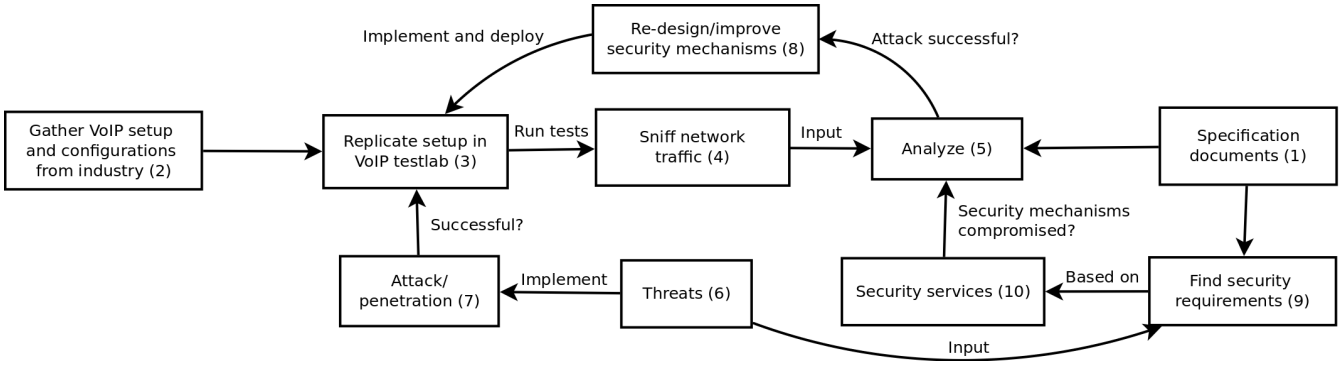



Fig. 2: Workflow for analysis of the SIP authentication method.

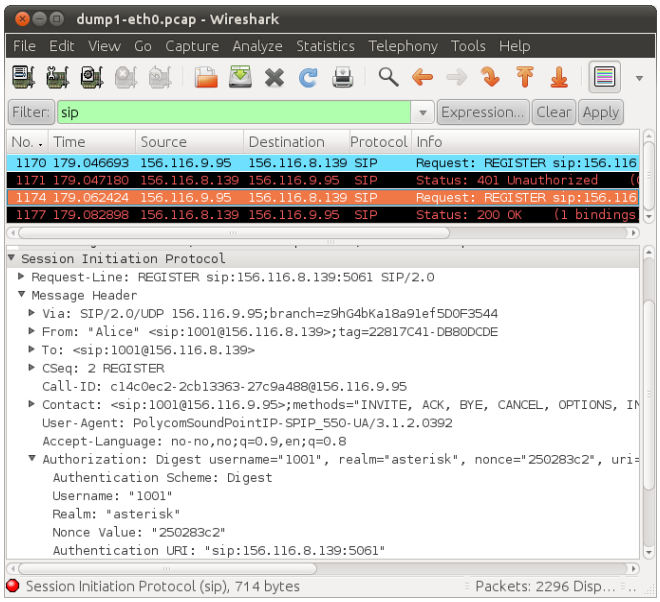



Fig. 3: Network analysis using the network tool Wireshark.

requirements (9) obtained from the SIP specification, we then checked if the authentication method (10) was compromised by the real-world attack. After careful analysis of the SIP headers we found that the SIP registration attack could be countered by a modification of the SIP authentication method (8).

We identify three scenarios where identity in SIP needs to be handled, as depicted in Fig. 1: Scenario *I* between the UA and the local SIP server; Scenario *II* between SIP servers; and Scenario *III* end-to-end.

Scenario I between the UA and the local SIP server is relevant when the UA comes online and before any outgoing calls can be placed. Then, the UA must register itself to a local SIP server. During the SIP register handshake, the server usually challenges the UA to authenticate. Before placing a call (sending a SIP `INVITE`), the UA might be challenged again by the server to authenticate. The most common authentication method used between UA and server today is the DAA.

Scenario II handles the authentication between SIP servers to achieve trust between SIP servers. It is not desirable to have SIP traffic handled by an unknown or untrusted SIP server that might have malicious intent. However, since most SIP servers today use some kind of SIP peering [6], the relationships between servers are often static and pre-defined. Therefore the identities between SIP servers are often predetermined by other security mechanisms than what are offered by SIP (like IPSec, TLS etc.).

Scenario III is about end-to-end authentication, which determines the identity of both the caller and the callee across different SIP domains. This is of particular importance and not easily attained in SIP. There is an increased threat and fear for both VoIP phishing and SPIT (Spam over Internet Telephony), that might seriously affect SIP-based VoIP services. By enforcing end-to-end authentication in SIP, these threats might be mitigated or prevented.

We list authentication mechanisms in SIP and their support in these three SIP scenarios in Table I on the facing page.

## IV. Digest Access Authentication

The SIP Digest Access Authentication (DAA) [24] is currently the most common authentication scheme for SIP. Other authentication schemes have emerged, but DAA is the only mandatory authentication scheme [8, Section 22]. DAA uses a challenge-response pattern, and relies on a shared secret between client and server. Since the DAA relies on a shared secret and is only meaningful for a specific realm, its usage is limited to Scenario I.

SIP is heavily influenced by the HTTP request-response model, where each transaction consists of a request that requires a particular response. The SIP messages are also similar in syntax and semantics to both HTTP and SMTP [10]. A SIP message consists of several headers and a body. The SIP header fields are textual, always in the format `<header_name>: <header_value>`. The header value can contain one or more parameters. We show an example SIP header message in Fig. 5.

Any SIP request can be challenged for authentication. We show an example SIP DAA handshake in Fig. 4, and refer to the protocol clauses with a number in parentheses. The initial

| Authentication mechanisms | Supported authentication scenarios | | | Supported SIP methods | |
|---|---|---|---|---|---|
| | scenario I | scenario II | scenario III | REGISTER | INVITE |
| Digest Access Authentication (DAA) | yes | no | no | yes | yes |
| Secure MIME (S/MIME) | no | no | yes | yes[a] | yes |
| Secure SIP (SIPS) using TLS | yes | yes | no[b] | yes | yes |
| P-Asserted Identity | no | yes | no | no | yes[c] |
| SIP Strong Identity | no | yes | no | no | yes[d] |
| Password Authenticated Key Exchange (PAKE) | yes | no | no | yes | yes |
| Generic Security Service API (GSS-API) | yes | yes | yes | yes | yes |
| Simple Authentication and Security Layer (SASL) | yes | yes | yes | yes | yes |

TABLE I: List of SIP authentication mechanisms and their support.

[a] Not intended to be used with SIP REGISTER, however there are no constrains in the SIP specification for using S/MIME in addition to DAA.
[b] SIPS only offers hop-by-hop confidentiality and authentication protection and thus no end-to-end protection.
[c] Does not provide an authentication method *per se*, but provide identity authentication in a trusted environment.
[d] The authentication service is handled by intermediate SIP servers to verify UAs across SIP domains.

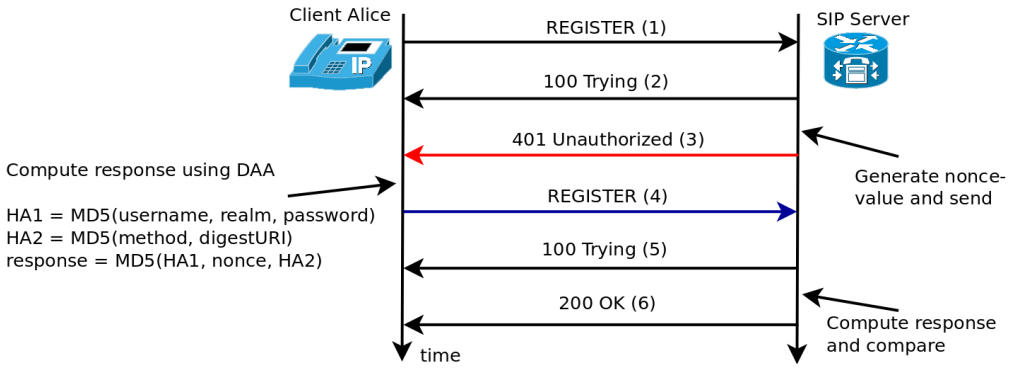



Fig. 4: The SIP Digest Access Authentication method during a SIP REGISTER transaction.

SIP REGISTER message (1) from Alice is not authorized and must be authenticated. The SIP server responds with a 401 Unauthorized status message (3) which contains a WWW-Authenticate header with details of the challenge, including a *nonce* value. The client computes the required SIP digest that is embedded in (4) as an Authorization header. The SIP server, upon receiving the Authorization header, must perform the same digest operation, and compare the result. If the results are identical, the client is authenticated, and a 200 OK message (6) is sent.

