Guido Boella
Dov M. Gabbay
Valerio Genovese
Leendert van der Torre

Fibred Security Language

Abstract. We study access control policies based on the **says** operator by introducing a logical framework called Fibred Security Language (FSL) which is able to deal with features like joint responsibility between sets of principals and to identify them by means of first-order formulas. FSL is based on a multimodal logic methodology. We first discuss the main contributions from the expressiveness point of view, we give semantics for the language (both for classical and intuitionistic fragment), we then prove that in order to express well-known properties like 'speaks-for' or 'hand-off', defined in terms of **says**, we do not need second-order logic (unlike previous approaches) but a decidable fragment of first-order logic suffices. We propose a model-driven study of the **says** axiomatization by constraining the Kripke models in order to respect desirable security properties, we study how existing access control logics can be translated into FSL and we give completeness for the logic.

Keywords: Logic, Fibring, Access Control, Language-based Security, Trust Management.

1. Introduction

Access control is a pervasive issue in security: it consists in determining whether the principal (a key, channel, machine, user, program) that issues a request to access a resource should be trusted on its request, i.e., if it is authorized. Authorization can be based in the simplest case on access control lists (ACL) associated with resources or with capabilities held by principals, but it may be complicated, for instance, by membership of groups, roles and delegation. Thus, logics are often introduced in access control to express policies and to enable reasoning about principals and their requests, and other general statements.

In many cases first-order/propositional logic suffices, but it does not in the case of distributed policies and delegation, e.g., "administrator says that Alice can be trusted when she says to delete $file_1$": Alice speaks for the administrator concerning the deletion of $file_1$, thus she should be trusted as much as the administrator.

Special Issue: New Ideas in Applied Logic
Edited by **Dov M. Gabbay and Jacek Malinowski**

Studia Logica (2009) 92: 395–436
DOI: 10.1007/s11225-009-9201-6

Logical approaches to comprehend, analyze, create and verify the policies and control mechanisms used to protect resources have been extensively studied in these years [1,3,7,8,17,22]. The interest of the community is also underlined by a number of research projects that have applied these logics for designing or motivating various languages and systems [4,6,10,25]. On the other hand, as reported in [14], there have been only few, limited efforts to study the logics themselves (e.g. [1,3,15]). In particular, the problems of the axiomatization of the well-known **says** operator has been studied only recently in [2,14]. Generally, the full expressiveness of the proposed logics is reached by employing second-order formalisms, this is due to the need of axiomatize important concepts like the speaks-for or hand-off [3,20].

In this paper we introduce a novel first-order multimodal logic for access control in distributed systems called Fibred Security Language (FSL). The contribution of FSL addresses the following research questions:

1. How to define a general language capable to embody and extend existing access control logics?
2. How to formalize a logic which provides axiomatizations of security properties that avoids undesired side effects and which at the same time ensure tractability?

Our methodology is centered on a first-order language based on the fibring approach of Gabbay [13] and goes in the direction of having a method to integrate different logics into a single system.

We use a multimodal approach to express axioms which in the literature have been expressed in (computationally intractable) second-order logics within a first-order logic. The reduction from second-order to first order can be of practical value in the objective of building theorem provers for logics for access control.

In Section 2 we present the use of the says modality and we comment the axioms of the says defined in [2]. In Section 3 we present FSL by showing its expressiveness. Section 4 is devoted to the introduction and formalization of the basic system FSL. In Section 5, we show how the second-order axioms introduced in Section 2 can be translated into first-order constraints on the multimodal-kripke semantics of FSL, we also discuss models that have desirable security properties. In Section 6 we study a particular fragment of FSL in which access control policies are expressed by Horn clauses. In Section 7 we discuss how FSL can be used as a framework to embody and extend existing access control logics, as an example we translate the logic SecPAL [7] into FSL. Section 8 ends the paper.

2. Modality Axioms

In this section we first summarize how the **says** operator is used in access control logics, and then we discuss which properties are desirable for this operator and which are not by presenting several key axioms that have been mainly studied in [2].

The access control logic we propose aims at distributed scenarios. Thus, to express delegation among principals, it is centered, like the access control logic of [21,22], on formulas such as "A **says** ψ" where A represents a principal, ψ represents a statement (a request, a delegation of authority, or some other utterance), and **says** is a modality. Note that it is possible to derive that A **says** ψ even when A does not directly utter ψ. For example, when the principal A is a user and one of its programs includes ψ in a message, then we may have A **says** ψ, if the program has been delegated by A. In this case, A **says** ψ means that A has caused ψ to be said, that ψ has been said on A's behalf, or that A supports ψ.

We assume that such assertions are used by a reference monitor in charge of making access control decisions for resources, like o. The reference monitor may have the policy that a particular principal A is authorized to perform $Do(o)$. This policy may be represented by the formula: $(A \textbf{ says } Do(o)) \rightarrow Do(o)$, which expresses that A **controls** $Do(o)$. Similarly, a request for the operation on o from a principal B may be represented by the formula: B **says** $Do(o)$. The goal of the reference monitor is to prove (or check) that these two formulas imply $Do(o)$, and grant access if it succeeds. While proving $Do(o)$ the reference monitor does not need that the principal B controls $Do(o)$. Rather it may exploit relations between A and B and some other facts. For example, it may knows that B has been delegated by A and thus, that B speaks for A as concerns $Do(o)$, in formulas:

$$(B \textbf{ says } Do(o)) \rightarrow (A \textbf{ says } Do(o))$$

This simple example does not show the subtleties arising from the formalization of the **says** operator, since expressing simple properties like controlling a resource or speaking for another principal may imply less desirable properties, leading to security risks, or even to inconsistent or degenerate logic systems [2].

Despite the pervasive employment of the **says** operator in several logics for access control, only recently in [2] different axiomatizations of the says have been studied and proposed. The objective is to explore the formal consequences and the security interpretations of several possible axiomati-

zations, and thus to help in identifying logics that are sufficiently strong but not inconsistent, degenerate, or otherwise unreasonable.

Following we present the relevant axioms underlined in [2] notice that some of them are second-order due to a quantification over formulas. In Section 5 we will show how, by exploiting the multimodal approach of FSL, we can translate them as first-order constraints on the kripke semantics. We write $\Box_A X$ as an abbreviation for A **says** X, where X ranges over formulas.

1. *B speaks for A* (notation $B \Rightarrow A$):

$$\forall X[\Box_B X \rightarrow \Box_A X].$$

This is the fundamental relation among principals in access control logics. If $B \Rightarrow A$, from the fact that principal B says something means that the reference monitor [20] can believe that principal A says the same thing. This relation serves to form chains of responsibility: a program may speak for a user, much like a key may speak for its owner, much like a channel may speak for its remote end-point.

2. *Restricted speaks for*

$$\forall X[\alpha(X) \wedge \Box_B X \rightarrow \Box_A X]$$

where $\alpha(X)$ be any formula and X a new variable.
Restriction of "speaks for" is similar to the one [20] introduces. In particular, if $\alpha(X) = \varphi \rightarrow X$, then the above formula would refer to B speaks for A on all consequences of φ [1].

3. A **controls** ψ, for some sepcified formula ψ

$$\Box_A \psi \rightarrow \psi$$

Intuitively it represents the direct control of A over a resource ψ. In this view it is desirable not to have a principal which controls *all* formulas, that is why we do not employ the universal quantifier.

4. *Hand-off axiom*

$$\Box_A \forall X[\Box_B X \rightarrow \Box_A X] \rightarrow \forall X[\Box_B X \rightarrow \Box_A X]$$

or more briefly:

$$\Box_A(B \Rightarrow A) \rightarrow (B \Rightarrow A)$$

Hand-off states that whenever A says that B speaks for A, then B does indeed speaks for A. This axiom allows every principal to decide which principals speak on its behalf, since it controls the delegation to other principals.
Sometimes this axiom follows from logic rules as in [2], sometimes it is assumed as an axiom. Note that the general axiom is too powerful, and thus risky for security: for example when A represents a group: if A **controls** $(B \Rightarrow A)$ then any member of A can add members to A. Thus, for instance, [3] does not adopt this axiom.

5. *Unit*

$$\forall X[X \rightarrow \Box_A X]$$

Unit is stronger than the necessitation rule. In classical logic (but not intuitionistic), adopting Unit implies that each principal either always says the truth or it says false: $(A \rightarrow B) \vee (B \rightarrow A)$. In the first case A speaks for any other principal, in the latter any other speaks for A. The policies described by this kind of systems are too manicheist.

6. *Escalation*

$$\forall X[\Box_A X \rightarrow X \vee \Box_A Y]$$

Escalation is not considered as a desirable property. It amounts to "if A **says** ψ then ψ *or*A **says** *anything*": from A **says** ψ, if ψ is not the case, may follow a statement "much falser" than ψ. As an example of its riskiness, consider that from $(A$ **controls** $\psi) \wedge (B$ **controls** $\psi)$ it allows to infer that if A **says** B **says** ψ then ψ follows. If the logic is not able to avoid escalation, the only cumbersome solution is to make A avoid saying that B says ψ unless he really wishes to say ψ. Thus we must be careful that it does not follow from other properties (like from Unit in classical logic).

According to [2] in classical logic, Unit implies Escalation. If we leave out Unit we can rely on intermediate systems between the basic modal logic and Escalation. For instance, one may require the standard axiom C4 from modal logic ($\Box_A \Box_A X \rightarrow \Box_A X$) without obtaining Escalation. However, these intermediate systems appear quite limited in their support of delegation and related concepts.

In trying to have an expressing logic without Escalation as a theorem, intuitionism seems to be the right semantics to employ. In fact, Abadi in [2] propose a logic (CDD) based on second-order intuitionistic semantics in order to have a sufficiently expressive logic without having dangerous

consequences like Escalation. In Section 4.2 we present predicate FSL which extends CDD expressiveness without using a second-order semantics.

3. The Fibred Security Language

From the expressivity point of view, FSL aims to extend previous logics for access control by introducing joint responsibility between principals and groups of principals as first-class citizen described by means of first-order formulas.

Although these properties are employed in general languages to describe policies [5], FSL is the first *logical* approach which embodies these features within a coherent semantical formalization of the well-known **says** operator.

In FSL, we enrich first-order logic with formulas of the kind

$$\{x\}\varphi(x) \textbf{ says } \psi \tag{I}$$

where $\{x\}\varphi(x)$ is a set-binding operator which represents the group composed by all the principals that satisfy $\varphi(x)$, **says** is a modal binary operator and ψ is a general formula.

