



A TABLEAUX METHOD FOR DOLEV-YAO MULTI-AGENT EPISTEMIC LOGIC

Luiz Cláudio Frederico Fernandez

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Mario Roberto Folhadela
Benevides

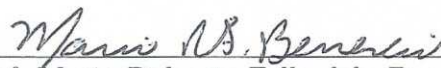
Rio de Janeiro
Abril de 2018

A TABLEUX METHOD FOR DOLEV-YAO MULTI-AGENT EPISTEMIC
LOGIC

Luiz Cláudio Frederico Fernandez

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO
ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE
ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE
JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A
OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE
SISTEMAS E COMPUTAÇÃO.

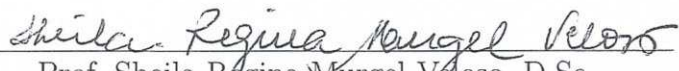
Examinada por:



Prof. Mario Roberto Folhadela Benevides, Ph.D.



Prof. Gerson Zaverucha, Ph.D.



Prof. Sheila Regina Murgel Veloso, D.Sc.

RIO DE JANEIRO, RJ – BRASIL

ABRIL DE 2018

Fernandez, Luiz Cláudio Frederico

A Tableaux Method for Dolev-Yao Multi-Agent Epistemic Logic/Luiz Cláudio Frederico Fernandez. – Rio de Janeiro: UFRJ/COPPE, 2018.

XI, 51 p.: il.; 29, 7cm.

Orientador: Mario Roberto Folhadela Benevides

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2018.

Referências Bibliográficas: p. 42 – 44.

1. tableaux. 2. epistemic. 3. logic. 4. security.

I. Benevides, Mario Roberto Folhadela. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

*To my family, friends and
girlfriend, for all the support.*

Acknowledgments

I would like to thank all my family, specially my parents, Doris and Claudionor, and my sister, Line, for all the love and understanding over these years.

To all my friends, that even from afar, I am always with them.

I also wish to thank my advisor, Mario Benevides, for the patient guidance and for all the opportunities since I was an undergraduate student.

To my coworkers from CAPGov/Lemobs, for all the support and encouragement.

I must also express my gratitude to Professor Ivan Varzinczak, for spending some time with our project and for introducing us to Simon Kramer.

Finally, a special thank you to my girlfriend, Anna Carolina, for the companionship in this long walk.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM MÉTODO TABLEAUX PARA LÓGICA EPISTÊMICA MULTI-AGENTE DOLEV-YAO

Luiz Cláudio Frederico Fernandez

Abril/2018

Orientador: Mario Roberto Folhadela Benevides

Programa: Engenharia de Sistemas e Computação

Dada a importância dos protocolos de segurança no nosso cotidiano, os esforços para desenvolver mecanismos e modelos para verificação de tais protocolos são sempre relevantes. Neste trabalho, nós propomos a Lógica Epistêmica Multi-Agente Dolev-Yao, uma extensão da Lógica Epistêmica Multi-Agente, destinada para a análise de protocolos de segurança e inspirada no modelo Dolev-Yao, o trabalho precursor sobre criptografia formal. Nós provamos a corretude e completude do nosso sistema, também demonstrando o seu uso. Em seguida, um método tableaux para essa lógica é apresentado, também incluindo sua corretude e completude. Por último, mostramos uma prova de terminação para o nosso método, além de alguns exemplos.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A TABLEAUX METHOD FOR DOLEV-YAO MULTI-AGENT EPISTEMIC LOGIC

Luiz Cláudio Frederico Fernandez

April/2018

Advisor: Mario Roberto Folhadela Benevides

Department: Systems Engineering and Computer Science

Given the increasing importance of security protocols in our daily lives, the efforts to develop mechanisms and models for verification of such protocols are always relevant. In this work, we propose the Dolev-Yao Multi-Agent Epistemic Logic, which is an extension of Multi-Agent Epistemic Logic, aimed to analyze security protocols and inspired by Dolev-Yao model, the seminal work in formal cryptography. We prove the soundness and completeness of our system, also demonstrating its use. Then, a tableaux method for this logic is presented, including the proofs of soundness and completeness. Finally, we provide a termination argument for our method and show some examples.

Contents

List of Figures	x
List of Tables	xi
1 Introduction	1
1.1 Objectives	2
1.2 Roadmap	2
2 Background	3
2.1 Dolev-Yao model	3
2.1.1 Public key protocols	3
2.1.2 Examples	4
2.1.3 Rules	6
2.2 Multi-agent epistemic logic	7
2.2.1 Language and semantics	7
2.2.2 Axiomatization	10
2.2.3 Example	10
2.3 Tableaux method	11
2.3.1 Semantics	12
2.3.2 Rules	12
2.3.3 Example	13
3 Dolev-Yao Multi-Agent Epistemic Logic	15
3.1 Language and semantics	15
3.2 Axiomatization	17
3.3 Soundness	18
3.4 Completeness	19
3.5 Examples	23
4 Tableaux Method for Dolev-Yao Multi-Agent Epistemic Logic	29
4.1 Semantics	29
4.2 Rules	29

4.3	Soundness	30
4.4	Completeness	31
4.5	Termination property	34
4.5.1	Classical and modal rules	34
4.5.2	Dolev-Yao Multi-Agent Epistemic Logic rules	35
4.6	Examples	35
5	Conclusion	39
5.1	Knowledge <i>de re/de dicto</i>	39
5.2	Common knowledge	40
5.3	Adding actions	40
	Bibliography	42
A	Kerberos Protocol	45
B	Andrew Secure RPC Handshake Protocol	49

List of Figures

2.1	Illustration of Example 2.1.1	4
2.2	Illustration of Example 2.1.2	5
2.3	Illustration of Example 2.1.3	6
2.4	Reflexive relation	8
2.5	Transitive relation	9
2.6	Symmetric relation	9
2.7	Epistemic model <i>Hexa</i> , based on DITMARSCH <i>et al.</i> [1]	11

List of Tables

4.1	Components of a type- α formula	32
4.2	Components of a type- β formula	32

Chapter 1

Introduction

Security protocols are increasingly present in our daily lives. They help us in banking transactions, communication, file downloads, that is, information access in general, protection. There are risks such as key or password cracking, the tracking of these actions, an user can impersonate another user and so on. A good implementation is a difficult issue given the saboteur's behavior possibilities.

Most security protocols are based on *one-way functions*, which is a good way of encryption, since it uses functions that are easy to compute, but hard to invert without knowing a specific complementary information. We can find works related to the logical verification of such specifications, also called *formal cryptography* [2–5], and the ones involving concepts from areas such as probability and process algebra, which consists the so-called *computational cryptography* [6–8]. There are even works that try to combine both approaches, pioneered by ABADI and ROGAWAY [9].

The seminal work known as *Dolev-Yao model* [10] initiated the formal approach in the early 1980's. Here, we consider a perfect encryption scheme and the models are obtained from functions of encryption and decryption. They show the vulnerability of the protocol by demonstrating a logical error in the specification. This has inspired several proposal, including this dissertation.

Multi-agent epistemic logics are designed to reason about knowledge of agents and groups [11]. As pointed by DITMARSCH *et al.* [1], nowadays, these logics are influenced by the development of modal logics and the system $\mathcal{S5}$ is the most popular one. Its use is relevant in many distinct areas, such as philosophy, economics, linguistics, cryptography and computer science.

There are many different automated theorem provers, including resolution, natural deduction and tableaux, from different approaches, namely *direct* or *indirect deduction* and *labeled deductive systems* [12]. In the latter, we have *prefixed tableaux* [13], which has a quite simple proof representation, similar to *Kripke semantics* [14].

1.1 Objectives

Our main objectives in this work are:

1. to present the *Dolev-Yao Multi-Agent Epistemic Logic*, namely $\mathcal{S5}_{DY}$, a novel multi-agent epistemic logic for reasoning about properties in protocols. It uses structured propositions, which is a new technique to deal with messages, keys and properties in security protocols in uniform manner, keeping the logic propositional; and
2. to provide a tableaux method for Dolev-Yao Multi-Agent Epistemic Logic, also discussing the soundness and completeness of the system and presenting its termination argument.

1.2 Roadmap

This dissertation is planned in the following manner:

- in the next chapter, we present the background of the proposal: the Dolev-Yao model for analysis of protocols, the multi-agent epistemic logic $\mathcal{S5}$ and the tableaux method;
- Chapter 3 introduces the $\mathcal{S5}_{DY}$, illustrating its use with some examples and showing soundness and completeness;
- in Chapter 4, we provide a sound and complete tableaux method for $\mathcal{S5}_{DY}$ and discuss its termination;
- finally, Chapter 5 concludes this dissertation and provides some final remarks.

Chapter 2

Background

This chapter presents the required topics for elaboration of our work. First, we present the Dolev-Yao model, including a brief explanation about public key protocols and some examples. Then, the formalization of multi-agent epistemic logic $\mathcal{S5}$ and, finally, the theorem prover known as tableaux method are given.

2.1 Dolev-Yao model

Introduced in [10], at the time of great discussion about the use of public key encryption in network communication, this work intends to show why a formal model is desirable to deal with security protocols.

Public key systems are efficient when we have a “passive” saboteur (also called eavesdropper, attacker, intruder and so on), one who only intercepts the communication and tries to decode the message. But NEEDHAM and SCHROEDER [15] already had pointed out that a not well specified protocol permits an “active” intruder, one who may fake his identity and manipulate the intercepted message, to succeed.

2.1.1 Public key protocols

In this system, based on [16, 17], we assume that every user X in the network has an *encryption function* E_X , which generates a pair (X, E_X) , inserted in a secure public directory, and a *decryption function* D_X , known only by X . The main requirements on the functions above are:

- $D_X(E_X(M)) = M$;
- for any user Y , knowing $E_X(M)$ and the directory containing all the public pairs does not reveal anything about M .

So, other users can communicate with X sending an encrypted message $E_X(M)$ and X can decrypt it using $D_X(E_X(M)) = M$, but only X gets M , even if $E_X(M)$ is accessible to them.

A message transmitted between two users is denoted by: the sender's name, the text (encrypted) and the receiver's name. One of the basic assumptions on the perfect public key system is that the functions are unbreakable.

2.1.2 Examples

To illustrate intruder's possible behaviours, let's consider the following examples.

Example 2.1.1. *In this case, the receiver always replies a message using the sender's public key. User A wants to send a plaintext M to user B :*

- a) A sends message $(A, E_B(M), B)$ to B [Figure 2.1(a)];
- b) Intruder Z intercepts the above message and sends message $(Z, E_B(M), B)$ to B [Figure 2.1(b)];
- c) B sends message $(B, E_Z(M), Z)$ to Z [Figure 2.1(c)];
- d) Z decodes $E_Z(M)$ and obtains M .

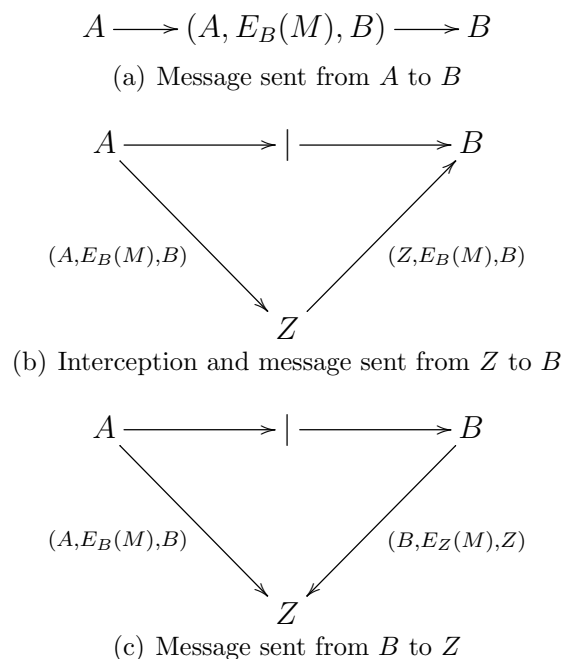


Figure 2.1: Illustration of Example 2.1.1

As suggested in [15], it's possible to improve this protocol encoding the sender's name concatenated with the plaintext.

Example 2.1.2. Now, the receiver uses the public key of the indicated user that is encrypted with the plaintext, not the sender's one (where MA is the concatenation of M and A):

- a) A sends message $(A, E_B(MA), B)$ to B [Figure 2.2(a)];
- b) Z intercepts the above message and sends message $(Z, E_B(MA), B)$ to B [Figure 2.2(b)];
- c) B sends message $(B, E_A(MB), Z)$ to Z [Figure 2.2(c)];
- d) Z cannot decode $E_A(MB)$ to obtain M .

