Journal of Logic and Computation, 8(3), 1998. ©Oxford University Press.
Preprint from Dip. Informatica e Sistemistica. Univ. of Rome "La Sapienza".

Tableau Methods for Formal Verification of Multi-Agent Distributed Systems

Fabio Massacci
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
via Salaria 133, I-00198 Roma
email:massacci@dis.uniroma1.it

Abstract

Formal verification is a key step in the development of trusted and reliable multi-agent distributed systems. This is particularly relevant when security concerns such as privacy, integrity and availability impose limitations on the operations that can be performed on sensitive data. The aim of access control is to limit what agents (humans, programs, softbots, etc.) of distributed systems can do directly or indirectly by delegating their powers and tasks. As the size of the systems and the sensitivity of data increase, the availability of automated reasoning methods becomes essential for logical analysis of access control.

This paper presents a prefixed tableau method for the calculus of access control developed at the Digital System Research Center by Abadi, Lampson et. al. This calculus is particularly interesting for a number of reasons. At first it was the basis for the development and the verification of an implemented system. Second, it poses many technical challenges for classical modal tableaux: it lacks the tree-model property, has some features of the universal modality, and can introduce delegation certificates between agents "on-the-fly" not compilable into axiom schemata.

1 Introduction

Formal verification is a major issue in the development of trusted and reliable computer systems[1]. The need of logical analysis and formal proofs is particularly important in large-scale multi-agent systems where agents (humans, programs, softbots, etc.) can make autonomous decisions and where sensitive data and operations are at stake. Key security concerns such as privacy, confidentiality and integrity may impose severe limitations on what an agent should be allowed to do.

Thus, access control plays a key role in the verification multi-agent distributed systems (see the review of Sandhu & Samarati [34] for an introduction). Its main purpose is to restrain the actions which legitimate or malicious agents may perform, either directly or indirectly (via other agents). It should be clearly distinguished from

[1]Formal (logical) analysis and verification are also required by the U.S. government [10] and the E.U. Commission [11] for systems to be legally labelled as "trusted".

other aspects of multi-agent distributed systems such as concurrency and coordination. These try to maximize the proper use of (public) resources while access control is focussed on the minimization of the improper use of (private or sensitive) resources.

Loosely speaking we may say that access control is the distributed systems counterpart of some aspects of human normative systems: “who can do what”. A human logging on a workstation, or a web-spider asking for permission to visit a restricted Internet site can be easily compared with a human crossing the passport control in an airport. In all cases, an access control decision is made by the agent in charge of the area. The verification of access control privileges and security policies is just a problem of jurisdiction in distributed systems where many agents have different goals and are able to make autonomous (but not necessarily sensible) decisions.

Different access control policies may be devised to answer the needs of different systems, such as those used by the military [4], by banks and commercial organizations [7] or by health care services [2]. A comprehensive field study of the policies employed in different (human) organizations in the U.S. can be found in the paper by Ferraiolo et al. [14].

Distributed systems face additional challenges (e.g. large scale, insecure communications, delegation of management etc.) which make the access control decision more complex and require additional tools. For instance we may decide to combine access control with authentication [40], or use formal methods to refine general security policies at various levels of management [32].

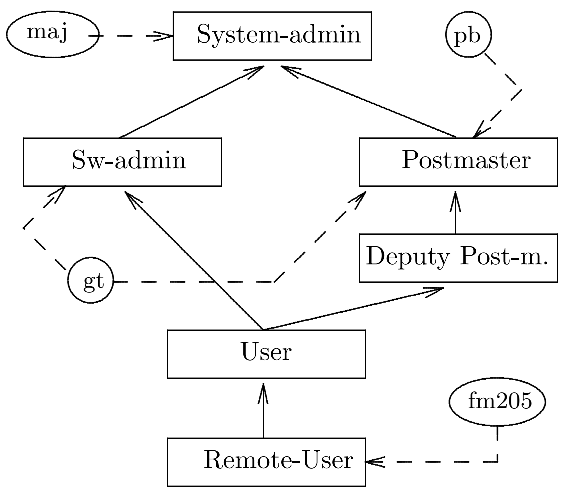
It is also important to notice that relations between agents in a “real” distributed system are seldom flat (all agents being equal). On the contrary, agents may be members of different groups or play different roles [14, 13]; the privileges of an agents usually depend on the groups they are members of; groups and roles are organized along hierarchies. Access control systems and policies become more complex and sophisticated [31, 5, 17, 35].

A very simple example is shown in Fig. 1. The right-hand side is the text of an information leaflet distributed to the users of the Computer Laboratory at the university of Cambridge (UK). It describes roles, privileges and tasks of various principals (in this case humans but they need not to be) of the laboratory computer system. On the left side we show a pictorial representation of the same structure. We represent roles as boxes and the hierarchies of privileges by straight line arrows. Principals having those roles are represented as ovals and the "has role" relation is represented by a dashed arrow. Here we have only the major roles of a local computer system and yet we have already to consider hierarchies, membership relations and other issues such as delegation.

Above all, as the size and the degree of automation of a system increase, a number of tasks is delegated to sophisticate software agents, which must be able to take autonomous decisions in a sensible way. The corresponding increase in complexity makes infeasible the human (and informal) verification of the system.

Hence formal methods, logics and automated reasoning techniques can be useful tools for the verification of security policies and access control procedures.

Our goal is to design and use tableau methods as an automated reasoning tool for access control in multi-agent distributed systems.



```
maj:  Titan Room Manager, knows
      about UNIX, DEC stations,
      Alphas, Ethernet, ATM,
      printer systems, backup
      systems and TEX.
gt:   Knows about UNIX, Suns, HP
      bobcats and snakes, GNU
      emacs, X, Ethernet, Lisp
      and similar languages.
      Deputy Postmaster.
pb:   Postmaster, knows about
      UNIX, Suns, X, mail, news,
      and wide area
      communications.
[...]  If you anticipate a need
to login from outside Cambridge,
you should consult maj or pb.
```

Figure 1: From "Computing Facilities at the Computer Laboratory"

1.1 Plan of the paper

In the rest of the paper we present the basic principles of access control (§2), review the use of logic for reasoning about obligations and permissions in multi-agent systems (§3.1) and discuss the reasoning services we would like with such formalizations (§3.2).

After a brief discussion of the motivations behind the choice of the DEC-SRC calculus (§4), we discuss its notation and the corresponding intuitions, its formal syntax, the Kripke semantics and the computational characteristics (§5). Then we give some examples of problems that can be formulated with the calculus (§6).

Next, we present the tableau calculus (§7) and some examples of deduction (§8) for the sample problems we have seen in §6. Then we prove the soundness and the (partial) completeness of the calculus (§9).

Last we discuss some tricky aspects for the integration of roles in our formalism (§10) and conclude by pointing to future works (§11).

2 The Basic Principles of Access Control

The principles of access control in distributed systems can be described with few abstractions: *subjects* (humans, programs etc.), *objects* (data, other programs etc.) and *privileges* which subjects detain on objects (e.g. read, write and execute in UNIX). The use of these abstractions is the basis of most formal models proposed in the literature, starting from the traditional access matrix developed by Lampson for operating systems [27, 33] to more advanced methods [13, 26, 32, 35].

Using an access matrix is relatively simple: we put subjects along the rows, objects along the columns and privileges in the cells. Then checking what a subjects can do is just a table look-up[2].

[2]This may be technically complicated by the fact that the matrix is usually represented in compact form by lists along rows (capabilities) or along columns (access control lists) [34].

The problem becomes more complicated if we admit operations over such a matrix and are interested in *transition analysis* [22, 33]. In other words, we study the evolution of the system and its safety properties: given some privileges and some operations to change them, prove that the system will not evolve in an undesirable state.

In the verification of multi-agent distributed systems we are also interested also in *policy analysis*: do the actual low level access control procedures (and the decisions of agents based on them) respect the overall policies of the organization?

With an access matrix this is immediate: all privileges of every agents are explicitly written down. We can just check them one by one.

The problem is that the high-level security policies of an organization are seldom represented (let alone understood) at this level of details. For instance, an extensive field study by Ferraiolo et al. [14] pointed out that permissions are assigned to (human) users according their roles in the organization. How do we know that the low-level privileges assigned to an agent are really the logical consequence of the high-level policy based on roles and groups defined in our organization? From this perspective, the access matrix is extremely poor and it does not capture explicitly the richness of the underlying security policies. A key feature of the new approaches [5, 13, 17, 35] is the attempt to model more closely the (hierarchical) relationships between the various subjects, such as those described in Fig. 1.

We can revise the abstractions in use, consider *agents* (or principals) and *operations* over objects as primitives, and use a logic able of expressing the following properties:

- *complex agents* which could be users or software agents but also groups, roles, and agents using roles[3] or acting as delegates of other agents;
- *membership* relations between agents and groups or roles; agents must be able to belong to different groups and to play different roles;
- *hierarchies* between groups and roles, where permissions are inherited (the more powerful has at least all privileges of the less powerful);
- *delegation certificates* so that agents may pass their privileges to other agents on-the-fly;
- *privilege attributions* which link agents with the operations that they can accomplish.
- *imperative statements*, representing requests coming from other agents, and which must be considered by the agent making the access control decision.
- *indirect requests* by an agent on behalf of somebody else (for instance A must be able to say that B made a request).

[3]For instance the principals $Alice$ and $Alice\,\mathsf{as}\,professor$ are not the "same agent" for the principal $Bob\,\mathsf{as}\,student$.

3 Logics for Reasoning About Access Control

The abstractions we just introduced lead naturally towards a formalization of the problem with multi modal logics (alternative approaches can be found in [22, 33, 32, 17]). The requests, actions and obligations of agents can be mapped into modalities, atomic imperative or objective statements into propositions; other modal operators can be used to represent hierarchies, knowledge, permission etc.

The use of modal logics for modeling the obligations or the jurisdiction of agents has a long tradition (for instance see the work of Kanger [24]) and this research area is still very active [8, 18, 25, 39]. Traditional logics of knowledge and belief have also been extended to deal with security problems [6, 18, 37] and to incorporate desires and intentions [16].

3.1 Logics for ability versus logics for obligations

In systems for reasoning about knowledge and obligations (for instance in Cuppens & Demolombe [8] or Glasgow et al. [18]), atomic propositions represent state of affairs and the main concern is confidentiality. We usually look for properties like

$$\text{“}P \text{ knows } \varphi\text{”} \;\supset\; \text{“}P \text{ is allowed to know } \varphi\text{”} \tag{1}$$

With a theory of ability, focusing on actions and obligations (for instance in Kanger [24] or Krogh [25] or van der Meyden [39]), we may be more interested in other properties:

$$\text{“}P \text{ sees to it that } \varphi\text{”} \;\supset\; \text{“}\varphi \text{ is permitted”} \tag{2}$$

If atomic propositions are imperative statements (as in Abadi et al. [1]) we may look for something like

$$\text{“}P \text{ says } \varphi\text{”} \wedge \text{“}P \text{ controls } \varphi\text{”} \;\supset\; \varphi \tag{3}$$

In logics for authentication (for instance the BAN logic [6] or its extension by Syverson & van Oorschot [37]) a formula like $P\,\texttt{controls}\,\varphi$ is usually called the *jurisdiction* of a principal[4]. Loosely speaking we may say that if P controls φ (or equally P has privileges or jurisdiction over φ) then P can (is allowed to) make φ happen.

