A Simple Logic for Authentication Protocol Design*

Levente Buttyán
Swiss Federal Institute of Technology
Institute for computer
Communications and Applications (ICA)
EPFL-DI-ICA, 1015 Lausanne, Switzerland
Levente.Buttyan@epfl.ch

Sebastian Staamann and Uwe Wilhelm
Swiss Federal Institute of Technology
Operating Systems Laboratory
EPFL-DI-LSE, 1015 Lausanne, Switzerland
Sebastian.Staamann@epfl.ch
Uwe.Wilhelm@epfl.ch

Abstract

In this paper, we describe a simple logic. The logic uses the notion of channels that are generalisations of communication links with various security properties. The abstract nature of channels enables us to treat the protocol at a higher abstraction level than do most of the known logics for authentication, and thus, we can address the higher level functional properties of the system, without having to be concerned with the problems of the actual implementation. The major advantage of the proposed logic is its suitability for the design of authentication protocols. We give a set of synthetic rules that can be used by protocol designers to construct a protocol in a systematic way.

1. Introduction

Cryptographic protocols are used to provide secure services in distributed systems. However, if the protocol is not designed carefully enough, it may contain flaws, which can be the ideal starting point for various attacks [8] [19] [6] [7]. Such flaws can be subtle and hard to find, so informal design and verification of the protocol is not enough. Formal methods seem to be better suited for solving this problem [15] [10].

One of the formal methods is to use logics for reasoning about the properties of a protocol. Many publications about such logics can be found in the academic literature (e.g., [6] [9] [5] [20] [17] [22] [12]). However, most of these logics (as well as other formal tools [13]) only concentrate on verification. We are not aware of any logics that aim at helping in the design process of cryptographic protocols.

The importance of design tools is emphasised in [15], where the layered approach, known as a general design principle, is mentioned. The layered approach is based on a "stack" of models with different abstraction levels. The protocol designer first uses a relatively abstract model to construct and verify the protocol (top layer). If the protocol is correct at that level, then the designer can switch to a less abstract, more detailed model (succeeding layer), which "implements" (refines) the abstract one. Finally, through the repeated execution of this process, a detailed specification or even the actual protocol code is produced. In this paper, implementation means refinement, so the result of the implementation is a more detailed (not necessarily final) protocol description. This protocol description can then be analysed with other tools, based on more detailed models.

The layered approach bridges the abstraction gap between the informal protocol description (e.g., plain English) and the final implementation (e.g., C source code) by introducing several smaller steps. We assume that it is easier in each layer to produce a correct implementation of the upper layer than to produce a final implementation of the protocol directly from the informal description. It also makes it easier to change the implementation later on (which might be necessary because of modifications to the original design assumptions), because there is a chance that the protocol does not have to be completely redesigned from the informal protocol description, but it might be sufficient to propagate the changes through the different layers.

In this paper, we describe a logic that is abstract enough to be the basis of a first design tool for authentication protocols. Thus, it fits into one of the highest layers in the layered approach described above. The logic can be used to reason about authentication protocols at a high abstraction level. Its clarity and economy (there are only seven inference rules in the logic) make it a tool easy to use, while it still captures all the important aspects of authentication protocols. Based on the logical constructs of the presented logic, we give a set of synthetic rules that can be used by the protocol designer to construct the desired protocol. First, the designer must identify the goals of the protocol and describe them

* ©1998 IEEE. Accepted for the 11th IEEE Computer Security Foundations Workshop. Research supported by the Swiss National Science Foundation under project number 5003-045364.

with the language of our logic. Then, by using the given synthetic rules, the designer can generate the whole protocol and the required assumptions in a systematic way. The result of this process is a formal description of the protocol in our language. The implementation of the formal protocol is not supported by our logic, but we believe that in most of the cases it is straightforward.

Our method of constructing protocols is very similar to the approach described in [4], where a weakest precondition calculus is introduced and used to generate the least restrictive set of required assumptions from the goals of the protocol and the protocol itself. Although the authors discuss the possibility of using their approach for protocol derivation, they do not actually use their results to derive protocols. Instead, they use the weakest precondition calculus for protocol verification.

The logic that we present in this paper belongs to the BAN logic [6] family. BAN logic is the best known and most influential logic in protocol verification. It allows us to describe the beliefs of trustworthy parties involved in authentication protocols and the evolution of these beliefs as a consequence of communication. It has been successfully applied to discover flaws in a variety of authentication protocols and has also been helpful in the understanding of the basic concepts of authentication. BAN logic is simple (it has about 15 inference rules), which might be one of the reasons for its popularity. The need for many universal assumptions in the model, however, is a minor disadvantage. In BAN logic, it is assumed that principals of the system are trustworthy and do not release secrets (this has not always been understood with its full meaning [16]); the applied encryption schemes are strong (i.e., encrypted messages can be decrypted only with the appropriate key); each encrypted message contains sufficient redundancy to allow a principal who decrypts it to verify that it has used the right key; and principals can recognise and ignore their own messages. Note, that the last three assumptions are rather technical, while the first one is more fundamental.

