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].

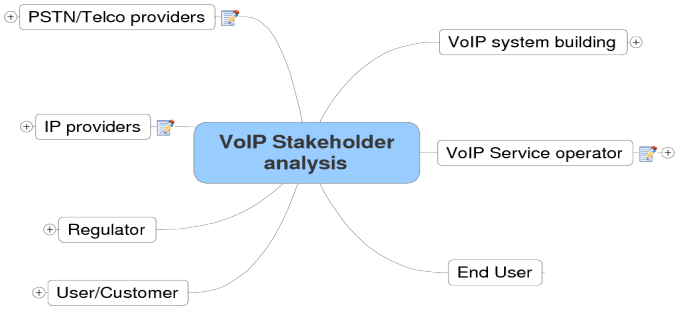

Fig. 2. VoIP Stakeholder analysis

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.

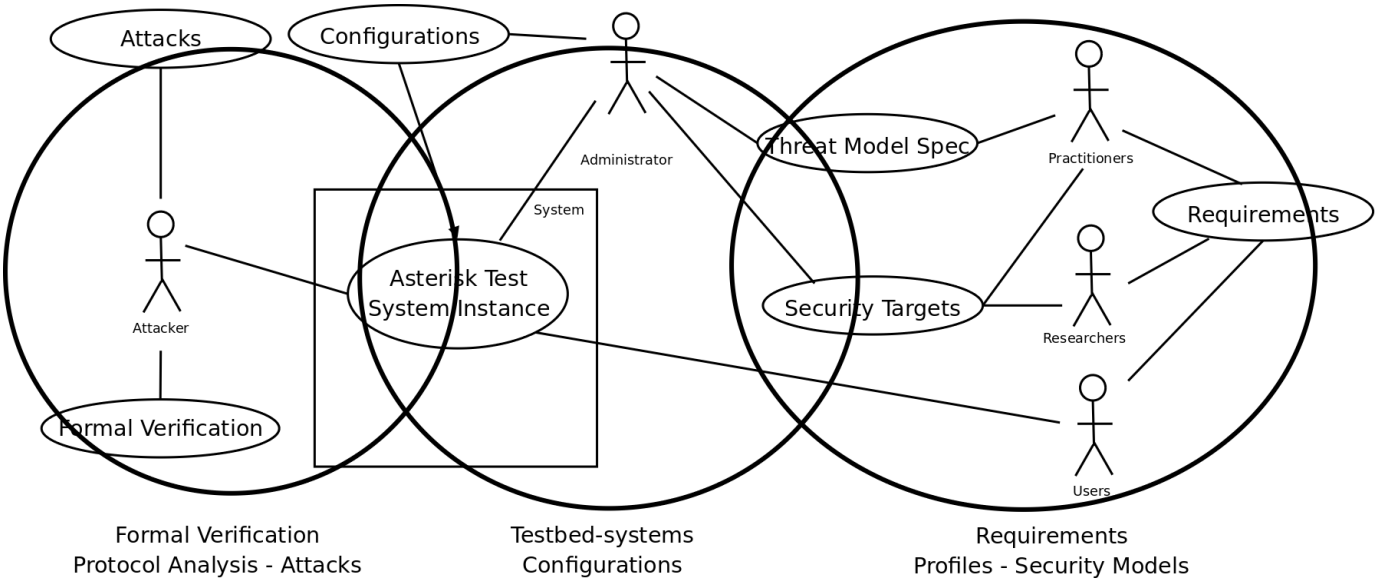

Fig. 1. EUX2010SEC research approach

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*

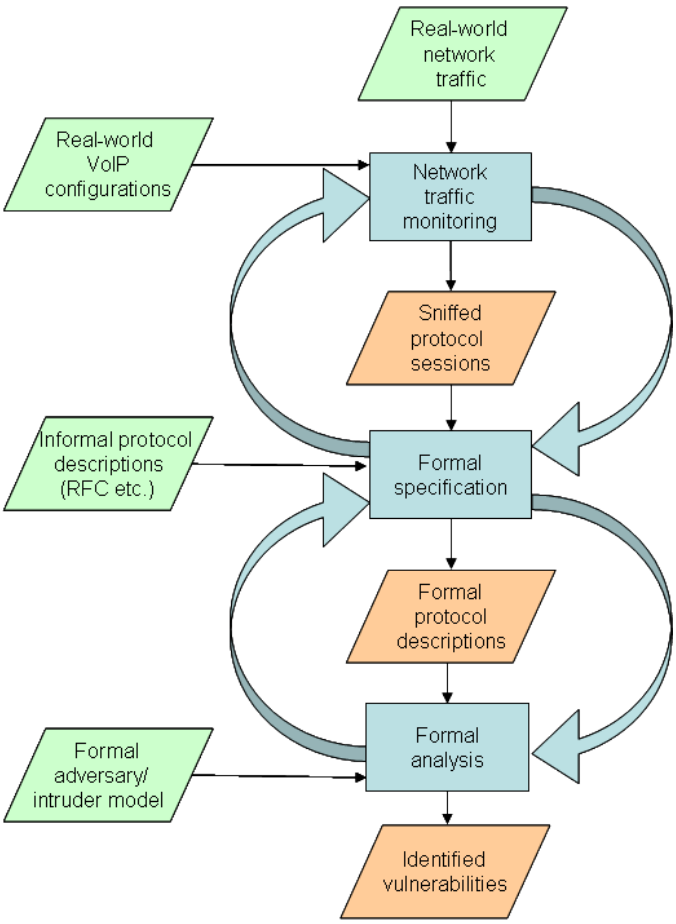

Fig. 3. Formal analysis of VoIP systems

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 $\mathscr{L}_P$	
	ε	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 μ
	$\mathsf{role}(a)$	a is a role in a protocol
(ii)	If φ, ψ, $\xi^{\mathcal{T}}$, $\xi^{\mathcal{A}}$, $\xi^{\mathcal{S}} \in \mathscr{L}_P$, then so are;	
	$\neg\varphi$, $\varphi \rightarrow \psi$	propositional logic
	$a \longrightarrow b : \varphi$	a sends the message φ to b