Still, if φ is an imperative sentence, we may wonder what it means to say that φ is true. The intuition is that φ is a request directed to the agent in charge of the access control decision. With this provision, we can interpret it as a propositional letter which can be true in a particular state (granted by the agent) or false (not granted).

To visualize the intuition behind the formulae we may compare a request such as $fm205\,\texttt{says}\,rm(test.c)$ to a command typed by the user on a UNIX terminal: `fm205@ely: rm test.c`. As we press enter, our imperative statement is evaluated by the agent in charge (in this case the operating system) according our access privileges and either executed or not (i.e. $rm(test.c)$ may be true or false).

Thus, the modal operator $P\,\texttt{says}\,\varphi$ is closer to the *Do* operator or the "P sees to it that φ" operator of Kanger [24]. When φ is not anymore a sentence describing a state of affairs but rather an imperative statement, we need to revise such definition.

[4]We refer to the BAN paper [6] or the work of Syverson & van Oorschot [37] for a discussion of the tricky notion of jurisdiction in computer security.

The key observation, from the viewpoint of access control, is that we should rather use an operator for "P tries φ", since P may not be allowed to do φ.

Another issue is the relations between agents, since most logics for obligations and multi-agent systems are flat: agents are on equal footing. This may be acceptable for modeling laws (in front of which we are supposed to be equal), but it is not so for an access control policy.

Hence our aim is to replace the reasoning based on schema (3) by the following one (more complex yet more realistic):

$$\text{“}P \text{ says } \varphi\text{”} \wedge \text{“}Q \text{ controls } \varphi\text{”} \wedge \text{ “}P \text{ is at least as powerful as } Q\text{”} \supset \varphi \qquad (4)$$

where the relation between P and Q is specified by an access control policy.

3.2 The Reasoning Services

Once we have formalized the information needed by an agent for an access control decision (e.g. group hierarchies, privileges, etc.) we also want to "do something" with it. In a nutshell we expect a number of reasoning services:

Consistency Check: check whether the assumptions have a model. Typically verify the compatibility of role hierarchies, users attributions etc. with the constraints.

Logical Consequence: decide whether a property φ logically follows from the assumptions. It could be used to decide whether a request should be granted or whether some delegation certificate is entailed even if is not explicitly written.

Model Generation: if a property φ does not follows from the assumptions, derive a counter model for it.

Compatibility: liberal systems may decide to grant a request provided that it is simply compatible (i.e. not inconsistent) with their security policies.

From this perspective tableau methods are extremely suitable since they provide a deductive machinery where all these services are available.

Axiom systems alone are inadequate for formal verification, because Hilbert systems require people to be good logicians and cannot be automated at all. The exclusive focus on axiomatization, without calculi for automated reasoning, may condemn logics for practical reasoning to be unused in practice and therefore fail in their very first aim.

4 The choice of the DEC-SRC logic

Among the various proposals, we focus on the expressive calculus developed at the Digital System Research Center (DEC-SRC) by Abadi, Lampson et al. [1] for a number of reasons:

- it provides a uniform and expressive framework for reasoning about access control in presence of delegation and hierarchies between agents;
- it has a simple and natural semantics based on Kripke models;

- it constitutes the basis of a real system [26, 40] whose properties have been checked (by hand) using the calculus;
- its features pose interesting technical challenges for deduction.

One of the characteristics which challenge "standard" tableau calculi is the presence of formulae used for modeling delegation certificates and hierarchical relationships between subjects (i.e. modalities). Those formulae have the same force of axiom schemata and are close to role-value-map constructs of AI languages [36].

The key difficulty is that we cannot "compile" them into tableau rules (nor axiom schemata) since their presence depends on the particular non logical axioms *and* the particular theorem we want to prove. Different axioms correspond to different security policies and we may not want to hardwire the policy in the logic. Moreover, access privileges may be passed on-the-fly by the agents. Thus a formula may contain a delegation of certain privileges from A to B (and thus be a valid request) another may not (and thus be invalid). Again we don't want to fix which delegation certificate are possible.

In a nutshell, some global properties of the underlying Kripke models can be discovered only *on-line* during the deduction process.

Another characteristic is the absence of the tree model property which also hinders a straightforward extension of tableau calculi. Finally it shares some properties of the universal modality [19, 20] which is not fully axiomatizable[5].

5 The DEC-SRC Calculus

To make the paper self-contained we present the major intuitions, the syntax and the semantics of the calculus and refer to Abadi et al. [1] for further discussions and to Lampson et al. [26, 40] for its applications. We only clarify some of its semantical features not discussed in [1].

5.1 Notation and Intuitions

The two building blocks of the calculus are operations and agents.

Operations over objects are represented by atomic imperative *statements* denoted by r (for request). Once again, they are propositional letters which can be true in a particular state of the system (request granted) or false (not granted). Complex statements are built with boolean connectives $\wedge, \neg, \supset$ etc.: for instance $\mathit{login}(\mathit{telnet}) \supset \mathit{login}(\mathit{ftp})$ whose intuitive interpretation is "if telnet access has been granted so has ftp access".

The names of agents, groups and cryptographic keys are represented by atomic *principals*, and denoted by a, b, K_a etc. Complex principals (P or Q) are built by conjunction "&" and quoting "|". The intuition is that $P\&Q$ is a principal with the privileges of both P and Q, whereas $P \mid Q$ corresponds to the principal P claiming to quote a request from Q. Notice that P may claim to quote Q even when Q never said anything.

[5]This may also explain why only a sound axiomatization has been devised in [1].

An example may better clarify the quoting construct. Suppose we received an email from $fm205$ asking for a request r. We often react as r had come directly from $fm205$ but, in reality, the request came from the mail agent *sendmail* which claimed to have received a mail from $fm205$ i.e. from the "composite" agent *sendmail* | $fm205$. We usually trust *sendmail* to forward all requests by $fm205$ and not to forge fake requests but we should not automatically rely on this assumption.

Other operators are possible i.e. $P\,\texttt{for}\,Q$ and $P\,\texttt{as}\,R$ [1]: the former is used when P is claiming to act as a delegate for Q; the latter when Q speaks using a role[6] R. They can be encoded using "|" and "&" (see [1] for a discussion), so we do not use them. For instance $P\ \texttt{for}\ Q \doteq (P\&d) \mid Q$ where d is an agent entrusted with delegation certificates. The incorporation of roles may be trickier since roles may have particular properties. So, we discuss their incorporation into this framework in more details in §10.

To represent *agent requests* we use the modal statement "$P\,\texttt{says}\,s$": principal P requests s to be granted. If P is a group then we follow Abadi et al. [1, 26] and interpret it as "somebody in group P says s". When P is a cryptographic key we mean that the request has been signed or encrypted with the corresponding key. So, following a common interpretation in computer security, we may identify a principal with a key[7].

Hierarchical relations between principals are constructed with the "speaks for" operator $P \Rightarrow Q$. The intuition is that P has at least all privileges of Q i.e. P can speak for Q. If P says s this would be as Q itself said s. Back to our example about the mail agent, in our day by day work we often assume that *sendmail* | $fm205 \Rightarrow fm205$.

The operator $\Rightarrow$ is also used for group membership: $P \Rightarrow G$ means that P has at least all privileges of group G. We also speak of a "certificate" when we have the formula $K \Rightarrow P$. Intuitively this formula binds K to P, so that all messages signed with K can be seen as coming from P.

Principals and statements are linked by *privileges attributions* [35]: the statement "$P\,\texttt{controls}\,s$" captures the intuition that principal P has access control over s. This is the *jurisdiction* of a principal (see §3).

In the DEC-SRC calculus `controls` is a defined symbol:

$$P\,\texttt{controls}\,s \quad\equiv\quad (P\,\texttt{says}\,s) \supset s$$

The intuition may become clearer by rephrasing it as "P controls s means that if P asks for s to be executed then s will indeed be executed by the agent in charge of the access control decision".

In future extensions, it may be desirable to replace material implication with an intuitionistic or a relevant implication. At present, a principal may pretend to control whatever she does not explicitly request i.e. $\neg(P\,\texttt{says}\,s) \supset P\,\texttt{controls}\,s$. Such situation may be a feature if we are interested in modelling "bluffing" principals. Anyhow, this is sufficient for modelling a wide range of situations.

To show how the logic can be used, we consider the case of the access control structure of Fig. 1.

[6]For a distinction between the security concepts of role and group see [13, 35].

[7]Since we are interested in access control and not in authentication or confidentiality we assume that the problems of associating keys with principals or finding whether keys are good etc. have been tackled already (see [6, 26, 37]).

$$\text{agents} = \begin{cases} maj \Rightarrow SysAdm \\ gt \Rightarrow (SwAdm\&DepPostM) \\ pb \Rightarrow PostMaster \\ fm205 \Rightarrow RemUsr \end{cases} \qquad \text{priv.} = \begin{cases} SysAdm \;\texttt{controls}\; (fm205 \Rightarrow Usr) \\ Usr \;\texttt{controls}\; login(telnet) \\ RemUsr \;\texttt{controls}\; login(ftp) \\ (fm205\&Usr) \;\texttt{controls}\; read(mail) \end{cases}$$

$$\text{groups} = \begin{cases} SysAdm \Rightarrow SwAdm \\ SysAdm \Rightarrow PostMaster \\ SwAdm \Rightarrow Usr \\ PostMaster \Rightarrow DepPostM \\ DepPostM \Rightarrow Usr \\ Usr \Rightarrow RemUsr \end{cases} \qquad \text{req.} = \begin{cases} maj \;\texttt{says}\; (fm205 \Rightarrow Usr) \\ fm205 \;\texttt{says}\; login(telnet) \\ fm205 \;\texttt{says}\; read(mail) \end{cases}$$

Figure 2: A Logical Formalization of Fig. 1

We use an assumption $P \Rightarrow Q$ for every direct subsumption relation between roles. So we have $SysAdm \Rightarrow SwAdm$ and $SwAdm \Rightarrow Usr$. The role membership of the various agents is also described with the $\Rightarrow$ operator, so we have $maj \Rightarrow SysAdm$ and $fm205 \Rightarrow RemUsr$. We do not need to specify in details all relations between agents and roles: the logic does the transitive closure of the speaks-for relation for us. For instance $pb \Rightarrow Usr$ is a logical consequence of our assumptions. We can verify it with the tableau procedure in §7.

We also introduce some privileges: every remote user can log in with ftp but only authorized users can use telnet. The privilege of reading one's mail is a possible example of the use of conjunction: the mail belongs to $fm205$ so he (or a delegate of him) is the only one who can read it, but he must have being logged in as a legitimate user. Notice that the system administrator can change the group membership of $fm205$. So, the possibility of logging from remote sites depends on maj's statements.

Some possible statements are also shown with the final formalization in Fig. 2.