The SIP DAA is almost identical to the HTTP digest access authentication [24]. As we will show later, too few attributes (SIP header values) are included in the digest computation, thus leaving some values unprotected. Formally, the DAA is expressed as follows:

$$\begin{aligned} HA1 &= \mathrm{MD5}(A1) \\ &= \mathrm{MD5}(username : realm : password) \\ HA2 &= \mathrm{MD5}(A2) = \mathrm{MD5}(method : digestURI) \\ response &= \mathrm{MD5}(HA1 : nonce : HA2) \end{aligned}$$

In this context, *A1* is the concatenated string of Alice's *username*, the *realm* (usually a hostname or domain name) and the shared secret *password* between Alice and the server. For *A2*, the *method* is the SIP method used in the current transaction, in the above example that would be REGISTER.

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    156.116.9.95;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@156.116.9.95:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response="
    ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",algorith
    m=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Fig. 5: The only attributes included in the digest response (blue) are depicted in green.

In a REGISTER transaction the *digestURI* is set to the URI in the *To:*-field. The digest authentication *response* is the hash of the concatenated values of *HA1*, the *nonce* received from the server, and *HA2*. A SIP REGISTER message with a computed digest embedded in the Authorization header is shown in Fig. 5. DAA provides only reply protection due to the nonce value and one-way message authentication. There is no encryption of the content, nor confidentiality support, except the shared secret *password* between client and server. All messages are sent in clear. DAA only works within a local

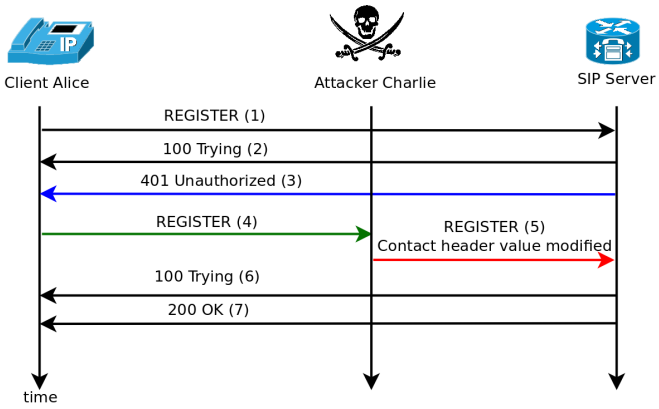



Fig. 6: The attacker Charlie can modify the `Contact` header value, and thereby have all Alice's calls redirected to him.

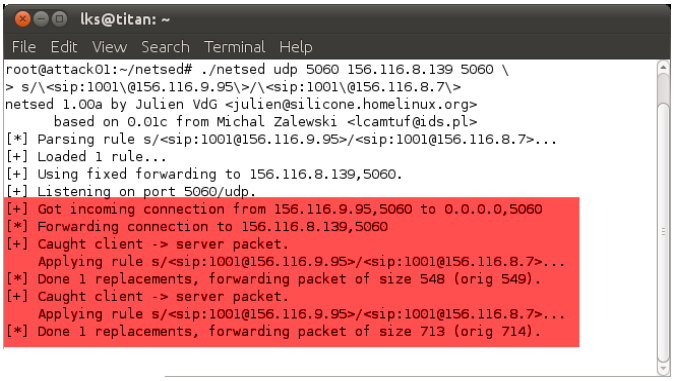



Fig. 7: The network packet stream editor NetSED modifies network packets in real time based on a regular expression (in red).

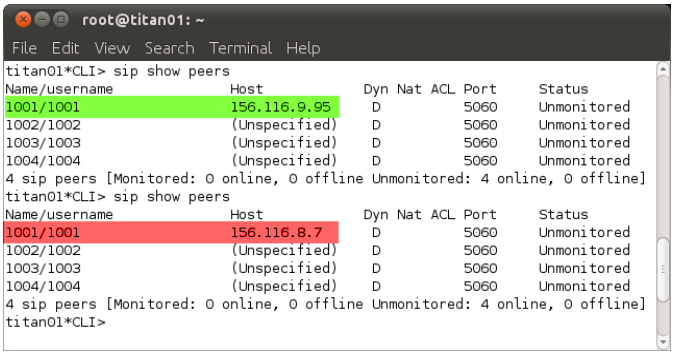



Fig. 8: Host name before (green) and after a successful attack (red), which makes Asterisk believe that Alice's phone (with number 1001) is reachable at an IP-address of the attacker's choice.

domain so cross-domain authentication is not supported, which implies that end-to-end authentication is not supported. There is no provision in the DAA for the initial secure arrangement between a client and server to establish the shared secret. However, DAA has low computation overhead compared to other methods [18].

### A. Attack on Digest Access Authentication

When a UA comes online it registers its contact point(s) to a *location service*. Contact points are the preferred methods a user can be contacted by, for example using SIP, mail, or IM. Usually, only a SIP URI contact method is present. The location service is responsible to redirect SIP requests (for VoIP calls) to the correct SIP end-point. For example, an incoming SIP call destined to `alice@CompanyA.org` does not contain information about which hostname or IP-address Alice's phone can be reached. Therefore, a SIP proxy will query the location service to receive Alice's phone's hostname or IP-address, and then forward the call to this address.

The binding of Alice's phone to a hostname or IP-address is done during the `REGISTER` transaction, as depicted in Fig. 4. Before the binding, or registration, the SIP server should ask the client to authenticate itself, as explained in the previous section. After a successful authentication, the client's hostname or IP-address is registered. A re-registration is normally done at regular intervals. This registration is repeated usually every 3-15 minutes, depending on the configuration. The client's preferred contact methods, including hostname or IP-address, is carried in the SIP header `Contact`, as depicted in Line 5 in Fig. 5. However, this SIP header value is sent in clear, and is not protected by DAA. Thus, the registration is vulnerable to a man-in-the-middle attack [25].

If an attacker modifies the hostname or IP-address in the *contactURI* header value during a `REGISTER` phrase, as depicted in Fig. 6, all requests, and hence calls, to the client will be diverted to a hostname or IP-address controlled by an attacker. Here, Alice cannot perceive that she is unreachable. An attacker can modify Alice's `REGISTER` session in real-time using NetSED [45] as depicted in Fig. 7. The SIP server (Asterisk), will not detect nor suspect that anything is wrong, and register Alice's phone number with the attackers IP address, as seen on Asterisk's terminal in Fig. 8. When Asterisk receives a call to Alice, the call will be forwarded to the attackers registered IP address. If this vulnerability is left incorrect, it constitutes a fatal flaw.

### B. Improving the Digest Access Authentication

The SIP digest authentication is weak, which is stated in both the SIP specification [8], and the digest specification [24]. Specifically, DAA only offers protection of the value in the `To` header called the `Request-URI` and the *method*, but no other SIP header values are protected.

A minor modification of DAA can counter the registration hijack attack [25], which is caused by having too few SIP header parameters protected by the digest. Since an attacker can modify and redirect all requests, we protect the header by including the `Contact` header value in the digest. By including the `Contact` value, which we name *contactURIs* in the digest, we effectively counter the registration hijack

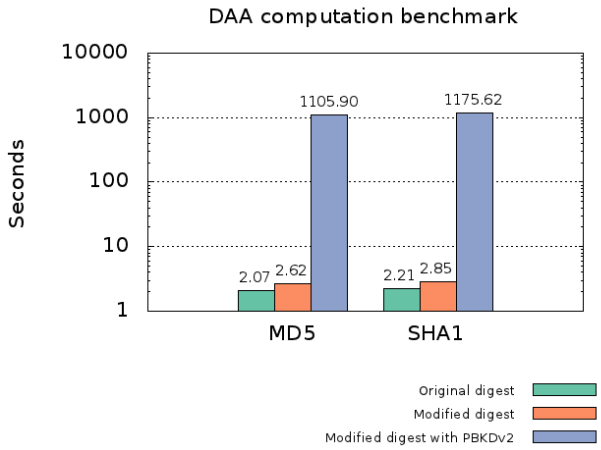



Fig. 9: The computation overhead for 100.000 iterations for original DAA, our modified DAA, and modified DAA with PBKDFv2 for both MD5 and SHA1.

attack.

We define $HA0$ with $contactURIs$. The new digest computation algorithm is as follows:

$$\begin{aligned} HA0 &= \text{MD5}(A0) = \text{MD5}(contactURIs) \\ HA1 &= \text{MD5}(A1) \\ &= \text{MD5}(username : realm : password) \\ HA2 &= \text{MD5}(A2) = \text{MD5}(method : digestURI) \\ response &= \text{MD5}(HA0 : HA1 : nonce : HA2) \end{aligned}$$

Weaknesses in the MD5 hash have been found. In particular we mention collision attacks where two different input values produce the same MD5 hash [46]. This weakness is not known to be exploitable to reveal a user's password [47]. Nonetheless, a stronger hash function, like SHA1 [48], is recommend.