BAN logic has been extended in many directions (e.g., [9] [3]). Each extension is based on a more general model (that needs less assumptions) than the model of the original logic, but as a consequence, each has a more complex logic. In GNY logic [9], for instance, it is not required to assume that principals are trustworthy and redundancy is always explicitly present in encrypted messages. GNY logic distinguishes between what a principal can possess and what it can believe in. It allows us to express different trust levels and implicit conditions behind protocol steps. However, GNY logic has more than 40 inference rules.

Our logic preserves the simplicity of BAN logic and adopts some concepts from GNY logic. As in BAN logic, we assume that encrypted messages contain sufficient redundancy, and principals can recognise and ignore their own messages. We consider the problem of redundancy and recognisability to be at a too low level and too implementation specific to deal with them in the first design phase. As in GNY logic, we do not assume that principals are trustworthy. We can also express different aspects of trust and implicit conditions. However, we do this in a slightly different way than GNY logic does.

The main notion of our logic is the channel. Channels are abstract views of various types of secure communication links between principals. Our logic does not have any constructs that deal with encryption or decryption operations, instead we use only channels with various access restrictions. This is similar to how *pi* calculus primitives for channels are used in [1]. However, we consider our method to be more intuitive, because channel formulae are notationally less complex and more closely aligned with how one usually talks about cryptographic protocols. A similar logic based on channels is introduced in [14] and [24]. However, these works do not address all of the problems treated in this paper. For instance, timeliness, which is fundamental in authentication protocols, is not addressed at all.

How the channels are realised is out of the scope of the logic, although this is a very important issue. One can directly use the results of [18] and [23] for channel implementation. In these papers, channels are used to model security services. For each type of channel, a set of production rules, that describes how to establish that channel, is introduced. One can use these rules to implement our channels. Furthermore, a detailed treatment about channels that are implemented by encryption can be found in [14]. We have provided examples for most of the channels that we use.

The outline of the paper is the following. In Section 2, we introduce our model and the notion of channels. In Section 3, we describe our logic and we illustrate how one can use it to generate synthetic rules. In Section 4, we give a detailed example how the logic can be applied to verify and design authentication protocols. In particular, we analyse and redesign the Woo and Lam symmetric key authentication protocol [25]. This protocol has flaws. Attacks that exploit these flaws are presented in [2] and [7]. We demonstrate, how our synthetic method generates a correction of the original protocol, which is resistant against these attacks. Finally, in Section 5, we conclude the paper.

2. The model

We consider a distributed system to be a set of principals (users, hosts, and processes) and channels. Principals can interact with each other according to the rules of some predefined protocols in order to accomplish a common task (e.g., to provide a service). Interactions are based on messages that are conveyed via channels. A channel is an abstraction of a communication facility that has certain access properties. The channel can represent a physical link, as well as a cryptographically secured logical connection between principals. Channels are the main abstractions that we consider in this paper. Their abstract nature enables us to design and analyse protocols without addressing the problem and the complexity of the actual implementation.

A channel is characterised by its set of readers and its set of writers (i.e., the set of principals that can receive messages via the channel and the set of principals that can send messages via the channel). The set of readers and the set of writers of channel C are denoted by $r(C)$ and $w(C)$, respectively.

As an example, let us consider two principals P and Q that have a shared secret K established between them and both know a symmetric key encryption algorithm A. Given the algorithm A and the key K, P and Q can securely exchange messages via C. These messages are incomprehensible for other principals in the system. We say, that there is a channel C between P and Q, for which $r(C) = w(C) = \{P, Q\}$ (i.e., only P and Q can send and receive messages via C).

To use a channel, a principal needs information about how to read and/or write from/to the channel. This information is denoted by C^r and C^w, respectively. We say, that principal P is a reader of the channel C, denoted by $P \in r(C)$, iff P possesses C^r, and similarly, principal P is a writer of the channel C, denoted by $P \in w(C)$, iff P possesses C^w. In the above example $C^r = C^w = (K, A)$ (i.e., the knowledge of the shared secret and the encryption algorithm is sufficient for P and Q to communicate in a secure way).

We assume that a principal can always detect a message arrival on any channel that it can read. To be more precise, we assume that if a message arrives on a readable channel, then the principal detects the arrival and the principal is able to determine on which channel the message arrived. We make this assumption to keep our model simple. Removing the problem of recognisability from the scope of the model enables us to concentrate on the important functional properties of the system. For similar reasons, we also assume that principals can recognise their own messages (if the message arrives on a channel that the principal can read) and ignore them. Real implementations of the channels can support these features by putting enough redundancy into the messages and using key identifiers. In this paper, we are not explicitly concerned with the details of the implementation of the channels. For this topic, we refer to [14].

Below, we give some basic types of channels as examples:

Public Channel. C is a public channel iff anybody in the system can write and read it (i.e., $r(C) = w(C) = \Omega$, where Ω is the set of all principals).

Authentic Channel. C is an authentic channel iff anybody can read it, but only one principal P can write it (i.e., $r(C) = \Omega$ and $w(C) = \{P\}$). Authentic channels can be implemented by using digital signatures [1].