5.2 Formal Syntax

The formal language is the described by the following syntax, where r is an atomic request (a propositional letter) and a is an atomic principal (an agent or a key):

$$\begin{array}{rcl} P, Q & ::= & a \mid P\&Q \mid (P \mid Q) \\ s, s' & ::= & r \mid \neg s \mid s \wedge s' \mid P \;\texttt{says}\; s \mid (P \Rightarrow Q) \end{array}$$

Other connectives are abbreviations, e.g. $s \supset s' \equiv \neg(s \wedge \neg s')$. We also assume that in $P \Rightarrow Q$ either P or Q is an atomic principal, w.l.o.g. since $P \Rightarrow Q$ is equivalent to $P \Rightarrow a \wedge a \Rightarrow Q$ for a new atomic a.

For sake of readability we try to omit parentheses whenever possible. So we make the following assumptions: $\mid$ and & bind more strongly than $\Rightarrow$; operators over principals have higher precedence than modal and propositional connectives; modal connectives such as `says` and `controls` have a higher precedences than propositional ones. Finally, we assume the usual precedence relation among propositional connectives.

So $\neg(A \mid B \;\texttt{says}\; s) \wedge A \mid B \Rightarrow B \supset B \mid A \;\texttt{says}\; s$ should read as follows:

$$(\neg ((A \mid B) \;\texttt{says}\; s) \wedge ((A \mid B) \Rightarrow B)) \supset ((B \mid A) \;\texttt{says}\; s)$$

A statement is *left (right) restricted* when speaks-for subformulae have the form $a \Rightarrow Q$ (respectively $P \Rightarrow a$) i.e. the left (right) principal is atomic. It is *weakly left (right) restricted* if all speaks-for statements under the scope of an even number of negation are left (right) restricted. This means that arbitrary statements $P \Rightarrow Q$ are admitted only under the scope of an odd number of negations[8]. It is *request restricted* when in each statement of the form $P\,\texttt{says}\,s$, the statement s is either an atomic request or a group membership (each possibly negated). For instance the logical formalization of the access control structure in Fig. 2 is left, right and request restricted.

In many cases, statements are right and request restricted. If "$\Rightarrow$" is used for hierarchies and group and role membership, as in Fig. 1, the rightmost principal is atomic. Moreover, in almost all systems [34], privileges attributions are represented by ACL (Access Control Lists). In the DEC-SRC language, an ACL for a request r is simply the conjunction of statements $\bigwedge_i P_i\,\texttt{controls}\,r$, where r is uninterpreted [26, 35]. If we add, among the possible privileges, the possibility to hand over delegation to other principals such as $P_i\,\texttt{controls}\,(Q_j \Rightarrow a_k)$, then we still have a right restricted language. Further examples can be found in §6.

5.3 Kripke Semantics

The semantics is based on Kripke models [15, 23]: a relation models the compatibility of a state with the requests made by a principal in the real world. Thus, a model is a pair $\langle W, \mathcal{I}\rangle$, where W is a non empty set of states (sometimes called worlds) and $(\cdot)^{\mathcal{I}}$ an interpretation such that for every atomic principal a we have $(a)^{\mathcal{I}} \subseteq W \times W$ and for every propositional letter r we have $(r)^{\mathcal{I}} \subseteq W$. Then $(\cdot)^{\mathcal{I}}$ is extended as follows:

$$\begin{array}{lcl}
(\neg s)^{\mathcal{I}} & \doteq & W - (s)^{\mathcal{I}} \\
(s \wedge s')^{\mathcal{I}} & \doteq & (s)^{\mathcal{I}} \cap (s')^{\mathcal{I}} \\
(P \Rightarrow Q)^{\mathcal{I}} & \doteq & \text{if } (Q)^{\mathcal{I}} \subseteq (P)^{\mathcal{I}} \text{ then } W \text{ else } \emptyset \\
(P\,\texttt{says}\,s)^{\mathcal{I}} & \doteq & \left\{ w \mid \forall v \in W \text{ if } \langle w, v\rangle \in (P)^{\mathcal{I}} \text{ then } v \in (s)^{\mathcal{I}} \right\} \\
(P \& Q)^{\mathcal{I}} & \doteq & (P)^{\mathcal{I}} \cup (Q)^{\mathcal{I}} \\
(P \mid Q)^{\mathcal{I}} & \doteq & \left\{ \langle w, v\rangle \mid \exists u\, \langle w, u\rangle \in (P)^{\mathcal{I}} \text{ and } \langle u, v\rangle \in (Q)^{\mathcal{I}} \right\}
\end{array}$$

We write $w \Vdash s$ for $w \in (s)^{\mathcal{I}}$ and say that w *satisfies* s. For simplicity we interchange a set of statements with their conjunction. A model $\langle W, \mathcal{I}\rangle$ *satisfies* a set of statements S iff every state in W satisfies S.

Definition 1 A statement s is *valid* if every model $\langle W, \mathcal{I}\rangle$ satisfies s. It is a *logical consequence* of a set of global assumptions G iff in every model $\langle W, \mathcal{I}\rangle$ which satisfies G every state w satisfies s.

In this framework the global assumptions are non-logical axioms describing the access control system: groups membership, privileges attributions etc.

Global assumptions can be added in the axiomatization of Abadi et al. [1] with the modal deduction theorem [23, 15], but their explicit representation is more effective because the modal deduction theorem leads to an exponential blow up [12].

[8]For instance the formula $\neg(a\,\texttt{says}\,((b\&c) \Rightarrow d))$ is weakly left restricted since the group membership $(b\&c) \Rightarrow d$ is under the scope of one negation.

5.4 Semantical Properties

It is worth noting that, in contrast with other operators, the semantics of $P \Rightarrow Q$ reflects global properties of the model. Indeed it is the only operator for which a statement must hold either everywhere or nowhere, since either $(P \Rightarrow Q)^{\mathcal{I}} = W$ or $(P \Rightarrow Q)^{\mathcal{I}} = \emptyset$.

This property is very close to the universal modality discussed by Goranko et. al [19, 20]. We can represent explicitly the relation between $\Rightarrow$ and the universal modality, denoted by $\mathit{everybody}\,\texttt{says}\,s$, as follows:

$$P \Rightarrow Q \equiv \mathit{everybody}\,\texttt{says}\,(P \Rightarrow_{loc} Q)$$

where we have the following two conditions:

$$\begin{array}{rcl} (P \Rightarrow_{loc} Q)^{\mathcal{I}} & \doteq & \{w \mid \forall v \in W \text{ if } \langle w, v\rangle \in (Q)^{\mathcal{I}} \text{ then } \langle w, v\rangle \in (P)^{\mathcal{I}}\} \\ (\mathit{everybody})^{\mathcal{I}} & \doteq & W \times W \end{array}$$

We can use this relation to introduce axiom schemata "on the fly". Indeed, enforcing $\mathit{everybody}\,\texttt{says}\,s$ in a world is equivalent to enforce s in the whole model and hence enforcing s as a non-logical axiom.

For instance $P \Rightarrow P \mid P$ forces the transitivity of relation $(P)^{\mathcal{I}}$, where P may be a complex principal. Yet, these global properties may or may not be present. As an example, suppose we have $A\,\texttt{says}\,(B \Rightarrow B \mid B)$. Transitivity of B follows only if $\neg(A\,\texttt{says}\,\bot)$ is the case. So B's properties depend on the particular global assumptions and theorems we are trying to prove.

Another "feature" is the *absence of the tree-model property* [38, 23]. For instance the following formula has no tree model at all:

$$\begin{array}{lll} a \Rightarrow a \mid a \wedge & \% & a \text{ is transitive} \\ \neg(a \Rightarrow b) \wedge & \% & b \text{ is not included in } a \\ b\,\texttt{says}\,\bot \wedge \neg(a\,\texttt{says}\,\bot) \wedge & \% & b \text{ is inconsistent and } a \text{ is not} \\ a\,\texttt{says}\,(b\,\texttt{says}\,\bot \wedge \neg a\,\texttt{says}\,\bot) & \% & \text{and so on (at least for } a) \end{array}$$

Yet, it is satisfiable in the state 1 of the model with $W = \{1, 2, 3\}$ and $(a)^{\mathcal{I}} = \{\langle 1, 3\rangle, \langle 3, 3\rangle\}$ and $(b)^{\mathcal{I}} = \{\langle 2, 2\rangle\}$. The key point is that this model has *two* clusters (connected components) so that state 1 satisfies the (local) `says` statements and state 2 satisfies the global $\neg(a \Rightarrow b)$.

In model theoretic terms, (un)satisfiability is not preserved under disjoint union as in "traditional" modal logics [38]. This is due to the "hidden" presence of the universal modality which makes impossible the complete characterization of the logic with an Hilbert system [19, 20, 38].

5.5 Undecidability

Validity is *undecidable* in general and a reduction to Thue systems (without details) is pointed out in [1]. A simpler proof uses the techniques of Schmidt-Schauss [36] for AI languages and reduces validity to the word problem of (semi)groups.

We map each element of the group a_g to an atomic principal a and composition "$\circ$" to quoting "$|$". Equations between words $p_g \cong q_g$ becomes conjunctions of speaks-for statements $P \Rightarrow Q \wedge Q \Rightarrow P$ (for short $P = Q$) for the corresponding principals P and Q.

To define groups we must add, for each atomic principal a, a new principal a^c corresponding to its converse. Then we add the equation such as $a \mid a^c = e$ where e is the atomic principal corresponding to the identity element e_g of the group.

Finally we introduce a new principal g and the global assumptions $g \mid a = g$ for every atomic a and the statement $g \,\mathsf{says}\, (P = Q)$ for every $p_g \cong q_g$ characterizing the group. Then one can prove that $g \,\mathsf{says}\, (P' = Q')$ is valid with those assumptions iff equation $p'_g \cong q'_g$ holds for the group (see Schmidt-Schauss [36] for the proof).

6 Examples of Access Control Problems

In this section we use the logic to formalize the information of agents who have to make the access control decision for a number of problems.

We have already shown the formalization of the access control structure of Fig. 1 in Fig. 2. Then we can consider the following problem:

Example 2 A software agent works as a firewall in a Computer Laboratory whose policy is formalized in Fig. 2. What should the agent do when maj grants to $fm205$ the membership in the role of $User$ and $fm205$ asks to login with telnet and tries to read his mail, should the agent grant $\mathit{login}(\mathit{telnet}) \wedge \mathit{read}(\mathit{mail})$?

If we abbreviate $\mathit{req} \doteq \mathit{login}(\mathit{telnet}) \wedge \mathit{read}(\mathit{mail})$, then the statement we would like to check is the following:

$$maj \,\mathsf{says}\, (fm205 \Rightarrow User) \wedge fm205 \,\mathsf{says}\, \mathit{req} \supset \mathit{req}$$

The global assumptions we use are those denoted by *agents*, *groups* and *priv.* in Fig. 2. If the statement above is a logical consequence of our assumptions then the agent in charge of the firewall could "safely" grant the request of $fm205$.

The next example is the following:

Example 3 The agent *Paymaster* is responsible for the automatic handling of invoices and payments in the department. *Alice* is responsible for the accounts, *Bob* for marketing and ordering of goods, and $Charlie$ is the storekeeper in charge of receiving and shipping of goods. *Alice* is trusted in forwarding *Bob* orders. *Sam* is the security agent responsible for the authentication server. The policy of the department is to pay after the store has certified the arrival of goods. Messages and certificates are shown in Fig. 3. Should the agent *Paymaster* actually *pay*?