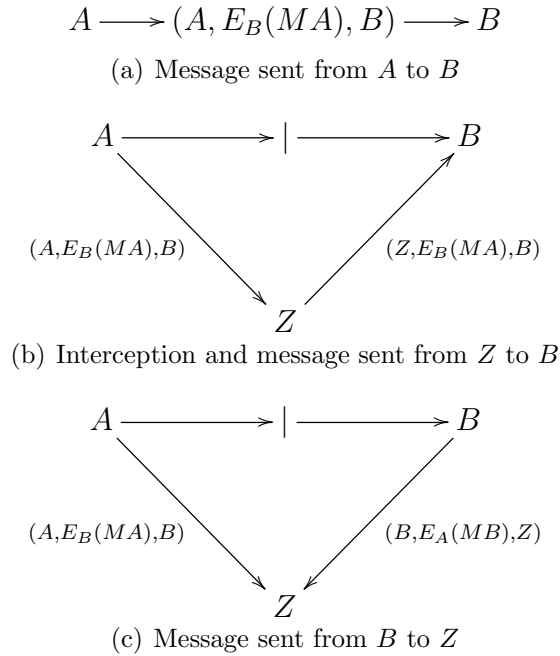


Figure 2.2: Illustration of Example 2.1.2

In the next example, the sender adds another layer of encryption to the message but, surprisingly, it becomes an insecure protocol.

Example 2.1.3. In this case, the receiver also uses the public key of the indicated user that is encrypted with the plaintext.

- a) A sends message $(A, E_B(E_B(M)A), B)$ to B [Figure 2.3(a)];
- b) B sends message $(B, E_A(E_A(M)B), A)$ to A [Figure 2.3(a)];
- c) Z intercepts the above message. Let's denote $E_A(M)B$ by \tilde{M} , then Z has $E_A(\tilde{M})$ from the intercepted message. Now, Z starts a communication with A sending him message $(Z, E_A(E_A(\tilde{M})Z), A)$ [Figure 2.3(c)];

- d) A sends message $(A, E_Z(E_Z(\tilde{M})A), Z)$ to Z [Figure 2.3(d)];
- e) Z decodes $E_Z(\tilde{M})$, obtains $E_A(M)$ and sends message $(Z, E_A(E_A(M)Z), A)$ to A [Figure 2.3(e)];
- f) A sends message $(A, E_Z(E_Z(M)A), Z)$ to Z [Figure 2.3(f)];
- g) Z decodes $E_Z(M)$ and obtains M .

$$A \longrightarrow (A, E_B(E_B(M)A), B) \longrightarrow B \quad A \longleftarrow (B, E_A(E_A(M)B), A) \longleftarrow B$$

(a) Message sent from A to B (b) Message sent from B to A

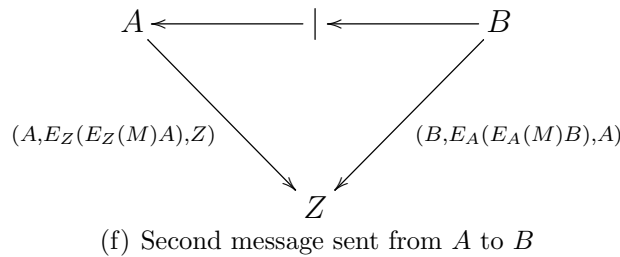
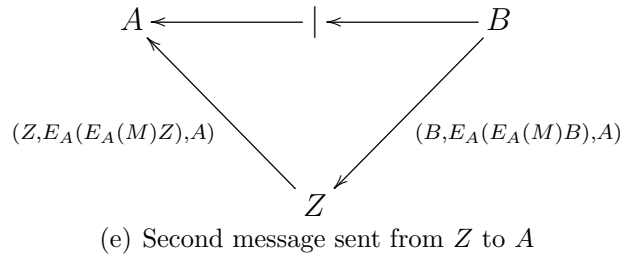
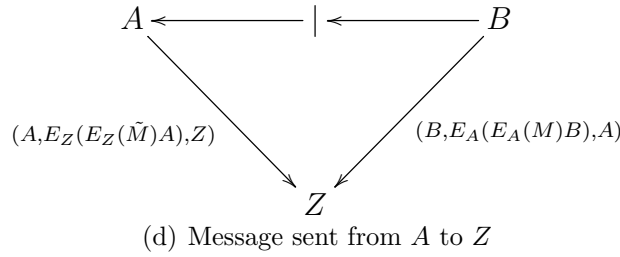
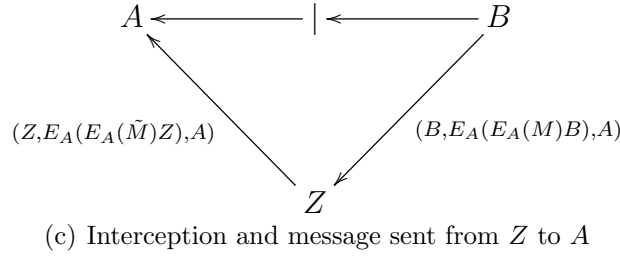


Figure 2.3: Illustration of Example 2.1.3

2.1.3 Rules

These rules are not formulated in the original paper, but we can easily obtain them from the theory presented there. They permit the intruder to make deductions from

the intercepted and sent messages.

Here, we are assuming an enumerable set $\mathcal{K} = \{k_1, \dots\}$ of keys, T as all the information that the intruder has and an encryption function $\{M\}_k$, which encrypts the message M under the key k .

The *entailment* relation $T \vdash M$ has the intuitive meaning that M can be computed from T . This relation is defined inductively, in a natural deduction-like system. Works such as [18, 19] have defined a similar notation:

$$\text{Reflexivity} \frac{M \in T}{T \vdash M}$$

$$\text{Encryption} \frac{T \vdash M \quad T \vdash k}{T \vdash \{M\}_k} \quad \text{Decryption} \frac{T \vdash \{M\}_k \quad T \vdash k}{T \vdash M}$$

$$\text{Pair-Composition} \frac{T \vdash M \quad T \vdash N}{T \vdash (M, N)}$$

$$\text{Pair-Decomposition} \frac{T \vdash (M, N)}{T \vdash M} \quad \frac{T \vdash (M, N)}{T \vdash N}$$

2.2 Multi-agent epistemic logic

The study of knowledge, *epistemology*, and modalities started with the early Greek philosophers [11, 20], but it was VON WRIGHT [21] who motivated a formal logical analysis of reasoning about knowledge, while HINTIKKA [22] extended these notions, resulting in the epistemic logic as we know it today.

Using Kripke structure, the multi-agent approach allows us to represent knowledge and belief of an agent or a group of agents. It's useful for many applications, such as puzzles, negotiations and protocols.

2.2.1 Language and semantics

The multi-agent epistemic logic $\mathcal{S5}$ presented below is based on [1, 11].

Definition 2.2.1. *The multi-agent epistemic language consists of an enumerable set Φ of propositional symbols, a finite set \mathcal{A} of agents, the Boolean connectives \neg and \wedge and a modality K_a for each agent a . The formulae are defined as follows, represented in BNF-notation:*

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid K_a\varphi$$

where $p \in \Phi$, $a \in \mathcal{A}$ and $K_a\varphi$ is intended to mean that “agent a knows φ ”.

We are considering the standard abbreviations and conventions:

- $\perp \quad \equiv \quad \neg\top$
- $\varphi_1 \vee \varphi_2 \quad \equiv \quad \neg(\neg\varphi_1 \wedge \neg\varphi_2)$
- $\varphi_1 \rightarrow \varphi_2 \quad \equiv \quad \neg(\varphi_1 \wedge \neg\varphi_2)$
- $\varphi_1 \leftrightarrow \varphi_2 \quad \equiv \quad (\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1) \quad \equiv \quad \neg(\varphi_1 \wedge \neg\varphi_2) \wedge \neg(\varphi_2 \wedge \neg\varphi_1)$
- $B_a\varphi \quad \equiv \quad \neg K_a\neg\varphi$

$B_a\varphi$ may be read as “agent a believes φ ”.

Definition 2.2.2. A multi-agent epistemic frame is a tuple $\mathcal{F} = \langle W, \sim_a \rangle$ where:

- W is a non-empty set of states;
- \sim_a is a reflexive, transitive and symmetric binary relation over W , for each $a \in \mathcal{A}$.

In this case, we say that \sim_a is an equivalence relation.

Formally, the property of *Reflexivity* is defined by:

$$\text{for all } w \in W, w \sim_a w$$

which means that any state is accessible from itself [Figure 2.4].

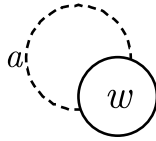


Figure 2.4: Reflexive relation

Transitivity is defined by:

$$\text{for all } w, v, u \in W, \text{ if } w \sim_a v \text{ and } v \sim_a u \text{ then } w \sim_a u$$

which means that, if there exists a state that is accessible via an intermediate then the former is also directly accessible [Figure 2.5].

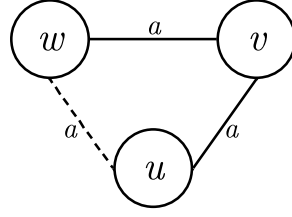


Figure 2.5: Transitive relation

Finally, *Symmetry* is defined by:

for all $w, v \in W$, if $w \sim_a v$ then $v \sim_a w$

that is, the relation is reversible [Figure 2.6].

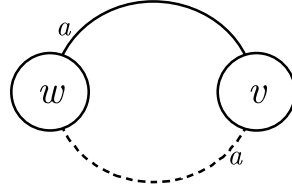


Figure 2.6: Symmetric relation

Definition 2.2.3. A multi-agent epistemic model is a pair $\mathcal{M} = \langle \mathcal{F}, V \rangle$, where \mathcal{F} is an epistemic frame and V is a valuation function $V : \Phi \rightarrow 2^W$. We call a rooted multi-agent epistemic model (\mathcal{M}, w) an epistemic state and we will often write \mathcal{M}, w rather than (\mathcal{M}, w) .

Definition 2.2.4. Let $\mathcal{M} = \langle W, \sim_a, V \rangle$ be a multi-agent epistemic model. The notion of satisfaction $\mathcal{M}, w \models \varphi$ is defined as follows (where iff stands for “if and only if”):

1. $\mathcal{M}, w \models \top$ always
2. $\mathcal{M}, w \models p$ iff $w \in V(p)$
3. $\mathcal{M}, w \models \neg\alpha$ iff $\mathcal{M}, w \not\models \alpha$
4. $\mathcal{M}, w \models \alpha_1 \wedge \alpha_2$ iff $\mathcal{M}, w \models \alpha_1$ and $\mathcal{M}, w \models \alpha_2$
5. $\mathcal{M}, w \models K_a\alpha$ iff for all $w' \in W$, if $w \sim_a w'$ then $\mathcal{M}, w' \models \alpha$

2.2.2 Axiomatization

The axioms and inference rules (also called *derivation rules*) for $\mathcal{S5}$ are given below:

Axioms

1. *All instantiations of propositional tautologies.*
2. $K_a(\varphi \rightarrow \psi) \rightarrow (K_a\varphi \rightarrow K_a\psi)$
3. $K_a\varphi \rightarrow \varphi$
4. $K_a\varphi \rightarrow K_aK_a\varphi$ [*positive introspection*]
5. $\neg K_a\varphi \rightarrow K_a\neg K_a\varphi$ [*negative introspection*]

Inference Rules

$$\text{Modus Ponens } \frac{\varphi \quad \varphi \rightarrow \psi}{\psi} \qquad \text{Universal Generalization } \frac{\varphi}{K_a\varphi}$$

$$\text{Substitution } \frac{\varphi}{\sigma\varphi}$$

where σ is a map uniformly substituting formulae for propositional variables.

2.2.3 Example

This example is found in [1]. There are three players, a (Anne), b (Bill) and c (Cath). Each one receives a card. They don't know each other's card and they can only see their own card; they know these facts, they know that they know them, and so on. We denote by $0_x, 1_x, 2_x$, for $x \in \{a, b, c\}$, each card that each player holds in their hand and each state by every possible deal of cards, for example, state 210 represents the state that Anne holds card 2, Bill holds card 1 and Cath holds card 0. Suppose that, in fact, $0_a, 1_b$ and 2_c are true. Leaving out all reflexive relations (to provide an economical representation of the model), we have the following epistemic model $Hexa = \langle W, \sim, V \rangle$ [Figure 2.7]:

- $W = \{012, 021, 102, 120, 201, 210\}$
- $\sim_a = \{(012, 021), (102, 120), (201, 210)\}$
- $\sim_b = \{(021, 120), (012, 210), (102, 201)\}$
- $\sim_c = \{(012, 102), (021, 201), (120, 210)\}$
- $V(0_a) = \{012, 021\}$, $V(1_a) = \{102, 120\}$, $V(2_a) = \{201, 210\}$

- $V(0_b) = \{102, 201\}$, $V(1_b) = \{012, 210\}$, $V(2_b) = \{021, 120\}$
- $V(0_c) = \{120, 210\}$, $V(1_c) = \{021, 201\}$, $V(2_c) = \{012, 102\}$

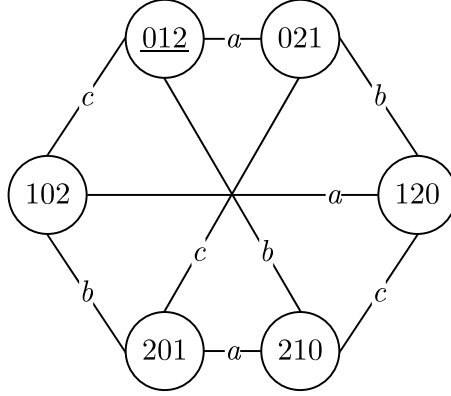


Figure 2.7: Epistemic model *Hexa*, based on DITMARSCH *et al.* [1]

These equivalence relations represent the players' ignorance about each other players' cards, since they can only see their own card. Thus, by the actual distribution of card, we have the epistemic state (*Hexa*, 012) and we can assert some formulae:

- $Hexa, 012 \models K_a 0_a$
- $Hexa, 012 \models K_b 1_b$
- $Hexa, 012 \models K_c 2_c$

So, in this epistemic state, it holds that 'Anne knows she holds card 0', 'Bill knows he holds card 1' and 'Cath knows she holds card 2'. We also have, for instance, that 'Bill doesn't know Anne's card' and 'Anne knows that Bill doesn't know her card', formally represented as follows:

- $Hexa, 012 \models \neg(K_b 0_a \vee K_b 1_a \vee K_b 2_a)$
- $Hexa, 012 \models K_a \neg(K_b 0_a \vee K_b 1_a \vee K_b 2_a)$

2.3 Tableaux method

We choose the tableau method theorem-proving for our proposal motivated by its Kripke model-like representation, which is very intuitive to work with. This system is a tree-structured *refutational method*, which, to prove a formula φ , we start the proof supposing $\neg\varphi$ and then we try to obtain unsatisfiable subformulae in each

branch from this negation. If *every* branch is unsatisfiable, then $\neg\varphi$ is unsatisfiable as well, therefore φ is valid. We can also consider it in the sense of *logical consequence* checking: for a database $DB = \{\phi_1, \dots, \phi_n\}$ and a question φ ,

$$DB \models \varphi$$

if and only if $(\phi_1 \wedge \dots \wedge \phi_n) \rightarrow \varphi$ is a tautology, that is, if its negation is a contradiction.