Confidential Channel. C is a confidential channel iff anybody can write it, but only one principal P can read it (i.e., $r(C) = \{P\}$ and $w(C) = \Omega$). Confidential channels can be established by encryption with the public key of P.

Dedicated Channel. C is a dedicated channel iff one principal P can read it and one principal Q can write it (i.e., $r(C) = \{P\}$ and $w(C) = \{Q\}$). Dedicated channels can be constructed by combining the properties of the authentic channel with the properties of the confidential channel.

Closed Group Channel. C is a closed group channel iff a set of principals can write it and the same set of principals can read it (i.e., $r(C) = w(C) = \mathcal{A}$, where $\mathcal{A}$ is a set of principals). Closed group channels can be implemented by using symmetric key encryption and distributing the key to a set of principals. If the size of the set of the principals is two, then we call the resulting closed group channel conventional secret channel:

Conventional Secret Channel. C is a conventional secret channel iff it can be used only by two principals P and Q (i.e., $r(C) = w(C) = \{P, Q\}$).

3. The logic

Our logic is a many-sorted modal logic that enables us to reason about authentication protocols. It models the behaviour of the principals and the channels in the system described in the previous section. It consists of a language and a small number of inference rules. The language is used to describe assumptions and events, as well as the goals of the protocol. The inference rules are used to derive new statements about the system.

One can use the logic for analysing existing authentication protocols. For this reason, protocol descriptions first have to be translated into a description in our notation. This usually means that all the encryption and decryption operations have to be eliminated from the description, and must be replaced by the appropriate channels. It is also required to interpret some messages and to replace certain parts of a message by formulae (we define this later) when it is necessary to capture implicit assumptions behind protocol steps.

The goal of the analysis is to construct a witnessing deduction, which is a derivation of the goals from the assumptions and the formal protocol description. We consider that the protocol is correct if such a deduction exists. The lack of a witnessing deduction means that the protocol may not be correct. Although, in this case, we cannot generate a complete attack scenario with our logic, the analysis reveals the

[1] In this case, recognising that a message arrived on the channel requires a successful verification of the signature.

possible weakness in the protocol, and we can construct an attack afterwards more easily.

The analysis technique is the same as in [6]: we annotate the protocol with logical formulae. We write formulae before the first message (these represent the initial assumptions) and after every message. In our logic, a message M that is received by Q on channel C is represented by the logical formula $Q \lhd C(M)$. We do not use the usual $P \rightarrow Q : M$ notation, because when a principal receives a message it does not necessarily know where the message comes from. However, certain channels may allow the principal to derive this information later on. Our notation of message reception also prevents us from attaching the same meaning of message M for both the sender and the receiver. Assuming that the sender's intention and the receiver's interpretation are the same can lead to subtle mistakes in the analysis as it is pointed out in [21]. We are interested in the receiver's interpretation. After receiving a message, the formula that represents the message becomes valid. From this recent valid formula and from previously derived valid formulae we can derive new valid formulae. In this way we can follow the evolution of various statements about the system.

The main advantage of our logic is that one can also use it in the design process of authentication protocols. For this reason, we have constructed some synthetic rules that are based on the inference rules of the logic. One can use these rules to generate the formal protocol description and the assumptions starting from the goals of the protocol. Due to the small number of the inference rules, we were able to keep the number of the synthetic rules small as well.

During the synthesis, the designer must choose synthetic rules that fit best the desired goals and the assumptions that have already been generated. It is possible that to reach a given goal, different synthetic rules can be used. This may result in different protocols for the same goals and assumptions. By applying a synthetic rule, the designer obtains new goals to be reached. Certain goals may be reached simply by assuming that the statements that represent them are true. Other goals may require a protocol message to be sent. If there are no more goals to reach, then the synthesis is finished. This process results in a number of assumptions and protocol messages, from which the construction of the formal protocol description is then straightforward.

Thus, using our logic, protocol designers can describe and analyse protocols at a high abstraction level, without dealing with the problems of implementation. If the protocol works at that level, then they can replace the high level constructs (the channels) by their implementations. Separating the design and the implementation phase significantly reduces the complexity of authentication protocol construction.

We demonstrate the analysis and the design processes in Section 4. In the remainder of this section, we concentrate on the logic itself.

3.1. The language

Hereafter we will use the following notations: P and Q range over principals; C is a channel; X represents a message, which can be data or formulae or both; $C(X)$ denotes message X on channel C; ϕ will be used to denote a formula.

The basic formulae are the following:

$P \lhd C(X)$: P sees $C(X)$. Someone has sent a message X via channel C and P can observe this. If P cannot read channel C, then P cannot recognise and understand this message (i.e., P cannot determine which channel was used and what was the message).

$P \lhd X \mid C$: P sees X via C. P received message X via channel C. This is possible only if someone has sent this message, and P can read this channel.

$P \lhd X$: P sees X. Someone has sent a message containing X via a channel that P can read.