The corresponding formalization is shown in Fig. 3. We used the same technique used for Fig. 2 in §5.1: group memberships are represented by statements of the form $\mathit{agent} \Rightarrow \mathit{department}$ for every agent and every department; key certificates have the form $\mathit{key} \Rightarrow \mathit{agent}$; signed messages are denoted by $\mathit{key} \,\mathsf{says}\, \mathit{message}$. The set of privileges is immediate.

Another example concerns the access control decision in presence of the standard hierarchy of "clearance" with read-down and write-up policies (see Sandhu & Samarati [34] for a discussion). For instance, with a read-down security policy an agent can only read documents with a clearance lower than his own. Although it seems a problem

$$\text{agents} = \left\{ \begin{array}{l} Alice \Rightarrow Account \\ Bob \Rightarrow Marketing \\ Charlie \Rightarrow Store \\ Sam \Rightarrow Server \end{array} \right. \qquad \text{priv.} = \left\{ \begin{array}{l} Marketing \texttt{ controls } order \\ Account \texttt{ controls } pay \\ Store \texttt{ controls } recvd \\ Alice \texttt{ controls } (Bob \texttt{ says } order) \\ Server \texttt{ controls } (K_A \Rightarrow Alice) \\ Server \texttt{ controls } (K_B \Rightarrow Bob) \end{array} \right.$$

$$\text{msg} = \left\{ \begin{array}{l} K_A \texttt{ says } Bob \texttt{ says } order \\ K_C \texttt{ says } recvd \\ K_S \texttt{ says } K_A \Rightarrow Alice \\ K_S \texttt{ says } K_C \Rightarrow Charlie \end{array} \right. \qquad \begin{array}{l} \text{cert.} = \left\{ \; K_S \Rightarrow Sam \right. \\ \\ \text{constr.} = \left\{ \; Account \texttt{ says } (order \wedge recvd \supset pay) \right. \end{array}$$

Figure 3: A Model with Keys for Example 3

$$\text{groups} = \left\{ \begin{array}{l} A \Rightarrow 15 \\ 15 \Rightarrow PG \\ PG \Rightarrow F \end{array} \right. \qquad \text{priv.} = \left\{ \begin{array}{l} A \texttt{ controls } view(m_1) \\ A \texttt{ controls } load(m_1) \\ 15 \texttt{ controls } load(m_1) \\ 15 \texttt{ controls } view(m_2) \\ F \texttt{ controls } stop(m_1) \\ Decoder \texttt{ controls } (K_S \Rightarrow Sam) \\ Decoder \texttt{ controls } (K_B \Rightarrow Bob) \\ Decoder \texttt{ controls } (K_C \Rightarrow Charlie) \end{array} \right.$$

$$\text{agents} = \left\{ \begin{array}{l} Sam \Rightarrow A \\ Bob \Rightarrow 15 \\ Charlie \Rightarrow F \end{array} \right.$$

$$\text{cert.} = \left\{ \begin{array}{l} K_D \Rightarrow Decoder \\ K_S \Rightarrow Sam \end{array} \right. \qquad \text{msg} = \left\{ \begin{array}{l} K_D \texttt{ says } K_B \Rightarrow Bob \\ K_B \texttt{ says } Sam \texttt{ says } (load(m_1) \wedge view(m_1)) \end{array} \right.$$

Figure 4: A Read-down Policy for "Cyber-sitter"

restricted to the military, its relevance spans far beyond this field. Video on demand is a good example.

Example 4 Consider the hierarchy established by a corporation for video on demand: F for films suitable for the whole family, PG for those requiring parental guidance, 15 for those limited to teenager at least 15 years old, and A for adults only. The users are *Sam* with A privileges, *Bob* with 15 and *Charlie* who has none. The agent *Alice*, sometimes called "Cyber-sitter" supervises the requests from the users which are entitled to different movies and tries to avoid that minors have access to unsuitable movies. She received a message and some delegation certificates from the *Decoder* (Fig. 4). Should $load(m_1) \wedge view(m_1)$ be granted with the equivalent of a read-down policy for viewing?

In other terms is $load(m_1) \wedge view(m_1)$ a logical consequence of the access control model described in Fig. 4? Again, recall that we represent the information from the viewpoint of *Alice*, the agent in charge of the access control decision. Even in this simple case is not easy to see that it is *not*. Hence the need of an automated reasoning method.

For a "real-life" example we take delegation without certificates from Abadi et al. [1, page 719] where a careful (and non trivial) Hilbert proof is given. The problem that an agent C has to face is the following:

Example 5 "A delegates to B who makes requests to C. For instance A may be a user with a sufficiently powerful smart-card, B a workstation and C a file server. [...]

When B wishes to make a request r on A's behalf, B sends the signed requests along with A's name... in the format $K_B\,\texttt{says}\,(A\,\texttt{says}\,r)$... When C receives the request r he has evidence that B has said A has requested r but not that A has delegated to B; then C consults the ACL for request r and determines whether the request should be granted. [...] A certification authority provides the certificates for the principals' public keys as needed. The necessary certificates are $K_S\,\texttt{says}\,(K_A \Rightarrow A)$ and $K_S\,\texttt{says}\,(K_B \Rightarrow B)$, where K_S is S's public key."

In symbols we are asking whether $G \models s \supset r$ where G and s are defined as follows:

$$\begin{aligned} G &\doteq \{\, K_S \Rightarrow S, S\,\texttt{controls}\,(K_B \Rightarrow B), Sc \Rightarrow A \,\} \cup ACL \\ s &\doteq K_B\,\texttt{says}\,(Sc\,\texttt{says}\,r) \wedge K_S\,\texttt{says}\,(K_B \Rightarrow B) \end{aligned}$$

Where ACL is a set of statements of the form $P\,\texttt{controls}\,s$. For instance we may assume that $(B \mid A)\,\texttt{controls}\,r \in ACL$.

7 Prefixed Tableaux

To define our tableau calculus we capitalize on the deductive machinery with prefixed tableaux which have been developed for modal and dynamic logic (see Fitting [15], Goré [21] and De Giacomo & Massacci [9] for further references).

Prefixed tableaux for the DEC-calculus use *prefixed statements*, i.e. pairs $\sigma : s$ where s is a statement and σ is an alternating sequence of integers n and atomic principals a called prefix and defined as $\sigma ::= n \mid \sigma.a.n$.

Remark 6 A key difference from "standard" prefixed tableaux for modal logics (such as [15, 21]) is that a prefix does not always start with 1. So a set of prefixes is a *forest of trees*, where arcs are labelled with atomic principals and nodes with integers.

With k initial prefixes we have, in graph-theoretic terminology, k connected components or clusters. With global assumptions and the operator $\Rightarrow$ we can impose an euclidean or transitive closure on a cluster but we cannot collapse two clusters.

The definition of prefixed tableau is standard [15, 21]. A *tableau* $\mathcal{T}$ for a statement s is a rooted (binary) tree where nodes are labelled with prefixed statements according the rules of the calculus in the usual fashion:

- $1 : s$ labels the root of the tree;
- if the antecedent of a rule labels a node in a path from the root to a leaf then we can extend the tree by adding to the leaf of the path one or more children, each labelled with a consequent of the rule.

What changes is the notion of branch:

Definition 7 A *branch* of a tableau $\mathcal{T}$ is a pair $\langle \mathcal{B}, G_{\mathcal{B}} \rangle$ where $\mathcal{B}$ is a path from the root to a leaf of $\mathcal{T}$ and $G_{\mathcal{B}}$ is a set of global assumptions.

Thus, each time we split a branch of the tree, we should also duplicate (in theory) the set of global assumption. This definition is essential because we need to modify

$$\langle|\rangle : \frac{\sigma : \neg(P \mid Q\,\mathsf{says}\,s)}{\sigma : \neg(P\,\mathsf{says}\,(Q\,\mathsf{says}\,s))} \qquad [|] : \frac{\sigma : P \mid Q\,\mathsf{says}\,s}{\sigma : P\,\mathsf{says}\,(Q\,\mathsf{says}\,s)}$$

$$\langle\&\rangle : \frac{\sigma : \neg(P\&Q\,\mathsf{says}\,s)}{\sigma : \neg(P\,\mathsf{says}\,s) \quad \sigma : \neg(Q\,\mathsf{says}\,s)} \qquad [\&] : \frac{\sigma : P\&Q\,\mathsf{says}\,s}{\begin{array}{c}\sigma : P\,\mathsf{says}\,s\\ \sigma : Q\,\mathsf{says}\,s\end{array}}$$

$$\text{Glob} : \frac{\vdots}{\sigma : s} \quad \text{if } \sigma \text{ is present in } \mathcal{B} \text{ and } s \in G_{\mathcal{B}}$$

$$\langle a\rangle : \frac{\sigma : \neg(a\,\mathsf{says}\,s)}{\sigma.a.m : \neg s} \text{ with } \sigma.a.m \text{ new in } \mathcal{B}$$

$$[a] : \frac{\sigma : a\,\mathsf{says}\,s}{\sigma.a.n : s} \text{ with } \sigma.a.n \text{ present in } \mathcal{B}$$

$$D(a) : \frac{\sigma : a\,\mathsf{says}\,s}{\sigma : \neg(a\,\mathsf{says}\,\neg s)} \text{ with some } \sigma.a.n \text{ present in } \mathcal{B}$$

Figure 5: Rules for "Modal" Connectives

the set of global assumptions during the deduction process and therefore different branches may end up with different and possibly inconsistent global assumptions.

The notions of prefixes present or new in a branch $\langle\mathcal{B}, G_{\mathcal{B}}\rangle$ are defined in the usual way [21]: a prefix σ is *present* if there is a node in $\mathcal{B}$ labelled by a prefixed statement $\sigma : s$ for some s; it is *new* if it is not present.

7.1 Tableaux Rules

The rules for propositional connectives are identical to those for modal and dynamic logics and therefore omitted (see [9, 15, 21]). The rules for conjunction, quoting, the use of global assumptions and the transitional rules for atomic principals are in Fig. 5. We assume that each rule is applied to a prefixed statement labeling a node in the branch $\langle\mathcal{B}, G_{\mathcal{B}}\rangle$.

Note the similarity of the rules for "|" and "&" operators with the corresponding rules for sequential and nondeterministic composition in dynamic logic [9].

To cope with "$\Rightarrow$" we introduce a new set of propositional atoms x_i (distinct from r) to mark unsaid statements as in Fig. 6. Since $P \Rightarrow Q$ implies that if $P\,\mathsf{says}\,s$ then $Q\,\mathsf{says}\,s$ for all s, its negation means that there is "something" (an unknown x_i) which principal P said but principal Q didn't. The first two rules correspond to the local features of the $\Rightarrow$ operator, whereas the last is due to its "universal" flavor. The $\langle U_{gr}\rangle$-rule combines both aspects.

Remark 8 For weakly left restricted statements we may use but do *not* need rules $\langle R_{gr}\rangle$ and $D(a)$ while for right restricted statements we do not need rule $[L_{gr}]$.