2.3.1 Semantics

The method presented below is based on the tableaux method for modal logics found in [23, 24]. As our main concern at the moment is to prove and model the deductions resulted from the bad behaviour of a particular agent, the intruder, certain changes were made to adapt the method to our needs, slightly abusing the notation.

Definition 2.3.1. *A branch θ of a tableau \mathcal{T} is closed if there is φ and $\neg\varphi$ for any formula φ .*

Definition 2.3.2. *A tableau \mathcal{T} is closed if every branch is closed.*

For $\mathcal{S5}$, we must use the sub-tableaux concept to obtain a refutation. A sub-tableau intends to simulate the possible world relation. So, if a sub-tableau is closed, the branch that originated it also closes. As we should use rules that creates a new sub-tableau or add a new formula to previously generated one, we need a mechanism to label it.

Definition 2.3.3. *A prefix is any expression used to name a tableau.*

Each tableau will have a different name, so a formula φ in a tableau refutation is unique, identified by (σ, φ) , where σ is the prefix.

To manage the creation of new tableaux and the addition of new formulae to a previously generated tableau, we denote ρ as the operator which applied on a formula (σ, φ) it will:

- create a new tableau σ' , starting with φ , if σ' is not a name for a previously generated tableau subordinated to the branch which φ holds; or
- add φ to the tableau specified by the prefix σ .

2.3.2 Rules

First, we present the propositional tableaux rules, for all formulae α and β :

$$\begin{array}{ccccc}
R_{\wedge} \frac{\alpha \wedge \beta}{\alpha} & R_{\text{Dneg}} \frac{\neg\neg\alpha}{\alpha} & R_{\wedge}^{\neg} \frac{\neg(\alpha \wedge \beta)}{\neg\alpha \quad \neg\beta} & R_{\rightarrow} \frac{\alpha \rightarrow \beta}{\neg\alpha \quad \beta} & R_{\rightarrow}^{\neg} \frac{\neg(\alpha \rightarrow \beta)}{\alpha \quad \neg\beta} \\
\beta & & & &
\end{array}$$

When rules R_{\wedge} , R_{Dneg} and R_{\rightarrow}^{\neg} are applied, we add the derived subformulae in the same branch of the original formula, while rules R_{\wedge}^{\neg} and R_{\rightarrow} splits the original branch.

The rules for $\mathcal{S5}$ are defined as follows, based on [24]:

$$\begin{array}{c}
R_{\pi} \frac{\neg K_a \alpha}{\rho(\mathcal{T}', \neg\alpha)}, \text{ where } \mathcal{T}' \text{ is a new tableau} \\
R_t \frac{K_a \alpha}{\alpha} \quad R_4^r \frac{\rho(\mathcal{T}'', K_a \alpha)}{K_a \alpha} \\
R_4 \frac{K_a \alpha}{\rho(\mathcal{T}'', K_a \alpha)}, \text{ where } \mathcal{T}'' \text{ is a previously generated tableau}
\end{array}$$

The subscripts t and 4 indicate the correspondence between axioms $K_a \varphi \rightarrow \varphi$ and $K_a \varphi \rightarrow K_a K_a \varphi$, respectively, and properties of accessibility relations.

2.3.3 Example

In order to illustrate the usage of tableaux method, we give the following examples:

Example 2.3.1. *Let's prove the theorem $\alpha \rightarrow (\beta \rightarrow (\alpha \wedge \beta))$:*

1. $\neg(\alpha \rightarrow (\beta \rightarrow (\alpha \wedge \beta)))$ [negation of the question]
 2. α [from 1, by R_{\rightarrow}^{\neg}]
 3. $\neg(\beta \rightarrow (\alpha \wedge \beta))$ [from 1, by R_{\rightarrow}^{\neg}]
 4. β [from 3, by R_{\rightarrow}^{\neg}]
 5. $\neg(\alpha \wedge \beta)$ [from 3, by R_{\rightarrow}^{\neg}]
-
6. $\neg\alpha \quad \neg\beta$ [from 5, by R_{\wedge}^{\neg}]

Since each of the branches is closed, we have a closed tableau.

Example 2.3.2. *Let's prove the theorem $(K_a \alpha \wedge K_a \beta) \rightarrow K_a(\alpha \wedge \beta)$:*

1. $\neg((K_a \alpha \wedge K_a \beta) \rightarrow K_a(\alpha \wedge \beta))$ [negation of the question]
2. $K_a \alpha \wedge K_a \beta$ [from 1, by R_{\rightarrow}^{\neg}]

3. $\neg K_a(\alpha \wedge \beta)$ [from 1, by R_{\neg}]

4. $K_a\alpha$ [from 2, by R_{\wedge}]

5. $K_a\beta$ [from 2, by R_{\wedge}]

Now, we generate a new tableau:

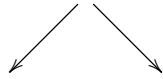
3.1. $\neg(\alpha \wedge \beta)$ [from 3, by R_{π}]

3.2. $K_a\alpha$ [from 4, by R_4]

3.3. $K_a\beta$ [from 5, by R_4]

3.4. α [from 3.2, by R_t]

3.5. β [from 3.3, by R_t]



3.6. $\neg\alpha \quad \neg\beta$ [from 3.1, by R_{\wedge}^{-}]

Since each of the branches is closed, we have a closed tableau.

Chapter 3

Dolev-Yao Multi-Agent Epistemic Logic

This section presents the $\mathcal{S5}_{DY}$, a multi-agent epistemic logic designed to analyze security protocols, based on Dolev-Yao model. We prove soundness and completeness of our system and also show some examples. An early version of this work was presented as a short paper at CSBC 2017 [25].

3.1 Language and semantics

There is a novelty in the language of $\mathcal{S5}_{DY}$: formulae are built from expressions and not only from propositional symbols. Intuitively, an expression is any piece of information that can be encrypted, decrypted or concatenated in order to be communicated.

Definition 3.1.1. *The Dolev-Yao multi-agent epistemic language consists of an enumerable set Φ of propositional symbols, a finite set \mathcal{A} of agents, an enumerable set of keys $\mathcal{K} = \{k_1, \dots\}$, the Boolean connectives \neg and \wedge and a modality K_a for each agent a . The expressions and formulae are defined as follows, represented in BNF-notation:*

$$E ::= p \mid k \mid (E_1, E_2) \mid \{E\}_k$$

where $k \in \mathcal{K}$ and $p \in \Phi$.

$$\varphi ::= e \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid K_a\varphi$$

where $e \in E$ and $a \in \mathcal{A}$.

We are also considering the standard abbreviations and conventions as specified in Definition 2.2.1.

Definition 3.1.2. A Dolev-Yao multi-agent epistemic frame is a tuple $\mathcal{F} = \langle S, \sim_a \rangle$ where:

- S is a non-empty set of states;
- \sim_a is a reflexive, transitive and symmetric binary relation over S , for each agent $a \in \mathcal{A}$.

Definition 3.1.3. A Dolev-Yao multi-agent epistemic model is a pair $\mathcal{M} = \langle \mathcal{F}, V \rangle$, where \mathcal{F} is a Dolev-Yao multi-agent epistemic frame and V is a valuation function $V : E \rightarrow 2^S$ satisfying the following conditions for all $m \in E$ and $k \in \mathcal{K}$:

1. $V(m) \cap V(k) \subseteq V(\{m\}_k)$
2. $V(\{m\}_k) \cap V(k) \subseteq V(m)$
3. $V(m) \cap V(n) = V((m, n))$

We call a rooted Dolev-Yao multi-agent epistemic model (\mathcal{M}, s) an epistemic state and again, we will often write \mathcal{M}, s rather than (\mathcal{M}, s) .

We will see that these conditions are necessary for the soundness and completeness proofs.

The first one ensures that, in any state, if we have a message m and a key k then we must be able to have the encrypted message $\{m\}_k$.

Condition 2 establishes that if we have an encrypted message $\{m\}_k$ and a key k then we must be able to decrypt it and obtain m .

Finally, the last condition says that, in any state, we have messages m and n if and only if we have the pair (m, n) .

Definition 3.1.4. Let $\mathcal{M} = \langle S, \sim_a, V \rangle$ be a Dolev-Yao multi-agent epistemic model. The notion of satisfaction $\mathcal{M}, s \models \varphi$ is defined as follows:

1. $\mathcal{M}, s \models \top$ *always*
2. $\mathcal{M}, s \models e$ *iff* $s \in V(e)$
3. $\mathcal{M}, s \models \neg\varphi$ *iff* $\mathcal{M}, s \not\models \varphi$
4. $\mathcal{M}, s \models \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{M}, s \models \varphi_1$ *and* $\mathcal{M}, s \models \varphi_2$
5. $\mathcal{M}, s \models K_a\varphi$ *iff* for all $s' \in S$, if $s \sim_a s'$ then $\mathcal{M}, s' \models \varphi$

3.2 Axiomatization

In the axiomatization of $\mathcal{S5}_{DY}$ we have the same axioms listed in Definition 2.2.2, plus the last three axioms of the following list:

Axioms

1. *All instantiations of propositional tautologies.*
2. $K_a(\varphi \rightarrow \psi) \rightarrow (K_a\varphi \rightarrow K_a\psi)$
3. $K_a\varphi \rightarrow \varphi$
4. $K_a\varphi \rightarrow K_aK_a\varphi$ [*positive introspection*]
5. $\neg K_a\varphi \rightarrow K_a\neg K_a\varphi$ [*negative introspection*]
6. $m \wedge k \rightarrow \{m\}_k$ [*encryption*]
7. $\{m\}_k \wedge k \rightarrow m$ [*decryption*]
8. $m \wedge n \leftrightarrow (m, n)$ [*pair composition & decomposition*]

Inference Rules

$$\text{Modus Ponens } \frac{\varphi \quad \varphi \rightarrow \psi}{\psi} \qquad \text{Universal Generalization } \frac{\varphi}{K_a\varphi}$$

$$\text{Substitution } \frac{\varphi}{\sigma\varphi}$$

where σ is a map uniformly substituting formulae for propositional variables.

Axioms 1, 2, 3, 4 and 5 are standard in multi-agent epistemic logics literature [11], while axioms 6, 7 and 8 enforce the semantical properties of the valuation function (conditions 1, 2 and 3 of Definition 3.1.3).

Lemma 3.2.1. *The following formulas are theorems of $\mathcal{S5}_{DY}$:*

1. $K_a m \wedge K_a k \rightarrow K_a \{m\}_k$
2. $K_a \{m\}_k \wedge K_a k \rightarrow K_a m$
3. $K_a m \wedge K_a n \leftrightarrow K_a (m, n)$

Proof. This proof is straightforward from axioms 2, 6, 7, 8, inference rule *Universal Generalization* and the fact that K_a distributes over conjunction, that is:

$$\vdash K_a(\varphi \wedge \psi) \leftrightarrow (K_a\varphi \wedge K_a\psi)$$

■

3.3 Soundness

We only prove the soundness of axioms 6, 7, 8. The others axioms and inference rules are standard in multi-agent epistemic logics and are well-known to be sound for the class of $\mathcal{S5}$ models.

Lemma 3.3.1. *The following axioms are sound with respect to the class of $\mathcal{S5}_{DY}$ models:*

1. $m \wedge k \rightarrow \{m\}_k$ [encryption]
2. $\{m\}_k \wedge k \rightarrow m$ [decryption]
3. $m \wedge n \leftrightarrow (m, n)$ [pair composition & decomposition]

Proof.