We implemented and tested our modified DAA by using the Python Twisted [49] networking engine, using both MD5 and SHA1. According to our test, the computation overhead by including $HA0$ with the $ContactURIs$ is minimal, as shown in Fig. 9. The difference between the original DAA and our modified DAA with MD5 for 100.000 authentication requests on a 2.2Ghz Intel CPU, is only 0.55 seconds, a negligible amount.

A modified DAA means a modification of the SIP standard. Since the SIP standard has seen widespread industry adoption, it can be difficult to re-deploy a non-standardized SIP DAA. To prevent a modification of the SIP standard, we can use the DAA parameter `auth-param` to store our modified digest response. The parameter `auth-param` is reserved "for future use" [24, page 12], and can be a part of the `Authorization` header.

SIP devices that do not support the updated and more secure digest, can and will ignore this value, and use the original DAA for authentication. However, we cannot recommend this approach, since an attacker could remove this value and force the usage of the original standardized DAA. We would prefer to modify the DAA digest computation to force an upgrade to the new improved DAA method, instead of compromising on security.

### C. Using a Password-Based Key Derivation Function

The improved DAA, described above, is still vulnerable to dictionary-based off-line (brute-force) attacks. The attacker can intercept the message exchange, and do an exhaustive (brute-force) search for the password. To increase the cost of such search, we add support for a key derivation technique with the purpose of increasing the cost of producing the digest from the shared secret, thereby also increasing the difficulty of the brute-force attack.

We introduce support for "Password Based Key Derivation Function version 2" (PBKDF2) as specified by Kaliski [50] from RSA Laboratories. PBKDF2 works by using a key derivation function ($KDF$) on the password ($P$) and salt ($S$) to derive the key ($DK$) as:

$$DK = \text{KDF}(P, S)$$

When applied to the DAA, $P$ is the shared secret and $S$ is the nonce issued from the SIP server. The $DK$ is derived by these required steps:

1) The maximum length $dkLen$ of the derived key $DK$ is given as:
$$dkLen > (2^{32} - 1) * hLen$$
where $hLen$ denotes the length in octets of the pseudo-random function output, which is 16 for MD5 and 20 for SHA-1. We implement and benchmark both MD5 and SHA-1. However, the use of MD5 is not recommended due to weaknesses and attacks found [51].
2) We let $l$ be the number of $hLen$-octet blocks in the derived key, rounding up, and $r$ the number of octets in the last block:
$$\begin{aligned} l &= \left\lceil \frac{dklen}{hLen} \right\rceil \\ r &= dkLen - (l-1) * hLen \end{aligned}$$
3) For each block of the derived key, the function $F$ is applied. The function $F$ take password $P$, salt $S$, the iteration count $c$ and the block index to compute the block:
$$\begin{aligned} T_1 &= F(P, S, c, 1) \\ T_2 &= F(P, S, c, 2) \\ &\dots \\ T_l &= F(P, S, c, l) \end{aligned}$$
Here, function $F$ is defined as the exclusive-or sum of the first $c$ iterates of the underlying pseudo-random function $PRF$ (using HMAC-SHA1 [52]) applied to the password $P$ and the concatenation of the salt $S$ and the block index $i$:

$$F(P, S, c, i) = U_1 \oplus U_2 \oplus ... \oplus U_c$$

where:

$$\begin{aligned} U_1 &= PRF(P, S \parallel INT(i)) \\ U_2 &= PRF(P, U_1) \\ &... \\ U_c &= PRF(P, U_{c-1}) \end{aligned}$$

Here, $INT(i)$ is a four-octet encoding of the integer $i$, most significant octet first.

4) Then the blocks are concatenated and the first $dkLen$ octets is extracted to produce a derived key $DK$:

$$DK = T_1 \parallel T_2 \parallel ... \parallel T_l < 0..r - 1 >$$

5) The derived key $DK$ is returned base64-encoded [53].

We implemented and tested DAA with PBKDFv2 with $c$ iterations set to the recommended value of 1000. Input to the PBKDFv2 is the password (shared secret) and the nonce from the SIP server. The new modified DAA replaces the $password$ with the derived key $DK$ from PBKDFv2, thus a modified DAA algorithm is as follows:

$$\begin{aligned} HA0 &= \text{MD5}(A0) = \text{MD5}(contactURIs) \\ HA1 &= \text{MD5}(A1) \\ &= \text{MD5}(username : realm : DK) \\ HA2 &= \text{MD5}(A2) = \text{MD5}(method : digestURI) \\ response &= \text{MD5}(HA0 : HA1 : nonce : HA2) \end{aligned}$$

As shown in Fig. 9, the computation overhead using PBKDFv2 is significant compared to the original DAA. The result is as expected, since each DAA computation using PBKDFv2 calls a HMAC function 1000 times. This increase the cost of an exhaustive brute-force search for the shared secret used by DAA, without a significant impact of deriving individual DK used by a UA to authenticate with DAA.

While DAA with PBKDFv2 reduces much of the risk of a brute-force dictionary attack, it does not provide us with means to authenticate the SIP server.

## V. Password Authenticated Key Exchange

In the following, we discuss how to add support for a variant of PAKE denoted as "Key Agreement Method 3" (KAM3) as a cryptographic protocol [26, page 17]. PAKE has the following attractive features: *1*) PAKE provides mutual authentication between UA and the SIP server, and thus a rogue SIP server can not claim that the authentication succeed without knowing the shared password. PAKE assures the UA that the SIP server knows the UA's encrypted password. *2*) Reuse of the shared password used by DAA as the UA's credential, which enables our approach to easily replace DAA used within a local SIP domain (scenario I). *3*) PAKE offers strong protection of the shared secret if the communication is eavesdropped, that prevents brute-force attacks, including dictionary-based off-line attacks, to which the DAA is vulnerable to.

Our approach follows the work of Oiwa et al. [54]. They use KAM3 to introduce a stronger authentication in HTTP and their initial design and specification is submitted to the IETF as an Internet Draft [55]. We have adapted their approach to SIP, since SIP closely resembles HTTP in both message structure and flow, and we need to prevent the `REGISTER` hijack attack presented earlier [1].

In KAM3, the UA and the SIP server compute cryptographic keys based on the shared password. These keys are exchanged, and a shared session secret is computed based on these keys. Each peer sends then a hash value computed of the session secret and some other values, to the requesting peer. The receiving peer computes the same hash value, and compares it with the received hash value. If these are identical, the sending peer is authenticated.

PAKE supports several authentication algorithms, which differ in their underlying mathematical groups and security parameters [55]. The only mandatory supported authentication algorithm, the *iso-kam3-dl-2048-sha256*, uses the 2048-bit discrete-logarithm defined in RFC3526 [56] and the SHA-256 hash function.

### A. Initial requirements

In the following section, we let $q$ an odd prime integer defining the number of elements in $F(q)$ which is a representation of a finite group. We let $g$ the generator of a subgroup of $r$ elements in $F(q)$. The one-way hash function is denoted as $H$.

Before the authentication starts, username and password must be set and configured. We compute a weak secret $\pi$ used by the client as a one-way hash of the values $realm$, $username$ and $password$:

$$\pi = H(realm, username, password)$$

Here, $realm$ is the protection domain where SIP authentication is meaningful for a set of $username$ and $password$. The server does not need to store the shared password directly, only a specially encrypted version $J(\pi)$, where $J$ is the password verification element derivation function defined as:

$$J(\pi) = g^{\pi} \bmod q$$

### B. PAKE message exchange

We need to extend the current SIP `REGISTER` handshake by one extra round-trip of SIP messages between the UA and the SIP server. These two extra messages are depicted in blue and numbered (4) and (5) in Fig. 10. A more detailed specification is given in the following paragraphs, where the numbers refer to the protocol clauses depicted in Fig. 10.