These rules are sound for the whole language (see further Thm. 13) but the completeness proof of Thm. 14 (see §9) for weakly left restricted fragments does not make use of the rules $\langle R_{gr}\rangle$ and $D(a)$. So we may or may not use them.

$$\langle R_{gr}\rangle : \frac{\sigma : P \Rightarrow a \quad \sigma : \neg(a\,\mathsf{says}\,s)}{\sigma : \neg(P\,\mathsf{says}\,s)} \qquad [L_{gr}] : \frac{\sigma : a\,\mathsf{says}\,s \quad \sigma : a \Rightarrow Q}{\sigma : Q\,\mathsf{says}\,s}$$

$$\langle U_{gr}\rangle : \frac{\sigma : \neg(P \Rightarrow Q)}{\begin{array}{c} n : P\,\mathsf{says}\,x_i \\ n : \neg(Q\,\mathsf{says}\,x_i)\end{array}} x_i \text{ and } n \text{ new} \qquad [U_{gr}] : \frac{\sigma : P \Rightarrow Q}{G_{\mathcal{B}} := G_{\mathcal{B}} \cup \{P \Rightarrow Q\}}$$

Figure 6: Rules for the speaks-for operator

Needless to say, we can introduce a number of optimizations as in modal and dynamic logic. For instance we may apply the following observation:

Observation 9 Rule $\langle a\rangle$ must be applied to a prefixed statement $\sigma : \neg(a\,\mathsf{says}\,s)$ only if there is no prefixed statement $\sigma.a.n : \neg s$ already present in the branch.

See [3, 9, 21] for further references and techniques.

Other optimizations are linked to this particular calculus. For instance the statement $P \Rightarrow (A\&B)$ is logically equivalent to $P \Rightarrow A \wedge P \Rightarrow B$. A rule transforming statements of the former type into the latter type can be useful since it may lead to right-restricted formulae. We can also simplify our proof by using the following observation:

Observation 10 Rule $\langle U_{gr}\rangle$ can be applied only once for each subformula $\neg(P \Rightarrow Q)$ in the branch, no matter its prefix.

Definition 11 A branch $\langle \mathcal{B}, G_{\mathcal{B}}\rangle$ is *contradictory* if $\mathcal{B}$ contains both $\sigma : s$ and $\sigma : \neg s$, for some s and σ. It is *open* if all possible rules have been applied and it is not contradictory. A tableau is *closed* if all branches are contradictory; it is *open* if at least one branch is open.

Definition 12 A *tableau proof* for statement s with global assumptions G is a closed tableau starting with the branch $\langle \{\, 1 : \neg s \,\}, G\rangle$.

We can now prove that the calculus is sound:

Theorem 13 (Strong Soundness) If s has a tableau proof with global assumptions G then s is a logical consequence of G.

As for completeness, we can only prove a weaker result, namely that the tableau calculus is strongly complete for the two main fragments introduced so far:

Theorem 14 (Strong WL-Completeness) If s is a logical consequence of G and $G \cup \{\neg s\}$ is weakly left restricted then s has a tableau proof.

Theorem 15 (Strong WRR-Completeness) If s is a logical consequence of G and $G \cup \{\neg s\}$ is weakly right and request restricted then s has a tableau proof.

If the calculus is complete for the fragment at hand, then a satisfiability witness is an open branch of the tableau starting with $\langle \{\, 1 : s \,\}, G\rangle$.

7.2 Decidability in Practice

A decision method (rather than a semi-decidable procedure which could be based on first order translations) is important for security analysis because satisfiability gives information on security weaknesses.

To obtain a decision method a sufficient condition, checkable in polynomial time, can be imposed on the global assumptions and the consequence s. Associate a graph to the global assumptions and the negation of the formula to be proved as follows:

- each atomic principal is represented by a node,
- for every $P \Rightarrow Q$ (under the scope of an even number of negation) draw an arc from the atomic principals in P to those in Q

If the resulting graph is *acyclic* then the tableau construction terminates by using loop checking with an extended notion of the Fischer-Ladner closure [9]. Notice that in the embedding of the word problem (§5.5), the principal g creates cycles.

In access control, acyclicity is not a restriction but a requirement: if $\Rightarrow$ is used for hierarchies of groups/roles then cycles are not allowed (see Ferraiolo et al. [13] or Sandhu et al. [35]).

8 Some Examples of Deduction

For sake of simplicity, we assume that we can directly reduce $\supset$ and `controls` with the obvious α and β rules rather than translating them back to $\wedge$, $\neg$ and `says` and then apply the corresponding rules.

The first example is the derivation of *hand-off axioms* [1, 26]:

$$\neg(a\,\texttt{says}\,\bot) \;\supset\; (a\,\texttt{controls}\,(P \Rightarrow a)).$$

We need a statement of this form whenever principal a must be able to hand over her privileges to P. It cannot be proved within the Hilbert system developed by [1] and must be added to the assumptions. The tableau derivation is shown in Fig. 7. Once again, the key step is the application of rule $[U_{gr}]$ which cannot be axiomatized.

To check that global assumptions must be linked to a branch, try the statement below using one set of global assumptions for the whole tableau. This statement is not valid, yet the tableau constructed by using only one set of global assumptions can be closed.

$$(b\,\texttt{says}\,(b \Rightarrow a) \wedge b\,\texttt{says}\,r \wedge b\,\texttt{says}\,\neg r) \supset a\&b\,\texttt{says}\,\bot$$

As a practical example we show the deduction necessary for Example 2 (page 12) in Fig. 8. We abbreviate *SysAdm* in *Sys*, *login*(*telnet*) in *telnet*, and *read*(*mail*) in *mail*. For sake of readability we omit the root of the tableau which is labelled by the negated goal formula:

$$^{(a)}\; 1 : \neg((fm\,\texttt{says}\,(telnet \wedge mail) \wedge maj\,\texttt{says}\,(fm \Rightarrow usr)) \supset (telnet \wedge mail))$$

Last, we show the deduction necessary for Example 5 (pag.13). We use the logic for the reasoning of the agent C and add a level of indirection to the original problem

(a)	$1 : \neg\,(\neg(a\,\texttt{says}\,\bot)\ \supset\ (a\,\texttt{controls}\,P \Rightarrow a))$	
	$G := \{\}$	
(b)	$1 : \neg(a\,\texttt{says}\,\bot)$	by α to (a)
(c)	$1 : \neg a\,\texttt{controls}\,(P \Rightarrow a)$	
(d)	$1 : a\,\texttt{says}\,(P \Rightarrow a)$	by reduce $\texttt{controls}$ to (c)
(e)	$1 : \neg(P \Rightarrow a)$	
(f)	$2 : P\,\texttt{says}\,x_1$	by $\langle U_{gr}\rangle$ to (e)
(g)	$2 : \neg(a\,\texttt{says}\,x_1)$	
(h)	$1.a.3 : \neg\bot$	by $\langle a\rangle$ to (b)
(i)	$1.a.3 : P \Rightarrow a$	by $[a]$ to (d)
(l)	$G_1 := \{\,P \Rightarrow a\,\}$	by $[U_{gr}]$ to (i)
(m)	$2 : P \Rightarrow a$	by *Glob* to (G_1)
(n)	$2 : \neg(P\,\texttt{says}\,x_1)$	by $\langle R_{gr}\rangle$ to (g, m)
	contradiction between (g, n)	

Figure 7: Tableau Proof of an Hand-off Axiom

by modeling explicitly the smart-card Sc. The statement we want to prove is the following:

$$(K_B\,\texttt{says}\,(Sc\,\texttt{says}\,r) \wedge K_S\,\texttt{says}\,(K_B \Rightarrow B) \wedge (B \mid A)\,\texttt{controls}\,r) \supset r$$

Therefore the tableau starts with

$$1 : \neg\,((K_B\,\texttt{says}\,(Sc\,\texttt{says}\,r) \wedge K_S\,\texttt{says}\,(K_B \Rightarrow B) \wedge (B \mid A)\,\texttt{controls}\,r) \supset r)$$

The tableau proof is shown in Fig. 9. In Fig. 9 only $[L_{gr}]$-rule is used. A derivation with only $\langle R_{gr}\rangle$-rule is possible.

This is an example of the "not compilability" of $P \Rightarrow Q$ into axiom schemata (or rules). Indeed the statement $K_B \Rightarrow B$ corresponds to the semantical property $B^{\mathcal{I}} \subseteq K_B^{\mathcal{I}}$, i.e. to the axiom schema $K_B\,\texttt{says}\,s \supset B\,\texttt{says}\,s$. The problem is that it is not always valid, since it depends on the server's statement $S\,\texttt{says}\,(K_B \Rightarrow B)$. Without the server's certificate, it doesn't hold. So, if we try to prove a different theorem without this certificate this axiom schema turns out to be invalid. The possibility of adding "on-line" properties is critical here, because delegations and groups membership depend on security policies and current certificates and therefore cannot be fixed a priori in the calculus.

9 Soundness and (partial) completeness

The overall structure of the proofs of soundness and completeness is similar to the corresponding proofs for modal logics [15, 21, 28] and dynamic logic [9]. We must "only" adapt the proofs to the new connectives.

At an high-level the soundness proof requires the following steps:

1. devise a mapping between prefixes in a tableau and states in a model;

$$\begin{tikzcd}
& \begin{array}{l} {}^{(a)}\ 1 : \ldots \\ {}^{(b)}\ 1 : fm\ \mathtt{says}\ (telnet \wedge mail) \\ {}^{(c)}\ 1 : maj\ \mathtt{says}\ (fm \Rightarrow Usr) \\ {}^{(d)}\ 1 : \neg(telnet \wedge mail) \\ {}^{(e)}\ 1 : maj \Rightarrow Sys \\ {}^{(f)}\ 1 : Sys\ \mathtt{says}\ (fm \Rightarrow Usr) \\ {}^{(g)}\ 1 : Sys\ \mathtt{controls}\ (fm \Rightarrow Usr) \end{array} \arrow[dl] \arrow[dr] & & \\
\begin{array}{l} {}^{(h_1)}\ 1 : \neg(Sys\ \mathtt{says}\ (fm \Rightarrow Usr)) \\ \times \end{array} & & \begin{array}{l} {}^{(h_2)}\ 1 : fm \Rightarrow Usr \\ {}^{(i)}\ 1 : Usr\ \mathtt{says}\ (telnet \wedge mail) \end{array} \arrow[dl] \arrow[dr] & \\
& \begin{array}{c} {}^{(l_1)}\ 1 : \neg telnet \\ \mathcal{T}_1 \end{array} & & \begin{array}{c} {}^{(l_2)}\ 1 : \neg mail \\ \mathcal{T}_2 \end{array}
\end{tikzcd}$$

$$\begin{tikzcd}
& \begin{array}{l} \mathcal{T}_1 \\ {}^{(m)}\ 1 : Usr\ \mathtt{controls}\ telnet \end{array} \arrow[dl] \arrow[dr] & \\
\begin{array}{l} {}^{(n_1)}\ 1 : \neg(Usr\ \mathtt{says}\ telnet) \\ {}^{(o)}\ 1.Usr.2 : \neg telnet \\ {}^{(p)}\ 1.Usr.2 : telnet \wedge mail \\ {}^{(q)}\ 1.Usr.2 : telnet \\ {}^{(s)}\ 1.Usr.2 : mail \\ \times \end{array} & & \begin{array}{l} {}^{(n_2)}\ 1 : telnet \\ \times \end{array}
\end{tikzcd}$$

$$\begin{tikzcd}
& & \begin{array}{l} \mathcal{T}_2 \\ {}^{(t)}\ 1 : (fm \& Usr)\ \mathtt{controls}\ mail \end{array} \arrow[dl] \arrow[dr] & \\
& {}^{(u_1)}\ 1 : \neg((fm \& Usr)\ \mathtt{says}\ mail) \arrow[dl] \arrow[dr] & & \begin{array}{l} {}^{(u_2)}\ 1 : mail \\ \times \end{array} \\
\begin{array}{c} {}^{(v_1)}\ 1 : \neg(fm\ \mathtt{says}\ mail) \\ \mathcal{T}_{21} \end{array} & & \begin{array}{c} {}^{(v_2)}\ 1 : \neg(Usr\ \mathtt{says}\ mail) \\ \mathcal{T}_{22} \end{array} &
\end{tikzcd}$$