1. suppose we have a Dolev-Yao multi-agent epistemic model \mathcal{M} and a state s such that

$$\mathcal{M}, s \Vdash m \wedge k$$

Then we have that

$$\mathcal{M}, s \Vdash m$$

and

$$\mathcal{M}, s \Vdash k$$

But this is if and only if $s \in V(m)$ and $s \in V(k)$. By condition 1 of Definition 3.1.3, we have that $s \in V(\{m\}_k)$ and thus

$$\mathcal{M}, s \Vdash \{m\}_k$$

and

$$\mathcal{M}, s \Vdash m \wedge k \rightarrow \{m\}_k$$

- 2 & 3. analogous to the above proof, but we use conditions 2 and 3 of Definition 3.1.3, respectively.

■

We will also see in the next chapter that the contrapositive of these axioms are true, which will help us later on.

3.4 Completeness

Now we prove the completeness of $\mathcal{S5}_{DY}$ by *Canonical Models*, based on [26]. First, we need some definitions:

Definition 3.4.1 (Maximal Consistent Set). *Given a system \mathcal{S} and a set of formulae Γ , we say:*

1. Γ is \mathcal{S} -inconsistent if for some subset $\{\alpha_1, \dots, \alpha_n\} \subseteq \Gamma$ we have

$$\vdash_{\mathcal{S}} \neg(\alpha_1 \wedge \dots \wedge \alpha_n)$$

and Γ is \mathcal{S} -consistent if it is not \mathcal{S} -inconsistent;

2. Γ is maximal if for any formula α , either $\alpha \in \Gamma$ or $\neg\alpha \in \Gamma$;
3. Γ is maximal \mathcal{S} -consistent if it is both maximal and \mathcal{S} -consistent. In this case, we say that Γ is a \mathcal{S} -MCS.

Next, we list and prove the MCS properties:

Proposition 3.4.1 (MCS Properties). *Let Γ be a \mathcal{S} -MCS. Then for all formulae ϕ and ψ :*

1. either $\phi \in \Gamma$ or $\neg\phi \in \Gamma$, but not both;
2. Γ is closed under Modus Ponens: if $\phi \in \Gamma$ and $\phi \rightarrow \psi$ then $\psi \in \Gamma$;
3. $\phi \vee \psi \in \Gamma$ iff either $\phi \in \Gamma$ or $\psi \in \Gamma$;
4. $\phi \wedge \psi \in \Gamma$ iff both $\phi \in \Gamma$ and $\psi \in \Gamma$;

In particular, if Γ is a $\mathcal{S5}_{DY}$ -MCS then for all messages m and $\{m\}_k$, pair (m, n) and key k :

5. all theorems of $\mathcal{S5}_{DY} \subseteq \Gamma$;
6. if $m \in \Gamma$ and $k \in \Gamma$ then $\{m\}_k \in \Gamma$;
7. if $\{m\}_k \in \Gamma$ and $k \in \Gamma$ then $m \in \Gamma$;
8. $(m, n) \in \Gamma$ iff both $m \in \Gamma$ and $n \in \Gamma$.

Proof.

1. by maximality, one of them must be in Γ ;

2. suppose $\psi \notin \Gamma$, then $\{\phi, \phi \rightarrow \psi, \neg\psi\} \subseteq \Gamma$, which is an absurd because $\{\phi, \phi \rightarrow \psi, \neg\psi\}$ is \mathcal{S} -inconsistent. Therefore $\psi \in \Gamma$;
- 3 & 4. analogous to property 2;
5. for all theorems $\omega \in \mathcal{S5}_{DY}$, $\vdash_{\mathcal{S5}_{DY}} \omega$. Suppose $\neg\omega \in \Gamma$, as Γ is $\mathcal{S5}_{DY}$ -consistent, $\Gamma \vdash_{\mathcal{S5}_{DY}} \neg\omega$, which is a contradiction. Then $\neg\omega \notin \Gamma$. By maximality, $\omega \in \Gamma$. Therefore all theorems of $\mathcal{S5}_{DY} \subseteq \Gamma$;
- 6, 7 & 8. follow straightforward from properties 2 and 5.

■

Now, our aim is to state and prove *Lindenbaum's Lemma*, which shows that any consistent set of formulae can be extended to a MCS:

Lemma 3.4.1 (Lindenbaum's Lemma). *For any \mathcal{S} -consistent set Σ , there is a set Σ^+ such that:*

- $\Sigma \subseteq \Sigma^+$; and
- Σ^+ is a \mathcal{S} -MCS.

Proof. Let $\phi_0, \phi_1, \phi_2, \dots$ be an enumeration of formulae of our language. We define the set Σ^+ as the union of a chain of \mathcal{S} -consistent sets as follows:

- $\Sigma_0 = \Sigma$; $\Sigma_{i+1} = \begin{cases} \Sigma_i \cup \{\phi_{i+1}\}, & \text{if it is } \mathcal{S}\text{-consistent} \\ \Sigma_i \cup \{\neg\phi_{i+1}\}, & \text{otherwise} \end{cases}$

Claim: Σ_j is \mathcal{S} -consistent for any j . We prove that by induction on j .

Base case: $\Sigma_0 = \Sigma$ is \mathcal{S} -consistent by hypothesis.

Induction hypothesis: suppose that Σ_j is \mathcal{S} -consistent.

Now, we want to show that Σ_{j+1} is also consistent. By construction, we have:

$$\Sigma_{j+1} = \begin{cases} \Sigma_j \cup \{\phi_{j+1}\}, & \text{if it is } \mathcal{S}\text{-consistent} \\ \Sigma_j \cup \{\neg\phi_{j+1}\}, & \text{otherwise} \end{cases}$$

By the above construction we have directly that Σ_{j+1} is also \mathcal{S} -consistent. Thus, Σ_i is \mathcal{S} -consistent for any i .

- $\Sigma^+ \cup_{i \geq 0} \Sigma_i$. Now we have to prove that Σ^+ is a \mathcal{S} -MCS.

Σ^+ is \mathcal{S} -consistent. Because otherwise some finite subset of the set $\Sigma_i \subseteq \Sigma^+$ would be \mathcal{S} -inconsistent, but we just proved that any Σ_i is \mathcal{S} -consistent. Therefore, by item 1 of the definition of **Maximal Consistent Set** (Definition 3.4.1), Σ^+ is \mathcal{S} -consistent.

Σ^+ is maximal. Because given any formula ϕ , either $\phi \in \Sigma_j$ or $\neg\phi \in \Sigma_j$, for some j . Then $\Sigma_j \subseteq \Sigma^+$. So, Σ^+ is maximal.

Therefore Σ^+ is a \mathcal{S} -MCS. ■

The Canonical Model for \mathcal{S} is defined as follows:

Definition 3.4.2 (Canonical Model). *The canonical model \mathfrak{M} over \mathcal{S} is the triple $\langle S^{\mathcal{S}}, \sim_a^{\mathcal{S}}, V^{\mathcal{S}} \rangle$, where:*

1. $S^{\mathcal{S}}$ is the set of all \mathcal{S} -MCS;
2. $\sim_a^{\mathcal{S}}$ is the canonical relation, a binary relation on $S^{\mathcal{S}}$, for each agent $a \in \mathcal{A}$, defined by $s \sim_a^{\mathcal{S}} r$ if for all formula ψ , if $K_a\psi \in s$ then $\psi \in r$;
3. $V^{\mathcal{S}}$ is the canonical valuation, defined as $V^{\mathcal{S}}(e) = \{s \in S^{\mathcal{S}} \mid e \in s\}$, where $e \in E$.

$\mathfrak{F} = (S^{\mathcal{S}}, \sim_a^{\mathcal{S}})$ is called the canonical frame.

Next, we prove the *Existence Lemma*, in order to prove later the *Truth Lemma*:

Lemma 3.4.2 (Existence Lemma). *Let $\Gamma \in S^{\mathcal{S}}$ be a \mathcal{S} -MCS such that $B_a\phi \in \Gamma$. Then, there exists a \mathcal{S} -MCS Σ such that $\{\varphi \mid K_a\varphi \in \Gamma\} \cup \{\phi\} \subseteq \Sigma$.*

Proof. We first prove that $\Sigma^- = \{\varphi \mid K_a\varphi \in \Gamma\} \cup \{\phi\}$ is \mathcal{S} -consistent.

Suppose that Σ^- is \mathcal{S} -inconsistent. Then, there exists a finite subset $\varphi_1, \dots, \varphi_n$ such that $\neg(\varphi_1 \wedge \dots \wedge \varphi_n \wedge \phi)$ is a theorem:

$$\begin{aligned} & \vdash_{\mathcal{S}} \neg(\varphi_1 \wedge \dots \wedge \varphi_n \wedge \phi) \\ & \vdash_{\mathcal{S}} \varphi_1 \wedge \dots \wedge \varphi_n \rightarrow \neg\phi && \text{[propositional tautology]} \\ & \vdash_{\mathcal{S}} K_a(\varphi_1 \wedge \dots \wedge \varphi_n \rightarrow \neg\phi) && \text{[inference rule } \textit{Universal Generalization}] \\ & \vdash_{\mathcal{S}} K_a\varphi_1 \wedge \dots \wedge K_a\varphi_n \rightarrow K_a\neg\phi && \text{[axiom 2]} \end{aligned}$$

By hypothesis, $K_a\varphi_1 \in \Gamma, \dots, K_a\varphi_n \in \Gamma$, so, by property 2 of the **MCS Properties** (Proposition 3.4.1), $K_a\neg\phi \in \Gamma$, and also, by duality and as Γ is \mathcal{S} -MCS, $\neg B_a\phi \in \Gamma$, which is a contradiction. Thus, Σ^- is \mathcal{S} -consistent. By **Lindenbaum's Lemma** (Lemma 3.4.1), there exists a \mathcal{S} -MCS extension Σ that extends Σ^- . ■

Lemma 3.4.3 (Truth Lemma). *For any formula ϕ , $\mathfrak{M}^S, s \Vdash \phi$ iff $\phi \in s$.*

Proof. By induction on the length of ϕ .

Base case:

$$\mathfrak{M}^S, s \Vdash e \text{ iff } s \in V^S(e) \text{ iff } e \in s$$

Induction hypothesis: it holds for $|\phi| < i$:

$$\mathfrak{M}^S, s \Vdash \phi \text{ iff } \phi \in s$$

Booleans: follows from the property 1 of the **MCS Properties** (Proposition 3.4.1).

Knowledge operator:

\Rightarrow Suppose

$$\mathfrak{M}^S, s \Vdash K_a \phi \text{ (i)}$$

and $K_a \phi \notin s$. Thus, by maximality, we have that $B_a \neg \phi \in s$. So, by **Existence Lemma** (Lemma 3.4.2) there exists a r such that

$$\{\varphi \mid K_a \varphi \in s\} \cup \neg \phi \subseteq r \text{ (ii)}$$

By definition of **Canonical Model** (Definition 3.4.2) $s \sim_a^S r$. From (i), for all s' , if $s \sim_a^S s'$ then

$$\mathfrak{M}^S, s' \Vdash \phi$$

By the induction hypothesis, $\phi \in s'$ for all s' and in particular $\phi \in r$, which is a contradiction with (ii). Thus, $K_a \phi \in s$

\Leftarrow Suppose $K_a \phi \in s$ and

$$\mathfrak{M}^S, s \not\Vdash K_a \phi$$

then there exists a r such that $s \sim_a^S r$ and

$$\mathfrak{M}^S, r \Vdash \neg \phi$$

But by induction hypothesis, $\neg \phi \in r$. By **Canonical Model** (Definition 3.4.2) if $s \sim_a^S r$, for all formula ψ , if $K_a \psi \in s$ then $\psi \in r$. So, $\phi \in r$, which is a contradiction. Thus,

$$\mathfrak{M}^S, s \Vdash K_a \phi$$

■

Lemma 3.4.4. *The canonical model relations \sim_a^S are reflexive, transitive and symmetric.*

Proof. This follows from the definition of \sim_a^S and this proof can be found in epistemic and modal logics literature [1, 11, 26]. ■

Theorem 3.4.1. *The canonical model $\mathfrak{M}^{S5_{DY}}$ is a $S5_{DY}$ model.*

Proof. First we prove that $\mathfrak{M}^{S5_{DY}}$ satisfies conditions 1, 2 and 3 of Definition 3.1.3:

- Suppose we have $s \in V(m) \cap V(k)$ for a generic state $s \in S^{S5_{DY}}$. So, we have that $s \in V(m)$ and $s \in V(k)$. Also,

$$\mathfrak{M}^{S5_{DY}}, s \Vdash m$$

and

$$\mathfrak{M}^{S5_{DY}}, s \Vdash k$$

which entails

$$\mathfrak{M}^{S5_{DY}}, s \Vdash m \wedge k$$

As $S^{S5_{DY}}$ is a $S5_{DY}$ -MCS, all the axioms of $S5_{DY}$ are valid in s . Using axiom 6 and inference rule *Modus Ponens*, we have

$$\mathfrak{M}^{S5_{DY}}, s \Vdash \{m\}_k$$

Therefore, by the **Truth Lemma** (Lemma 3.4.3), we have that $\{m\}_k \in s$, that is, $s \in V(\{m\}_k)$. Thus, $V(m) \cap V(k) \subseteq V(\{m\}_k)$ (condition 1 of Definition 3.1.3).

- The proofs of conditions 2 and 3 of Definition 3.1.3 are analogous to the above proof, but we use axioms 7 and 8, respectively.