The UA registers to a SIP *location service* (SIP server). The initial SIP `REGISTER` message (1) from the UA is not authorized, and must be authenticated. The SIP server responds with a `401 Unauthorized` status message (2), which contains a `WWW-Authenticate` header with details of the challenge, including *realm* and *algorithm*. The UA

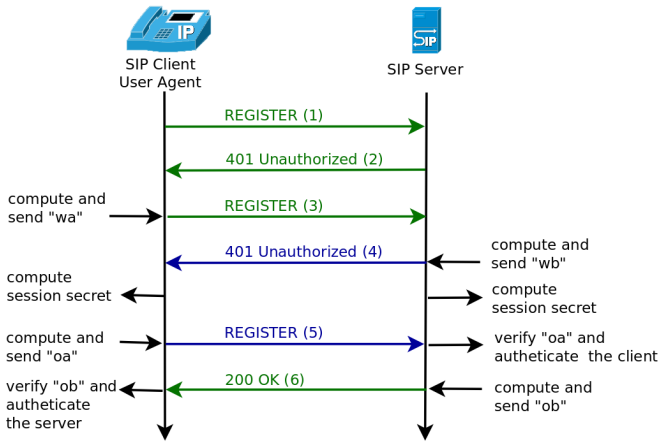



Fig. 10: SIP `REGISTER` message flow with mutual authentication security using PAKE.

constructs a cryptographic value $w_a$ generated from a random integer $s_a$:

$$w_a = g^{s_a} \bmod q$$

This value is sent in a new SIP `REGISTER` message (3) to the SIP server. The SIP server proceeds to generate and send another cryptographic value $w_b$, which is generated from $J(\pi)$, the received value $w_a$ and a random integer $s_b$:

$$w_b = (J(\pi) \times w_a^{H(1,w_a)})^{s_b} \bmod q$$

At the next step, each peer computes a session secret $z$. The UA derives $z$ based on $\pi$, $s_a$, $w_a$ and $w_b$:

$$z = w_b^{(s_a + H(2,w_a,w_b))/(s_a * H(1,w_a) + \pi) \bmod r} \bmod q$$

Likewise, the SIP server derives $z$ based on $s_b$, $w_a$ and $w_b$ using the following function:

$$z = (w_a \times g^{H(2,w_a,w_b)})^{s_b} \bmod q$$

The session secret $z$ matches only if both peers have used the secret credentials generated from the same shared secret. The above equations are directly derived from the PAKE HTTP authentication specifications [55]. The next step is to validate the value of $z$ at the communicating peer.

The UA sends a third SIP `REGISTER` message (5) and includes the value $o_a$ which is a hash value computed as:

$$o_a = H(4, w_a, w_b, z, contactURIs)$$

Here, $contactURIs$ is the value of the UA's `Contact` SIP header value. This value is integrity-protected to prevent register hijacking attacks as presented in [1]. The SIP server, upon receipt of $o_a$, performs the same hash operation, and compares the results. If these results are identical, the UA is authenticated. The SIP server then sends a final message (6), with the value $o_b$ computed as:

$$o_b = H(3, w_a, w_b, z, contactURIs)$$

When the UA receives $o_b$, it verifies this value by computing its hash value. If the results are identical, the SIP server is authenticated to the UA. After a complete message exchange, the UA is authenticated to the SIP server, and the SIP server has been authenticated to the UA.

### C. SIP message support for PAKE

We embed the cryptographic values derived in the previous section as base64-encoded [53] SIP header values. We re-use the SIP DAA headers to carry PAKE authentication data, so that PAKE can be used as a drop-in replacement for DAA. A SIP `REGISTER` message with a DAA `Authorization` header is depicted in Fig. 15. Again, we refer to the protocol clauses with a number in parentheses as depicted in Fig. 10.

The UA first sends a SIP `REGISTER` without any authentication credentials (1). The SIP server responds with a `401 Unauthorized` status message (2), which contains a `WWW-Authenticate` header with header values *realm* and *algorithm*:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: Mutual realm="asterisk",
  algorithm="iso-kam3-dl-2048-sha256"
```

The UA then computes $w_a$ and sends it to the SIP server using a new SIP `REGISTER` message (3), with the required values embedded in the `Authorization` header:

```
SIP/2.0 REGISTER
Authorization: Mutual user="alice",
  algorithm="iso-kam3-dl-2048-sha256",
  wa="Q29tcHV0ZWQgd2E...ljaCBcyBsb25nCg=="
```

The next required values in the authentication mechanism $w_b$, $o_a$ and $o_b$ are embedded and sent using these two SIP headers.

## VI. Security Programming Interfaces

A modified PAKE authentication can more easily replace the current digest (DAA) authentication used in SIP, since they both rely on a shared secret and use the same SIP headers. PAKE also introduces a stronger authentication than DAA. However, a more flexible authentication mechanism is desired. Different VoIP scenarios require different security requirements, and the communicating peers should be able to negotiate the best possible authentication mechanism supported.

Instead of adding numerous different authentication mechanisms to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. The industry might also be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Adding support to a security programming interface will require only small changes to the SIP standard.

A security programming interface provides a generic interface for application layer protocols like SIP, with a layer of abstraction for different security services like authentication,

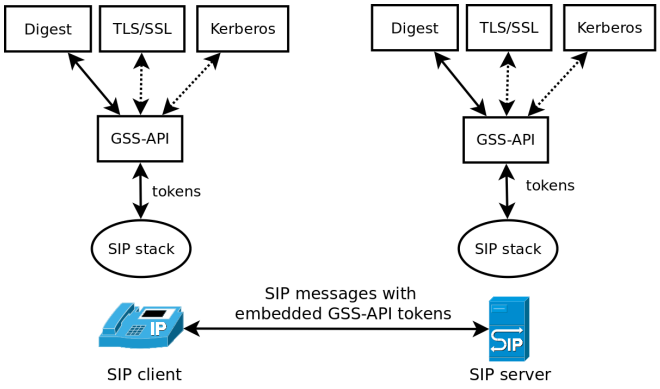



Fig. 11: The GSS-API interface in SIP.

integrity or confidentiality. Using a security programming interface, an application does not need to support or implement every authentication method, but use the provided security API [57]. Support for two security programming interfaces, the "Generic Security Services API" (GSS-API) and "Simple Authentication and Security Layer" (SASL), are added to SIP. Both are developed by the IETF, have been extensively tested, and are now classified as mature standards by the IETF.

### A. Generic Security Services API

The GSS-API is not a communication protocol in itself, but relies on the application to encapsulate, send, and extract data messages called "tokens" between the client and server. The tokens' content are opaque from the viewpoint of the calling application, and contain authentication data, or, once the authentication is complete, portion of data that the client and server want to sign or encrypt. The tokens are passed through the GSS-API to a range of underlying security mechanisms, ranging from secret-key cryptography, like Kerberos [58], to public-key cryptography, like the Simple Public-Key GSS-API Mechanism (SPKM) [59]. The GSS-API interface to SIP is depicted in Fig. 11. For an application, the use of the GSS-API becomes a standard interface to request authentication, integrity, and confidentiality services in a uniform way. However, GSS-API does not provide credentials needed by the underlying security mechanisms. Both server and client must acquire their respective credentials before GSS-API functions are called.

To establish peer entity authentication, a security context is initialized and established. After the security context has been established, additional messages can be exchanged, that are integrity and, optionally, confidentially protected. To initiate and manage a security context, the peers use the *context-level* GSS-API calls. The client calls `GSS_Init_sec_context()` that produces a "output_token" that is passed to the server. The server then calls `GSS_Accept_sec_context()` with the received token as input. Depending on the underlying security mechanism, additional token exchanges may be required in the course of context establishment. If so, `GSS_S_CONTINUE_NEEDED` status is set and additional tokens are passed between the client and server until a security context is established, as depicted in Fig. 14.

After a security context has been established, *per-message* GSS-API calls can be used to protect a message by adding a Message Integrity Code (MIC) with `GSS_GetMIC()` and verifying the message with `GSS_VerifyMIC()`. To encrypt and decrypt messages, the peers can use `GSS_Wrap()` and `GSS_Unwrap()`. Thus, two different token types exist:

1) *Context-level tokens* are used when a context is established.
2) *Per-message tokens* are used after a context has been established, and are used to integrity or confidentiality protect data.

In addition to send and receive tokens, the application is responsible to distinguish between token types. This is necessary because different tokens types are sent by the application to different GSS-API functions. But since the tokens are opaque to the application, the application must use a method to distinguish between the token types. In our solution, we use explicit tagging of the token type that accompanies the token message.

*1) SIP message support for GSS-API:* When a SIP client is authenticated to a server using DAA, the authentication handshake data is encapsulated in the `WWW-Authenticate` header from server to client, and the `Authorization` header from client to server. We reuse these headers for GSS-API support, and instead of encapsulate DAA data, we send the GSS-API tokens. An example of both DAA `Authorization` header and the new `Authorization` header with GSS-API data is depicted in Fig. 12.

During the initialization of a security context it is necessary to identify the underlying security mechanism to be used. The caller initiating the context indicates at the start of the token the security (authentication) mechanism to be used. The security mechanism is denoted by a unique Object Identifier (OID). For example, the OID for the Kerberos V5 mechanism is `1.2.840.113554.1.2.2`. However, the initiating peer cannot know which security mechanism the receiving peer supports. If an unsupported "mech_type" is requested, the authentication fails. The GSS-API standard resolves this by recommending to manually standardizing on a fixed "mech_type" within a domain. Since SIP addresses are designed to be global [6], and not confined to a local domain, a GSS-API *negotiation* mechanism is required. The SPNEGO is such a GSS-API negotiation mechanism.

The "Simple and Protected GSSAPI Negotiation Mechanism" (SPNEGO [60] is a pseudo security mechanism that enables peers to negotiate a common set of one or more GSS-API security mechanisms. The GSS-API stack with SPNEGO is shown in Fig. 13. The client sends a prioritized list of supported authentication mechanisms to the server. The server then chooses the preferred authentication method based on the received list from the client. The client initiates `GSS_Init_sec_context()` as with an ordinary GSS-API security mechanism, but requests that SPNEGO is used as

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response="ccbde1c3c129b3dcaa14a4d5e35
    519d7",uri="sip:CompanyA",algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP 192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: GSSAPI ttype="context"
    token="0401000B06092A864886F712010202DACD139402AAF44350CDE32"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Fig. 12: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying GSS-API data to the right.

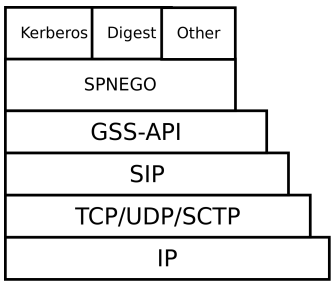



Fig. 13: The GSS-API protocol stack with the SPNEGO negotiation mechanism and underlying security mechanisms.

the underlying GSS-API mechanism ("mech_type"). The SPNEGO handshake between client and server is communicated by sending and receiving tokens. After the handshake, the client and server initiate and set up a security context (authentication) using the agreed GSS-API security mechanism.

*2) SIP authentication using GSS-API and SPNEGO:* When discussing PAKE authentication earlier, we added one round-trip of SIP messages between the UA and the SIP server. When using GSS-API with the SPNEGO, the number of SIP messages going back and forth depends on the underlying authentication mechanism. We therefore extend the SIP `REGISTER` handshake with an arbitrary number of round-trips, until the underlying authentication mechanism has completed communication.

In the following paragraphs, the numbers in parentheses refer to the numbers in Fig. 14. When a client comes online and registers itself to a "location service" (SIP server), it does so by sending a SIP `REGISTER` message (1). We define the token type in the variable `ttype`. In the following messages, the `ttype` is set to "context" indicating that these tokens are *context-level tokens*. The first message (1) does not contain any `Authorization` header. The server responds with an empty `WWW-Authenticate` header (3):