Intuitively, we read Formula I as: "The group composed by all the principals that satisfy $\varphi(x)$ *supports* ψ".

In FSL is then possible to let principals jointly (as a group) support a statement, which is a desirable feature in several access control models as underlined in [5].

Previous approaches are limited in the representation of principals, in [21] principals are propositional atoms distributed in a lattice-based structure which can be combined with classical meet and join operators, in [7,17] a formula can be supported by at most one principal and it is not possible to make a group of principals jointly support a formula. In [22] groups of principals can be described by propositional atoms but their employment is limited to static and dynamic thresholds.

The proposed view on access control logics offers a general methodology to define policies and freedom in crafting logics. In fact we can let $\varphi(x)$ and ψ belong to two different languages $\mathbb{L}_p$ and $\mathbb{L}_e$ as language of principals and security expressions respectively which refers to two different systems (semantics).

For instance we can think of formulas in $\mathbb{L}_p$ be SQL queries and formulas in $\mathbb{L}_e$ be Delegation Logic [22] expressions.

In order to formally specify how to evaluate expressions like I, we formalize the **says** modality by using the fibring methodology [13] which, depending

on the chosen languages (and systems), provides a formal tool to combine logics in a common framework which is coherent and does not collapse.

In this paper, to show the full expressiveness of our approach, we decide to make $\mathbb{L}_p = \mathbb{L}_e = \mathbb{L}$, where $\mathbb{L}$ is a classical first order language, whereas the relying system S is intuitionistic modal logic[1]; this is predicate FSL formally presented in Section 4.2. This approach offers us to iterate the **says** modality and to have extremely complex formulas in which free variables are shared between different levels of nesting of the **says**. In fact, in Formula I, $\varphi(x)$ and ψ can share variables and φ may include occurrences of the **says** operator.

In the following we discuss the epressivity power of FSL by means of examples, for a complete formalization of the framework we refer to Section 4.

The formula $\{x\}\varphi(x)$ is used to select the set of principals making the assertion **says**. To select a single principal whose name is A we write:

$$\{x\}(x = A) \textbf{ says } s$$

The following formula means that the group of *users* asks to delete $file_1$:

$$\{x\}user(x) \textbf{ says } delete(file_1)$$

Since $\varphi(x)$ and ψ can share variables, we can put restrictions on the variables occurring in ψ. E.g., the set of all users who all own file(s) y asks to delete the file(s) y.

$$\alpha(y) = \{x\}(user(x) \wedge own(x, y)) \textbf{ says } delete(y)$$

However, the formula above is satisfactory only in the particular situation where we are talking about the set of all users who assert **says** at once as a group (committee).

We can as well express that each member of a set identified by a formula can assert **says** separately. E.g., each user deletes individually the files he owns:

$$\forall x(user(x) \wedge own(x, y)) \rightarrow \{z\}(z = x) \textbf{ says } delete(y)$$

Note that the latter formula usually implies the former but not vice versa[2].

[1] To see why intuitionistic logic is preferred over classical we refer to Section 2.

[2] In fact, it could be sensible to have situations in which, if all the members of a group say something then the whole group says it but not vice versa, formally

$$\forall t(\varphi(t) \rightarrow t \textbf{ says } \psi) \rightarrow \{x\}\varphi(x) \textbf{ says } \psi$$

For instance, a committee may approve a paper that not all of its members would have accepted.

Operations on principals

We can express the fact that two principals A and B together says **s**:

$$\{x\}(x = A \vee x = B) \textbf{ says s}$$

which corresponds to

$$\{A, B\} \textbf{ says s}$$

If we want to express that the intersection of two different kind of principals (T_1,T_2) **says** ψ:

$$\exists x(T_1(x) \wedge T_2(x)) \rightarrow \{y\}(y = x) \textbf{ says } \psi$$

For instance, T_1 could be *club_member* and T_2 *adult*.

In this view we can also have negation in selecting principals:

$$\{x\}(x \neq A) \textbf{ says s}$$

Variables over principals

The possibility to have variables which range over principals allows first of all attribute-based (as opposed to identity-based) authorization as in [7]. Attribute-based authorization enables collaboration between parties whose identities are initially unknown to each other. The authority to assert that a subject holds an attribute (such as being a student) may then be delegated to other parties, who in turn may be characterized by attributes rather than identity. In the example below, a shop gives a discount to students. The authority over the student attribute is delegated to holders of the university attribute, and authority over the university attribute is delegated to known principal, the Board of Education (BE).

Shop says x is entitled to discount if x is a student.

$$\textit{Shop} \textbf{ says } (\textit{student}(x) \rightarrow \{y\}(x = y) \textbf{ controls } \textit{discount})$$

Shop says x can say z is a student if x is a university

$$\textit{Shop} \textbf{ says } (\textit{university}(x) \rightarrow \{y\}(x = y) \textbf{ controls } \textit{student}(z))$$

Shop says that the Board of Education can say x is a university

$$\textit{Shop} \textbf{ says } (BE \textbf{ controls } \textit{university}(x))$$

We may have more complicated policies involving more that two principals, like in the following example [22].

$$\{y\}(y = A)\ \mathbf{says}\ ((\{y\}(y = C)\ \mathbf{says}\ fraudulent(x)) \wedge \{y\}(y = D)\ \mathbf{says}\ expert(C)) \rightarrow fraudulent(x))$$

Since φ in $\{x\}\varphi(x)$ **says** ψ can be any formula, it can contain even occurrences of the **says** operator. This allows to refer to principals who made previous assertions of the **says** operator. For example, we can express the following: the members of the board who said to write a file they own, ask to delete it.

In symbols

$$\{x\}[\{u\}member\text{-}board(u)\ \mathbf{says}\ ((member\text{-}board(x) \wedge file\text{-}owner(y, x)) \rightarrow write(y))]\ \mathbf{says}\ delete(y)$$

Like in [7] delegation can be restricted to principals respecting some requirements: Fileserver is a trusted principal who delegates file reading authorizations only to the owners of files:

$$\forall x\ own(x, y) \rightarrow (Fileserver\ \mathbf{says}\ (\{z\}(z = x)\ \mathbf{says}\ read(y) \rightarrow Fileserver\ \mathbf{says}\ read(y)))$$

Variables over principals allow width-bounded delegation. Suppose A wants to delegate authority over *is a friend* fact to Bob. She does not care about the length of the delegation chain, but she requires every delegator in the chain to satisfy some property, e.g. to possess an email address. Principals with the *is a delegator* attribute are authorized by A to assert *is a friend* facts, and to transitively re-delegate this attribute, but only amongst principals with a matching email address.

A says x can say y is a friend if x is a delegator

$$A\ \mathbf{says}\ ((delegator(x) \rightarrow (\{y\}(x = y)\ \mathbf{says}\ friend(z))) \rightarrow friend(z)$$

A says B is a delegator

$$A\ \mathbf{says}\ delegator(B)$$

A says x can say y is a delegator if x is a delegator, y possesses email.

$$A\ \mathbf{says}\ ((delegator(x) \wedge has\text{-}email(y)) \rightarrow (\{w\}(w = x)\ \mathbf{controls}\ delegator(y)))$$

As with depth-bounded delegation, this property cannot be enforced in SPKI/SDSI, DL or XrML.

Restrictions on says

Some authors restrict $\Rightarrow$ to a set of propositions [19] $P \Rightarrow_T Q$ means that the proposition s in P **says** $s \rightarrow Q$ **says** s must belong to T.

We can put some restrictions on the variables:

$$(\{x\}(user(x) \wedge owns(x,y)) \textbf{ says } delete(y)) \rightarrow \\ (\{z\}(super\text{-}user(z)) \textbf{ says } delete(y)$$

Moreover we can use the following to restrict the scope of speaks for:

$$\alpha(X) \wedge \Box_B X \rightarrow \Box_A X$$

If $\alpha(X) = \varphi \rightarrow X$ then B speaks for A only on consequences of φ. The restricted speaks for is strictly related with delegation, if for instance $B \Rightarrow_T A$ we say that B is delegated by A on T. If we want to limit the delegation chain to one step such that we do not permit B to delegate another principal C on T, we add the following constraint:

$$(C \Rightarrow_T B \Rightarrow_T A) \rightarrow (C = B)$$

Separation of duties

One of the main concerns in security is the separation of duties: for example the principal(s) signing an order cannot be the same principals who approve it:

$$\neg(\{x\}(\{y\}(x = y) \textbf{ says } sign(project)) \textbf{ says } \\ approving(project))$$

In this formula we exploit the possibility to define the principal in terms of the **says** operator.

As noticed in [7] separation of duties requires using negation. For instance, in FSL we can express the following: "A member m of the Program Committee can not accept a paper P_1 in which one of its authors says that he has published a paper with him after 2007"

$$\neg(\{m\}[PC(m) \wedge \{y\}author\_of(y, P_1) \textbf{ says } \exists p(paper(p) \wedge \\ author\_of(m,p) \wedge author\_of(y,p) \wedge year(p) \geq 2007)] \textbf{ says } accept(P_1))$$

Roles

When roles are considered, it emerges the question whether we consider roles types or instances. We distinguish here among roles instances which can be principals by themselves or properties of other principals. So a sentence like "A, who plays a role x of type R, **says** ψ" becomes:

$$\forall x(x = A \wedge \mathit{role\text{-}played\text{-}by}(x, y) \wedge R(y)) \rightarrow \{z\}(z = y)\ \mathbf{says}\ \psi$$

As concerns hierarchies, if we have:

$$\forall x\ \mathit{super\text{-}user}(x) \rightarrow \mathit{user}(x)$$

then we may want to express the following constraint

$$\forall x\ \mathit{super\text{-}user}(x) \rightarrow (\{z\}(x = z)\ \mathbf{says}\ \mathbf{s}) \rightarrow (\forall x\ \mathit{user}(x) \rightarrow (\{z\}(x = z)\ \mathbf{says}\ \mathbf{s})$$

In [3] if A says something in a role, then it is true that he is playing a role. However, he admits that there should be some requirements to play a role.

For instance, we require that a super-user is a technician:

$$\forall x\ \mathit{super\text{-}user}(x) \rightarrow \mathit{technician}(x)$$

then we can say

$$\forall x\ (x = A \wedge \mathit{super\text{-}user}(x)) \rightarrow (\{z\}(x = z)\ \mathbf{says}\ \mathbf{s})$$

but there can be no super-user x if A is not a technician.