Together with Lemma 3.4.4, we are done. ■

Theorem 3.4.2. *Let Σ be a $S5_{DY}$ -consistent set of formulae. Then, Σ is satisfiable.*

Proof. By **Existence Lemma** (Lemma 3.4.2), there exists a $S5_{DY}$ -MCS Σ^+ such that $\Sigma \subseteq \Sigma^+$ and, by **Truth Lemma** (Lemma 3.4.3), $\mathfrak{M}^{S5_{DY}}, \Sigma^+ \models \Sigma$. ■

3.5 Examples

Let's return to the examples in subsection 2.1.2. At the moment, the protocols actions are represented in a kind of extra or metalanguage. The mentioned theorems refer to Lemma 3.2.1.

We have three agents, A, B and Z . Assuming that $k_{XY} = k_{YX}$ for every agent X and Y , KB stands for *Knowledge Base* and *i.k.* for *initial knowledge*:

Example 3.5.1. *Returning to Example 2.1.1, A wants to send a message m to user B . The receiver always replies a message using the key shared with the sender:*

$$\begin{array}{l}
0. \quad KB_0 = \{K_A k_{AB}, K_B k_{AB}, K_B k_{BZ}, K_Z k_{BZ}, K_A m\} \quad \text{i.k.} \\
\\
\begin{array}{c}
KB_0 \vdash K_A \{m\}_{k_{AB}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{AB}(\{m\}_{k_{AB}}) \\
\text{---} \\
\downarrow \text{Z intercepts} \\
KB_1 := KB_0 \cup K_Z \{m\}_{k_{AB}} \\
\downarrow \text{send}_{ZB}(\{m\}_{k_{AB}}) \\
KB_2 := KB_1 \cup K_B \{m\}_{k_{AB}}
\end{array} \\
1. \\
2. \\
\\
KB_2 \vdash K_B m \quad \text{thrm. 2} \\
\\
\begin{array}{c}
KB_2 \vdash K_B \{m\}_{k_{BZ}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{BZ}(\{m\}_{k_{BZ}}) \\
KB_3 := KB_2 \cup K_Z \{m\}_{k_{BZ}}
\end{array} \\
3. \\
\\
KB_3 \vdash K_Z m \quad \text{thrm. 2}
\end{array}$$

Intruder Z knows m .

Example 3.5.2. *Returning to Example 2.1.2, A also sends an encrypted message to agent B , but now the receiver always replies a message using the key shared with the indicated agent that is encrypted with the plaintext:*

$$0. \quad KB_0 = \{K_A k_{AB}, K_B k_{AB}, K_B k_{BZ}, K_Z k_{BZ}, K_A m\} \quad i.k.$$

$$KB_0 \vdash K_A(k_{AB}, m) \quad thrm. 3$$

$$KB_0 \vdash K_A\{(k_{AB}, m)\}_{k_{AB}} \quad thrm. 1$$

$$\begin{array}{c} send_{AB}(\{(k_{AB}, m)\}_{k_{AB}}) \downarrow \\ \text{---} \end{array}$$

$$\begin{array}{c} Z \text{ intercepts} \downarrow \end{array}$$

$$1. \quad KB_1 := KB_0 \cup K_Z\{(k_{AB}, m)\}_{k_{AB}}$$

$$\begin{array}{c} send_{ZB}(\{(k_{AB}, m)\}_{k_{AB}}) \downarrow \end{array}$$

$$2. \quad KB_2 := KB_1 \cup K_B\{(k_{AB}, m)\}_{k_{AB}}$$

$$KB_2 \vdash K_B(k_{AB}, m) \quad thrm. 2$$

$$KB_2 \vdash K_B m \quad thrm. 3$$

$$KB_2 \vdash K_B\{(k_{AB}, m)\}_{k_{AB}} \quad thrm. 1$$

$$\begin{array}{c} send_{BZ}(\{(k_{AB}, m)\}_{k_{AB}}) \downarrow \end{array}$$

$$3. \quad KB_3 := KB_2 \cup K_Z\{(k_{AB}, m)\}_{k_{AB}}$$

$$KB_3 \not\vdash K_Z m$$

Intruder Z does not know m.

Example 3.5.3. *Returning to Example 2.1.3, A sends a double-encrypted message to agent B and B replies, but the intruder intercepts the message and starts communicating with A. The receiver also replies a message using the key shared with the indicated agent that is encrypted with the plaintext:*

$$0. \quad KB_0 = \{K_A k_{AB}, K_A k_{AZ}, K_B k_{AB}, K_Z k_{AZ}, K_A m\} \quad i.k.$$

$$KB_0 \vdash K_A \{m\}_{k_{AB}} \quad thrm. 1$$

$$KB_0 \vdash K_A(k_{AB}, \{m\}_{k_{AB}}) \quad thrm. 3$$

$$KB_0 \vdash K_A \{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}} \quad thrm. 1$$

$$send_{AB}(\{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}}) \downarrow$$

$$1. \quad KB_1 := KB_0 \cup K_B \{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}}$$

$$KB_1 \vdash K_B(k_{AB}, \{m\}_{k_{AB}}) \quad thrm. 2$$

$$KB_1 \vdash K_B \{m\}_{k_{AB}} \quad thrm. 3$$

$$KB_1 \vdash K_B m \quad thrm. 2$$

$$KB_1 \vdash K_B \{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}} \quad thrm. 1$$

$$send_{BA}(\{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}}) \downarrow$$

— — —

$$Z \text{ intercepts} \downarrow$$

$$2. \quad KB_2 := KB_1 \cup K_Z \{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}}$$

$$KB_2 \vdash K_Z \{\tilde{m}\}_{k_{AB}} = K_Z \{(k_{AB}, \{m\}_{k_{AB}})\}_{k_{AB}}$$

$$KB_2 \vdash K_Z \{(k_{AZ}, \{\tilde{m}\}_{k_{AB}})\}_{k_{AZ}} \quad thrm. 1$$

$$send_{ZA}(\{(k_{AZ}, \{\tilde{m}\}_{k_{AB}})\}_{k_{AZ}}) \downarrow$$

$$3. \quad KB_3 := KB_2 \cup K_A \{(k_{AZ}, \{\tilde{m}\}_{k_{AB}})\}_{k_{AZ}}$$

$$KB_3 \vdash K_A(k_{AZ}, \{\tilde{m}\}_{k_{AB}}) \quad thrm. 2$$

$$KB_3 \vdash K_A \{\tilde{m}\}_{k_{AB}} \quad thrm. 3$$

$$KB_3 \vdash K_A \tilde{m} \quad thrm. 2$$

$$\begin{array}{l}
KB_3 \vdash K_A\{(k_{AZ}, \{\tilde{m}\}_{k_{AZ}})\}_{k_{AZ}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{AZ}(\{(k_{AZ}, \{\tilde{m}\}_{k_{AZ}})\}_{k_{AZ}}) \\
4. \quad KB_4 := KB_3 \cup K_Z\{(k_{AZ}, \{\tilde{m}\}_{k_{AZ}})\}_{k_{AZ}}
\end{array}$$

$$KB_4 \vdash K_Z(k_{AZ}, \{\tilde{m}\}_{k_{AZ}}) \quad \text{thrm. 2}$$

$$KB_4 \vdash K_Z\{\tilde{m}\}_{k_{AZ}} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_Z\tilde{m} \quad \text{thrm. 2}$$

$$KB_4 \vdash K_Z\tilde{m} = K_Z(k_{AB}, \{m\}_{k_{AB}})$$

$$KB_4 \vdash K_Z\{m\}_{k_{AB}} \quad \text{thrm. 3}$$

$$\begin{array}{l}
KB_4 \vdash K_Z\{(k_{AZ}, \{m\}_{k_{AB}})\}_{k_{AZ}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{ZA}(\{(k_{AZ}, \{m\}_{k_{AB}})\}_{k_{AZ}}) \\
5. \quad KB_5 := KB_4 \cup K_A\{(k_{AZ}, \{m\}_{k_{AB}})\}_{k_{AZ}}
\end{array}$$

$$KB_5 \vdash K_A(k_{AZ}, \{m\}_{k_{AB}}) \quad \text{thrm. 2}$$

$$KB_5 \vdash K_A\{m\}_{k_{AB}} \quad \text{thrm. 3}$$

$$KB_5 \vdash K_A m \quad \text{thrm. 2}$$

$$\begin{array}{l}
KB_5 \vdash K_A\{(k_{AZ}, \{m\}_{k_{AZ}})\}_{k_{AZ}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{AZ}(\{(k_{AZ}, \{m\}_{k_{AZ}})\}_{k_{AZ}}) \\
6. \quad KB_6 := KB_5 \cup K_Z\{(k_{AZ}, \{m\}_{k_{AZ}})\}_{k_{AZ}}
\end{array}$$

$$KB_6 \vdash K_Z(k_{AZ}, \{m\}_{k_{AZ}}) \quad \text{thrm. 2}$$

$$KB_6 \vdash K_Z\{m\}_{k_{AZ}} \quad \text{thrm. 3}$$

$$KB_6 \vdash K_Z m \quad \text{thrm. 2}$$

Intruder Z knows m.

Inspired by *BAN Logic* [2], we also tested our system on some well-known protocols: the *Kerberos* [27] and the *Andrew Secure RPC Handshake* [28] protocols. These analyses can be found in appendices A and B, respectively.

Chapter 4

Tableaux Method for Dolev-Yao Multi-Agent Epistemic Logic

Now, we present the tableaux method for $\mathcal{S5}_{DY}$. Here we provide a set of rules that allow us to verify if a malicious user can obtain private messages from a communication network, for example, deriving this information from the messages he received or intercepted.

We also prove the soundness and completeness of our method, we briefly explain the termination argument and, finally, we give some examples.

4.1 Semantics

The semantics for our method are the same as in subsection 2.3.1.

4.2 Rules

The rules for our method are the same as in subsection 2.3.2, for all formulae α and β :

$$\begin{array}{c} R_{\wedge} \frac{\alpha \wedge \beta}{\alpha} \\ \beta \end{array} \quad R_{\text{Dneg}} \frac{\neg\neg\alpha}{\alpha} \quad R_{\wedge}^{\neg} \frac{\neg(\alpha \wedge \beta)}{\neg\alpha \quad \neg\beta} \quad R_{\rightarrow} \frac{\alpha \rightarrow \beta}{\neg\alpha \quad \beta} \quad R_{\rightarrow}^{\neg} \frac{\neg(\alpha \rightarrow \beta)}{\alpha} \\ \neg\beta$$
$$R_{\pi} \frac{\neg K_a \alpha}{\rho(\mathcal{T}', \neg\alpha)}, \text{ where } \mathcal{T}' \text{ is a new tableau}$$
$$R_t \frac{K_a \alpha}{\alpha} \quad R_4^r \frac{\rho(\mathcal{T}'', K_a \alpha)}{K_a \alpha}$$
$$R_4 \frac{K_a \alpha}{\rho(\mathcal{T}'', K_a \alpha)}, \text{ where } \mathcal{T}'' \text{ is a previously generated tableau}$$

And now we add the following ones:

$$\begin{array}{cccc}
R_{\text{Dec}} \frac{\{m\}_k}{m} & R_{\text{Enc}}^- \frac{\neg\{m\}_k}{\neg m \quad \neg k} & R_{\text{Pair}} \frac{(m, n)}{m} & R_{\text{Pair}}^- \frac{\neg(m, n)}{\neg m \quad \neg n} \\
& & n &
\end{array}$$

where $m, \{m\}_k, n, (m, n) \in E$ and $k \in \mathcal{K}$.

4.3 Soundness

The soundness proof for our method is inspired by [23]. First, we need some definitions:

Definition 4.3.1. *Let Γ be a set of formulae:*

1. *we denote $s \Vdash \Gamma$ to represent $s \Vdash \alpha$, for all $\alpha \in \Gamma$;*
2. *we say Γ is satisfiable if there exists a model \mathcal{M} and some possible state $s \in S$ such that $s \Vdash \Gamma$;*
3. *a tableau branch is satisfiable if the set of all its formulae is satisfiable. A tableau is satisfiable if at least one branch is satisfiable.*

Lemma 4.3.1. *The rules of tableaux method preserve satisfiability. That is, if a tableau \mathcal{T} is satisfiable then the tableau resulting from the application of a rule to \mathcal{T} is satisfiable.*

Proof. Let \mathcal{T} be a satisfiable tableau. By property 3 of Definition 4.3.1, \mathcal{T} has at least one satisfiable branch, although it could have unsatisfiable ones. So, or the rule is applied to a satisfiable branch or to an unsatisfiable one.

First case: if the rule is applied to an unsatisfiable branch, each originally satisfiable branch remains unchanged. Therefore, the tableau resulting from the application of a rule is satisfiable.

Second case: if the rule is applied to a satisfiable branch θ , which consists of a set of formulae Γ and some specific formulae γ and δ which the rule is applied. As θ is satisfiable, by property 2 of Definition 4.3.1, there exists a model \mathcal{M} and a possible state $s \in S$ such that $s \Vdash \Gamma$, in particular, $s \Vdash \gamma$ and $s \Vdash \delta$. Let's θ' be the new branch obtained by the application of an inference rule to θ . We have the following cases for each possible structure of γ and/or δ :

- for γ or δ of type $\neg\neg\alpha$, $\alpha \wedge \beta$, $\neg(\alpha \vee \beta)$, $\neg(\alpha \rightarrow \beta)$, $\alpha \vee \beta$, $\neg(\alpha \wedge \beta)$, $\alpha \rightarrow \beta$, $K_a\alpha$ or $\neg K_a\alpha$, the proof can be found in tableaux for modal logics literature [13, 23].