```
REGISTER SIP/2.0
WWW-Authenticate: GSSAPI ttype="context"
  token=""
```

The client then calls `GSS_Init_sec_context()` with SPNEGO as underlying GSS-API mechanism to negotiate a common authentication mechanism (4). The GSS-API "mech_type" is set to SPNEGOs OID `1.3.6.1.5.5.2`. The token data might be in binary format, depending on the security mechanism used. Since the SIP headers are in ASCII string format, the token data is base64 encoded:

```
SIP/2.0 401 Unauthorized
Authorization: GSSAPI ttype="context"
  token="0401000B06092A864886F7120..."
```

The server retrieves the GSS-API data, the token, and passes this to the SPNEGO GSS-API mechanism. In this first initial token, the client embeds authentication data for its first preferred authentication mechanism. This way, should the server accept the clients preferred mechanism, we avoid an extra SIP message round trip. If the client's preferred method was accepted by the server, the server passes the relevant authentication data to the selected authentication mechanism in a `401` SIP message (5). The selected authentication method continues to pass tokens between client and server as many times as necessary to complete the authentication (6-7-N) and establish a security context. Once the security context is established, it sends a `200 OK` SIP message (N+2). Should the server have some last GSS-API data to be communicated to the client to complete the security context, it can be carried in a `WWW-Authenticate` header embedded in the `200 OK` message:

```
SIP/2.0 200 OK
WWW-Authenticate: GSSAPI ttype="context"
  token="dd02c7c2232759874e1c20558701..."
```

If the client's preferred mechanism is not the server's most preferred mechanism, the server outputs a negotiation token and sends it to the client embedded in a new `401` SIP message (5). The client processes the received SIP message and passes the authentication data to the correct authentication mechanism. The GSS-API then continues as described in the previous paragraph.

### *B. Simple Authentication and Security Layer*

The Simple Authentication and Security Layer (SASL), defined in RFC4422 [28], provides an interface for authentication and an authentication negotiation mechanism. It provide the same security services as GSS-API and is implemented and used in several popular communications protocols applications

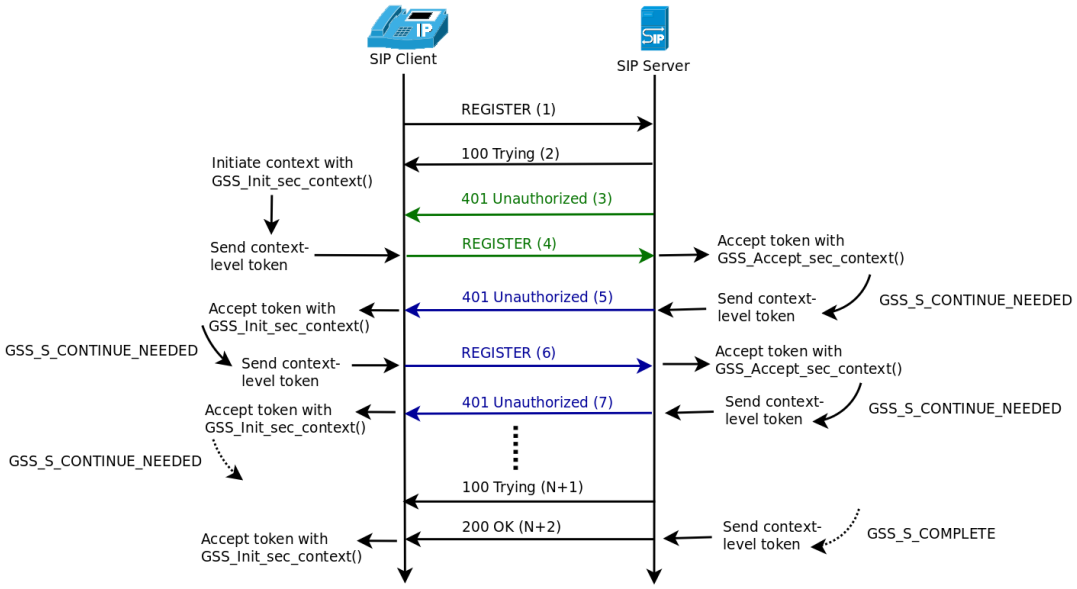



Fig. 14: SIP `REGISTER` message flow with GSS-API security context establishment (authentication).

like IMAP, SMTP and LDAP[1].

As with GSS-API, the SASL framework does not provide authentication mechanisms in itself, but supports different underlying authentication mechanisms through a standardized interface[2]. SASL does not provide a transport layer and thus relies on the application, to encapsulate, send and extract SASL messages between client and server, which in our case is the SIP protocol. The SASL messages sent between client and server contain authentication data, and are opaque from the viewpoint of the calling application (SIP). The application only needs to add support to a SASL software library implementation, and thus have support to a range of underlying authentication mechanisms the library supports.

While the GSS-API is primarily intended for use with applications, SASL is used in, and intended for, communication protocols. The functionalities offered by the GSS-API and SASL are alike, but the SASL specification is more high-level, and allows more freedom in implementing the SASL requirements. SASL also supports more underlying security mechanisms than the GSS-API. By using the "GS2" mechanism family, the GSS-API can be used as an underlying security mechanism in SASL. However, the GSS-API negotiation mechanism SPNEGO cannot be used due to security concerns [61, Section 14].

*1) SIP message support for SASL:* In SASL terminology, the description on how to encapsulate SASL negotiation and SASL messages for a given protocol, is called a "SASL profile". The SIP protocol stack with SASL is shown in Fig. 16. We create a SASL profile for SIP by reusing the `WWW-Authenticate` and `Authorization` SIP headers used by the digest authentication, shown earlier. Instead of encapsulating DAA data, we embed SASL messages, as depicted in Fig. 15.

As with the GSS-API, we need to increase the number of messages going back and forth between the SIP client and server. The number of messages depends on the required message exchange needed by the used underlying authentication mechanism.

In the following paragraphs, the numbers in parentheses refer to the SIP message numbers in Fig. 14. The SASL specification only outlines a very high-level method of how the server should advertise its supported mechanisms to the client. We implement the mechanism negotiation in the first three messages in the SIP `REGISTER` handshake (1-4). The UA starts by requesting authentication from the SIP server, with no `Authorization` header (1). The SIP server responds with a `401 Unauthorized` SIP message (3), with the supported and available mechanisms embedded in the `WWW-Authenticate` header:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: SASL
  negotiate="DIGEST-MD5 NTLM GS2-KRB5"
```

The client selects the best mechanism from the received list that it supports and sends a new SIP `REGISTER` message (4). This message includes an `Authorization` header requesting authentication with "GS2-KRB5" as the preferred mechanism. The initial authentication data is embedded base64 encoded to the *data* parameter:

```
SIP/2.0 REGISTER
```

[1]The Carnegie Mellon University's implementation: http://asg.web.cmu.edu/sasl/ and the GNU SASL library: http://www.gnu.org/software/gsasl/ are two popular and freely available SASL libraries.