$\sharp(X)$: X is fresh. X has never been said before the current run of the protocol. This is usually true for nonces.

$P \mid\!\sim X$: P once said X. P at some time sent a message that contained X. We do not know exactly when the message was sent.

$P \parallel\!\sim X$: P has recently said X. This means that P uttered X in the current run of the protocol.

If ϕ is a formula, then the following is also a formula:

$P \models \phi$: P believes ϕ. P believes that ϕ is true. This does not mean that ϕ is really true, but P acts as though.

Further formulae can be derived by using the conventional logical operators from propositional logic. If ϕ_1 and ϕ_2 are formulae, then the following are also formulae:

$\phi_1 \wedge \phi_2$: ϕ_1 and ϕ_2.

$\phi_1 \vee \phi_2$: ϕ_1 or ϕ_2.

$\phi_1 \rightarrow \phi_2$: ϕ_1 implies ϕ_2.

We also use notations from set theory. The meaning of these notations is straightforward in our language (e.g. $P \in \mathcal{A}$, where $\mathcal{A}$ is a set of principals, is a formula, which means that principal P is an element of the set $\mathcal{A}$).

We can demonstrate the expressive power of our language by showing how certain aspects of trust can be described with the constructs defined above. In our language, the belief of principal P in the honesty of principal Q is modelled by the following formula:

$$P \models ((Q \mid\!\sim \phi) \to (Q \models \phi))$$

This means that P believes that if Q has recently said ϕ, then Q believes ϕ. In short, P believes that "Q says only what it believes". P does not necessarily believe that what Q says is true, it only believes that Q believes it. However, it is possible that P does not share all of Q's beliefs. Another aspect of trust, namely competency, can be expressed by the following formula:

$$P \models ((Q \models \phi) \to \phi)$$

This means that P believes that if Q believes ϕ, then ϕ is true. If we combine these two formulae, then we obtain the following one:

$$P \models ((Q \mid\!\sim \phi) \to \phi)$$

This means that P believes that if Q has recently said ϕ, then ϕ is true. In short, P believes "what Q says is true" (i.e., P believes that Q is honest and competent).

We can restrict the scope of these beliefs by giving more details about ϕ. P can believe, for instance, that Q is an authentication server and it is competent only in establishing new secret channels between principals. We can express this by the following formula:

$$\begin{aligned} P \models ((Q \models (r(C) = w(C) = \{A, B\})) \to \\ \to (r(C) = w(C) = \{A, B\})) \end{aligned}$$

At the same time, P may not believe, for instance, that Q is competent in recognising freshness, which means that

$$P \models ((Q \models \sharp(X)) \to \sharp(X))$$

does not hold.

3.2. Inference rules

Seeing rules:

(S1) If a principal P receives a message X via a channel C, and P can read this channel, then P recognises that the message has arrived on C and P can see the message.

$$\frac{P \lhd C(X), P \in r(C)}{P \models (P \lhd X \mid C), P \lhd X}$$

(S2) If a principal P sees a compound message (X, Y), then it sees also parts of the message (i.e., X and Y).

$$\frac{P \lhd (X, Y)}{P \lhd X, P \lhd Y}$$

Interpretation rules:

(I1) If a principal P believes that a channel C can be written only by a set of principals $\mathcal{W}$, then P believes that if it receives a message via C, then someone from the set $\mathcal{W}$ except P itself ($\mathcal{W} \setminus \{P\}$) said X.

$$\frac{P \models (w(C) = \mathcal{W})}{P \models ((P \lhd X \mid C) \to \bigvee_{\forall Q_i \in \mathcal{W} \setminus \{P\}} (Q_i \mid\!\sim X))}$$

(I2) If a principal P believes that another principal Q has said a compound message (X, Y), then it believes that Q has said the parts of the message as well (i.e., X and Y).

$$\frac{P \models (Q \mid\!\sim (X, Y))}{P \models (Q \mid\!\sim X), P \models (Q \mid\!\sim Y)}$$

Freshness rules:

(F1) If a principal P believes that another principal Q said a message X and P also believes that X is fresh, then P believes that Q has recently said X.

$$\frac{P \models (Q \mid\!\sim X), P \models \sharp(X)}{P \models (Q \mid\!\mid\!\sim X)}$$

(F2) If a principal P believes that a part of a compound message X is fresh, then it believes that the whole message (X, Y) is fresh.

$$\frac{P \models \sharp(X)}{P \models \sharp(X, Y)}$$

Rationality rule:

(R1) Our rationality rule is the well-known K axiom of modal logics [11] in rule form: if a principal P believes that ϕ_1 implies ϕ_2 and the principal believes that ϕ_1 is true, then it believes that ϕ_2 is also true.

$$\frac{P \models (\phi_1 \to \phi_2), P \models \phi_1}{P \models \phi_2}$$

3.3. Theorems

In the following, we give three simple theorems of the logic that we will use in the construction of the synthetic rules.