We recall that the prefixed statement (a) (omitted here and shown in the text) is the negation of the goal formula. We obtain (b), (c), and (d) from (a) by a number of α-rules. The prefixed statement (e) is introduced from the assumptions by rule *Glob*. From (b, e) we derive (f) by the $[L_{gr}]$-rule. (g) is also introduced by rule *Glob* and the direct reduction of `controls` in (g) yields (h_1) and (h_2). One branch closes immediately. In the other branch we apply rule $[L_{gr}]$ to (h_2, b) and obtain (i). Next we apply rule β to (d) and split the tableau with (l_1) and (l_2).
As for $\mathcal{T}_1$ we first apply rule *Glob* to introduce (m) and then reduce `controls` with a β-rule. Again, one branch closes immediately and, in the other branch, we apply rule $\langle Usr \rangle$ to (n_1) yielding (o). Since the prefix $1.Usr.2$ is present we can apply rule $[Usr]$ to (i) and add (p). Then we apply rule α to (p) and we are done.

To close $\mathcal{T}_2$ we apply rule *Glob* to obtain (t) and then reduce `controls`. We apply $\langle \& \rangle$ to (u_1) splitting further the tableau. The resulting tableau $\mathcal{T}_{21}$ and $\mathcal{T}_{22}$ are substantially identical to $\mathcal{T}_1$, replacing *telnet* with *mail*.

Figure 8: Deduction for the request of Example 2

$$\begin{array}{ll}
{}^{(a)} & 1 : K_B\,\mathsf{says}\,(Sc\,\mathsf{says}\,r) \\
{}^{(b)} & 1 : K_S\,\mathsf{says}\,(K_B \Rightarrow B) \\
{}^{(c)} & 1 : B \mid A\,\mathtt{controls}\,r \\
{}^{(d)} & 1 : \neg r \\
{}^{(e)} & 1 : K_S \Rightarrow S \\
{}^{(f)} & 1 : S\,\mathsf{says}\,(K_B \Rightarrow B) \\
{}^{(g)} & 1 : S\,\mathtt{controls}\,(K_B \Rightarrow B)
\end{array}$$

$$\swarrow \quad \searrow$$

$$\begin{array}{ll}
{}^{(h_1)} & 1 : \neg(S\,\mathsf{says}\,(K_B \Rightarrow B)) \\
 & \times
\end{array}
\qquad\qquad
\begin{array}{ll}
{}^{(h_2)} & 1 : K_B \Rightarrow B \\
{}^{(i)} & 1 : B\,\mathsf{says}\,(Sc\,\mathsf{says}\,r)
\end{array}$$

$$\swarrow \quad \searrow$$

$$\begin{array}{ll}
{}^{(l_1)} & 1 : \neg(B \mid A\,\mathsf{says}\,r) \\
{}^{(m)} & 1 : \neg(B\,\mathsf{says}\,(A\,\mathsf{says}\,r)) \\
{}^{(n)} & 1.B.2 : \neg(A\,\mathsf{says}\,r) \\
{}^{(o)} & 1.B.2 : Sc\,\mathsf{says}\,r \\
{}^{(p)} & 1.B.2 : Sc \Rightarrow A \\
{}^{(q)} & 1.B.2 : A\,\mathsf{says}\,r \\
 & \times
\end{array}
\qquad\qquad
\begin{array}{ll}
{}^{(l_2)} & 1 : r \\
 & \times
\end{array}$$

We omit again the negated theorem in the root. The prefixed statements (a), (b), (c), and (d) are obtained by α-rules from the negated theorem. (e) is obtained by rule *Glob* and (f) by rule $[L_{gr}]$ from (b, e). The prefixed statement (g) is added by rule *Glob* too. Then we reduce `controls` in (g) yielding two branches.

One branch closes immediately. In the other branch, we derive (i) by rule $[L_{gr}]$ from (a, h_2) and then split again the branch by reducing (c). Next we apply rule $\langle | \rangle$ to (l_1) yielding (m). The prefixed statement (n) is introduced by rule $\langle B \rangle$ from (m). Since $1.B.2$ is now present we can apply rule $[B]$ to (i) and add (o). Then we introduce (p) by rule *Glob* and apply $[L_{gr}]$ to (o, p).

Figure 9: Tableau Proof of Delegation without Certificates

2. prove a safe extension lemma (any tableau rule applied to a satisfiable formula preserves satisfiability with this mapping);

3. prove a safe closure lemma (no satisfiable branch is ignored).

The first and third steps are very similar to dynamic and multi modal logic [9] taking into account that branches are pairs $\langle \mathcal{B}, G_{\mathcal{B}} \rangle$ in our framework.

For the second step, which is also the longest, we can capitalize on the proof for dynamic logic for the operators $|$ and & (see again [9]). The only substantially new cases are given by the rules for the operator $\Rightarrow$. The argument requires to apply both the local properties and the universal properties of $\Rightarrow$. We use local properties for $\langle R_{gr} \rangle$ and $[L_{gr}]$ and global properties for $[U_{gr}]$. The rule $\langle U_{gr} \rangle$ relies heavily on both kind of properties and on the fact that the propositional letter x_i is a new letter.

For completeness, we also have an established path:

1. apply a systematic and fair procedure to the tableau;

2. if the tableau does not close, then choose an open branch to build a model for the initial formula $\neg s$.

The difficult part of the proof is the construction of the model and the mayor differences with the proof for modal and dynamic logics is that the presence of the $\Rightarrow$ relation imposes constraints on the accessibility relation which are not known a priori.

So we must work in two stages.

1. At first we construct a pre-model as in dynamic logic, by using prefixes σ as states, assigning the valuation to agents such that $(a)^{\mathcal{I}_0} = \{ \langle \sigma, \sigma.a.n \rangle \}$ and to proposition according the prefixed statements presents in the branch $(r)^{\mathcal{I}_0} = \{ \sigma \mid \sigma : r \in \mathcal{B} \}$.

2. Then we start an iterative *closure phase* where we construct the final valuation $(\cdot)^{\mathcal{I}}$ from the pre-valuation. For each constraint $a \Rightarrow P$ we add all prefixes from $(P)^{\mathcal{I}}$ to $(a)^{\mathcal{I}}$. We must repeat this process until we arrive at a fixed point.

Then we start the hard part of the proof which requires to show that if a prefixed statement occurs on the branch then that statement is satisfied in the state named by that prefix. This requires a double induction: on the number of steps of the closure phase and, on the size of the statements. Only the inner induction (on the size of the statement) is the usual argument for modal and dynamic logic.

9.1 Soundness

Definition 16 Let $\mathcal{B}$ be a branch and $\langle W, \mathcal{I} \rangle$ a model, a *mapping* is a function $\imath()$ from prefixes to states such that for all σ and $\sigma.a.n$ present in $\mathcal{B}$ it is $\langle \imath(\sigma), \imath(\sigma.a.n) \rangle \in (a)^{\mathcal{I}}$.

Definition 17 A tableau branch $\langle \mathcal{B}, G_{\mathcal{B}} \rangle$ is SAT in the model $\langle W, \mathcal{I} \rangle$ iff

1. $\langle W, \mathcal{I} \rangle$ satisfies $G_{\mathcal{B}}$;

2. there is a mapping $\imath(\cdot)$ such that for every $\sigma : s$ present in $\mathcal{B}$ one has $\imath(\sigma) \Vdash s$.

A tableau is SAT if one branch is such.

We can now give a proof of the safe extension lemma.

Theorem 18 (Safe Extension) If $\mathcal{T}$ is a SAT tableau for the model $\langle W, \mathcal{I}\rangle$, then the tableau $\mathcal{T}'$ obtained by an application of a tableau rule is also SAT for the same model.

Proof Let $\langle \mathcal{B}, G_{\mathcal{B}}\rangle$ be the SAT branch of the tableau and $\imath()$ the corresponding mapping. If the rule in not applied on $\mathcal{B}$ then clearly the new tableau is still SAT. If the rule is applied on $\mathcal{B}$ then we must show that the new branch $\langle \mathcal{B}', G'_{\mathcal{B}}\rangle$ is still SAT, possibly by changing the mapping. For the propositional connectives and the modal operator $\texttt{says}$ the proof is identical to that of logic K in [28, 15]. For quoting and conjunction we use the techniques used for dynamic logic in [9].

The only new cases are for the reduction of the operator $\Rightarrow$.

At first consider rule $\langle R_{gr}\rangle$ and suppose that $\sigma : P \Rightarrow a$ and $\sigma : \neg(a \,\texttt{says}\, s)$ are present in $\mathcal{B}$. Then by hypothesis $\imath(\sigma) \Vdash P \Rightarrow a$ and $\imath(\sigma) \not\Vdash a \,\texttt{says}\, s$. therefore for some w one has $\langle \imath(\sigma), w\rangle \in (a)^{\mathcal{I}}$ and $w \not\Vdash s$. We also have that $(a)^{\mathcal{I}} \subseteq (P)^{\mathcal{I}}$ so that $\langle \imath(\sigma), w\rangle \in (P)^{\mathcal{I}}$ and thus $\imath(\sigma) \not\Vdash P \,\texttt{says}\, s$ by definition of semantic entailment. Dually for $[L_{gr}]$ we use the fact that $\imath(\sigma) \Vdash a \Rightarrow P$ implies that $(P)^{\mathcal{I}} \subseteq (a)^{\mathcal{I}}$. If $\sigma : a \,\texttt{says}\, s$ is present then for all w such that $\langle \imath(\sigma), w\rangle \in (a)^{\mathcal{I}}$ one has $w \Vdash s$. Hence we also have $\imath(\sigma) \Vdash P \,\texttt{says}\, s$.