	$\mathsf{Bel}_a(\varphi)$	a believes φ
	$\mathsf{Hash}[\varphi]$	hash φ
	$\mathsf{Enforce}_{t^A}(\varphi)$	enforce agent t^A to do φ
	$\mathsf{protocol}[\mu, N, \xi^{\mathcal{T}}, \xi^{\mathcal{A}}, \xi^{\mathcal{S}}, \Phi]$	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

$\mathsf{protocol}\mathrm{SIP}, 0,$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B),$
$\quad \mathsf{role}(A) \wedge \mathsf{role}(S) \wedge \mathsf{role}(B),$
$\quad \mathsf{role}(A),$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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.

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

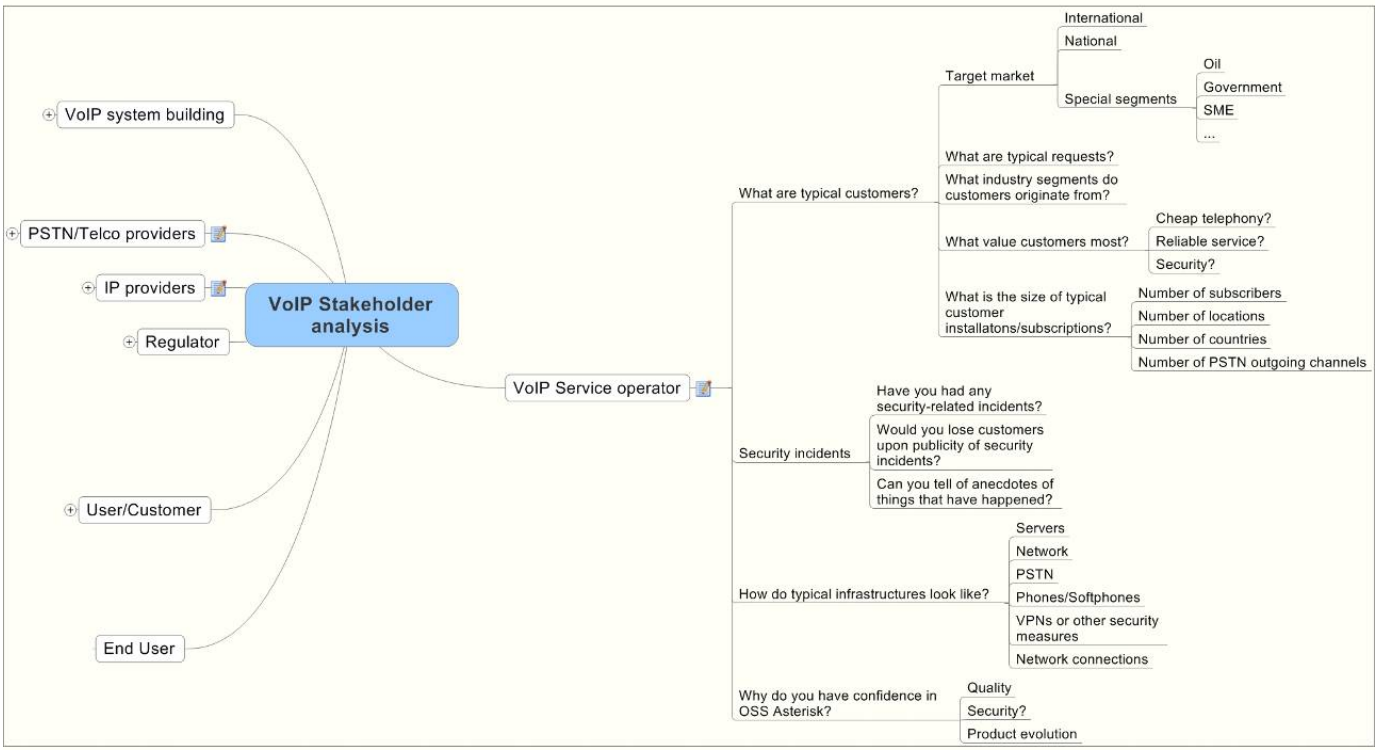

Fig. 6. Stakeholder interview outline for "VoIP system operator"

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.