Theorem 1 *If a principal P receives a message X via a channel C, then it believes that the message was said by someone, who the principal believes to be able to write channel C, except itself.*

$$\frac{P \lhd C(X), P \in r(C), P \models (w(C) = \mathcal{W})}{P \models (\bigvee_{\forall Q_i \in \mathcal{W}\setminus\{P\}}(Q_i \mid\!\sim X))}$$

Proof: The proof can be easily constructed by using the first seeing rule (S1), the first interpretation rule (I1), and the rationality rule (R1).

Theorem 2 *If a principal P believes that another principal Q is honest and P believes that Q has recently said ϕ, where ϕ is a formula, then P believes that Q believes ϕ.*

$$\frac{P \models ((Q \mid\!\mid\!\sim \phi) \to (Q \models \phi)), P \models (Q \mid\!\mid\!\sim \phi)}{P \models (Q \models \phi)}$$

Proof: This is a substitution instance of the rationality rule.

Theorem 3 *If a principal P believes that another principal Q is honest and competent, and P believes that Q has recently said ϕ, where ϕ is a formula, then P believes ϕ.*

$$\frac{P \models ((Q \mid\!\mid\!\sim \phi) \to \phi), P \models (Q \mid\!\mid\!\sim \phi)}{P \models \phi}$$

Proof: This is a substitution instance of the rationality rule.

3.4. Synthetic rules

Reversing the inference rules of the logic and the theorems derived before, we construct synthetic rules that can be used in a systematic way when designing protocols. The construction of the synthetic rules is straightforward, except (Syn 4) and the first part of (Syn 5) that we will discuss in more detail later. The list of the synthetic rules given here is not exhaustive; one can derive new rules from new theorems. For our purposes (i.e., to show how to use the synthetic rules for protocol construction), the list given below is sufficient.

The synthetic rules have the following general form:

$$\begin{array}{ll} G & \\ \hookrightarrow & G_1 \\ \hookrightarrow & G_2 \\ \hookrightarrow & \ldots \end{array}$$

which means that in order to reach the goal G all the given new goals $G_1, G_2, \ldots$ have to be reached. A goal G can have the form G'/G'', which means that either G' or G'' have to be reached.

(Syn1) (derived from (S1)) To recognise that a message X arrived via a channel C, a principal P has to receive $C(X)$ and it has to be able to read C.

$$\begin{array}{ll} P \models (P \lhd X \mid C) & \\ \hookrightarrow & P \lhd C(X) \\ \hookrightarrow & P \in r(C) \end{array}$$

(Syn2) (derived from (S1) and (S2)) To see a message X, a principal P has to see a message (X, Y) that contains X or it has to receive X via a channel C.

$$\begin{array}{ll} P \lhd X & \\ \hookrightarrow & P \lhd (X, Y) \ / \ P \models (P \lhd X \mid C) \end{array}$$

(Syn3) (derived from (I2)) To believe that a principal Q said X, a principal P has to believe that Q said a message (X, Y) that contains X.

$$\begin{array}{ll} P \models (Q \mid\!\sim X) & \\ \hookrightarrow & P \models (Q \mid\!\sim (X, Y)) \end{array}$$

(Syn4) (derived from Theorem 1) To believe that a principal Q said X, a principal P has to receive X via a channel C that it can read and that it believes can be written only by Q, or P and Q. Furthermore, Q has to see X.

$$\begin{array}{ll} P \models (Q \mid\!\sim X) & \\ \hookrightarrow & P \lhd C(X) \\ \hookrightarrow & P \in r(C) \\ \hookrightarrow & P \models (w(C) = \{Q\}) \ / \\ & P \models (w(C) = \{P, Q\}) \\ \hookrightarrow & Q \lhd X \end{array}$$

(Syn5) (derived from Theorem 1 and (F1)) To believe that a principal Q has recently said X, the following is required:

if X is a formula and P believes that Q is honest (i.e., $P \models ((Q \mid\!\mid\!\sim \phi) \to (Q \models \phi))$), then P has to receive X via a channel C that it can read and that it believes can be written only by Q, or P and Q. Furthermore, P has to believe that X is fresh and Q has to believe X,

$$
\begin{array}{lll}
P \models (Q \mid\!\sim X) & & \\
 & \hookrightarrow & P \lhd C(X) \\
 & \hookrightarrow & P \in r(C) \\
 & \hookrightarrow & P \models (w(C) = \{Q\}) \ / \\
 & & P \models (w(C) = \{P, Q\}) \\
 & \hookrightarrow & P \models \sharp(X) \\
 & \hookrightarrow & Q \models X
\end{array}
$$

otherwise P has to believe that Q said X and X is fresh.

$$
\begin{array}{lll}
P \models (Q \mid\!\sim X) & & \\
 & \hookrightarrow & P \models (Q \mid\!\sim X) \\
 & \hookrightarrow & P \models \sharp(X)
\end{array}
$$

(Syn6) (derived from (F2)) To believe that a message X is fresh, a principal P has to believe that some part X' of X is fresh.

$$
\begin{array}{lll}
P \models \sharp(X) & & \\
 & \hookrightarrow & P \models \sharp(X')
\end{array}
$$