Parameterized roles can add significant expressiveness to a role-based system and reduce the number of roles [7,16,24]. If we model roles as instances they can have attributes. For instance the example in [7] "NHS[3] says x can access health record of patient if x is a treating clinician of patient" can be modeled as:

$$(\mathit{clinician\text{-}role}(x) \wedge \mathit{patient}(p) \wedge \mathit{record}(r, p) \wedge \mathit{treats}(x, p)) \rightarrow (\{w\}(w = x)\ \mathbf{says}\ \mathit{access}(r) \rightarrow \mathit{NHS}\ \mathbf{says}\ \mathit{access}(r)))$$

In FSL, we can model the operator used to represent a principal A in the role B $(A \mid B)$ in [3] in the following way.

$$(A \mid B)\ \mathbf{says}\ \mathbf{s} \equiv A\ \mathbf{says}\ (B\ \mathbf{says}\ \mathbf{s})$$

In order to match the predicate role-played-by with the above definition we can add the following (where x is a role):

$$\forall x, y\ \mathit{role\text{-}played\text{-}by}(x, y) \rightarrow ((x\ \mathbf{says}\ \mathbf{s}) \rightarrow y\ \mathbf{says}\ (\{z\}(z = x)\ \mathbf{says}\ \mathbf{s}))$$

[3]National Health Service.

Discretionary access control

Discretionary access control allows users to pass on their access rights to other users at their own discretion. For instance we can express: "FileServer says user can say x can access resource if user can access resource"[7]

$$\forall x\ user(x) \wedge user(z) \rightarrow (Fileserver\ \mathbf{says}\ \{w\}(\{y\}(w = y = x)\ \mathbf{controls}\ access(u))\ \mathbf{controls}\ \{t\}(t = z)\ \mathbf{controls}\ access(u))$$

Groups

In FSL you have to possibility to express how the set $\{x|\varphi(x)$ holds$\}$ says what it says, e.g. If $\varphi(x) = (x = A_1) \vee (x = A_2) \vee (x = A_3)$ then if at least one of $\{A_i\}$ **says** ψ is enough for the group to **say** ψ we add:

$$\{x\}\varphi(x)\ \mathbf{says}\ \psi \leftrightarrow \bigvee_{1 \leq i \leq 3} \{x\}(x = A_i)\ \mathbf{says}\ \psi.$$

This represents the fact that each principal in the group can speak for the whole group. We can as well express that every group has a spokesman (maybe several ones dependent on issues), that one cannot be a spokesman for two different groups and that a group controlling an issue cannot control issues inconsistent with the definition of the group. We can define groups using what they say as part of the definition, put restriction on what they further say or control.

1. *Every group has a spokesman.*
   This is an axiom schema in φ. Let $\mathbf{spoke}(\varphi, y)$ be

   $$\mathbf{spoke}(\varphi, y) = (\forall X[\{x\}\varphi(x)\ \mathbf{says}\ X \leftrightarrow \{x\}(x = y)\ \mathbf{says}\ X])$$

   We then take the axiom as $\exists y\ \mathbf{spoke}\ (\varphi, y)$.
2. *One cannot be a spokesman for two different groups.*

   $$\forall y[\ \mathbf{spoke}\ (\varphi_1, y) \wedge\ \mathbf{spoke}\ (\varphi_2, y) \rightarrow \forall x[\varphi_1(x) \leftrightarrow \varphi_2(x)]]$$

3. A group cannot control issues inconsistent with the definition of the group

   $$\frac{\vdash \varphi \wedge \psi \rightarrow \bot}{\vdash [(\{x\}\varphi(x)\ \mathbf{says}\ \psi) \rightarrow \psi] \rightarrow \bot}$$

4. A group says ψ iff someone in the role of *Board* says ψ

$$\{x\}\varphi(x) \textbf{ says } \psi \leftrightarrow \bigvee_i \{x\}(x = A_i) | Board \textbf{ says } \psi.$$

5. A group says ψ iff someone with property P says ψ

$$\{x\}\varphi(x) \textbf{ says } \psi \leftrightarrow (\bigvee_i \{x\}(x = A_i) \wedge P(x)) \textbf{ says } \psi.$$

The following additional axiom expresses that the group identified by the extension of $\{x\}\varphi(x)$ says ψ if at least two members says ψ:

$$\{x\}(\textstyle\bigvee_i x = A_i) \textbf{ says } \psi \leftrightarrow$$
$$\textstyle\bigvee_{i \neq j} [\{x\}(x = A_i) \textbf{ says } \psi \wedge \{x\}(x = A_j) \textbf{ says } A_j]$$

More generally, majority voting in $\{x\}\varphi(x)$ **says** ψ, is just an axiom.

$$\{x\}\varphi(x) \textbf{ says } \psi \leftrightarrow \bigvee_i \{x\}\varphi_i(x) \textbf{ says } \psi$$

where $\varphi_i(x)$ are all formulas $(\forall x \varphi_i(x) \rightarrow \varphi(x))$ defining majorities in the set $\{x\}\varphi(x)$.

Majority vote is an example of threshold-constrained trust SPKI/SDSI [11]. The concept of k-of-n threshold subjects means that at least k out of n given principals must sign a request and it is used to provide a fault tolerance mechanism. RTT has the language construct of "threshold structures" for similar purposes [39]. As in SecPAL [7] there is no need for a dedicated threshold construct, because threshold constraints can be expressed directly.

4. The basic system FSL

This section introduces our basic system FSL step by step from a semantical point of view. First, in Section 4.1 we introduce modalities indexed by propositional atoms, then we take into account classical and intuitionistic models for the propositional setting and finally, in Section 4.2, we give a fibred semantics for modalities indexed by first-order formulas.

The FSL system can be defined with any logic $\mathbb{L}$ as a *Fibred Security System based on* $\mathbb{L}$. We will motivate the language for the cases of $\mathbb{L}$ = classical logic and $\mathbb{L}$ = intuitionistic logic.

Basically adding the **says** connective to a system is like adding many modalities. So to explain and motivate FSL technically we need to begin with examining options for adding modalities to $\mathbb{L}$.

4.1. Adding modalities

We start by adding modalities to classical propositional logic. We are going to do it in a special way. The reader is invited to closely watch us step-by-step,

Our approach is semantic.

Let S be a nonempty set of possible worlds. For every subset $U \subseteq S$ consider a binary relation $R_U \subseteq S \times S$.

This defines a multimodal logic, containing at most 2^S modalities $\Box_U$, $U \subseteq S$. The models are of the form $(S, R_U, t_0, h), U \subseteq S$. In this view, if $U = \{t | t \vDash \varphi_U\}$ for some φ_U we get a modal logic with modalities indexed by formulas of itself. This requires now a formal definition.

Definition 4.1 (Language). Consider (classical or intuitionistic) propositional logic with the connectives $\wedge, \vee, \rightarrow, \neg$ and a binary connective $\Box_\varphi \psi$, where φ and ψ are formulas.[4] The usual definition of wff is adopted.

Definition 4.2. We define classical Kripke models for this language.

1. A model has the form

$$\mathbf{m} = (S, R_U, t_0, h), U \subseteq S$$

   where for each $U \subseteq S, R_U$ is a binary relation on S. $t_0 \in S$ is the actual world and h is an assignment, giving for each atomic q a subset $h(q) \subseteq S$.

2. We can extend h to all formulas by structural induction:
   - $h(q)$ is already defined, for q atomic
   - $h(A \wedge B) = h(A) \cap h(B)$
   - $h(\neg A) = S - h(A)$
   - $h(A \rightarrow B) = (S - h(A)) \cup h(B)$
   - $h(A \vee B) = h(A) \cup h(B)$
   - $h(\Box_\varphi \psi) = \{t |$ for all s $(tR_{h(\varphi)}s \rightarrow s \in h(\psi))\}$
3. $\mathbf{m} \vDash A$ iff $t_0 \in h(A)$.

Let us now do the same for intuitionistic logic. Here it becomes more interesting. An intuitionistic Kripke model has the form

$$\mathbf{m} = (S, \leq, t_0, h),$$

[4] There are many such connectives, e.g. φ **says** $\psi, \varphi > \psi$ (conditional), $\bigcirc(\varphi/\psi)$ relative obligation, etc. The semantics given to it will determine its nature.

where $(S, \leq)$ is a partially ordered set, $t_0 \in S$ and h is an assignment to the atoms such that $h(q) \subseteq S$. We require that $h(q)$ is a closed set, namely

$$x \in h(q) \text{ and } x \leq y \Rightarrow y \in h(q)$$

Let D be a set, we can add for each $U \subseteq D$ a binary relation R_U on S. This semantically defines an intuitionistic modality, $\Box_U$.

In intuitionistic models we require the following condition to hold for each formula A, i.e. we want $h(A)$ to be closed:

$$x \in h(A) \text{ and } x \leq y \Rightarrow y \in h(A)$$

This condition holds for A atomic and propagates over the intuitionistic connectives $\wedge, \vee, \rightarrow, \neg, \bot$. To ensure that it propagates over $\Box_U$ as well, we need an additional condition on R_U. To see what this condition is supposed to be, assume $t \vDash \Box_U A$. This means that

$$\forall y(tR_U y \Rightarrow y \vDash A)$$

Let $t \leq s$. If $s \nvDash \Box_U A$, then for some z such that $sR_U z$ we have $z \nvDash A$. This situation will be impossible if we require

$$t \leq s \wedge sR_U z \Rightarrow tR_U z \qquad (*)$$

Put differently, if we use the notation:

$$R'_U(x) = \{y | xR_U y\}$$

then

$$x \leq x' \Rightarrow R'_U(x) \supseteq R'_U(x') \qquad (*)$$

We now want to concentrate on what happens if U is defined by a formula φ_U, i.e. $U = h(\varphi_U)$. This will work only if U is closed

- $t \in U \wedge t \leq s \Rightarrow s \in U$.

So from now on, we talk about modalities associated with closed subsets of S.

We can now define our language. This is the same as defined in Definition 4.1. We now define the semantics.

Definition 4.3. A model has the form

$$\mathbf{m} = (S, \leq, R_U, t_0, h), U \subseteq S$$

where $(S, \leq)$ is a partial order, $t_0 \in S$, and each $U \subseteq S$ is a closed set and so is $h(q)$ for atomic q. R_U satisfies condition (*) above. We define the notion $t \vDash A$ for a wff by induction, and then define

$$h(A) = \{t | t \vDash A\}$$

So let's define $\vDash$:

- $t \vDash q$ iff $t \in h(q)$
- $t \vDash A \wedge B$ iff $t \vDash A$ and $t \vDash B$
- $t \vDash A \vee B$ iff $t \vDash A$ or $t \vDash B$
- $t \vDash A \rightarrow B$ iff for all $s, t \leq s$ and $s \vDash A$ imply $s \vDash B$
- $t \vDash \neg A$ iff for all s, $t \leq s$ implies $s \nvDash A$
- $t \nvDash \bot$
- $t \vDash \Box_{\varphi}\psi$ iff for all s such that $tR_{h(\varphi)}s$ we have $s \vDash \psi$. We assume by induction that $h(\varphi)$ is known.
- $\mathbf{m} \vDash A$ iff $t_0 \vDash A$.