- R_{Dec} : for γ of type m and for δ of type k , where $m, k \in \Gamma$, since $s \Vdash m$ and $s \Vdash k$, that is, $s \Vdash m \wedge k$, by the soundness of axiom 1 of Lemma 3.3.1, we have $s \Vdash \{m\}_k$. Therefore, θ' is satisfiable.
- R_{Enc}^- : for γ or δ of type $\neg\{m\}_k$ and $s \Vdash \neg\{m\}_k$. By the contrapositive of axiom 1 of Lemma 3.3.1 and its soundness, we have $s \Vdash \neg(m \wedge k)$ and also $s \Vdash \neg m \vee \neg k$. Suppose $s \Vdash \neg m$, then θ' is satisfiable. Suppose $s \Vdash \neg k$, then θ' is also satisfiable. Therefore, θ' is satisfiable.
- the cases for rules R_{Pair} and R_{Pair}^- are analogous to the cases for rules R_{Dec} and R_{Enc}^- , respectively, but using axiom 3 of Lemma 3.3.1.

■

The soundness of our tableaux method follows straightforward from the above lemma. If a formula $\neg\alpha$ has a closed tableaux, then it is unsatisfiable. Therefore α must be a valid formula.

4.4 Completeness

As pointed by COSTA [23], in [29] it is proven the completeness of tableaux method for classical logic, based on the construction of a completed tableaux and showing that when we cannot build a closed tableau for a formula $\neg\alpha$, we have what is necessary to build a counter-model for α , therefore, α is not valid. Then, FITTING [13] extended this approach by adding the notion of *prefixed tableaux*, with the definition of a *completed tableau* and proving that if a formula α is valid, then every completed tableau for $\neg\alpha$ is closed. The completeness proof found in [23] is inspired by this approach and is also the base for the proof below.

Let's begin with some definitions:

Definition 4.4.1. *Formulae of the form $X \wedge Y$, $\neg(X \vee Y)$, $\neg(X \rightarrow Y)$, $\neg\neg X$, (m, n) or occurrences of m and k are called type- α formulae, while every formulae of the form $X \vee Y$, $\neg(X \wedge Y)$, $X \rightarrow Y$, $\neg(m, n)$ or $\neg\{m\}_k$ are called type- β formulae. The components α_1 and α_2 from a type- α formula and the components β_1 and β_2 from a type- β formula are given in the tables bellow:*

α	α_1	α_2
$X \wedge Y$	X	Y
$\neg(X \vee Y)$	$\neg X$	$\neg Y$
$\neg(X \rightarrow Y)$	X	$\neg Y$
$\neg\neg X$	X	X
(m, n)	m	n
m	$\{m\}_k$	$\{m\}_k$
k		

Table 4.1: Components of a type- α formula

β	β_1	β_2
$X \vee Y$	X	Y
$\neg(X \wedge Y)$	$\neg X$	$\neg Y$
$X \rightarrow Y$	$\neg X$	Y
$\neg(m, n)$	$\neg m$	$\neg n$
$\neg\{m\}_k$	$\neg m$	$\neg k$

Table 4.2: Components of a type- β formula

Definition 4.4.2. A branch θ of a tableau σ is called complete if it satisfies the following conditions (where Σ is a set of formulae of θ and γ a specific formula):

1. if $(\sigma, \alpha) \in \Sigma$, then $(\sigma, \alpha_1) \in \Sigma$ and $(\sigma, \alpha_2) \in \Sigma$;
2. if $(\sigma, \beta) \in \Sigma$, then $(\sigma, \beta_1) \in \Sigma$ or $(\sigma, \beta_2) \in \Sigma$;
3. if $(\sigma, K_a \gamma) \in \Sigma$, then $(\sigma', \gamma) \in \Sigma$ for every tableau σ' that occurs in Σ and is accessible from σ ;
4. if $(\sigma, \neg K_a \gamma) \in \Sigma$, then $(\sigma', \gamma) \in \Sigma$ for some tableau σ' that is accessible from σ ;
5. every branch of any tableau which is accessible from θ is complete or closed as well.

Definition 4.4.3. We say that a tableau \mathcal{T} is completed if every branch of σ is complete or closed.

So, if a branch θ of a tableau \mathcal{T} is complete and open, then we have at least one open branch (that is also complete) per subordinated tableaux to θ .

Theorem 4.4.1. Every complete and open branch of a tableau is satisfiable.

Proof. Let θ be a complete and open branch of a tableau \mathcal{T} and Σ be a set of formulae of θ and of the tableaux $\mathcal{T}_1, \mathcal{T}_2, \dots$ (which are recursively subordinated to θ). We construct a model \mathcal{M} where S is the set of tableaux $\{\mathcal{T}, \mathcal{T}_1, \mathcal{T}_2, \dots\}$, \sim_a is built from the pairs $(\mathcal{T}_1, \mathcal{T}_2)$, such that \mathcal{T}_2 is subordinated to \mathcal{T}_1 and satisfying the following conditions, where E is an expression and the prefixes $\sigma, \sigma_1, \sigma_2, \dots$ are associated to $\{\mathcal{T}, \mathcal{T}_1, \mathcal{T}_2, \dots\}$, respectively:

1. if $(\sigma, E) \in \Sigma$, then $V(\sigma, E) = T$;
2. if $(\sigma, \neg E) \in \Sigma$, then $V(\sigma, E) = F$;
3. if $(\sigma, E) \notin \Sigma$ and $(\sigma, \neg E) \notin \Sigma$, then $V(\sigma, E) = T$ can have any value. Let's choose F by default.

Now, for any $(\sigma, \gamma) \in \Sigma$, we have $s \Vdash \gamma$, where γ is a formula and s a possible state associated to σ . According to γ structure:

- for $(\sigma, p), (\sigma, \alpha), (\sigma, \beta), (\sigma, K_a \gamma)$ and $(\sigma, \neg K_a \gamma)$ the proof is found in [23]. We only show the case for rules presented in Definition 4.2;
- The pair $(\sigma, \{m\}_k) \in \Sigma$, for some prefix σ . By condition 1 of Definition 4.4.2, we have $(\sigma, m) \in \Sigma$ and $(\sigma, k) \in \Sigma$ and by the induction hypothesis $s \Vdash m$ and $s \Vdash k$ and also $s \Vdash m \wedge k$, by the soundness of axiom 1 of Lemma 3.3.1, we have $s \Vdash \{m\}_k$;
- The pair $(\sigma, \neg\{m\}_k) \in \Sigma$, for some prefix σ . By condition 2 of Definition 4.4.2, we have $(\sigma, \neg m) \in \Sigma$ or $(\sigma, \neg k) \in \Sigma$ and by the induction hypothesis $s \Vdash \neg m$ or $s \Vdash \neg k$ and also $s \Vdash \neg m \vee \neg k$ and $s \Vdash \neg(m \wedge k)$, by the soundness of the contrapositive of axiom 1 of Lemma 3.3.1, we have $s \Vdash \neg\{m\}_k$;
- the cases for rules R_{Pair} and R_{Pair}^- are analogous to the cases for rules R_{Dec} and R_{Enc}^- , respectively, but using axiom 3 of Lemma 3.3.1.

Therefore, our model satisfies Σ . ■

Theorem 4.4.2. *If a formula γ is valid, then γ has a proof by tableaux method.*

Proof. Let \mathcal{T} be a completed tableau, started with $\neg\gamma$. If it is open, then $\neg\gamma$ is satisfiable by theorem 4.4.1. So, γ cannot be valid. Therefore, if γ is valid, then \mathcal{T} is closed and γ has a proof by tableaux method. ■

4.5 Termination property

For the tableaux rules presented in Definition 2.3.2, MASSACCI [24] provides the termination argument below, adapted for our semantics.

4.5.1 Classical and modal rules

To guarantee the termination of the proof search it's used the “loop checking” approach, a combination of techniques to apply any rule only after check if it was not applied already to the same antecedent. First we need some definitions:

Definition 4.5.1. *In a branch θ of a tableau σ , a prefixed formula (σ, γ) is reduced for a rule in θ :*

- *if the rule generates (σ', γ') and (σ', γ') is in θ ; or*
- *if the rule splits the tableau into (σ_1, γ_1) and (σ_2, γ_2) and at least one of those is in θ .*

The formula (σ, γ) is fully reduced in θ if it is reduced for all applicable rules and σ is (fully) reduced if all prefixed formula (σ, γ) are (fully) reduced as well.

So, for tableaux method for logic \mathcal{K} , the following technique is sufficient to terminate:

Technique 4.5.1. *Apply a rule to a prefixed formula (σ, γ) in θ only if the formula is not already reduced according to Definition 4.5.1, except for the knowledge operator.*

But for our case, the “loop checking” concept is required. Let's begin with the definition of a *copy* of a prefix:

Definition 4.5.2. *A prefix σ is a copy of a prefix σ_0 for branch θ if for every formula γ one has $(\sigma, \gamma) \in \theta$ iff $(\sigma_0, \gamma) \in \theta$.*

Now we define what is a π -reduced prefix:

Definition 4.5.3. *A prefix is π -reduced in θ if it is reduced for all rules except R_π . A branch θ is π -completed if:*

- *all prefixes are π -reduced in θ ;*
- *for every σ that is not fully reduced there is a fully reduced copy σ_0 shorter than σ .*

So, the idea is to restrict the usage of R_π to formulae belonging to copies. The following technique together with Technique 4.5.1 prove that we will always have a π -completed branch:

Technique 4.5.2. *Select the prefixed formulae with the shortest prefix.*

As we have a π -completed branch, the next technique guarantees termination:

Technique 4.5.3. *Check if the prefix of a π -formula is not a copy of a shorter prefix before reducing it.*

4.5.2 Dolev-Yao Multi-Agent Epistemic Logic rules

As rules R_{Dec} , R_{Enc}^- , R_{Pair} , R_{Pair}^- always yield a smaller conclusion than the premises, that is, they are considered *analytic* rules, the argument explained in subsection 4.5.1 is not interfered.

4.6 Examples

In order to illustrate the usage of our tableaux method, we present the following examples:

Example 4.6.1. *Let's consider a generic database $DB = [k, m, n, \{(m, n)\}_k \rightarrow p]$. We want to know if $DB \vdash (n, p)$:*

1. k [DB]
 2. m [DB]
 3. n [DB]
 4. $\{(m, n)\}_k \rightarrow p$ [DB]
 5. $\neg(n, p)$ [negation of the question]
-
6. $\neg n$ $\neg p$ [from 5, by R_{Pair}^-]

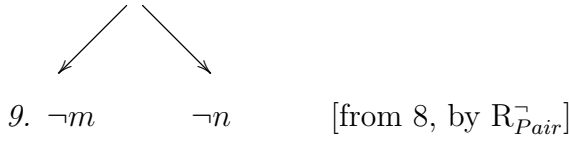
The left branch is closed and from the right branch we have:

-
7. $\neg\{(m, n)\}_k$ p [from 4, by R_{\rightarrow}]

Now, the last right branch is closed. From the last left branch we have:

-
8. $\neg(m, n)$ $\neg k$ [from 7, by R_{Enc}^-]

Again, the last right branch is closed and, finally, from the last left branch:



Thus, we have a closed tableau.

Example 4.6.2. Let's prove theorem 1 from Lemma 3.2.1:

1. $\neg(K_a m \wedge K_a k \rightarrow K_a \{m\}_k)$ [negation of the question]
2. $K_a m \wedge K_a k$ [from 1, by R_{\supset}]
3. $\neg K_a \{m\}_k$ [from 1, by R_{\supset}]
4. $K_a m$ [from 2, by R_{\wedge}]
5. $K_a k$ [from 2, by R_{\wedge}]

Now, we generate a new tableau:

- 3.1. $\neg \{m\}_k$ [from 3, by R_{π}]
 - 3.2. $K_a m$ [from 4, by R_4]
 - 3.3. $K_a k$ [from 5, by R_4]
 - 3.4. m [from 3.2, by R_t]
 - 3.5. k [from 3.3, by R_t]
- $$\begin{array}{ccc}
 & \swarrow & \searrow \\
 3.6. \neg m & & \neg k & \quad \text{[from 3.1, by } R_{Enc}^- \text{]}
 \end{array}$$

Since each of the branches closes, we have a closed tableau.

Example 4.6.3. Let's prove theorem 2 from Lemma 3.2.1:

1. $\neg(K_a \{m\}_k \wedge K_a k \rightarrow K_a m)$ [negation of the question]
2. $K_a \{m\}_k \wedge K_a k$ [from 1, by R_{\supset}]
3. $\neg K_a m$ [from 1, by R_{\supset}]

Now, we generate a new tableau:

- 3.1. $\neg m$ [from 3, by R_{π}]

3.2. $K_a\{m\}_k$	[from 2, by R_\wedge]
3.3. K_ak	[from 2, by R_\wedge]
3.4. $\{m\}_k$	[from 3.2, by R_t]
3.5. k	[from 3.3, by R_t]
3.6. m	[from 3.4 and 3.5, by R_{Dec}]

Thus, we have a closed tableau.