For rule $[U_{gr}]$ suppose that $\sigma : P \Rightarrow Q$ occurs in $\mathcal{B}$. Then $\imath(\sigma) \Vdash P \Rightarrow Q$ and therefore $(P \Rightarrow Q)^{\mathcal{I}} = W$ for the global property of $\Rightarrow$ because it contains at least $\imath(\sigma)$ and therefore it is not empty. Therefore adding it to $G_{\mathcal{B}}$ as done by the $[U_{gr}]$ does not change the satisfiability of the branch w.r.t. the first condition on $G_{\mathcal{B}}$ imposed by Def. 17.

If $\sigma : \neg(P \Rightarrow Q)$ is present in the branch then by hypothesis $\imath(\sigma) \Vdash \neg(P \Rightarrow Q)$. Hence there is a $\langle w, w^*\rangle \in (Q)^{\mathcal{I}}$ with $\langle w, w^*\rangle \notin (P)^{\mathcal{I}}$. Set $\imath(n) = w$ for the new prefix n and $(x_i)^{\mathcal{I}} = W - \{ w^* \}$ for the new propositional variable x_i. Clearly $\imath(n) \Vdash P \,\texttt{says}\, x_i$ but $\imath(n) \not\Vdash (Q \,\texttt{says}\, x_i)$.

The soundness theorem (Thm. 13) follows with a standard argument (see Fitting [15] or Goré [21]).

9.2 Completeness

The completeness proof is given for weakly left restricted statements and its extension to right and request restricted statements is sketched.

At first we apply a systematic and fair proof search procedure (see Goré [21] for a formal definition) to our initial tableau. At the end of this process (possibly ad infinitum) we either close the tableau or get an open branch. Then we can show that an open branch can be used to build a model for the initial formula.

Theorem 19 (Model Existence) If $\langle \mathcal{B}, G_{\mathcal{B}}\rangle$ is an open branch with weakly left restricted statements only, then it is SAT in a model $\langle W, \mathcal{I}\rangle$.

Proof Construct a pre-model $\langle W, \mathcal{I}_0\rangle$ as follows:

$$\begin{array}{rcl} W & \doteq & \{ \sigma \mid \sigma \text{ is present in } \mathcal{B} \} \\ (a)^{\mathcal{I}_0} & \doteq & \{ \langle \sigma, \sigma.a.n\rangle \mid \sigma \text{ and } \sigma.a.n \text{ are present in } \mathcal{B} \} \\ (r)^{\mathcal{I}_0} & \doteq & \{ \sigma \mid \sigma : r \in \mathcal{B} \} \end{array}$$

Incorporate the constraints due to $\Rightarrow$ and build $(\cdot)^{\mathcal{I}}$ from $(\cdot)^{\mathcal{I}_0}$ as follows:

- set $(\cdot)^{\mathcal{I}_{k+1}}{:=}(\cdot)^{\mathcal{I}_k}$;
- for every formula $\sigma : a \Rightarrow P$ occurring in $\mathcal{B}$
  - compute $(P)^{\mathcal{I}_k}$;
  - if $\langle \sigma, \sigma^* \rangle \in (P)^{\mathcal{I}_k}$ then add $\langle \sigma, \sigma^* \rangle$ to $(a)^{\mathcal{I}_{k+1}}$;
- repeat until a fix-point is reached[9].

We denote th final result as $(\cdot)^{\mathcal{I}}$.

After this *closure phase* we must prove that $\mathcal{B}$ is SAT in the model $\langle W, \mathcal{I} \rangle$ (according Def. 17) with the mapping $\imath(\sigma) = \sigma$: if $\sigma : s$ is present in $\mathcal{B}$ then $\imath(\sigma) \Vdash s$ by induction on the construction of s.

Indeed, once we have this result, it is easy to prove that for all $s \in G_{\mathcal{B}}$ and for all $\sigma \in W$ one has $\sigma \Vdash s$. Indeed by use saturation w.r.t. the *Glob* rule we have that for all σ present in $\mathcal{B}$ and $s \in G_{\mathcal{B}}$ we have that $\sigma : s \in \mathcal{B}$.

The proof is similar to those used for converse propositional dynamic logic and modal logics and works by induction on the size of s (see [9, 21]).

The new connective $\Rightarrow$ can be dealt without much difficulties. By construction we satisfy the local properties of $\Rightarrow$. For the global condition we use mutual saturation between *Glob* and $[U_{gr}]$-rules. If $\sigma : a \Rightarrow P$ occurs in $\mathcal{B}$ then $[U_{gr}]$-implies that $a \Rightarrow P \in G_{\mathcal{B}}$. By *Glob* we have that for all $s \in G_{\mathcal{B}}$ and all prefixes σ in $\mathcal{B}$ one has $\sigma : s$ present in the branch. Hence every σ satisfies the local condition i.e. W satisfies the global condition and therefore $(P)^{\mathcal{I}} \subseteq (a)^{\mathcal{I}}$.

The difficult case is $\sigma : a \,\mathtt{says}\, s$ since we must prove that for all prefixes $\langle \sigma, \sigma^* \rangle \in (a)^{\mathcal{I}}$ the prefixed statement $\sigma^* : s$ is present in $\mathcal{B}$ so that we can apply the induction hypothesis to $\sigma^* : s$, get $\sigma^* \Vdash s$ and then the claim.

The difference with "traditional" proofs [15, 28] is that some prefixes σ^* are introduced in $(a)^{\mathcal{I}}$ during the closure phase. Hence we can have

- $\sigma^* = \sigma.a.n$ for some n or
- $\langle \sigma, \sigma^* \rangle \in (P)^{\mathcal{I}_k}$ before the closure phase and $(P)^{\mathcal{I}_k} \subseteq (a)^{\mathcal{I}_{k+1}} \subseteq (a)^{\mathcal{I}}$ after a step of the closure phase

The proof goes by induction on the number of closure steps and needs a series of intermediate lemmata and propositions.

The base case requires the following property:

Lemma 20 If $\langle W, \mathcal{I}_0 \rangle$ is the pre-model of $\mathcal{B}$ and $\sigma : P \,\mathtt{says}\, s$ is present in $\mathcal{B}$ then for all σ^* such that $\langle \sigma, \sigma^* \rangle \in (P)^{\mathcal{I}_0}$ then the prefixed formula $\sigma^* : s$ is also present in $\mathcal{B}$.

The proof here also require induction on the size of P. Thus is P is an atomic principal a we simply apply saturation w.r.t. rule $[a]$

[9] A fix point is always reached since the operation we perform is monotone on a bounded lattice.

For complex principals we must first show that if $\langle\sigma,\sigma^*\rangle \in (P)^{\mathcal{I}}$ then we can "travel" from σ to σ^* using only atomic principals as intermediate steps. To this extent we introduce the definition of the initial segment of P (its head):

$$\begin{array}{rcl} head(a) & = & \{\, a \,\} \\ head(P\&Q) & = & head(P) \cup head(Q) \\ head(P \mid Q) & = & head(P) \end{array}$$

Then we use the following proposition to prove the main lemma:

Proposition 21 If $\langle\sigma,\sigma^*\rangle \in (P)^{\mathcal{I}_0}$ then σ^* has the form $\sigma.a.n.a_1.n_1 \ldots .a_k.n_k$ and $a \in head(P)$.

Proposition 22 If $\sigma : P\,\mathtt{says}\,s$ is present in $\mathcal{B}$, $\langle\sigma,\sigma^*\rangle \in (P)^{\mathcal{I}_0}$ and the prefix σ^* has the form $\sigma.a.n.a_1.n_1 \ldots .a_k.n_k$ then there are principal $Q_1, \ldots Q_h$ such that $\sigma.a.n : Q_1\,\mathtt{says}\,\ldots Q_h\,\mathtt{says}\,s$ is present in $\mathcal{B}$ and $\langle\sigma.a.n,\sigma^*\rangle \in (Q_1 \mid \cdot \mid Q_h)^{\mathcal{I}_0}$.

Notice that the presence of a number of Q_i is simply due to the various possibilities in which quoting was associated in P. If we had a sort of normal form, for instance always associating quoting to the right, then we would only have one Q.

From these results we can prove the main result by induction. For the base case we use Lemma 20. For the inductive step we must show that these properties are preserved for the partial $(\cdot)^{\mathcal{I}}$ we construct after each step of the iteration. Thus the inductive hypothesis is simply Lemma 20 where the partial $(\cdot)^{\mathcal{I}}$ replace $(\cdot)^{\mathcal{I}_0}$.

The key point is to observe that whenever $\sigma : a \Rightarrow P$ is present then by saturation $\sigma : a\,\mathtt{says}\,s$ implies $\sigma : P\,\mathtt{says}\,s$. So we apply the inductive hypothesis to get $\sigma^* : s$ from $\sigma : P\,\mathtt{says}\,s$. Therefore, when the prefix σ^* was added in $(a)^{\mathcal{I}}$ in the closure phase the prefixed statement $\sigma^* : s$ was present.

Once this theorem has been proved, the completeness theorem (Thm. 14) follows with a standard argument. See Fitting [15] for the proof.

For the right and request restricted fragment of the language the key point is that we only have literals or statements of the form $P \Rightarrow a$ under the scope of $\mathtt{says}$. The operator $\Rightarrow$ does not create problems given its global nature and the only difficult part is due to literals l (r or $\neg r$).

The previous proof does not work since $\sigma : P\,\mathtt{says}\,s$ for non atomic P cannot propagate over $\sigma : P \Rightarrow a$. However we can prove the dual of the induction step of the previous proof: if $\sigma : \neg(a\,\mathtt{says}\,l)$ and $P \Rightarrow a$ is also present then there is a $\langle\sigma,\sigma^*\rangle \in (P)^{\mathcal{I}}$ such that $\sigma^* : \neg l$.

This means that all $\sigma : P\,\mathtt{says}\,l'$ are consistent with each $\sigma : \neg(a\,\mathtt{says}\,l)$. At this stage we need to use the $D(a)$-rule, to prove that those P statements are consistent also with each $\sigma : a\,\mathtt{says}\,l''$. By $D(A)$ we obtain $\sigma : \neg(a\,\mathtt{says}\,\neg l'')$ and then apply the dual property.

Since all l, l', l'' are literals this is enough: all $a\,\mathtt{says}\,l$ are consistent among themselves, and each of them with all $P\,\mathtt{says}\,l'$. This means that when we add a $\langle\sigma,\sigma^*\rangle$ from $(a)^{\mathcal{I}}$ to $(P)^{\mathcal{I}}$ in the closure phase we can always extend the valuation of the unspecified l' or l'' in σ^* so that the result is still a model. Again, this only works for *request restricted* statements.

This is sufficient for the proof of the the completeness theorem for weakly right and request restricted fragments (Thm. 15).

10 Incorporating Roles

New security paradigms have stressed the importance of an explicit treatment of roles in the design and verification of access control systems and policies [13, 17, 35]. For instance, roles simplify the design of the system and make possible to use engineering principles such as least privilege, separation of duties, etc. [13, 35].

Roles are typically assumed by principals to accomplish some particular duties. A simple example is the command `su` (super-user) in UNIX systems. By executing `pb@ely% su webmaster` the user *pb* assume the role of *webmaster* to accomplish some tasks (such as changing the department home page) which require a particular set of privileges. As mentioned already, the distinction between roles and groups can be tricky and we refer to the literature for further discussion [13, 35]. An intuitive understanding of the concept of role and of the difference between the principals P and $P \,\texttt{as}\, R$ is sufficient here.