It is our intention to read $\Box_{\varphi}\psi$ as φ **says** ψ.

4.2. Predicate FSL

Intuitively, a predicate FSL fibred model is represented by a set of models linked together by means of a *fibring function*, every model has an associated domain D of elements together with a set of formulas that are true in it. In the FSL meta-model, the evaluation of the generic formula $\alpha = \{x\}\varphi(x)$ **says** ψ is carried out in two steps, first evaluating φ and then ψ in two different models. Suppose $\mathbf{m_1}$ is our (first order) starting model in which we identify $U \subseteq D$ as the set of all the elements that satisfy φ. Once we have U we can access one or more worlds depending on the *fibring function* $\mathbf{f} : \mathcal{P}(D) \rightarrow \mathcal{P}(M)$ which goes from sets of elements in domain D to sets of models. At this point, for every model $\mathbf{m_i} \in \mathbf{f}(U)$ we must check that ψ is *true*, if this is the case then α is true in the meta-model.

The fact that in the same expression we evaluate different sub-formulas in different models is the core idea of the fibring methodology [13]. Think about a group of administrators that have to set up security policies for their company. From a semantical point of view, if we want to check if ψ holds in the depicted configuration by the administrators, we must

1. Identify all the admins (all the elements that satisfy $admin(x)$).
2. Access the model that all the admins as a group have depicted.
3. Check in that model if ψ is *true* or *false*

Let $\mathbb{L}$ denote classical or intuitionistic predicate logic.[5] We assume the usual notions of variables, predicates, connectives $\wedge, \vee, \rightarrow, \neg$, quantifiers $\forall, \exists$ and the notions of free and bound variables.

Let $\mathbb{L}^+$ be $\mathbb{L}$ together with two special symbols:

- A binary (modality), **says**
- A set-binding operator $\{x\}\varphi(x)$ meaning the set of all x such that $\varphi(x)$

Note that semantically at the appropriate context $\{x\}\varphi(x)$ can behave like $\forall x\varphi(x)$ and sometimes in other contexts, we will use it as a set.

Definition 4.4. The language FSL has the following expressions:

1. All formulas of $\mathbb{L}^+$ are level 0 formulas of FSL.
2. If $\varphi(x)$ and ψ are formulas of $\mathbb{L}^+$ then $\alpha = \{x\}\varphi(x)$ **says** ψ are level 1 'atomic' formulas of FSL. If $(x, x_1, \ldots, x_n)$ are free in φ and $y_1, \ldots, y_m$ are free in ψ then $\{x_1, \ldots, x_n, y_1, \ldots, y_m\}$ are free in α. The variable x in φ gets bound by $\{x\}$. The formula of level 1 are obtained by closure under the connectives and quantifiers of $\mathbb{L}^+$.
3. Let $\varphi(x)$ and ψ be FSL formulas of levels r_1 and r_2 respectively, then $\alpha = \{x\}\varphi$ **says** ψ is an 'atomic' formula of FSL of level $r = \max(r_1, r_2) + 1$.
4. Formulas of level n are closed under classical logic connectives and quantifiers of all 'atoms' of level $m \leq n$.

Definition 4.5 (FSL classical fibred model of level n).

1. Any classical model with domain D is a FSL model of level 0.
2. Let $\mathbf{m}$ be a classical model of level 0 with domain D and let for each subset $U \subseteq D, \mathbf{f}^n(U)$ be a family of models of level n (with domain D). Then $(\mathbf{m}, \mathbf{f}^n)$ is a model of level $n + 1$.

Definition 4.6 (Classical satisfaction for FSL). We define satisfaction of formulas of level n in classical models of level $n' \geq n$ as follows.

First observe that any formula of level n is built up from atomic predicates of level 0 as well as 'atomic' formulas of the form $\alpha = \langle\{x\}\varphi(x)\rangle\psi$, where φ and ψ are of lower level.

We therefore first have to say how we evaluate $(\mathbf{m}, \mathbf{f}^n) \vDash \alpha$.

We assume by induction that we know how to check satisfaction in $\mathbf{m}$ of any $\varphi(x)$, which is of level $\leq n$.

[5] Classical predicate logic and intuitionistic predicate logic have the same language. The difference is in the proof theory and in the semantics.

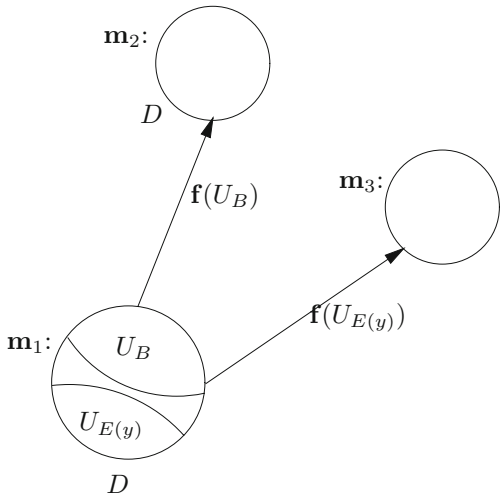

Figure 1.

We can therefore identify the set $U = \{d \in D \mid \mathbf{m} \vDash \varphi(d)\}$.
Let $\mathbf{m}' \in \mathbf{f}^n(U)$. We can now evaluate $\mathbf{m}' \vDash \psi$, since ψ is of level $\leq n-1$.
So we say

$$(\mathbf{m}, \mathbf{f}^n) \vDash \alpha \text{ iff for all } \mathbf{m}' \in \mathbf{f}^n(U), \text{ we have } \mathbf{m}' \vDash \psi.$$

We need to add that if we encounter the need to evaluate $\mathbf{m} \vDash \{x\}\beta(x)$, then we regard $\{x\}\beta(x)$ as $\forall x \beta(x)$.

Example 4.7. Figure 1 is a model for

$$\alpha(y) = \{x\}[\{u\}B(u) \textbf{ says } (B(x) \rightarrow A(x,y))] \textbf{ says } F(y)$$

In Figure 1, $\mathbf{m}_1$ is a single model in $\mathbf{f}^1(U_B)$ and $\mathbf{m}_3$ is a single model in $\mathbf{f}^1(U_{E(y)})$, as defined later.

The set U_B is the extension of $\{x\}B(x)$ in $\mathbf{m}_1$.

To calculate the set of pairs (x, y) such that $E(x, y) = \{u\}B(u)$ **says** $(B(x) \rightarrow A(x, y))$ holds in $\mathbf{m}_1$, we need to go to $\mathbf{m}_2$ in $\mathbf{f}(U_B)$ and check whether $B(x) \rightarrow A(x, y)$ holds in $\mathbf{m}_2$, x, y are free variables so we check the value under fixed assignment.

We now look at $E(y) = \{x\}E(x, y)$ for y fixed, we collect all elements d in D such that $\mathbf{m}_2 \vDash B(d) \rightarrow A(d, y)$. Call this set $U_{E(y)}$.

To check $\alpha(y) = \{x\}E(x, y)$ **says** $F(y)$ in $\mathbf{m}_1$ we have to check whether $F(y)$ holds in $\mathbf{m}_3$.

We now define intuitionistic models for FSL. This will give semantics for the intuitionistic language.

Definition 4.8. We start with intuitionistic Kripke models which we assume for simplicity have a constant domain. The model $\mathbf{m}$ has the form $(S, \leq, t_0, h, D)$ where D is the domain and $(S, \leq, t_0)$ is a partial order with first point t_0 and h is an assignment function giving for each $t \in S$ and each m-place atomic predicate P a subset $h(t, P) \subseteq D^m$ such that $t_1 \leq t_2 \Rightarrow h(t_1, P) \subseteq h(t_2, P)$

We let $h(P)$ denote the function $\lambda t\ h(t, P)$. For $t \in S$ let

$$\begin{aligned} S_t &= \{s \mid t \leq s\} \\ h(t, P) &= h(P) \restriction S_t \\ \leq_t &= \leq \restriction S_t \end{aligned}$$

Where $\restriction$ represents the standard domain restriction.

Let $\mathbf{m}_t = (S_t, \leq_t, t, h_t, D)$.

Note that a formula φ holds at $\mathbf{m} = (S, \leq, t_0, h, D)$ iff $t_0 \vDash \varphi$ according to the usual Kripke model definition of satisfaction.

1. A model of level 0 is any model $\mathbf{m}$: $\mathbf{m} = (S, \leq, t_0, h, D)$.
2. Suppose we have defined the notion of models of level $m \leq n$, (based on the domain D).

We now define the notion of a model of level $n + 1$

Let $\mathbf{m}$ be a model of level 0 with domain D. We need to consider not only $\mathbf{m}$ but also all the models $\mathbf{m}_t = (S_t, \leq_t, t, h_t, D)$, for $t \in S$. The definitions will be given simultaneously for all of them.

By an intuitionistic 'subset' of D in $(S, \leq, t_0, h, D)$, we mean a function $\mathbf{d}$ giving for each $t \in S$, a subset $\mathbf{d}(t) \subseteq D$ such that $t_1 \leq t_2 \Rightarrow \mathbf{d}(t_1) \subseteq \mathbf{d}(t_2)$.

Let $\mathbf{f}_t^n$ be a function associating with each $\mathbf{d}_t$ and $t \in S$ a family $\mathbf{f}_t^n(\mathbf{d}_t)$ of level n models, such that $t_1 \leq t_2 \Rightarrow \mathbf{f}_{t_1}^n(\mathbf{d}_{t_1}) \supseteq \mathbf{f}_{t_2}^n(\mathbf{d}_{t_2})$. Then $(\mathbf{m}_t, \mathbf{f}_t)$ is a model of level $n + 1$ where $\mathbf{d}_t = \mathbf{d} \restriction S_t$.

Definition 4.9 (Satisfaction in fibred intuitionistic models). We define satisfaction of formulas of level n in models of level $n' \geq n$ as follows.

Let $(\mathbf{m}_t, \mathbf{f}_t^n)$ be a level n model. Let $\alpha = \{x\}\varphi(x)$ **says** ψ is of level n. We assume we know how to check satisfaction of $\varphi(x)$ in any of these models.