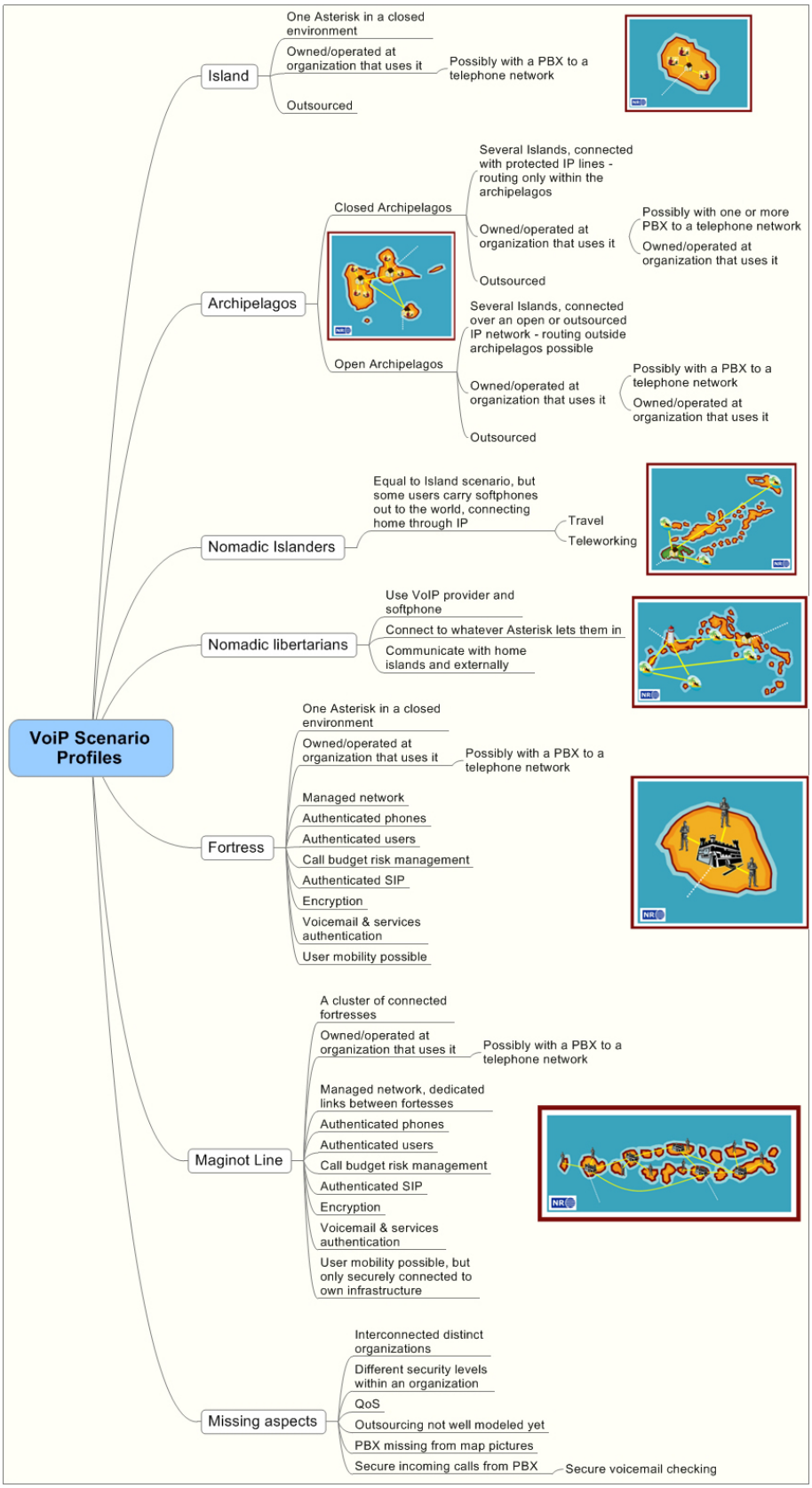

Fig. 7. Island metaphors for VoIP scenario profiles

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

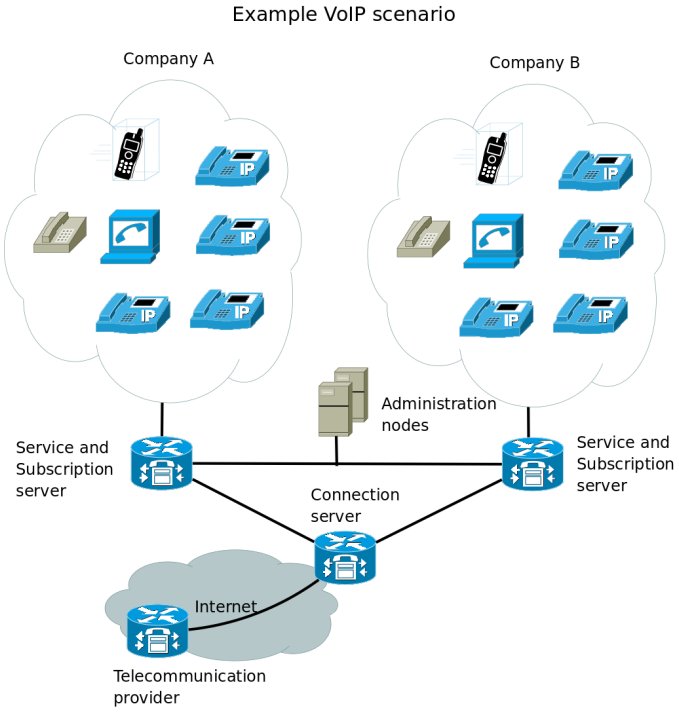

Fig. 9. A real-life VoIP scenario replicated in testbed