(Syn7) (derived from Theorem 2) To believe that a principal Q believes a formula ϕ, a principal P has to believe that Q is honest and that Q has recently said ϕ.

$$
\begin{array}{lll}
P \models (Q \models \phi) & & \\
 & \hookrightarrow & P \models (Q \mid\!\sim \phi) \\
 & \hookrightarrow & P \models ((Q \mid\!\sim \phi) \to (Q \models \phi))
\end{array}
$$

(Syn8) (derived from Theorem 3) To believe a formula ϕ, a principal P has to believe that a principal Q is honest and competent, and that Q has recently said ϕ.

$$
\begin{array}{lll}
P \models \phi & & \\
 & \hookrightarrow & P \models (Q \mid\!\sim \phi) \\
 & \hookrightarrow & P \models ((Q \mid\!\sim \phi) \to \phi)
\end{array}
$$

(Syn9) (derived from (R1)) To believe a formula ϕ, a principal P has to believe a formula ϕ' and the implication $\phi' \to \phi$.

$$
\begin{array}{lll}
P \models \phi & & \\
 & \hookrightarrow & P \models \phi' \\
 & \hookrightarrow & P \models (\phi' \to \phi)
\end{array}
$$

The rules (Syn 4) and the first part of (Syn 5) require some explanation. Let us start with (Syn 4), which is based on the reverse of Theorem 1. The first three of the new goals in (Syn 4) follow directly from Theorem 1. We require the last one to ensure that in the protocol that is generated using this synthetic rule, principals say only messages that they see (i.e., what they possess). In [4], where a similar approach is taken to derive the least restrictive set of assumptions for a given goal and protocol, this is called "required precondition". For similar reasons, in the first part of (Syn 5), we require the last of the new goals to ensure that the protocol, that is generated using this synthetic rule, does not override the honesty assumption (i.e., honest principals say only what they believe).

4. Example – Analysing and correcting the Woo and Lam authentication protocol

In this section, we demonstrate how our logic can be used for analysis and design purposes. In particular, we analyse the Woo and Lam symmetric key authentication protocol [25]. This protocol has flaws [2] [7]. We show that our logic discovers a known flaw in the protocol. We also demonstrate, how our synthetic method generates a correction of the original protocol, which is resistant against the attacks that are presented in [2] and [7].

4.1. Analysis

In the Woo and Lam protocol, A authenticates itself to B with the help of an authentication server S. In the first message, A sends its identifier to B, who responds with a freshly generated nonce N_b. A encrypts the nonce with the key K_{as}, which is shared between A and S, and sends the result to B. B cannot decrypt this message, so it sends it to S together with A's identifier encrypted under the key K_{bs}, which is shared between B and S. S decrypts the message and sends back to B the nonce N_b encrypted with B's key. Now, B can decrypt the message and it can verify if it received back its challenge. If so, then it concludes that it talks with A.

$A \to B : A$

$B \to A : N_b$
$A \to B : \{N_b\}_{K_{as}}$
$B \to S : \{A, \{N_b\}_{K_{as}}\}_{K_{bs}}$
$S \to B : \{N_b\}_{K_{bs}}$

In our logic, the protocol is described in the following way:

$B \lhd A$
$A \lhd N_b$
$B \lhd C_{as}(N_b)$
$S \lhd C_{bs}(A, C_{as}(N_b))$
$B \lhd C_{bs}(N_b)$

We identified the following assumptions:

(A1) $A \in r(C_{as})$
A can read from channel C_{as}.

(A2) $S \in r(C_{as})$
S can read from channel C_{as}.

(A3) $A \models (w(C_{as}) = \{A, S\})$
A believes that only A and S can write C_{as}.

(A4) $S \models (w(C_{as}) = \{A, S\})$
S believes that only A and S can write C_{as}.

(A5) $B \in r(C_{bs})$
B can read from channel C_{bs}.

(A6) $S \in r(C_{bs})$
S can read from channel C_{bs}.

(A7) $B \models (w(C_{bs}) = \{B, S\})$
B believes that only B and S can write C_{bs}.

(A8) $S \models (w(C_{bs}) = \{B, S\})$
S believes that only B and S can write C_{bs}.

(A9) $A \models ((S \mid\!\sim \phi) \to (S \models \phi))$
A belies that S is honest.

(A10) $A \models ((S \models (B \mid\!\sim X)) \to (B \mid\!\sim X))$
A believes that S is competent in deciding whether B said message X.

(A11) $B \models ((S \mid\!\sim \phi) \to (S \models \phi))$
B believes that S is honest.

(A12) $B \models ((S \models (A \mid\!\sim X)) \to (A \mid\!\sim X))$
B believes that S is competent in deciding whether A said message X.

(A13) $B \models \sharp(N_b)$
B believes that its nonce N_b is fresh.