Example 4.6.4. Let's prove theorem 3 from Lemma 3.2.1, using the fact that $K_am \wedge K_an \leftrightarrow K_a(m, n) \equiv (K_am \wedge K_an \rightarrow K_a(m, n) \wedge K_a(m, n) \rightarrow K_am \wedge K_an)$:

1. $\neg(K_am \wedge K_an \leftrightarrow K_a(m, n))$	[negation of the question]
2. $\neg(K_am \wedge K_an \rightarrow K_a(m, n))$	$\neg(K_a(m, n) \rightarrow K_am \wedge K_an)$

As we can see, the initial tableau splitted into a left and right branch. From the left branch we have:

2.1.1. $K_am \wedge K_an$	[from left-hand side of 2, by R_\rightarrow]
2.2.1. $\neg K_a(m, n)$	[from left-hand side of 2, by R_\rightarrow]
2.3.1. K_am	[from 2.1, by R_\wedge]
2.4.1. K_an	[from 2.1, by R_\wedge]

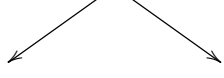
Now, we generate a new tableau:

2.2.1.1. $\neg(m, n)$	[from 2.2, by R_π]
2.2.1.2. K_am	[from 2.3, by R_4]
2.2.1.3. K_an	[from 2.4, by R_4]
2.2.1.2. m	[from 2.2.2, by R_t]
2.2.1.3. n	[from 2.2.3, by R_t]
2.2.1.4. $\neg m$ $\neg n$	[from 2.2.1, by R_{Pair}^-]

Thus, all this left branch of 2 is closed. Now, from the right branch we have:

2.1.r. $K_a(m, n)$ [from right-hand side of 2, by R_{\rightarrow}^-]

2.2.r. $\neg(K_a m \wedge K_a n)$ [from right-hand side of 2, by R_{\rightarrow}^-]



2.3.r. $\neg K_a m$ $\neg K_a n$ [from 2.2, by R_{Pair}^-]

From the left-hand side of 2.3 we generate a new tableau:

2.3.r.1.l. $\neg m$ [from left-hand side of 2.3, by R_{π}]

2.3.r.2.l. $K_a(m, n)$ [from 2.1., by R_4]

2.3.r.3.l. (m, n) [from 2.3.2., by R_t]

2.3.r.4.l. n [from 2.3.3., by R_{Pair}]

2.3.r.5.l. m [from 2.3.3., by R_{Pair}]

Thus, the left branch is closed. From the right-hand side of 2.3 we generate a new tableau:

2.3.r.1.r. $\neg n$ [from right-hand side of 2.3, by R_{π}]

2.3.r.2.r. $K_a(m, n)$ [from 2.1., by R_4]

2.3.r.3.r. (m, n) [from 2.3.2., by R_t]

2.3.r.4.r. m [from 2.3.3., by R_{Pair}]

2.3.r.5.r. n [from 2.3.3., by R_{Pair}]

Since each of the branches is closed, we have a closed tableau.

Chapter 5

Conclusion

In this work, we presented a new epistemic logic for reasoning about security protocols, the $\mathcal{S5}_{DY}$. This logic introduces a new semantics based on structured propositions. Instead of building formulas from atomic propositions, they are built from expressions. The latter, are any piece of information that can appear in protocols: keys, messages, agents and properties or some combination of this information in pairs, encrypted messages and so forth. We propose this new semantics and an axiomatization for this logic. Proofs for soundness and completeness are given as well.

It is also provided a tableaux method for $\mathcal{S5}_{DY}$. This theorem prover is based on prefixed tableaux, we made an extension of this concept with our semantics and we proved soundness and completeness, according to satisfiability results and analytic tableaux conditions, respectively.

We believe that this work contributes to the growing demand for security studies, by integrating concepts of logic in intuitive way and using knowledge formalisms.

Now we list some possible extensions to our work.

5.1 Knowledge *de re/de dicto*

The knowledge operator K_a is meant to capture the standard notion of knowledge *de dicto* that a has about a propositional sentence. An example of a sentence is $K_a K_b m$, of which the intuition is that agent a knows (that it is the case that) agent b knows message m . What do we mean by an agent to know the message m ? Does he know the content of the message or the message itself?

We can extend the language with a new knowledge operator $\check{K}_a m$ in order to capture the notion of knowledge *de re* that a has of m [30], that is, $\check{K}_a m$ denotes the fact that agent a knows the content of m . For instance, now we can express $K_a \check{K}_b m$, of which the intuition is that agent a knows (that it is the case that) agent b knows (the content of) message m .

5.2 Common knowledge

We also can extend our work by expressing notions of common knowledge [1, 11], obtaining the logic $\mathcal{S5}_{DY}^{CK}$ by adding the operator $C_G\varphi$ (φ is common knowledge for agents in group G). A proposal for an axiomatization could be:

- Axioms and rules of $\mathcal{S5}_{DY}$.
- $E_G\varphi \leftrightarrow \bigwedge_{a \in G} K_a\varphi$
- $C_G(\varphi \rightarrow \psi) \rightarrow (C_G\varphi \rightarrow C_G\psi)$
- $C_G\varphi \rightarrow (\varphi \wedge E_G C_G\varphi)$
- $C_G(\varphi \rightarrow E_G\varphi) \rightarrow (\varphi \rightarrow C_G\varphi)$ [+ induction]

5.3 Adding actions

As we said in subsection 3.5, the protocols actions are in a kind of extra or metalanguage, but it would be interesting to formalize these actions. In *dynamic epistemic logic* [1], for instance, we can reason about information change. We already had outlined some concepts using *propositional dynamic logic* [31, 32]. The \mathcal{PDL}_{DY} modality $[m, \mathcal{K}]\alpha$ is intended to mean that “ α holds once the intruder have message m and the keys in set \mathcal{K} ”. Some possible validities are:

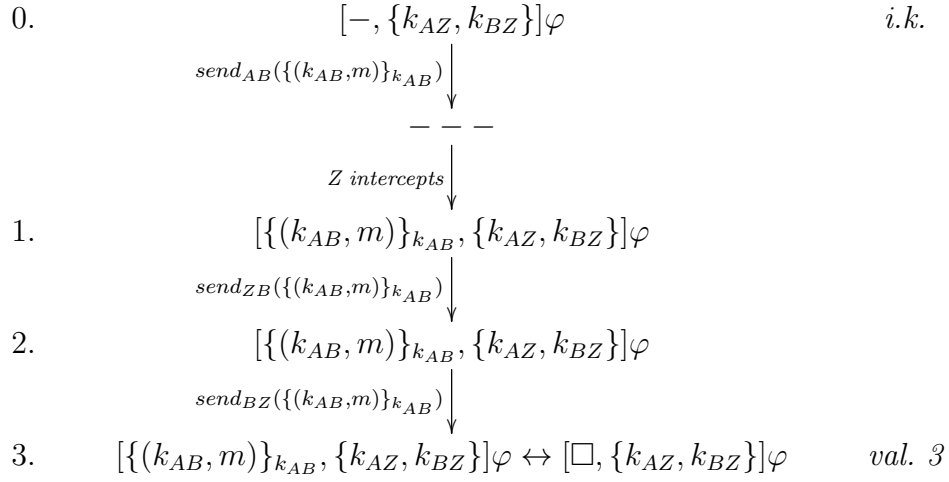
1. $[m, \mathcal{K}]\varphi \leftrightarrow [\{m\}_k, \mathcal{K}]\varphi$, if $k \in \mathcal{K}$
2. $[(m, n), \mathcal{K}]\varphi \leftrightarrow [m, \mathcal{K}]\varphi \wedge [n, \mathcal{K}]\varphi$
3. $[\{m\}_k, \mathcal{K}]\varphi \leftrightarrow [\square, \mathcal{K}]\varphi$, if $k \notin \mathcal{K}$

where \square may be read as “undecryptable” and we can define

$$\vdash [m, \mathcal{K}]\varphi \leftrightarrow [\square, K]\varphi$$

as a *Secrecy*.

Example 5.3.1. *Returning to Example 2.1.2, agent A wants to send an encrypted message to agent B, and the receiver always replies a message using the key shared with the indicated agent that is encrypted with the plaintext. The initial knowledge only refers to the intruder. We assume that $k_{XY} = k_{YX}$ for every agent X and Y:*



Intruder Z does not know M

It is still in its incipient stages, but it would allow us to track each step of a protocol and recognize where exactly an error of specification occurs.

Bibliography

- [1] DITMARSCH, H. V., VAN DER HOEK, W., KOOI, B. *Dynamic Epistemic Logic*. Springer Publishing Company, Incorporated, 2007. ISBN: 1402058381, 9781402058387.
- [2] BURROWS, M., ABADI, M., NEEDHAM, R. “A Logic of Authentication”, *ACM Transactions on Computer Systems*, v. 8, n. 1, pp. 18–36, 1990.
- [3] BOUREANU, I., COHEN, M., LOMUSCIO, A. “Automatic verification of temporal-epistemic properties of cryptographic protocols”, *Journal of Applied Non-Classical Logics*, v. 19, n. 4, pp. 463–487, 2009. doi: 10.3166/jancl.19.463-487.
- [4] CHEN, Q., SU, K., LIU, C., et al. “Automatic Verification of Web Service Protocols for Epistemic Specifications under Dolev-Yao Model”. In: *2010 International Conference on Service Sciences*, pp. 49–54, May 2010. doi: 10.1109/ICSS.2010.33.
- [5] SYVERSON, P. “The use of logic in the analysis of cryptographic protocols”. In: *Proceedings. 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 156–170, May 1991. doi: 10.1109/RISP.1991.130784.
- [6] ABADI, M., GORDON, A. D. “A Calculus for Cryptographic Protocols: The Spi Calculus”. In: *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pp. 36–47. ACM, 1997.
- [7] GOLDWASSER, S., MICALI, S. “Probabilistic Encryption.” *J. Comput. Syst. Sci.*, v. 28, n. 2, pp. 270–299, 1984.
- [8] BLUM, M., MICALI, S. “How to Generate Cryptographically Strong Sequences of Pseudo-random Bits”, *SIAM J. Comput.*, v. 13, n. 4, pp. 850–864, nov. 1984. ISSN: 0097-5397. doi: 10.1137/0213053.
- [9] ABADI, M., ROGAWAY, P. “Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption)”. In: *Proceedings of the*

International Conference IFIP on Theoretical Computer Science, Exploring New Frontiers of Theoretical Informatics, TCS '00, pp. 3–22, London, UK, UK, 2000. Springer-Verlag. ISBN: 3-540-67823-9.

- [10] DOLEV, D., YAO, A. C. “On the Security of Public Key Protocols”, *IEEE Transactions on Information Theory*, v. 29, n. 2, pp. 198–208, 1983.
- [11] FAGIN, R., HALPERN, J. Y., MOSES, Y., et al. *Reasoning about Knowledge*. MIT Press, 1995.
- [12] GABBAY, D. M. *Labelled Deductive Systems*. Oxford University Press, 1996.
- [13] FITTING, M. *Proof Methods for Modal and Intuitionistic Logics*. Solid Earth Sciences Library. Springer, 1983. ISBN: 9789027715739.
- [14] KRIPKE, S. A. “A Completeness Theorem in Modal Logic”, *Journal of Symbolic Logic*, v. 24, n. 1, pp. 1–14, 1959.
- [15] NEEDHAM, R. M., SCHROEDER, M. D. “Using encryption for authentication in large networks of computers”, *Commun. ACM*, , n. 12, pp. 993–999, 1978.
- [16] DIFFIE, W., HELLMAN, M. “New directions in cryptography”, *IEEE Transactions on Information Theory*, v. IT-22, pp. 664–654, 1976.
- [17] RIVEST, R. L., SHAMIR, A., ADLEMAN, L. “A method for obtaining digital signatures and public-key cryptosystems”, *Commun. ACM*, v. 21, pp. 120–126, 1978.
- [18] ABADI, M., ROGAWAY, P. “Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption)”, *IFIP International Conference on Theoretical Computer Science*, 2000.
- [19] AYALA-RINCÓN, M., FERNÁNDEZ, M., SOBRINHO, D. N. “Elementary Deduction Problem for Locally Stable Theories with Normal Forms”. In: *Proceedings Seventh Workshop on Logical and Semantic Frameworks, with Applications, LSFA 2012, Rio de Janeiro, Brazil, September 29-30, 2012.*, pp. 45–60, 2012. doi: 10.4204/EPTCS.113.7.
- [20] BLACKBURN, P., BENTHEM, J. F. A. K. V., WOLTER, F. *Handbook of Modal Logic, Volume 3 (Studies in Logic and Practical Reasoning)*. New York, NY, USA, Elsevier Science Inc., 2006. ISBN: 0444516905.
- [21] VON WRIGHT, G. H. “An Essay in Modal Logic”, 1951. doi: 10.1017/S0031819100026176.