We can assume that

$$\mathbf{d}_t = \{x \in D \mid t \vDash \varphi(x) \text{ in } (\mathbf{m}_t, \mathbf{f}_t^n)\}$$

is defined. Then $t \vDash \alpha$ iff for all models $\mathbf{m}_t'$ in $\mathbf{f}_t^n(\mathbf{d}_t)$ we have $\mathbf{m}_t' \vDash \psi$.

5. Kripke Models for Axioms

In this section we show one advantage in employing the multimodal semantics. First, we translate the most important second-order axioms appeared in [2] to first-order constraints on the kripke models. Then discuss models in which we have desirable security axioms like *Unit* and *C4*, but not necessary *Escalation*.

Example 5.1 (Two intuitionistic modalities). Let us examine the case of two intuitionistic modalities in more detail, call them $\Box_A$ and $\Box_B$ and their accessibility relations R_A and R_B. So our Kripke model has the form $(S, \leq, R_A, R_B, t_0, h)$. We know for $\mu = A$ or $\mu = B$ that we have in the model

$$t \leq s \wedge sR_\mu z \rightarrow tR_\mu z. \qquad (*)$$

What other conditions can we impose on $\Box_\mu$?

1. *The axiom Unit* $X \rightarrow \Box_\mu X$
   corresponds to the condition
   $$xR_\mu y \rightarrow x \leq y \qquad (*1)$$
2. *The axiom C4* $\Box_A \Box_A x \rightarrow \Box_A$
   corresponds to the condition, for finite models[6]
   $$xR_A y \wedge yR_A z \rightarrow zR_A y$$
3. *The axiom speaks-for* $\Box_B X \rightarrow \Box_A X$
   corresponds to the condition
   $$xR_A y \rightarrow xR_B y \qquad (*2)$$
4. Note that $\Box_B X \rightarrow \Box_A X$ is taken in (*2) as an axiom schema. If we want to have $t \vDash \forall X(\Box_B X \rightarrow \Box_A X)$ i.e. we want $\Box_B \varphi \rightarrow \Box_A \varphi$ to hold at the point $t \in S$ for all wff φ, we need to require (*2) to hold above t, i.e.
   $$\forall x, y(t \leq x \wedge xR_A y \rightarrow xR_B y) \qquad (*3)_t$$
5. Consider now an axiom called *hand-off A to B*.
   $$\Box_A(\forall X(\Box_B X \rightarrow \Box_A X)) \rightarrow \forall X(\Box_B X \rightarrow \Box_A X)$$

[6] For infinite models, C4 gives you density of the accessibility relationship.

This axiom has a second order propositional quantifier in it.
The antecedent of the axiom wants $\Box_A(\forall X \Box_B X \to \Box_A X))$ to hold at t_0. This means in view of (3) above that $(*4_a)$ needs to hold

$$\forall t(t_0 R_A t \to (*3)_t) \qquad (*4_a)$$

The axiom says that if the antecedent holds at t_0 so does the consequent, i.e.

$$t_0 \vDash \forall X(\Box_B X \to \Box_A X).$$

We know the condition for that to hold is $(*3)_{t_0}$. Thus the condition for Hand-off A to B is

$$\forall t[t_0 R_A t \to (*3)_t] \to (*3)_{t_0} \qquad (*4)$$

The important point to note is that although the axiom is second order (has $\forall X$ in it both in the antecedent and consequence), the condition on the model is first order[7].

6. Concerning *Escalation*:

$$\Box_A X \to X \vee \Box_A \bot$$

its condition is

$$\exists y(x R_A y) \to x R_A x \qquad (*5)$$

To check whether we can have hand-off from A to B without escalation for A, for some choice of R_A and R_B, we need to check whether we can have (*4) without having (*5), for some wise choice of R_A and R_B.

7. Consider a Kripke model $(S, \leq, t_0)$ which is nonending and dense, i.e.
   - $\forall x \exists y(x \lneqq y)$
   - $\forall xy(x \lneqq y \to \exists z(x \lneqq z \lneqq y))$

   In this model let

   $$x R_A y \text{ be } x \lneqq y$$
   $$x R_B y \text{ be } x \leq y.$$

   We have here that $(*3)_t$ holds for any t because it says

   $$\forall xy(t \leq x \wedge x \lneqq y \to x \leq y)$$

   Therefore (*4) also holds. This is hand-off from A to B.
   However, escalation does not hold because

   $$\exists y(x \lneqq y) \to x \lneqq x$$

   is false.

[7]Notice that we use first-order but we get a language more expressive than CDD[2] which is second-order.

DEFINITION 5.2. Let $(S, \leq, t_0, h)$ be a Kripke model. By a modal function $\mathbf{E}$ we mean a function giving to each point $t \in S$ a set of points $\mathbf{E}(t)$ such that

1. $t \lneqq s$ for all $s \in \mathbf{E}(t)$.
2. $t_1 \leq t_2 \rightarrow \mathbf{E}(t_1) \leq \mathbf{E}(t_2)$ where $\mathbf{E}(t_1) \leq \mathbf{E}(t_2)$ means $\forall x \in \mathbf{E}(t_2) \exists y \in \mathbf{E}(t_1)(y \leq x)$.
3. $x \in \mathbf{E}(t) \wedge x \leq y \rightarrow y \in \mathbf{E}(t)$

The intention is that $t \models \Box\psi$ iff for all $s \in \mathbf{E}(s), s \models \psi$. The conditions on $\mathbf{E}(t)$ ensure that the axiom (Unit) $A \rightarrow \Box A$ holds.

DEFINITION 5.3. $(S, \leq, t_0, \mathbf{E})$ is $\mathbf{E}$-dense iff the following holds:

- If $x \in \mathbf{E}(t)$, then for some $y, y \in \mathbf{E}(t) \wedge x \in \mathbf{E}(y)$ [8].

In the following, in order to show the existence of dense systems, we present some representative kripke models.

EXAMPLE 5.4 (Dense Kripke Models). Let $(T, <)$ be a dense order, for example it can be the rational numbers with usual ordering of "smaller than". We construct two modal models out of $(T, <)$ by adding two different possible sets $\mathbf{E}$ for it.

1. Let $\mathbf{f}$ be an increasing function such that
   (a) $x < \mathbf{f}(x)$
   (b) $x < y \rightarrow \mathbf{f}(x) < \mathbf{f}(y)$

   For each t, let $\mathbf{E}_{\mathbf{f}}(t)$ be

   $$\mathbf{E}_{\mathbf{f}}(t) = \{s \mid \mathbf{f}(t) < s\}$$

   For example we can take $\mathbf{f}(t) = t + 1$
2. For each t, let

   $$\mathbf{E}^2(t) = \{s \mid t < s\}$$

   Let $\mathbf{m}_1$ be the model $(T, <, \mathbf{E}_f)$ and $\mathbf{m}_2$ be the model $(T, <, \mathbf{E}^2)$

THEOREM 5.5. *The C4 axiom* $\Box\Box A \rightarrow \Box A$ *fails in* $\mathbf{m}_1$ *and holds in* $\mathbf{m}_2$.

[8] Notice that this corresponds to axiom C4 $\Box\Box A \rightarrow \Box A$

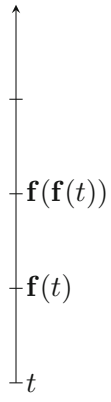

Figure 2.

PROOF. Consider Figure 2. First we show that $C4$ holds in $\mathbf{m}_2$.

Assume $t \models \Box\Box A$. We show $t \models \Box A$, let $t < s$, show $s \models A$. Choose y such that $t < y < s$, then since $t \models \Box\Box A$, we have $y \models \Box A$ and hence $s \models A$.

We now show density $C4$ fails in $\mathbf{m}_1$.

Consider Figure 2 and let A hold from after $\mathbf{f}(\mathbf{f}(t))$. Then $t \models \Box\Box A$, since $\mathbf{f}(t) \models \Box A$.

However, $t \not\models \Box A$, because there are points y such that $\mathbf{f}(t) < y < \mathbf{ff}(t)$ and at such points, $y \not\models A$. ■

The model $\mathbf{m}_2$ is very simple, it satisfies the axiom $A \vee \Box(B \rightarrow A)$. We want a general model of density which has no additional commitments. For this purpose we combine the models $\mathbf{m}_1$ and $\mathbf{m}_2$ in a certain way.

EXAMPLE 5.6 (Special Model). We construct a special model that will satisfy $C4$. We start with the model $\mathbf{m}_1$ of Example 5.4. This model does not satisfy $C4$ but it is also not dense. So we correct this by combining it with $\mathbf{m}_2$, add points and make it dense. Our starting point is the model $(T, <, \mathbf{f})$ of example 5.4. The problem is illustrated in Figure 3, the point y between $\mathbf{f}(t)$ and $\mathbf{ff}(t)$ is not reachable by any wff of the form $\Box\Box A$ but it is reached by $\Box A$. So by making true from $\mathbf{f}(\mathbf{f}(t))$ and A false at y, we get that $\Box\Box A = \top$ but $\Box A = \bot$.

What we need is another copy of the linear order $(T, <, \mathbf{f})$ leading sideways from t to y, as shown in the Figure 3. Call the main linear order

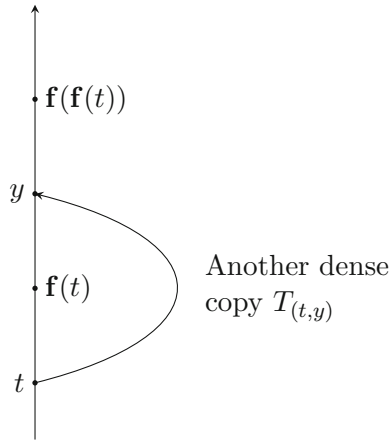

Figure 3.

$(T, <, \mathbf{f})$. Then we can call the copy running from t to y by $(T_{(t,y)}, <_{(t,y)}, \mathbf{f}_{(t,y)})$ which is an identical copy but we have many such pairs (t, y) and we need to keep good accounting.

Now in $T_{(t,y)}$, we have the same problem. We will have points t_1, y_1 such that $\mathbf{f}_{(t,y)}(t_1) < y_1 < \mathbf{f}_{(t,y)}(\mathbf{f}_{(t,y)}(t_1))$. So we need another copy of $(T, <, \mathbf{t})$ to go from t_1 to y_1, as in Figure 4. We call this copy $T_{(t,y)(t_1,y_1)}$. This process can continue a countable numbers of times for any pairs (t,y). At the end we get a dense order satisfying C4, without any additional specific axioms holding.