[2]A list of registered SASL mechanisms is maintained by IANA: http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: Digest
    username="alice",realm="asterisk",nonce="3b7a1395",response=
    "ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",
    algorithm=MD5
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

```
1.  REGISTER sip:CompanyA SIP/2.0
2.  Via: SIP/2.0/UDP
    192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
3.  From: Alice <sip:alice@CompanyA>;tag=1234648905
4.  To: Alice <sip:alice@CompanyA>
5.  Contact: "Alice" <sip:alice@192.168.1.102:5060>
6.  Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
7.  CSeq: 19481 REGISTER
8.  User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
9.  Authorization: SASL mechanism="DIGEST-MD5"
    data="YAzgusSGGeRFGw9nfUvOAxcEDzZCBmKY1H2ElnegaccBcx3DUSkGNW
    Y4qfiSwcXwjLtoqW0eBNog7ixHN"
10. Max-Forwards: 70
11. Expires: 3600
12. Content-Length: 0
```

Fig. 15: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying SASL data to the right.

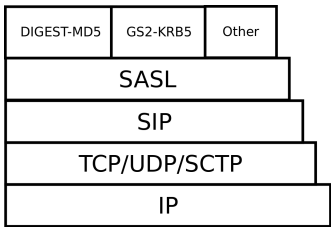



Fig. 16: The SIP SASL stack is similar to the SIP GSS-API stack with underlying security mechanisms.

```
Authorization: SASL mechanism="GS2-KRB5",
  data="SUZZT1VDQU5SR...JUPVVQU5FUkQK="
```

The server retrieves the SASL data, and passes the message to the SASL library which handles the authentication. The selected authentication method continues to pass SASL messages between client and server as many times as necessary to complete the authentication (messages 5-6 are repeated). Once the authentication is complete, the SIP server sends a `200 OK` SIP message. Should the server have some last SASL data to be communicated to the client to complete the authentication, it can be carried in a `WWW-Authenticate` header embedded in the `200 OK` message (N+2):

```
SIP/2.0 200 OK
WWW-Authenticate: SASL mechanism="GS2-KRB5",
  data="TFoG9rP56zrVH...YaAOndwPew6NdxKr"
```

As soon as the `200 OK` message is received and processed, the client is authenticated to the SIP server. Since the mechanism negotiation is not integrity-protected, the UA is vulnerable to a "down-grade" attack. An attacker can intercept and modify the negotiation messages so that the least favorable authentication method is used.

## VII. Migration towards a secure authentication

We propose a two step migration towards a secure authentication in SIP. While our attack on the DAA could be countered by including the SIP header value `ContactURI` in the digest, it did not provide any protection against off-line dictionary attacks. We implemented and showed that the use of "Password-Based Key Derivation Function version 2" (PBKDFv2) on the shared secret to make dictionary- and brute-force attacks significant harder to execute on the DAA. However, this method does not authenticate the SIP server, only the client.

Our *first* migration step suggests to replace the DAA with a modified "Password Authenticated Key Exchange" (PAKE) that is more secure than the DAA, introduce mutual authentication and re-use the shared secret used by the DAA. These properties make PAKE a preferred mechanism over the DAA with PBKDFv2. However, using PAKE does not leave any room for future extensions nor modification of authentication in SIP once implemented.

The *second* migration step takes the limitations of the previous mechanisms into consideration, and is seen as the most viable way of solution. The last authentication method introduces support for a GSS-API/SASL security layer which enables SIP to transparently support and use more secure authentication methods in a unified and generic way without the need for later changes to the SIP protocol specification.

Support for the GSS-API/SASL security layer in SIP, have the following attractive properties that address real-world security concerns:

1) Mature, stable and industry adopted standards: The industry might be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Both the GSS-API and SASL are stable, mature standards that have been adopted by the industry. Thus, implementing GSS-API or SASL should not be considered a drastic nor radical change by the relevant standardizing bodies (the IETF) nor the VoIP industry.
2) Minimal changes to the SIP standard required: The authentication data re-use the existing SIP DAA headers, so minimal changes to the SIP *message contents* are required. Also, minimal changes are required to the SIP *message flow*, since the authentication handshake is just extended with a number of required SIP message round-trips to complete the new authentication exchange.
3) Flexible and adaptive to new requirements and future changes: Instead of adding numerous different authen-

tication mechanisms to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. By adding support to a security layer in SIP, adding new or modifying existing underlying authentication mechanisms does not need any redesign of the SIP specification standard. In this case, only the GSS-API/SASL software library needs to be updated. Thus, authentication in SIP becomes adaptive to future extensions.

## VIII. Conclusion and Future Work

We have seen that the widely deployed authentication method DAA in SIP is weak and vulnerable to attacks. Moreover, we have confirmed and verified that the attack analyzed earlier [25] can be performed on the SIP protocol in real-time. We have examined this authentication method, and proposed a solution to counter the serious registration attack. By including more SIP header parameters in the authentication digest this attack can be countered.

The original SIP designers focused on functionality and compliance at the cost of security. A more thorough investigation of the SIP DAA in the design phase would have revealed the vulnerability presented here, and the vulnerability could have been prevented early on. Our remedy presented here solves an serious problem with the DAA.

Therefore, we wanted to replace DAA with support for a better, more robust authentication scheme. We have added support for a improved authentication mechanism that can easily replace DAA based on a modified PAKE algorithm. This new authentication mechanism adds support for mutual authentication and is more secure than DAA. We have also shown that the modified PAKE authentication can easily function as a drop-in replacement for DAA. However, a more flexible authentication mechanism is desired in the long-term. Different VoIP installations have different security requirements that may require different security services.

We introduced a security programming interface, which provides a security abstraction layer. This abstraction layer adds support to a range of underlying authentication mechanism in a unified way. As long as SIP supports the security layer, new authentication mechanisms can be added later, without requiring any change to the SIP protocol. Support for two security layers where added, the GSS-API and SASL. We recommend the use of SASL, as SASL has more industry deployment, have support for more underlying authentication mechanisms, and are specifically designed for communications protocols.

We envisage a two-step migration towards a stronger authentication scheme in SIP. First, the modified PAKE authentication is implemented and deployed. Second, the long-term solution is to deploy SASL with support for a range of underlying authentication mechanisms.

Future work will look into implementing a proof of concept for PAKE-enabled UA and SIP server, including overhead evaluation benchmarks for the new authentication algorithm. We also plan to evaluate different SASL security mechanism and their implications for SIP, and decide which authentication mechanisms should be mandatorily supported through SASL.

We plan to co-operate with the IETF and the "kitten" WG to further elaborate GSS-API and SASL support for SIP. We hope our work will gain acceptance and industrial deployment, so that the previously mentioned security attacks can be countered.

## Acknowledgment

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054).

## References

[1] L. Strand and W. Leister, "Improving SIP authentication," in *Proceedings of the Tenth International Conference on Networking (ICN2011)*. Xpert Publishing Services, Jan 2011, pp. 164 – 169.

[2] L. Strand, J. Noll, and W. Leister, "Generic security services API authentication support for the session initiation protocol," in *Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT2011)*. Xpert Publishing Services, Mar 2011, pp. 117 – 122.

[3] L. Strand, W. Leister, and A. Duric, "Migration towards a more secure authentication in the session initiation protocol," in *The Fifth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE2011)*. Xpert Publishing Services, Aug 2011.

[4] "Det norske markedet for elektroniske kommunikasjonstjenester 2009 (The Norwegian market for electronic communication services 2009)," Post- og teletilsynet (The Norwegian Post and Telecommunications Authority), 2010. [Online]. Available: http://www.npt.no/ikbViewer/Content/119027/Ekomrapport_2009_.pdf [Accessed: 1. Jul 2011]

[5] Telecommunication Development Sector (ITU-D), "The world in 2010," ITU-T ICT facts and figures, 2010.

[6] L. Strand and W. Leister, "A Survey of SIP Peering," in *NATO ASI - Architects of secure Networks (ASIGE10)*, May 2010.

[7] International Telecommunication Union, "7 kHz Audio-Coding within 64 kbits/s," ITU-T Recommendation G.722, 1993.

[8] 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), Internet Engineering Task Force, Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026, 6141. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt [Accessed: 1. Jul 2011]

[9] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (Standard), Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761, 6051, 6222. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt [Accessed: 1. Jul 2011]

[10] H. Sinnreich and A. B. Johnston, *Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol*, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006.

[11] H. Dwivedi, *Hacking VoIP: Protocols, Attacks, and Countermeasures*, 1st ed. No Starch Press, Mar. 2009.

[12] D. Endler and M. Collier, *Hacking Exposed VoIP: Voice over IP Security Secrets and Solutions*. McGraw-Hill Osborne Media, November 2006.

[13] A. M. Hagalisletto and L. Strand, "Designing attacks on SIP call set-up," *International Journal of Applied Cryptography*, vol. 2, no. 1, pp. 13–22(10), July 2010.