- [22] HINTIKKA, J. *Knowledge and belief: an introduction to the logic of the two notions*. Contemporary philosophy. Cornell University Press, 1962.
- [23] COSTA, M. M. D. C. *Introdução à Lógica Modal Aplicada à Computação*. UFRGS, 1992.
- [24] MASSACCI, F. “Single Step Tableaux for Modal Logics”, *J. Autom. Reasoning*, v. 24, n. 3, pp. 319–364, 2000. doi: 10.1023/A:1006155811656.
- [25] BENEVIDES, M. R. F., FERNANDEZ, L. C. F., OLIVEIRA, A. C. C. M. “Epistemic Logic Based on Dolev-Yao Model”. In: *Anais do XXXVII Congresso da Sociedade Brasileira de Computação - II ETC*. Sociedade Brasileira de Computação, 2017.
- [26] BLACKBURN, P., RIJKE, M. D., VENEMA, Y. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001. doi: 10.1017/CBO9781107050884.
- [27] MILLER, S. P., NEUMAN, B. C., SCHILLER, J. I., et al. “Kerberos authentication and authorization system”. In: *In Project Athena Technical Plan*, 1987.
- [28] SATYANARAYANAN, M. “Integrating Security in a Large Distributed System”, *ACM Trans. Comput. Syst.*, v. 7, n. 3, pp. 247–280, ago. 1989. ISSN: 0734-2071. doi: 10.1145/65000.65002.
- [29] SMULLYAN, R. *First-order Logic*. Ergebnisse der Mathematik und ihrer Grenzgebiete. Springer-Verlag, 1968.
- [30] KRAMER, S. “Cryptographic Protocol Logic: Satisfaction for (Timed) Dolev-Yao Cryptography”, *Journal of Logic and Algebraic Programming*, v. 77, n. 1–2, 2008.
- [31] FISCHER, M. J., LADNER, R. E. “Propositional dynamic logic of regular programs”, *Journal of Computer and System Sciences*, v. 18, n. 2, pp. 194 – 211, 1979. ISSN: 0022-0000. doi: [https://doi.org/10.1016/0022-0000\(79\)90046-1](https://doi.org/10.1016/0022-0000(79)90046-1).
- [32] HAREL, D., TIURYN, J., KOZEN, D. *Dynamic Logic*. Cambridge, MA, USA, MIT Press, 2000. ISBN: 0262082896.

Appendix A

Kerberos Protocol

Based on [15], the Kerberos protocol was developed for Project Athena at MIT. It is used to provide a shared key between two users when a server is requested to do so, using timestamps.

Considering two users A and B , an authentication server S (also treated as an agent), T_X as the timestamp generated by agent X and the lifetime L , we can represent this protocol by the following steps (assuming that every user already has a shared key with the server):

1. A wants to communicate with B , so he sends a message to S stating it;
2. S replies with an encrypted message containing T_S , L , k_{AB} , and an encrypted message that only B can read (since it was encrypted under a shared key between S and B), which also contains the timestamp, the lifetime, and the shared key requested (this message is also called *ticket*);
3. A forwards the message destined to B together with a timestamp encrypted under k_{AB} ;
4. B receives the first message and is able to check the T_S and L . If it has been created recently enough, he uses the k_{AB} to decrypt the second message. Then, he can take the communication from there, using T_A .

Supposing that an intruder Z intercepts the message sent from A to B and he already got from S what is necessary to communicate with B , let's analyze this protocol using $\mathcal{S5}_{DY}$:

$$0. \quad KB_0 = \{K_AA, K_AB, K_Ak_{AS}, K_AT_A, K_Bk_{BS}, K_BT_B,$$

$$K_ST_S, K_SL, K_Sk_{AB}, K_Sk_{BS}, \quad \text{i.k.}$$

$$K_ZT_Z, K_Zk_{ZB}, K_Z\{(T_S, L', k_{ZB}, Z)\}_{k_{BS}}\}$$

$$1. \quad \begin{array}{l} KB_0 \vdash K_A(A, B) \quad \text{thrm. 3} \\ \text{send}_{AS}((A, B)) \downarrow \\ KB_1 := KB_0 \cup K_S(A, B) \end{array}$$

$$KB_1 \vdash K_SA \quad \text{thrm. 3}$$

$$KB_1 \vdash K_SB \quad \text{thrm. 3}$$

$$KB_1 \vdash K_S(T_S, L, k_{AB}, A) \quad \text{thrm. 3}$$

$$KB_1 \vdash K_S\{(T_S, L, k_{AB}, A)\}_{k_{BS}} \quad \text{thrm. 1}$$

$$KB_1 \vdash K_S(T_S, L, k_{AB}, B, \{(T_S, L, k_{AB}, A)\}_{k_{BS}}) \quad \text{thrm. 3}$$

$$2. \quad \begin{array}{l} KB_1 \vdash K_S\{(T_S, L, k_{AB}, B, \{(T_S, L, k_{AB}, A)\}_{k_{BS}})\}_{k_{AS}} \quad \text{thrm. 1} \\ \text{send}_{SA}(\{(T_S, L, k_{AB}, B, \{(T_S, L, k_{AB}, A)\}_{k_{BS}})\}_{k_{AS}}) \downarrow \\ KB_2 := KB_1 \cup K_A\{(T_S, L, k_{AB}, B, \{(T_S, L, k_{AB}, A)\}_{k_{BS}})\}_{k_{AS}} \end{array}$$

$$KB_2 \vdash K_A(T_S, L, k_{AB}, B, \{(T_S, L, k_{AB}, A)\}_{k_{BS}}) \quad \text{thrm. 2}$$

$$KB_2 \vdash K_A\{(T_S, L, k_{AB}, A)\}_{k_{BS}} \quad \text{thrm. 3}$$

$$KB_2 \vdash K_Ak_{AB} \quad \text{thrm. 3}$$

$$KB_2 \vdash K_A(A, T_A) \quad \text{thrm. 3}$$

$$KB_2 \vdash K_A\{(A, T_A)\}_{k_{AB}} \quad \text{thrm. 1}$$

$$\begin{array}{c}
KB_2 \vdash K_A(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \quad \text{thrm. 3} \\
\text{send}_{AB}(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \downarrow \\
\text{---} \\
Z \text{ intercepts} \downarrow \\
3. \quad KB_3 := KB_2 \cup K_Z(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}})
\end{array}$$

Now Z has two possibilities. The first one is to send the same intercepted message to B :

$$\begin{array}{c}
3. \quad KB_3 := KB_2 \cup K_Z(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \\
\text{send}_{ZB}(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \downarrow \\
4. \quad KB_4 := KB_3 \cup K_B(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}})
\end{array}$$

$$KB_4 \vdash K_B\{(T_S, L, k_{AB}, A)\}_{k_{BS}} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_B\{(A, T_A)\}_{k_{AB}} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_B(T_S, L, k_{AB}, A) \quad \text{thrm. 2}$$

$$KB_4 \vdash K_B k_{AB} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_B(A, T_A) \quad \text{thrm. 2}$$

$$KB_4 \vdash K_B T_A \quad \text{thrm. 3}$$

$$\begin{array}{c}
KB_4 \vdash K_B\{T_A\}_{k_{AB}} \quad \text{thrm. 1} \\
\text{send}_{BZ}(\{T_A\}_{k_{AB}}) \downarrow \\
5. \quad KB_5 := KB_4 \cup K_Z\{T_A\}_{k_{AB}}
\end{array}$$

$$KB_5 \not\vdash K_Z T_A$$

Intruder Z does not know T_A .

Or he can send a concatenation of the ticket he previously got from S and the encrypted message:

$$3. \quad KB_3 := KB_2 \cup K_Z(\{(T_S, L, k_{AB}, A)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}})$$

$$KB_3 \vdash K_Z\{(A, T_A)\}_{k_{AB}} \quad \text{thrm. 3}$$

$$KB_3 \vdash K_Z(\{(T_S, L', k_{ZB}, Z)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \quad \text{thrm. 3}$$

$$\text{send}_{ZB}(\{(T_S, L', k_{ZB}, Z)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}}) \downarrow$$

$$4. \quad KB_4 := KB_3 \cup K_B(\{(T_S, L', k_{ZB}, Z)\}_{k_{BS}}, \{(A, T_A)\}_{k_{AB}})$$

$$KB_4 \vdash K_B\{(T_S, L', k_{ZB}, Z)\}_{k_{BS}} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_B\{(A, T_A)\}_{k_{AB}} \quad \text{thrm. 3}$$

$$KB_4 \vdash K_B(T_S, L', k_{ZB}, Z) \quad \text{thrm. 2}$$

$$KB_4 \not\vdash K_B k_{AB}$$

$$KB_4 \not\vdash K_B(T_A, A)$$

$$KB_4 \not\vdash K_Z T_A$$

Since B is not able to continue the communication, intruder Z cannot know T_A .

Appendix B

Andrew Secure RPC Handshake Protocol

The Andrew Secure RPC protocol can be used when an user wants to refresh his key. So, in this scenario, let's consider that a handshake between user A and server S is made when a shared key k_{AS} already exists and A wants to obtain a new key k'_{AS} . We can represent this protocol by the following steps:

1. A sends a nonce N_A encrypted under the key shared with S to state that he wants a new one;
2. S returns this nonce concatenated with N_S , also encrypted;
3. A returns only N_S to the server, encrypted under k_{AS} ;
4. after check the last message, S can send the new shared key k'_{AS} concatenated with N'_S , where the latter "is an initial sequence number to be used in subsequent communication" [2], and encrypted under the first shared key.

Since there is no indication of who originated the third message, the server will reply this message using the key shared with the sender. Let's suppose that an intruder Z intercepts this message, we can also analyze this protocol:

$$0. \quad KB_0 = \{K_A k_{AS}, K_A k_{AZ}, K_A N_A, K_A A,$$

$$K_S k_{AS}, K_S k_{ZS}, K_S N_S, K_S k'_{AS}, K_S N'_S \quad i.k.$$

$$K_Z k_{AZ}, K_Z k_{ZS}\}$$

$$KB_0 \vdash K_A(A, N_A) \quad thrm. 3$$

$$\begin{array}{l}
KB_0 \vdash K_A\{(A, N_A)\}_{k_{AS}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{AS}(\{(A, N_A)\}_{k_{AS}}) \\
1. \quad KB_1 := KB_0 \cup K_S\{(A, N_A)\}_{k_{AS}}
\end{array}$$

$$KB_1 \vdash K_B(A, N_A) \quad \text{thrm. 2}$$

$$KB_1 \vdash K_B N_A \quad \text{thrm. 3}$$

$$KB_1 \vdash K_B(N_A, N_S) \quad \text{thrm. 3}$$

$$\begin{array}{l}
KB_1 \vdash K_B\{(N_A, N_S)\}_{k_{AS}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{SA}(\{(N_A, N_S)\}_{k_{AS}}) \\
2. \quad KB_2 := KB_1 \cup K_A\{(N_A, N_S)\}_{k_{AS}}
\end{array}$$

$$KB_2 \vdash K_A(N_A, N_S) \quad \text{thrm. 2}$$

$$KB_2 \vdash K_A N_S \quad \text{thrm. 3}$$

$$\begin{array}{l}
KB_2 \vdash K_A\{N_S\}_{k_{AS}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{AS}(\{(N_S)\}_{k_{AS}}) \\
\text{---} \\
\downarrow \text{Z intercepts} \\
3. \quad KB_3 := KB_2 \cup K_Z\{N_S\}_{k_{AS}} \\
\downarrow \text{send}_{ZB}(\{(N_S)\}_{k_{AS}}) \\
4. \quad KB_4 := KB_3 \cup K_S\{N_S\}_{k_{AS}}
\end{array}$$

$$KB_4 \vdash K_S N_S \quad \text{thrm. 2}$$

$$KB_4 \vdash K_S(k'_{AS}, N'_S) \quad \text{thrm. 3}$$

$$\begin{array}{l}
KB_4 \vdash K_S\{(k'_{AS}, N'_S)\}_{k_{ZS}} \quad \text{thrm. 1} \\
\downarrow \text{send}_{SZ}(\{(k'_{AS}, N'_S)\}_{k_{ZS}}) \\
5. \quad KB_5 := KB_4 \cup K_Z\{(k'_{AS}, N'_S)\}_{k_{ZS}}
\end{array}$$

$$KB_5 \vdash K_Z(k'_{AS}, N'_S) \quad \text{thrm. 2}$$

$$KB_5 \vdash K_Z k'_{AS} \quad \text{thrm. 3}$$

$$KB_5 \vdash K_Z N'_S \quad \text{thrm. 3}$$

$$KB_5 \vdash K_Z \{(k'_{AS}, N'_S)\}_{k_{AZ}} \quad \text{thrm. 1}$$

$$\begin{array}{c}
\text{send}_{ZA}(\{(k'_{AS}, N'_S)\}_{k_{AZ}}) \downarrow \\
6. \quad KB_6 := KB_5 \cup K_A \{(k'_{AS}, N'_S)\}_{k_{AZ}}
\end{array}$$

$$KB_6 \vdash K_A(k'_{AS}, N'_S) \quad \text{thrm. 2}$$

$$KB_6 \vdash K_A k'_{AS} \quad \text{thrm. 3}$$

$$KB_6 \vdash K_A N'_S \quad \text{thrm. 3}$$

Now, intruder Z is able to decrypt any message eventually sent by A or S and encrypted under k'_{AS} .