However if we want to do the job in one step we need to take for $T_{(t,y)}$ a model which has no problems at all with density, namely the model $\mathbf{m}_2$. If we do that then some additional axioms will hold but we get a simple and quick construction. So let's go for it.

We said $(T, <)$ can be the rational numbers with the usual ordering. It helps us to actually take the rationals and make $\mathbf{f}$ more specific. So let $\mathbf{f}(t) = t + 1$, for each pair of points (t, y) such that $t < \mathbf{f}(t) = t + 1 < y$ let $\mathbf{g}(t, y) = (\mathbf{f}(t) + y)/2 = (t + 1 + y)/2$

Then we have the situation in Figure 5. Let us extract from the particulars of $(T, <)$ and regard it as a dense linear ordering without a first and without a last element. Viewed as such let us call it $\tau = (T, <, \mathbf{f})$. We let τ^* be $(T, <, \mathbf{f}, \mathbf{g})$ where $\mathbf{g}$ satisfies abstractly the following properties:

- $t < \mathbf{f}(t) < y$ implies $\mathbf{f}(t) < \mathbf{g}(t, y) < y$
- $\mathbf{f}(t) < y_1 < y_2$ implies $\mathbf{g}(t, y_1) < \mathbf{g}(t, y_2)$

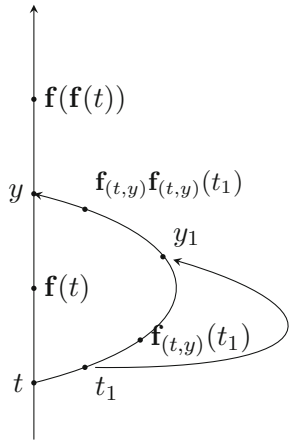

Figure 4.

Later on we would want copies of τ and τ^* to be used in various contexts. We write $\tau_1 = (T_i, <_i, \mathbf{f}_i)$ to indicate the $i-th$ isomorphic copy of τ. Similarly τ_i^*.

Now, with each pair (t, y) such that $\mathbf{f}(t) < y$, let

$$T_{(t,y)} = \{(t, y, z) \mid t < z < \mathbf{g}(t, y)$$

and define $<_{(t,y)}$ on $T_{(t,y)}$ by

$$(t, y, z) <_{(t,y)} (t, y, z') \text{ iff } z < z'$$

In the abstract we can take a copy $\tau_{(t,y)}$ of τ instead of the specific $T_{(t,y)}$.

Now let our new model (S, R) have as set of worlds S and relation R defined as follows.

- $S = T \cup \bigcup_{\mathbf{f}(t)<y} T_{(t,y)}$
- R is the transitive closure of R_1, which is the union of the following four components
  1. $<$ on T
  2. $<_{(t,y)}$ on $T_{(t,y)}$, for $t < y$
  3. $\{(t, (t, y, z)) \mid z \in T_{(t,y)}\}$
  4. $\{((t, y, z), \mathbf{g}(t, y)) \mid z \in T_{(t,y)}\}$

  Figure 6 explains the ordering. The direction of the upward arrow gives the order. To turn (S, R) into a modal model we need to define $\mathbf{E}(t)$, for any $t \in S$

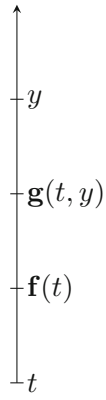

Figure 5.

1. $\mathbf{E}$(t,y,z), for $z \in T_{(t,y)}$ is defined as

$$\mathbf{E}(t, y, z) = \{s \mid (t, y, z)Rs\}$$

In Figure 6, $\mathbf{E}(t, y, z)$ are all points above (t, y, z) in $T_{(t,y)}$ and then the point $\mathbf{g}(t, y)$ in T and all points above it in T.

2. $\mathbf{E}$(t), for $t \in T$ is defined as the transitive R closure of all points in T above $\mathbf{f}(t)$ as well as all points in $T_{(t,y)}$ for any $\mathbf{f}(t) < y$

$$\text{for } (t \in T), \mathbf{E}(t) = \bigcup_{\mathbf{f}(t)<y} T_{(t,y)} \cup \{z \in T \mid \mathbf{f}(t) < z\}$$

Note that any $x \in T$ such that $t < x < \mathbf{f}(t)$ is not in $\mathbf{E}(t)$, nor is any $(x, y, z) \in T_{(x,y)}$, for any $\mathbf{f}(x) < y$. This completes the definition of the model $\mathbf{m} = (S, R, E)$

THEOREM 5.7 (Density of the special model). *The special model of Example 5.4 satisfies the density condition of Definition 5.3.*

PROOF. Assume $x \in \mathbf{E}(t)$, we are looking for a $y \in \mathbf{E}(t)$ such that $x \in \mathbf{E}(y)$. We make a case analysis based on Figure 6:

- case 1: $t \in T$ and for some $x' \in T$ we have $x \in T_{(x',u)}$, for $\mathbf{f}(x') < u$, or $x = x'$ In this case any $(t, y, z) \in T_{(t,x')}$ will do the job.
- case 2: $t \in T$ and $x \in T_{(t,y)}$ then any $y \in T_{(t,y)}$ such that $tRy \wedge yRx$ will do.

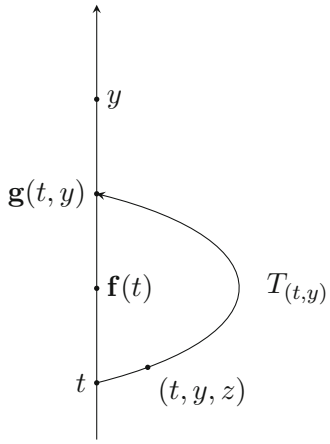

Figure 6.

- case 3: $t \in T_{(s,y)}$, then $x \in \mathbf{E}(t)$ iff tRx and $t \neq x$ and so any y such that $tRy \wedge yRx$ will do

The above exhaust all cases and we get that the model is dense. ■

REMARK 5.8. In models of Definition 5.3 we have *Unit* and *C4* hold but not necessarily *Escalation*.

6. FSL using Horn clauses

Many authorization logics rely on Datalog databases to carry out computation [9]. In [23] it has also been argued that Datalog with constraints could be seen as the logical foundation for trust management. We reject this view by underlining that there exist very expressive access control logics that do not employ Datalog [17,18].

More generally, ordinary logic programs rely on Horn clauses which are interesting mainly for their computational tractability. We now show that a fragment of FSL can be seen as an ordinary logic program.

Suppose we want to reason on ACL policies expressed with horn clauses of the form

$$b_1 \wedge b_2 \wedge b_3 \ldots \wedge b_n \rightarrow h_1$$ [9]

[9]Depending on the application, Horn clauses can be propositional or first-order.

Then we can stick to formulas like

$$\{x\}H1(x) \textbf{ says } H2 \qquad (*)$$

where $H1$ is an horn clause with no free variables in the body, whereas $H2$ are general Horn clauses. From a semantical perspective, we have that for each model we have a domain D and a list of horn clauses that are true in the model. Suppose we want to evaluate (*) in a model $\mathbf{m_1}$ with domain D, $H1$ is then used to select $U \subseteq D$

$$U = \{d \in D | \models H1(d)\}$$

and $H2$ is checked to be true in all models $\mathbf{m}_i \in \mathbf{f(U)}$.

The previous formula has a structure that is the same as a classical first order Horn clause, for instance, if we have for a world $\mathbf{m}$:

$$\mathbf{m} \models \underbrace{\{x\}(\bigwedge_i \varphi_i \rightarrow \varphi(x))}_{B1} \textbf{ says } \underbrace{(\bigwedge_i \psi_j \rightarrow \psi)}_{H1} \qquad \text{(II)}$$

which is true iff

$$\forall n\, (mR_{\{x\}H1}n \rightarrow n \models \bigwedge_j \psi_j \rightarrow \psi) \qquad \text{(III)}$$

that, translated into a classical first-order formula becomes

$$\forall n\, (mR_{\{x\}B1^*}n \rightarrow [\bigwedge_j \psi_j^*(n) \rightarrow \psi^*(n)]) \qquad \text{(IV)}$$

Generally if we have $t \models \psi(x, y)$ we can translate this into a pure first-order formula with $\psi^*(t, x, y)$ where $\psi^*(t, x, y)$ is obtained from $\psi(x, y)$ adding t as a free variable which represents the world into which $\psi(x, y)$ is forced. Notice that the equation IV below is an horn clause, in fact by considering n as a free variable we have

$$mR_{\{x\}B1^*}n \bigwedge_j \psi_j^*(n) \rightarrow \psi^*(n) \qquad \text{(V)}$$

so we can consider FSL formulas of the kind $\Box_{B1}$ **says** $H1$ as comparable with an horn clause.

7. The FSL Methodology

From a methodological perspective, FSL introduces concepts that can be easily exploited to strengthen existing authentication logics appeared in the literature. The main aim of this section is to take into account a representative case study in order to show how the key ideas of the FSL semantics can be employed to enrich some well-know authentication logics. As already underlined in Chapter 3, with the FSL methodology we introduce a language in which formulas have the form:

$$\Box_{\varphi}\psi$$

in which we identify two (not necessarily different) languages $\mathbb{L}_p$ and $\mathbb{L}_E$ such that $\varphi \in \mathbb{L}_p$ and $\psi \in \mathbb{L}_E$. The meaning of the formula above is that all the principals in a given domain that satisfy φ *support* ψ.

Now take an authorization logic **Q** with language $\mathbb{L}_Q$, we can extend it applying the FSL methodology looking at $\Box_{\varphi}\psi$ in two different ways:

1. Let $\psi \in \mathbb{L}_E = \mathbb{L}_Q$ and $\psi \in \mathbb{L}_p$ where expressions in $\mathbb{L}_p$ are classical first-order formulas used to index principals.
2. We first translate expressions of $\mathbb{L}_Q$ into predicate FSL language with the modal operator **says** such that $\mathbb{L}_p = \mathbb{L}_e = \mathbb{L}_{FSL}$. Where $\mathbb{L}_{FSL}$ in the language used in predicate FSL.

In the following we refer to extensions 1 and 2 as T-extension (Trivial extension) and C-extension (Complete extension) respectively. Case by case, depending on the logic that we are studying, it could be sensible to carry out both extensions. It must be underlined that the extensions are incremental so a logic extended with the complete methodology is more expressive than a logic extended with the trivial one.

7.1. The SecPAL case study

In this section we briefly present SecPAL [7] a declarative authorization language that strikes a careful balance between syntactic and semantic simplicity, policy expressiveness, and execution efficiency. Then we extend SecPAL with FSL using the C-extension methodology.