[14] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, *SIP Security*. WileyBlackwell, Mar. 2009.

[15] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf [Accessed: 1. Nov 2011]

[16] D. York, *Seven Deadliest Unified Communications Attacks*. Syngress, Apr. 2010.

[17] P. Park, *Voice over IP Security*. Cisco Press, Sep. 2008.

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

[19] D. Kuhn, "Sources of failure in the public switched telephone network," *Computer*, vol. 30, pp. 31–36, 1997.

[20] A. D. Keromytis, *Voice over IP Security - A Comprehensive Survey of Vulnerabilities and Academic Research*, 1st ed. New York, NY: Springer New York, 2011, vol. 1.

[21] International Telecommunication Union (ITU), "Security Architecture For Open Systems Interconnection (OSI)," The International Telegraph and Telephone Consultative Comittee (CCITT), X.800 Standard X.800, 1991.

[22] "Research project: EUX2010SEC – Enterprise Unified Exchange Security"." [Online]. Available: http://www.nr.no/pages/dart/project_flyer_eux2010sec [Accessed: 1. Nov 2011]

[23] L. Fritsch, A.-K. Groven, L. Strand, W. Leister, and A. M. Hagalisletto, "A Holistic Approach to Open Source VoIP Security: Results from the EUX2010SEC Project," *International Journal on Advances in Security*, no. 2&3, pp. 129–141, 2009.

[24] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617 (Draft Standard), Internet Engineering Task Force, Jun. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt [Accessed: 1. Jul 2011]

[25] A. M. Hagalisletto and L. Strand, "Formal modeling of authentication in SIP registration," in *Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08*. IEEE Computer Society, August 2008, pp. 16–21.

[26] International Organization for Standardization and ISO, "ISO/IEC 11770-4:2006: Information technology – Security techniques – Key management – Part 4: Mechanisms based on weak secrets," 2006.

[27] J. Linn, "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743 (Proposed Standard), Internet Engineering Task Force, Jan. 2000, updated by RFC 5554. [Online]. Available: http://www.ietf.org/rfc/rfc2743.txt [Accessed: 1. Jul 2011]

[28] A. Melnikov and K. Zeilenga, "Simple Authentication and Security Layer (SASL)," RFC 4422 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4422.txt [Accessed: 1. Jul 2011]

[29] J. Undery, "Ieft draft: SIP authentication: SIP digest access authentication," IETF, Tech. Rep., Jul. 2001.

[30] C. Yang, R. Wang, and W. Liu, "Secure authentication scheme for session initiation protocol," *Computers & Security*, vol. 24, no. 5, pp. 381–386, Aug. 2005.

[31] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC 3853 (Proposed Standard), Internet Engineering Task Force, Jul. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3853.txt [Accessed: 1. Jul 2011]

[32] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," RFC 5246 (Proposed Standard), Internet Engineering Task Force, Aug. 2008, updated by RFCs 5746, 5878, 6176. [Online]. Available: http://www.ietf.org/rfc/rfc5246.txt [Accessed: 1. Jul 2011]

[33] C. Jennings, J. Peterson, and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks," RFC 3325 (Informational), Internet Engineering Task Force, Nov. 2002, updated by RFC 5876. [Online]. Available: http://www.ietf.org/rfc/rfc3325.txt [Accessed: 1. Jul 2011]

[34] J. Peterson and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)," RFC 4474 (Proposed Standard), Internet Engineering Task Force, Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4474.txt [Accessed: 1. Jul 2011]

[35] F. Palmieri, "Improving authentication in voice over IP infrastructures," in *Advances in Computer, Information, and Systems Sciences, and Engineering*, K. Elleithy, T. Sobh, A. Mahmood, M. Iskander, and M. Karim, Eds. Springer Netherlands, 2006, pp. 289 – 296.

[36] F. Palmieri and U. Fiore, "Providing true end-to-end security in converged voice over IP infrastructures," *Computers & Security*, vol. 28, no. 6, pp. 433–449, Sep. 2009.

[37] Y. Liao and S. Wang, "A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves," *Computer Communications*, vol. 33, no. 3, pp. 372–380, Feb. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366409002631 [Accessed: 1. Jul 2011]

[38] Y. Liao, "Secure password authenticated key exchange protocols for various enviroments," Ph.D. dissertation, Tatung University, Dec. 2009.

[39] International Telecommunication Union, "H.323 security: Framework for security in H-series (H.323 and other H.245-based) multimedia systems," ITU-T Recommendation H.235.0, 2005.

[40] ——, "H.323 security: Framework for secure authentication in RAS using weak shared secrets," ITU-T Recommendation H.235.5, 2005.

[41] M. Spencer, B. Capouch, E. Guy, F. Miller, and K. Shumard, "IAX: Inter-Asterisk eXchange Version 2," RFC 5456 (Informational), Internet Engineering Task Force, Feb. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5456.txt [Accessed: 1. Jul 2011]

[42] R. Rivest, "The MD5 Message-Digest Algorithm," RFC 1321 (Informational), Internet Engineering Task Force, Apr. 1992, updated by RFC 6151. [Online]. Available: http://www.ietf.org/rfc/rfc1321.txt [Accessed: 1. Jul 2011]

[43] L. Strand, "VoIP lab as a research tool in the EUX2010SEC project," Norwegian Computing Center, Department of Applied Research in Information Technology, Tech. Rep. DART/08/10, April 2010.

[44] "Asterisk: The Open Source PBX & Telephony Platform." [Online]. Available: http://www.asterisk.org/ [Accessed: 1. Nov 2011]

[45] "NetSED: The network packet stream editor." [Online]. Available: http://silicone.homelinux.org/projects/netsed/ [Accessed: 1. Nov 2011]

[46] X. Wang and H. Yu, "How to break MD5 and other hash functions," *IN EUROCRYPT*, vol. 3494, 2005.

[47] P. Hawkes, M. Paddon, and G. G. Rose, "Musings on the wang et al. md5 collision," Cryptology ePrint Archive, Report 2004/64, 2004.

[48] D. Eastlake 3rd and P. Jones, "US Secure Hash Algorithm 1 (SHA1)," RFC 3174 (Informational), Internet Engineering Task Force, Sep. 2001, updated by RFCs 4634, 6234. [Online]. Available: http://www.ietf.org/rfc/rfc3174.txt [Accessed: 1. Jul 2011]

[49] "Twisted Matrix Labs." [Online]. Available: http://twistedmatrix.com [Accessed: 1. Nov 2011]

[50] B. Kaliski, "PKCS #5: Password-Based Cryptography Specification Version 2.0," RFC 2898 (Informational), Internet Engineering Task Force, Sep. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2898.txt [Accessed: 1. Jul 2011]

[51] S. Turner and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms," RFC 6151 (Informational), Internet Engineering Task Force, Mar. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6151.txt [Accessed: 1. Jul 2011]

[52] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," RFC 2104 (Informational), Internet Engineering Task Force, Feb. 1997, updated by RFC 6151. [Online]. Available: http://www.ietf.org/rfc/rfc2104.txt [Accessed: 1. Jul 2011]

[53] S. Josefsson, "The Base16, Base32, and Base64 Data Encodings," RFC 4648 (Proposed Standard), Internet Engineering Task Force, Oct. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4648.txt [Accessed: 1. Jul 2011]

[54] Y. Oiwa, H. Watanabe, and H. Takagi, "Pake-based mutual http authentication for preventing phishing attacks," *CoRR*, vol. abs/0911.5230, 2009. [Online]. Available: http://arxiv.org/abs/0911.5230 [Accessed: 1. Jul 2011]

[55] Y. Oiwa, H. Watanabe, H. Takagi, Y. Ioku, and T. Hayashi, "Mutual Authentication Protocol for HTTP," Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://tools.ietf.org/html/draft-oiwa-http-mutualauth-08 [Accessed: 1. Jul 2011]

[56] T. Kivinen and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)," RFC 3526 (Proposed Standard), Internet Engineering Task Force, May 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3526.txt [Accessed: 1. Jul 2011]

[57] D. Todorov, *Mechanics of User Identification and Authentication: Fundamentals of Identity Management*, 1st ed. Auerbach Publication, Jun. 2007.