The goal of the protocol is to convince B that it talks with A. We cannot express this directly in our language. One might think that the goal of the protocol is captured by the formula $B \models (A \mid\!\mid\!\sim N_b)$. In fact, it is not. Let us consider the following attack against the protocol [7], in which a legitimate, but malicious principal M wants to masquerade as A to B. Let us assume that A starts a session with M by sending its identifier to M. After receiving this message, M starts a session with B by sending A's identifier to B. Thus, B thinks that A started the session. B generates the nonce N_b and sends it to A, but M intercepts this message. M sends the nonce N_b to A. A thinks that this is the challenge from M, encrypts it with K_{as} and sends the result to M. Now, M can forward this message to B. Since this is the message that is expected by B, after the message exchange with the server, B will conclude that it talks with A. The problem is that A has really said the nonce N_b, so B can believe this with good reason. But A intended to say it to M. Considering these arguments, we express the goal of the Woo and Lam protocol with the following formula:

$$B \models (A \mid\!\mid\!\sim (B, N_b))$$

which means that B believes that A has recently said the nonce N_b to B.

Starting the analysis from the last message and using Theorem 1, we can obtain that $B \models (S \mid\!\sim N_b)$. Using the first freshness rule (F1) and the last assumption (A13), we get that $B \models (S \mid\!\mid\!\sim N_b)$. We cannot use any inference rule to derive more. Thus, the goal cannot be derived from the assumptions and the formal protocol description. The problem is that there is nothing in the last message that indicates that A said the nonce. The description of the attack that exploits this weakness can be found in [2].

4.2. Design

We correct the Woo and Lam protocol using our synthetic rules. Since we want the resulting protocol to be similar to the original one, we select the various synthetic rules accordingly during the synthesis. We start from the goal $B \models (A \mid\!\mid\!\sim (B, N_b))$. Using the second part of (Syn5):

$$\begin{array}{lll} B \models (A \mid\!\mid\!\sim (B, N_b)) & & \\ & \hookrightarrow & B \models (A \mid\!\sim (B, N_b)) \\ & \hookrightarrow & B \models \sharp(B, N_b) \end{array}$$

Using (Syn6):

$$\begin{array}{lll} B \models \sharp(B, N_b) & & \\ & \hookrightarrow & B \models \sharp(N_b) \end{array}$$

This is assumption (A13), so this goal is reached. We continue with $B \models (A \mid\!\sim (B, N_b))$. Using (Syn8):

$$
\begin{array}{lll}
B \models (A \mid\!\sim (B, N_b)) & & \\
 & \hookrightarrow & B \models (S \mid\!\models\!\sim (A \mid\!\sim (B, N_b))) \\
 & \hookrightarrow & B \models ((S \mid\!\models\!\sim (A \mid\!\sim (B, N_b))) \\
 & & \rightarrow (A \mid\!\sim (B, N_b)))
\end{array}
$$

The second new goal is reached by assumptions (A11) and (A12) (i.e., B believes that S is honest and competent in deciding whether A said a message). We continue with the goal $B \models (S \mid\!\models\!\sim (A \mid\!\sim (B, N_b)))$. Using the first part of (Syn5):

$$
\begin{array}{lll}
B \models (S \mid\!\models\!\sim (A \mid\!\sim (B, N_b))) & & \\
 & \hookrightarrow & B \lhd C_{bs}(A \mid\!\sim (B, N_b)) \\
 & \hookrightarrow & B \in r(C_{bs}) \\
 & \hookrightarrow & B \models (w(C_{bs}) = \{B, S\}) \\
 & \hookrightarrow & B \models \sharp(A \mid\!\sim (B, N_b)) \\
 & \hookrightarrow & S \models (A \mid\!\sim (B, N_b))
\end{array}
$$

The second and the third new goals are reached by assumption (A5) and (A7), respectively. The fourth new goal is reached by (Syn6) and assumption (A13). The first new goal can be reached by sending the formula $A \mid\!\sim (B, N_b)$ via channel C_{bs}. We continue with the last goal $S \models (A \mid\!\sim (B, N_b))$. Using (Syn4):

$$
\begin{array}{lll}
S \models (A \mid\!\sim (B, N_b)) & & \\
 & \hookrightarrow & S \lhd C_{as}(B, N_b) \\
 & \hookrightarrow & S \in r(C_{as}) \\
 & \hookrightarrow & S \models (w(C_{as}) = \{A, S\}) \\
 & \hookrightarrow & A \lhd (B, N_b)
\end{array}
$$

The first new goal is a protocol message. The next two goals can be reached by assumption (A2) and (A4). The last goal can be reached by sending the message (B, N_b) to A. So, our correction of the Woo and Lam protocol is the following:

$$
\begin{array}{l}
A \lhd (B, N_b) \\
S \lhd C_{as}(B, N_b) \\
B \lhd C_{bs}(A \mid\!\sim (B, N_b))
\end{array}
$$

This protocol can be implemented in various ways. Here, we give an implementation that is similar to the original protocol:

$$
\begin{array}{l}
A \rightarrow B : A \\
B \rightarrow A : B, N_b \\
A \rightarrow B : \{B, N_b\}_{K_{as}} \\
B \rightarrow S : A, \{B, N_b\}_{K_{as}} \\
S \rightarrow B : \{A, B, N_b\}_{K_{bs}}
\end{array}
$$

One can easily verify that this protocol is resistant against the attacks presented in [2] and [7].