SecPAL's syntax is close to natural language, and the semantics consists of just three deduction rules. The language can express many common policy idioms using constraints, controlled delegation, recursive predicates, and negated queries. The execution strategy is based on translation to Datalog

with Constraints, and table-based resolution. In [7] it is proven that the execution strategy is sound, complete, and always terminates, despite recursion and negation, as long as simple syntactic conditions are met.

In SecPAL formulas are expressed by mean of *assetions* of the following shape:

$$A \textbf{ says } fact \text{ if } fact_1, \ldots, fact_n, c$$

where A is called the *issuer*, facts are sentences that state properties on principals, $fact_1, \ldots, fact_n$ are called conditional facts and c is a constraint of the free variables that appear in the assertion. Assertions are similar to Horn clauses, with the difference that (1) they are qualified by some principal A who issues and vouches for the asserted claim; (2) facts are nested, using the verb phrase **can say**, by means of which delegation rights are specified; and (3) variables in the assertion are constrained by c, a first-order formula that can express e.g. temporal, inequality, path and regular expression constraints.

SecPAL, in addition to the **says** operator, has two other constructs: **can say** and **can act as**. Intuitively, the assertion:

$$\begin{gathered} Alice \textbf{ says } x \textbf{ can say } can\_read(y, file1) \text{ if} \\ can\_read(x, dir), file1 \leq dir, marked\_confidential(file1) \neq Yes \end{gathered}$$

means that *Alice* delegates a principal x to have the right to let a principal y read $file1$ if x has read access to the directory dir which contains $file1$, due that $file1$ is not marked as confidential. In the assertion above the **can say** express a delegation property, i.e. y can speak for x on reading $file1$.

Relating the construct **can act as**, the following SecPAL assertion:

$$A \textbf{ says } B \textbf{ can act as } C$$

intuitively represents the fact that from A point of view every fact concerning C also apply to B.

SecPAL semantics

We now describe the formal semantics of SecPAL which consists of three deduction rules that directly reflect the intuition suggested by the syntax. This proof-theoretic approach enhances simplicity and clarity even if in practice the semantics is not used in the query computation, this is due to the fact that SecPAL programs are first translated into Datalog and then evaluated[10].

[10]For a complete description of the evaluation process and Datalog translation see [7].

Let a substitution θ be a function mapping variables to constants and variables, and let ϵ be the empty substitution. Substitutions are extended to constraints, predicates, facts, claims, assertions etc. in the natural way, and are usually written in postfix notation. We write $vars(X)$ for the set of free variables occurring in a phrase of syntax X.

Each deduction rule consists of a set of premises and a single consequence of the form $\mathcal{AC}, D \models A$ **says** $fact$ where $vars(fact) = \emptyset$ and the delegation flag D is 0 or ∞. Intuitively, the deduction relation holds if the consequence can be derived from the assertion context $\mathcal{AC}$. If $D = 0$, no instance of the rule (can say) occurs in the derivation[11].

(cond)

$$\frac{\begin{array}{c}(A \textbf{ says } \mathsf{if}\ fact_1, \ldots, fact_n, c) \in AC \\ AC, D \models A \textbf{ says } fact_i\theta \ for\, all\, i \in \{1 \ldots k\} \\ \models c\theta \qquad\qquad vars(fact\theta) = \emptyset\end{array}}{AC, \infty \models A \textbf{ says } fact\theta}$$

(can say)

$$\frac{\begin{array}{c}AC, \infty \models A \textbf{ says } B \textbf{ can say}_D\, fact \\ AC, D \models B \textbf{ says } fact\end{array}}{AC, \infty \models A \textbf{ says } fact}$$

(can act as)

$$\frac{\begin{array}{c}AC, D \models A \textbf{ says } B \textbf{ can act as } C \\ AC, D \models A \textbf{ says } C\, verbphrase\end{array}}{AC, D \models A \textbf{ says } B\, verbphrase}$$

Rule (cond) allows the deduction of matching assertions in AC with all free variables substituted by constants. All conditional facts must be deducible with the same delegation flag D as in the conclusion. Furthermore, the substitution must also make the constraint ground and valid.

Rule (can say) deduces an assertion made by A by combining a **can say** assertion made by A and a matching assertion made by B. This rule applies only if the delegation flag in the conclusion is ∞. The matching assertion made by B must be proved with the delegation flag D read from A's **can say** assertion. Therefore, if D is 0, then the matching assertion must be proved without any application of the (can say) rule. If on the other and D is ∞, then B can redelegate. Note that by nesting the $\mathsf{can\ say}_0$ operator, we can

[11] So D si a parameter which limits the delegation depth.

limit the delegation depth. In the following Alice delegates the authority over is *a friend* fact to *Bob* and allows *Bob* to re-delegate at most one level further.

$$Alice \textbf{ says } Bob \textbf{ can say}_0\ firend(x)$$
$$Alice \textbf{ says } Bob \textbf{ can say}_0\ x \textbf{ can say}_0 friend(y)$$

Rule (can act as) asserts that all facts applicable to C also apply to B, when B can act as C is derivable. A corollary is that can act as is a transitive relation.

SecPAL C-extension

We extend SecPAL[7] with FSL using C-extension methodology, first we translate SecPAL in modal logic and then we enrich the expressiveness of SecPAL fibring it with FSL.

We argue that SecPAL can be seen as a subset of predicate FSL in which formulas have the following structure:

$$\Box_A \psi$$

where A is a single principal. We look at the **says** SecPAL constructs as a modal operator, so that A **says** ψ becomes $\Box_A \psi$. More generally, SecPAL *assertions* of the kind

$$A \textbf{ says } fact \text{ if } fact_1, \ldots, fact_n, c$$

are represented as

$$\Box_A fact_1 \wedge \ldots \wedge \Box_A fact_n \wedge c \rightarrow \Box_A fact_1$$

We relate the delegation operator "**can say**$_\infty$"[12] to the "*speaks for*" relationship, so that we translate

$$A \textbf{ says } B \textbf{ can say}_\infty fact$$

into

$$\Box_A B \Rightarrow_{fact} A$$
$$\Box_A (B \Rightarrow_{fact} A) \rightarrow B \Rightarrow_{fact} A$$

Where the second formula represents *hand-off* for restricted *speaks-for* and it is necessary to prove that the (can say) rule translated into modal logic is a theorem.

[12] If $D \neq \infty$, see Section 3 to see how in FSL we can constraint delegation depth.

We argue that the "**can act as**" operator is strictly linked with *syntactical* substitution of principals. For instance,

$$A \textbf{ says } B \textbf{ can act as } C$$

is translated into $\Box_A subs(B, C)$

which intuitively means that for every fact asserted by A about C, A asserts the same fact about B. In a more formal way we have

$$\Box_A subs(B, C) \wedge \Box_A \psi \rightarrow \Box_A \psi\{C/D\}$$

We refer to the translation of SecPAL into a modal flavor as MSecPAL (Modal SecPAL). Now that we have translated SecPAL primitives into MSecPAL which is a subset of FSL, we can show how the rules describing SecPAL semantics are theorems of MSecPAL translation.

We notice that the (cond) rule can be seen as an application of *modus ponens*, for instance take MSec-PAL formula $\Box_A(Y \rightarrow X)$ which mirrors a simple assertion in Sec-PAL with just one conditional fact (Y) and no constraints. It is clear that the following rule

$$\frac{\Box_A(Y \rightarrow X) \wedge \Box_A Y}{\Box_A X}$$

is a theorem of MSec-PAL considering that $\Box_A(Y \rightarrow X) \rightarrow (\Box_A Y \rightarrow \Box_A X)$.

Concerning the (can say) rule we have already underlined that with *hand-off* for limited *speaks for* it is possible to have the following as a theorem

$$\frac{\Box_A(B \Rightarrow_{fact} A) \wedge \Box_B fact}{\Box_A fact}$$

For the last rule (can act as) it is straightforward to see that with the definition of *Subs* given above the MSec-PAL translation of the rule is a theorem of the modal logic. We now propose an extension of MSecPAL called Fibred-MSecPAL (F-MSecPAL) rooted into FSL which introduces the following properties:

- Generalizing the notion of principal into formulas of a chosen logic. Notice that because of our fibred approach the logic identifying principals could be different from the extended language.
- Variables and predicates **can be shared** between $\mathbb{L}_p$ and $\mathbb{L}_e$.
- The **says** operator can be iterated (e.g. A **says** B **says** ...)

F-MSecPAL formulas have the following shape:

$$\Box_{\{x\}\varphi(x)}\psi$$

In which $\{x\}\varphi \in \mathbb{L}_p$ and $\psi \in \mathbb{L}_e$.

We recall that, from a semantical perspective, the $\{x\}\varphi(x)$ selects a subset of principals which is used to index a family of MSecPAL models.

In F-MSecPAL, as in FSL, the formula $\{x\}\varphi(x)$ **says** ψ stands for the whole group, identified by the extension of $\{x\}\varphi(x)$, asserting ψ. If we want to express that a group asserting *fact* stands for all members asserting *fact* separately, we have to add an ad-hoc axiom:

$$\{x\}\varphi(x)\,\mathbf{says}\,\psi \leftrightarrow \bigwedge_i \{x\}(x = x_i)\,\mathbf{says}\,\psi \qquad (*)$$

where x_i are elements of the extension of $\{x\}\varphi(x)$.

In F-MSecPAL we can have formulas like

$$\{x\}\forall y[admin(y) \rightarrow y\ \mathbf{says}\ Good(x)]\ \mathbf{says}\ fact$$

in which principals are selected through another F-MSecPAL formula (the fibred approach has been iterated).

8. Conclusion and Future Work

We have presented a logical formalism called FSL based on fibred multimodal first-order logic. The proposed framework extends existing logics for access control by introducing sets of principals described by formulas as first-class citizen that can jointly support statements. FSL is based on a general methodology to combine logics and use them within a same language called fibring [13]. Thanks to the proposed semantics based on multimodalties indexed by first-order formulas, we showed that second-order logics are not necessary to model common axioms for the **says** like 'speaks-for' or 'hand-off'.

For instance, in [2] the presented calculus for the proposed (second-order) access control logic can not be employed in practical theorem proving due to its complexity. On the contrary, there exists works in which first-order languages are constrained in order to get nice computational results in the derivation time [18,22].

We studied how security axioms can be translated into first-order constraints on Kripke models by introducing a model-driven study of logics for access control as underlined in [14].

As ongoing work, we are formalizing the extension of other well known logics like DL [22], DEBAC [8] and DKAL [17] with the FSL methodology and then to translate them into predicate FSL. In this view, FSL can be studied as a general framework to compare and integrate different logics for access control.