[58] L. Zhu, K. Jaganathan, and S. Hartman, "The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2," RFC 4121 (Proposed Standard), Internet Engineering Task Force, Jul. 2005, updated by RFC 6112. [Online]. Available: http://www.ietf.org/rfc/rfc4121.txt [Accessed: 1. Jul 2011]

[59] C. Adams, "The Simple Public-Key GSS-API Mechanism (SPKM)," RFC 2025 (Proposed Standard), Internet Engineering Task Force, Oct. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc2025.txt [Accessed: 1. Jul 2011]

[60] L. Zhu, P. Leach, K. Jaganathan, and W. Ingersoll, "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism," RFC 4178 (Proposed Standard), Internet Engineering Task Force, Oct. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4178.txt [Accessed: 1. Jul 2011]

[61] S. Josefsson and N. Williams, "Using Generic Security Service Application Program Interface (GSS-API) Mechanisms in Simple Authentication and Security Layer (SASL): The GS2 Mechanism Family," RFC 5801 (Proposed Standard), Internet Engineering Task Force, Jul. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5801.txt [Accessed: 1. Jul 2011]

# PART III:

# Appendices

# Appendix i

# List of terms and acronyms

A list of SIP relevant acronyms and definitions can be found in Section 6 in RFC3261 [77]. An extensive list telephony and telecom definitions can be found in *Newton's Telecommunications Dictionary* [53].

| Acronym | Description |
|---|---|
| AoR | Address-of-Record. An AoR represents a "long-term identity" of a user on the form a simple SIP (or SIPS) URI. Example `sip:alice@example.com`. An location service within the local domain must map the AoR URI to one or more Contact URI(s) where the user might be reachable (for example `sip:alice@192.168.1.123:5060`). |
| B2BUA | Back-to-Back User Agent. A SIP server that operates between both end points of a SIP dialog. It terminates the SIP dialog and generates a new dialog between both UAs. The B2BUA must therefore maintain dialog state and participate in all messages sent between the participating UAs. The open source PBX Asterisk is a B2BUA. |
| CDR | Call Detail/Data Record. A call accounting log record generated by user/customer traffic often used to bill the customer for service. |
| DAA | Digest Access Authentication. A weak authentication method originally developed for HTTP. Later adopted by the SIP protocol for authentication a SIP client to a SIP location service. Developed by the IETF and specified in RFC2617 [19]. |
| GSS-API | Generic Security Services Application Program Interface. An API originally developed by the Kerberos community to add an abstraction layer for applications to access security services, primarily authentication. Developed by IETF, and currently maintained by the "kitten" Working Group. Specified in RFC2743 [46]. |
| H.323 | A recommendation from ITU-T that defines (several) protocols to provide VoIP functionality. Today, the H.323 recommendation is more or less abandoned in favor of the SIP/RTP protocol pair developed by the IETF. SIP/RTP is today regarded as the de facto industry standard for VoIP. |

| | |
|---|---|
| IETF | The Internet Engineering Task Force. A loosely defined group of individuals (academics as well as industry professionals) that create and promotes Internet standards. |
| ITU-T | The Telecommunication Standardization Sector. Coordinates the productions of standards covering all fields of telecommunications on behalf of the International Telecommunication Union (ITU). |
| OID | Object Identifier. An identifier used to name an object. An OID consist of a node in a hierarchically-assigned namespace. They are basically strings of numbers and used in a variety of protocols. GSS-API uses OID to identify the security mechanism used. |
| PAKE | Password Authenticated Key Exchange. Based on a shared password, two or more parties can establish cryptographic keys based on an exchange of message. A PAKE mechanism based on [34] is used in this thesis to provide more secure authentication in SIP. |
| PBKDFv2 | Password-Based Key Derivation Function version 2. A key derivation function that apply a pseudorandom function (usually HMAC-SHA-1) to the input password and salt. The pseudorandom function is iterated a number of times (>1000) and produces a derived key that are significant harder to brute-force. Developed by "RSA Data Security", and specified in RFC2898 [40]. |
| PBX | Private Branch eXchange. A small telephone switch that make connection between local telephones (usually within a company) and to the PSTN. Today these devices are rapidly replaced by VoIP gateways. |
| POTS | Plain Old Telephone Service. Used to denote the absolute basic telephony service like receive and place calls. No extra functionality or features like call waiting or call forwarding. |
| PSTN | Public Switched Telephone Network. The PSTN refers to the entire interconnected collection of local, long distance and international circuit-switched telephone networks world wide. |
| RFC | Request For Comments. Standardizations documents published by the IETF. Several classifications exists, and not all RFCs are classified as "Internet Standards". A RFC is always associated with a unique serial number, for example "RFC3261". |
| RTP | Real-time Transport Protocol. A container protocol suitable to transport real-time data, such as audio and video. Often used in conjunction with the signaling protocol SIP. Developed by the IETF and specified in RFC3550 [80]. |
| SASL | Simple Authentication and Security Layer. Provides a security layer for applications and network protocols to access security services in a unified and generic way. Authentication and data security services are supported. Developed by IETF, and currently maintained by the "kitten" Working Group. Specified in RFC4422 [50]. |

| | |
|---|---|
| SIP | Session Initiation Protocol. A request-response protocol developed by the IETF for initiating and managing multimedia sessions. Core functionality specified in RFC3261 [77], additional functionality defined in numerous other RFCs. |
| SPNEGO | Simple and Protected GSS-API Negotiation Mechanism. A GSS-API "pseudo mechanism" that enables peers to negotiate a common set of one or more GSS-API security mechanisms. Developed by the IETF and specified in RFC4178 [102]. |
| UAC/UAS | User Agent Client / User Agent Server. UAC is the client component in a SIP node responsible for sending SIP request to a UAS. UAS is the SIP server component responsible for receiving SIP requests from a UAC. A SIP node have both UAC and UAS capabilities. UAC and UAS combined is often called a User Agent (UA). |
| UC | Unified Communication. There is no clear and precise definition of UC in the literature. A recurring keyword is *integration*. Integration of real-time (audio calls, video conferencing) and non-real time communications (IM, presence, email) with business processes and requirements (calendar, data sharing). UC is not a single product, but a set of products that are integrated to provide a unified user interface. |
| URI | Uniform Resource Identifier. A name used to identify a resource on the Internet. Example: `sip:alice@example.com`. More details can be found in RFC3986 [4] and in Section 19.1 in RFC3261 [77]. |
| VoIP | Voice over IP. A merge of telecom and data communication, where traditionally telephony service use the Internet as transport. Additional new services like instant messaging, calender integration, mobility, presence can also be supported with VoIP. SIP and RTP are two popular protocols that provide VoIP service. |

# Appendix ii

# SIP request methods and response codes

SIP is based on a request/response transaction model. Each transaction consists of a request that invokes a particular method on the server and at least one response. The different requests are called methods, and the six basic request methods are shown in the table below:

| **SIP method** | **Defined in** | **Support auth.** | **Description** |
|---|---|---|---|
| ACK | RFC3261 [77] | No | Acknowledge a request/session. |
| BYE | RFC3261 [77] | Yes | Terminate a session (call). |
| CANCEL | RFC3261 [77] | Yes | Cancel any pending requests. |
| INVITE | RFC3261 [77] RFC6026 [86] | Yes | Initiate a session (call). |
| OPTIONS | RFC3261 [77] | No | Query servers about their capabilities. |
| REGISTER | RFC3261 [77] | Yes | Register contact information for a UA to a location service. |

Additional SIP requests methods have been defined later, shown in the following table:

| **SIP method** | **Defined in** | **Description** |
|---|---|---|
| INFO | RFC6086 [33] | Send application level info between endpoints. |
| MESSAGE | RFC3428 [8] | Send an instant message. |
| NOTIFY | RFC3265 [72] | Send a notification of a new event that the UA may be subscribed to. |
| PRACK | RFC3262 [76] | PRovisional ACKnowledgement. Used for example if the INVITE transaction take some time to generate a final response. |
| PUBLISH | RFC3903 [54] | Publish event state used within the SIP Events framework |
| REFER | RFC3515 [85] | Redirect the recipient to a resource provided in the (REFER) request. For example call transfer. |
| SUBSCRIBE | RFC3265 [72] | Request current state and state updates from a UA |
| UPDATE | RFC3311 [73] | Update a parameters of a session, but not the state of the dialog. |

SIP response messages are numerical and have been borrowed from the HTTP standard. Six classes have been defined, identified by the first digit[1]. Only the first class is provisional and non-final – the remaining classes are "final" and represent a conclusion to the transaction.

| **Status code** | **Example** | **Description** | **Final** |
|---|---|---|---|
| `1xx` | `100 Trying` | Informal (provisional). Request received and is being processed. | non-final |
| `2xx` | `200 OK` | Success. The request has been received, processed and completed successfully. | final |
| `3xx` | `300 Multiple choices` | Redirection. The request need further action at another location. | final |
| `4xx` | `401 Unauthorized` | Client error. The request contains errors and cannot be completed. | final |
| `5xx` | `503 Service unavailable` | Server error. The request contains errors and cannot be completed at this location. | final |
| `6xx` | `600 Busy everywhere` | Global failure. The request has failed and should not be retried. | final |

[1]An exhaustive list of SIP response codes is maintained by IANA: `http://www.iana.org/assignments/sip-parameters`

# Colophon

This dissertation was set by the author using the official UiO template in LaTeX. The included papers use different LaTeX templates as required by the conference/journal.

This document version: $Rev : 200$