A role will be typically used by a principal in the format $P \,\texttt{as}\, R$, where principal P claims to be using role R, e.g. $pb \,\texttt{as}\, webmaster$. This can still be encoded by modelling role assumption "$\texttt{as}$" with quoting "$|$" as we suggested in §5.1.

Yet, in the formalism and deduction method presented so far, roles are just "principals like any other". After the encoding, the design of properties and global assumptions characterizing them is up to the user of our method.

It is possible to deal with them explicitly, along the lines proposed by Abadi et al. [1] for an axiomatic approach or by Massacci [29] for a tableau-based deduction.

To this extent we must introduce distinguished symbols for atomic roles $R_1 \ldots R_n$ and define their properties by adding suitable axiom schemata or tableau rules.

The first step is establishing the relationships between normal agents and roles. A possible consideration is the following: if *Bob* does not restrict his privileges when making a request to *Alice*, then *Alice* should conclude that *Bob* wants to use all possible roles entrusted to him. This assumption avoids a "denial of service" from *Alice* and can be easily characterized by the axiom schema below:

$$P \Rightarrow P \,\texttt{as}\, R$$

We can also require *idempotency* and enforce the following axioms:

$$R \,\texttt{as}\, R \Rightarrow R \qquad\qquad R \Rightarrow R \,\texttt{as}\, R$$

There seems to be no special reason why $fm205 \,\texttt{as}\, user$ should be different from $fm205 \,\texttt{as}\, user \,\texttt{as}\, user$.

Other properties, adopted in [1], may be less intuitive and appealing. For instance consider commutativity of roles, represented by the axioms below:

$$R_1 \,\texttt{as}\, R_2 \Rightarrow R_2 \,\texttt{as}\, R_1 \qquad\qquad R_2 \,\texttt{as}\, R_1 \Rightarrow R_1 \,\texttt{as}\, R_2$$

Commutativity may lead to undesirable results when combined with the need of avoiding denial of service. Indeed we can expect to have $P \,\texttt{as}\, R_1 \,\texttt{as}\, R_2$ to have the same privileges of $P \,\texttt{as}\, R_2$ and hence by commutativity of $P \,\texttt{as}\, R_2$. So we may lose the possibility of "downgrading" our privileges which may be advisable when performing sensitive operation (for instance we may want to act as `user` rather than `root` while deleting files). Moreover, if R_1 and R_2 are incompatible roles we may also derive an inconsistency.

The direct embedding preserves the flexibility of the calculus but may generate many irrelevant properties and therefore may hinder the effectiveness of the deduction method.

Another possibility is to design deduction rules for roles, which take into account their specific properties. The trade off is that we may hardwire desirable and less desirable security properties into the deduction methods.

A combination of both methods may be worth investigating.

11 Conclusions

The combination of logical languages and decision methods based on logic may be a step towards the development of practical reasoning tools for formal verification of multi-agent distributed systems.

A claim that we do *not* make is that logic and semantic tableaux should be used for run-time decisions on access control. Although possible, this may lead to unacceptable slow-downs. On the contrary, logic and tableaux (or similar logic-based methods) should be used for off-line *verification* and prototyping, for checking that access protocols respect security policies. This work is a step in this direction.

The major contribution of this paper is the development of tableau methods for the calculus for access control by Abadi, Lampson et al. [1, 26, 40]. We also clarified some model theoretic features of the calculus. The completeness results presented here extend those in [1] and provide the basis for a full fledged automatization. The method extends the deduction capabilities of [1] as we can prove important properties which must be added as non-logical axiom [1, 26].

This tableau method requires novel techniques such as passing from a tree-like tableau to a forest of prefixes, a run time update of global assumptions and a modification of the notion of branch. Future research is in the direction of providing a fully automated verifier, possibly using the results of Beckert and Goré [3], and extending the formalism to first order logic for expressing quantified properties over objects.

An important side-effect of these deductive techniques (discussed in [30]) is an EXPTIME-tableau for multi-modal logics with the universal modality [19, 20].

Acknowledgments

Most of this work was done while the author was at the Computer Laboratory at the University of Cambridge (UK). I'm indebted to L. Paulson and the Computer Laboratory for their hospitality in Cambridge and to my advisor L. Carlucci Aiello for her encouragement and support.

I would like to thank M. Abadi for discussing the features of the DEC-SRC calculus, clarifying some of its tricky aspects and commenting previous versions of this work, and the Computer Security group (Cambridge) and the Applied Logic group (IRIT) for many useful discussions. Precise comments from the anonymous referees have also greatly improved the quality of this paper.

A preliminary, much shorter version of this paper appearead in [30].

This work has been financially supported by grants from ASI, CNR and MURST.

References

[1] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, 1993.

[2] R. Anderson. A security policy model for clinical information systems. In *Proceedings of the 15th IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 1996.

[3] B. Beckert and R. Goré. Free variable tableaux for propositional modal logics. In *Proceedings of the International Conference on Analytic Tableaux and Related Methods (TABLEAUX-97)*, volume 1227 of *Lecture Notes in Artificial Intelligence*, pages 91–108. Springer-Verlag, 1997.

[4] D. Bell and L. La Padula. Secure computer systems: unified exposition and MULTICS. Report ESD-TR-75-306, The MITRE Corporation, March 1976.

[5] E. Bertino, S. Jajodia, and P. Samarati. Supporting multiple access control policies in database systems. In *Proceedings of the 15th IEEE Symposium on Security and Privacy*, pages 94–109. IEEE Computer Society Press, 1996.

[6] M. Burrows, M. Abadi, and R. Needham. A logic for authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990. Also available in *Proceedings of the Royal Society of London*, volume 426, pages 233–271, 1989 and Research Report SRC-39, DEC - System Research Center, 1989.

[7] D. Clark and D. Wilson. A comparison of commercial and military computer security policies. In *Proceedings of the 6th IEEE Symposium on Security and Privacy*,pages 184–194. IEEE Computer Society Press, 1987.

[8] F. Cuppens and R. Demolombe. A deontic logic for reasoning about confidentiality. In *3rd International Workshop on Deontic Logic in Computer Science*, Sesimbra, Portugal, 1996.

[9] G. De Giacomo and F. Massacci. Tableaux and algorithms for propositional dynamic logic with converse. In *Proceedings of the 13th International Conference on Automated Deduction (CADE-96)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 613–628. Springer-Verlag, 1996. Extended version to appear in *Information and Computation.*

[10] Department of Defense. Trusted computer systems evaluation criteria. Technical Report, DoD 5200,28-STD, United States of America, 1990.

[11] Directorate General XIII. Information technology security evaluation criteria (ITSEC). Technical Report, European Economic Community, 1990.

[12] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge.* The MIT Press, 1995.

[13] D. Ferraiolo, J. Cugini, and R. Kuhn. Role-based access control (RBAC): Features and motivations. In *Proceedings of the Annual Computer Security Applications Conference*. IEEE Computer Society Press, 1995.

[14] D. Ferraiolo, D. Gilbert, and N. Lynch. An examination of federal and commercial access control policy needs. In *Proceedings of the 16th NIST-NCSC National (U.S.) Computer Security Conference*, pages 107–116, 1993.

[15] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics.* Reidel, Dordrecht, 1983.

[16] M. Georgeff and A. Rao. The semantics of intention maintenance for rational agents. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 704–710. Morgan Kaufmann, Los Altos, 1995.

[17] L. Giuri and P. Iglio. A formal model for role based access control with constraints. In *Proceedings of the 10th IEEE Computer Security Foundations Workshop*, pages 136–145. IEEE Computer Society Press, 1996.

[18] J. Glasgow, G. MacEwen, and P. Panangaden. A logic for reasoning about security. *ACM Transactions on Computer Systems*, 10(3):226–264, 1992.

[19] V. Goranko. Modal definability in enriched languages. *Notre Dame journal of Formal Logic*, 31(1), 1990.

[20] V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.

[21] R. Goré. Tableaux method for modal and temporal logics. Technical Report TR-ARP-15-5, Australian National University, 1995. To appear as chapter on the "Handbook of Tableau Methods".

[22] M. Harrison, W. Ruzzo, and J. Ullman. Protection in operating systems. *Communications of the ACM*, 19(8):461–471, 1976.

[23] G. Hughes and M. Cresswell. *An Introduction to Modal Logic.* Methuen, 1968.

[24] S. Kanger. Law and logic. *Theoria*, 38(3):105–132, 1972.

[25] C. Krogh. Obligations in multiagent systems. In *Proceedings of the 5th Scandinavian Conference on Artificial Intelligence (SCAI-95)*, pages 29–31. ISO Press, Amsterdam, 1995.

[26] 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, 1992.

[27] B. Lampson. Protection. *ACM Opearting Systems Reviews*, 8(1):18–24, 1974.

[28] F. Massacci. Strongly analytic tableaux for normal modal logics. In *Proceedings of the 12th International Conference on Automated Deduction (CADE-94)*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 723–737. Springer-Verlag, 1994.

[29] F. Massacci. Reasoning about security: a logic and a decision methods for role-based access control. In *Proceeding of the International Joint Conference on Qualitative and Quantitative Practical Reasoning (ECSQARU/FAPR-97)*, volume 1244 of *Lecture Notes in Artificial Intelligence*, pages 421–435. Springer-Verlag, 1997.

[30] F. Massacci. Tableaux methods for access control in distributed systems. In D. Galmiche, editor, *Proceedings of the International Conference on Analytic Tableaux and Related Methods (TABLEAUX-97)*, volume 1227 of *Lecture Notes in Artificial Intelligence*, pages 246–260. Springer-Verlag, 1997.

[31] C. McCollum, J. Messing, and L. Notargiacomo. Beyond the pale of MAC and DAC - defining new forms of access control. In *Proceedings of the 9th IEEE Symposium on Security and Privacy*, pages 190–200, 1990.

[32] J. Moffet and M Sloman. Policy hierarchies for distributed systems management. *IEEE Journal on Selected Areas in Communications*, 11(9), 1993.

[33] R. Sandhu. The typed access matrix model. In *Proceedings of the 11th IEEE Symposium on Security and Privacy*, pages 122–136. IEEE Computer Society Press, 1992.

[34] R. Sandhu and P. Samarati. Access control: Principles and practice. *IEEE Communications Magazine*, pages 40–48, September 1994.

[35] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access controls models. *IEEE Computer*, 29(2), February 1996.

[36] M. Schmidt-Schauss. Subsumption in KL-ONE is undecidable. In *Proceedings of the 1st International Conference on the Principles of Knowledge Representation and Reasoning (KR-89)*, pages 421–431, 1989.

[37] P. Syverson and P. van Oorschot. On unifying some cryptographic protocols logics. In *Proceedings of the 13th IEEE Symposium on Security and Privacy.* IEEE Computer Society Press, 1994.

[38] J. van Benthem. Correspondence theory. In *Handbook of Philosophical Logic*, volume II, pages 167–247. Reidel, 1986.

[39] R. van der Meyden. The dynamic logic of permission. *Journal of Logic and Computation*, 6(3):465–479, 1996.

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