5. Conclusion

We presented a simple logic that can be used in the design process of authentication protocols. The logic is based on the notion of channels that are abstract views of different types of secure communication facilities. Channels are general enough to describe many of the well-known cryptographically protected communication schemes. Due to this notion, our logic has not more than seven main inference rules. This makes the logic economic and clear, while it still captures all the important aspects of authentication protocols. We can express, for instance, different aspects of trust that principals have in other principals. Based on the logic, we presented a set of synthetic rules that can be used in a systematic way, when designing authentication protocols. We demonstrated how to use the logic and the synthetic rules for verification and design of authentication protocols at a high abstraction level without having to be concerned with implementation details. As an example, we analysed and corrected the Woo and Lam symmetric key authentication protocol.

6. Acknowledgements

We would like to thank Shawn Koppenhöfer for commenting on an early version of this paper. We are also grateful to the anonymous referees for their useful suggestions.

References

[1] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 36–47, 1997.

[2] M. Abadi and R. Needham. Prudent engineering practice for cryptographic protocols. In *Proceedings of the IEEE CS Symposium on Research in Security and Privacy*, pages 122–136, 1994.

[3] M. Abadi and M. R. Tuttle. A semantics for a logic of authentication. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 201–216, 1991.

[4] J. Alves-Foss and T. Soule. A weakest precondition calculus for analysis of cryptographic protocols. In *DIMACS Workshop on Design and Formal Verifictaion of Security Protocols*, 1997.

[5] P. Bieber. A logic of communication in a hostile environment. In *Proceedings of the Computer Security Foundations Workshop III*, pages 14–22, June 1990.

[6] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, February 1990.

[7] M. Debbabi, M. Mejri, N. Tawbi, and I. Yahmadi. A new algorithm for the automatic verification of authentication protocols: From specifications to flaws and attack scenarios. In *DIMACS Workshop on Design and Formal Verifictaion of Security Protocols*, 1997.

[8] D. E. Denning and G. M. Sacco. Timestamps in key distribution protocols. *Communications of the ACM*, 24(8):198–208, 1981.

[9] L. Gong, R. Needham, and R. Yahalom. Reasoning about belief in cryptographic protocols. In *Proceedings of the IEEE CS Symposium on Research in Security and Privacy*, pages 234–248, 1990.

[10] S. Gritzalis, N. Nikitakos, and P. Georgiadis. Formal methods for the analysis and design of cryptographic protocols: A state-of-the-art review. In *Proceedings of the IFIP Working Conference on Communications and Multimedia Security, Vol. 3.*, pages 119–132, 1997.

[11] G. E. Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. Routledge, 1996.

[12] A. Huima and T. Aura. Using a multimodal logic to express conflicting interests in security protocols. In *DIMACS Workshop on Design and Formal Verifictaion of Security Protocols*, 1997.

[13] R. Kemmerer, C. Meadows, and J. Millan. Three systems for cryptographic protocol analysis. *Journal of Cryptology*, 7(2):79–130, 1994.

[14] B. Lampson, M. Abadi, M. Burrows, and E. Wobber. Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems*, 10(4):265–310, November 1992.

[15] C. A. Meadows. Formal verification of cryptographic protocols: A survey. In *Advances in Cryptology - ASIACRYPT'94*, pages 135–150. Springer-Verlag, 1995.

[16] D. M. Nessett. A critique of the burrows, abadi and needham logic. *Operating Systems Review*, 24(2):35–38, April 1990.

[17] P. V. Rangan. An axiomatic basis of trust in distributed systems. In *Proceedings of the IEEE CS Symposium on Research in Security and Privacy*, pages 204–211, April 1988.

[18] R. A. Rueppel. A formal approach to security architectures. In *EUROCRYPT'91*, pages 387–398, 1991.

[19] G. J. Simmons. How to (selectively) broadcast a secret. In *Proceedings of the IEEE CS Symposium on Research in Security and Privacy*, pages 108–113, 1985.

[20] P. F. Syverson. Formal semantics for logics of cryptographic protocols. In *Proceedings of the Computer Security Foundations Workshop III*, pages 32–41, June 1990.

[21] P. F. Syverson. A new look at an old protocol. *Operating Systems Review*, 30(3):1–4, 1996.

[22] P. F. Syverson and P. C. van Oorschot. On unifying some cryptographic protocol logics. In *Proceedings of the IEEE CS Symposium on Research in Security and Privacy*, pages 14–28, 1994.

[23] D. Trček and B. J. Blažič. Formal language for security services base modelling and analysis. *Computer Communications*, 18(12):921–928, December 1995.

[24] E. Wobber, M. Abadi, M. Burrows, and B. Lampson. Authentication in the TAOS operating system. *ACM Transactions on Computer Systems*, 12(1):3–32, February 1994.

[25] T. Y. C. Woo and S. S. Lam. Authentication for distributed systems. *Computer*, 25(1):39–52, 1992.