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-

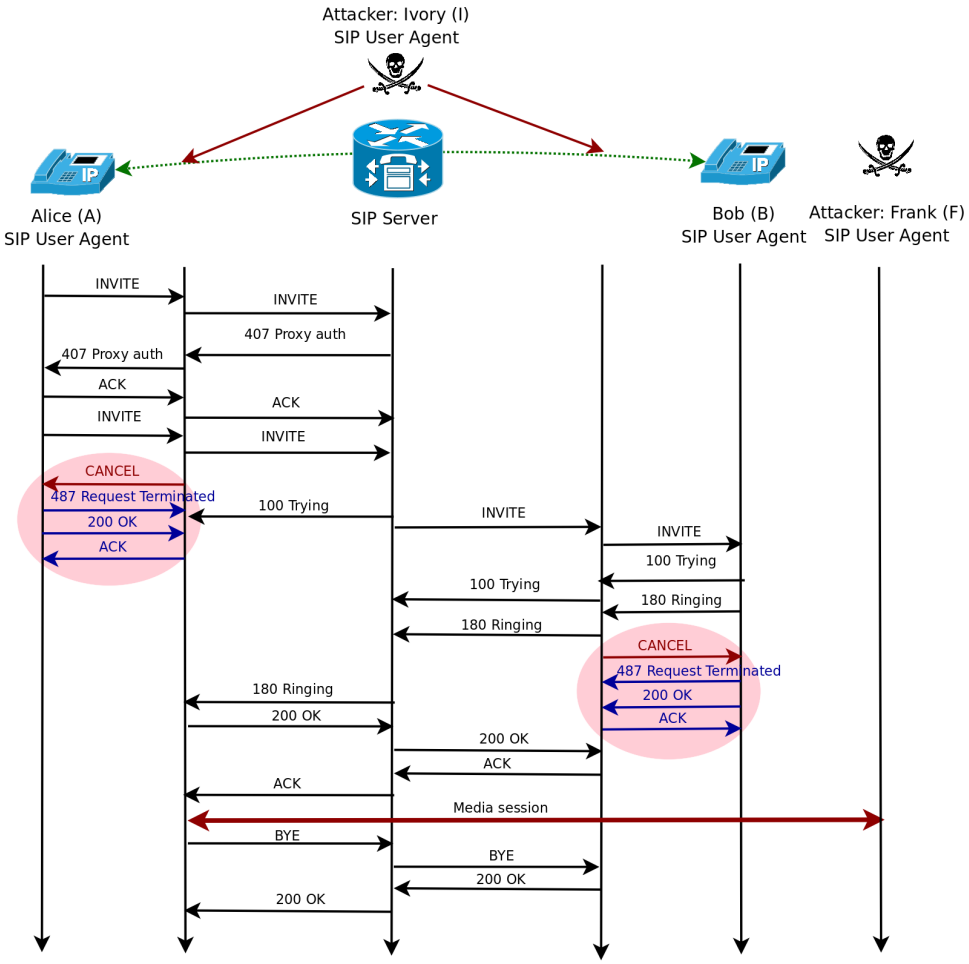

Fig. 8. Hijacking the initiator and the responder.

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.