We are also working in crafting calculi for different fragments of FSL, in particular we are concentrating on the propositional intuitionistic fragment and on the more general predicate FSL. Relating the propositional fragment, we believe that calculi for conditional logics can be adapted to deal with the says modality, therefore for predicate FSL calculus we plan to use Labelled Deductive Systems [12].

A. Appendix

A.1. Axiomatisation and completeness of FSL

We prove completeness for FSL with increasing domains and for FSL with constant domains (FSL and FSL_{CD}). Well-formed formulas (wffs) are defined recursively as follows:

- Atoms of the form $P(t_1 \dots t_n)$[13] are wffs.
- $\bot$ is a wff.
- If α and β are wff, then so are $(\neg\alpha), (\alpha \wedge \beta), (\alpha \vee \beta), (\alpha \to \beta), (\forall x \alpha), (\exists x \alpha)$.
- If $\varphi(x)$ and ψ are wff, the so is $\{x\}\varphi(x)\,\mathbf{says}\,\psi$.

A.1.1. Axiom system for predicate FSL

1. All axioms and rules for intuitionistic logic
2. Extensionality axiom:

$$\forall x(\varphi_1(x) \leftrightarrow \varphi_2(x)) \to \\ (\{x\}\varphi_1(x)\ \mathbf{says}\ \psi \leftrightarrow \{x\}\varphi_2(x)\ \mathbf{says}\ \psi)$$

3. Modality axiom:

$$\frac{\vdash \bigwedge_i \alpha_i \to \beta}{\vdash \bigwedge_i \{x\}\varphi\ \mathbf{says}\ \alpha_i \to \{x\}\varphi\ \mathbf{says}\ \beta}$$

[13] where $t_1 \dots t_n$ are classical first-order terms.

4. Constant domains axioms[14]:
   (a) $\forall y\{x\}\varphi$ **says** $\beta(y) \rightarrow \{x\}\varphi$ **says** $\forall y\beta(y)$
   (b) $\forall y(\psi \vee \beta(y)) \rightarrow (\psi \rightarrow \forall y\beta(y))$
5. Additional Axioms:
   here we put all the axioms we need to craft our logic like the ones in Section 2
   (a) $A \rightarrow \{x\}\varphi$ **says** A
   (b) $\forall t(\varphi(t) \rightarrow t$ **says** $\psi) \rightarrow \{x\}\varphi(x)$ **says** ψ

A.1.2. Definitions and Lemmas

Definition A.1 (Consistent and Complete Theory). Suppose we have a theory (Δ, Θ) of sentences[15].

- (Δ, Θ) is consistent, if we do not have for some $\alpha_i \in \Delta$, $\beta_j \in \Theta$

$$\vdash \bigwedge_i \alpha_i \rightarrow \bigvee_j \beta_j$$

- (Δ, Θ) is complete in the language with variables $\mathcal{V}$ iff for all ψ in the language, we have

$$\psi \in \Delta \textit{ or } \psi \in \Theta$$

Definition A.2 (Saturated Theory). A theory (Δ, Θ) is saturated in a language with variables $\mathcal{V}$ iff the following holds:

1. (Δ, Θ) is consistent
2. $\exists x A(x) \in \Delta$, then for some $y \in \mathcal{V}$, $A(y) \in \Delta$.
3. $\forall x A(x) \notin \Delta$, then for some $y \in \mathcal{V}$, $A(y) \notin \Delta$
4. $A \vee B \in \Delta$ iff $A \in \Delta$ or $B \in \Delta$.
5. If for some $\beta_j \in \Theta$

$$\Delta \vdash A \vee \beta_j \Rightarrow A \in \Delta$$

   with A in the language with variables $\mathcal{V}$.

[14] y not free in ψ or φ.

[15] intuitively, Δ is the set of formulas that are true in the model and Θ is the set of formulas that are false in the model.

Definition A.3 (Constant Domain Theory). A theory (Δ, Θ) is said to be constant domain (CD) theory in language $\mathcal{V}$ iff for any $\forall x A(x)$ and any $\beta_j \in \Theta$ such that

$$\Delta \nvdash \forall x A(x) \vee \bigvee_j \beta_j$$

then for some y

$$\Delta \nvdash A(y) \vee \bigvee_j \beta_j$$

Lemma A.4. *Assume the CD axiom* $\forall x(\beta \vee A(x) \rightarrow (\beta \vee \forall x A(x)))$*, then if* (Δ, Θ) *is a consistent CD theory and* $\Delta' = \Delta \cup \{\alpha_1, \ldots, \alpha_n\}$*,* $\Theta' = \Theta \cup \{\gamma_1, \ldots, \gamma_m\}$ *and* (Δ', Θ') *is consistent then* (Δ', Θ') *is a CD theory*

Proof. Assume

$$\Delta \cup \bigwedge_i \alpha_i \nvdash (\bigvee_j \beta_j) \vee (\bigvee_j \gamma_j) \vee \forall x A(x)$$

we can assume x not in β_j,α_j, γ_j hence

$$\Delta \nvdash \textstyle\bigwedge_i \alpha_i \rightarrow \forall x(\bigvee_j \beta_j) \vee (\bigvee_j \gamma_j) \vee \forall x A(x)$$
$$\Delta \nvdash \forall x(\textstyle\bigwedge_i \alpha_i \rightarrow (\bigvee_j \beta_j) \vee (\bigvee_j \gamma_j) \vee A(x))$$

hence for some y

$$\Delta \nvdash \bigwedge_i \alpha_i \rightarrow \beta \vee A(y) \vee \gamma_j$$

hence $\Delta' \nvdash \beta \vee A(y) \vee \gamma_j$ ∎

Lemma A.5. *Let* (Δ, Θ) *be a saturated theory. Let* Δ' *be*

$$\{\psi | (\{x\}\varphi(x) \textbf{ says } \psi) \in \Delta\}$$

Assume

$$(\{x\}\varphi(x) \ \textbf{\textit{says}} \ \beta) \in \Theta$$
$$\Delta' \nvdash \beta \vee \forall x A(x)$$

then for some y

$$\Theta \nvdash \beta \vee A(y)$$

Proof. The proof is by contradiction, suppose it is not the case that

$$\Theta \nvdash \beta \vee A(y)$$

then, for each y there exists a finite $\Delta_y^{'} \subseteq \Delta^{'}$ such that

$$\vdash \bigwedge \Delta_y^{'} \rightarrow \beta \vee A(y)$$

hence, with $\alpha \in \Delta_y^{'}$

$$\nvdash \bigwedge \{x\}\varphi(x) \textbf{ says } \alpha \rightarrow \{x\}\varphi \textbf{ says } \beta \vee A(y)$$

hence, for all y

$$\{x\}\varphi \textbf{ says } \beta \vee A(y) \in \Delta$$

Since Δ is saturated we get:

$$\forall y\{x\}\varphi \textbf{ says } (\beta \vee A(y)) \in \Delta$$

hence

$$\{x\}\varphi \textbf{ says } \forall y(\beta \vee A(y)) \in \Delta$$

hence

$$\forall y(\beta \vee A(y)) \in \Delta^{'}$$

but then

$$\beta \vee \forall y A(y) \in \Delta^{'}$$

which is a contradiction. ∎

Lemma A.6. *Let (Δ, Θ) be a consistent CD theory, then (Δ, Θ) can be extended to a saturated theory $(\Delta^{'}, \Theta^{'})$ in the same language with $\Delta \subseteq \Delta^{'}$ and $\Theta \subseteq \Theta^{'}$*

Proof. The proof is by induction on (Δ_n, Θ_n) the theory, let $\Delta_o = \Delta$ and $\Theta_0 = \Theta$.

Assume (Δ_n, Θ_n) is defined, $\Theta_n - \Theta$ is finite and (Δ_n, Θ_n) is CD. Let β_{n+1} be the $(n+1)th$ wff of the language. Then either $(\Delta_n, \Theta_n \cup \beta_{n+1})$ is consistent or is not consistent, if it is consistent let

$$\Delta_{n+1} = \Delta_n$$
$$\Theta_n + 1 = \Theta_n \cup \{\beta\}$$

If it is inconsistent then $(\Delta_n \cup \{\beta\}, \Theta_n)$ must be consistent so let

$$\Delta_{n+1} = \Delta_n \cup \{\beta\}$$
$$\Theta_n + 1 = \Theta_n$$

In any case $(\Delta_{n+1}, \Theta_{n+1})$ is CD.

Now let $(\Delta, \Theta) = \bigcup_n (\Delta_n, \Theta_n)$, this theory is the saturated theory. ∎

DEFINITION A.7. Let S be the set of all complete theories in the predicate language FSL. If the logic is CD then all the theories are in the language with variables $\mathcal{V}$, if the logic is not CD, then assume that each theory leaves us an infinite number of variables from $\mathcal{V}$ not in the theory. We can write (Δ, Θ) as Δ because for a saturated theory (Δ, Θ), we have $\Theta = \{\beta | \Delta \nvdash \beta\}$.

Define two relations on S

1. (set inclusion) $\Delta \subseteq \Delta'$
2. For every $\{x\}\varphi(x)$ let $\Delta R_{\{x\}\varphi(x)} \Delta'$ iff for all ψ such that $\{x\}\varphi \,\mathbf{says}\, \psi \in \Delta$ we have $\psi \in \Delta'$.

LEMMA A.8. *Suppose $\Delta \nvdash \alpha \to \beta$, then for some $\Delta' \supseteq \Delta$, $\Delta' \vdash \alpha$ and $\Delta' \nvdash \beta$.*

PROOF. From hypothesis we have

$$\Delta \cup \{\alpha\} \nvdash \beta$$

and $\Delta \cup \{\alpha\}$ can be completed to be a saturated theory Δ' such that

$$\Delta' \nvdash \beta$$

In case of logic CD, this can be done in the same language with variables $\mathcal{V}$. If the logic is not CD, then since there is an infinite number of variables not in Δ, Δ' can use some of them, still leaving infinitely out of Δ. ■

LEMMA A.9. *Assume $\Delta \nvdash \forall x \varphi(x)$, if the logic is not CD, then for some u not in the language of Δ, we have $\Delta \nvdash \varphi(x)$. Δ can be extended in a saturated Δ' by adding the variable u and more variables such that $\Delta' \nvdash \varphi(u)$, and still infinitely numbers of variables are not in Δ'. If the logic is CD, such a u is in the logic of Δ and $(\Delta, \{\varphi(u)\})$ can be extended to a complete and saturated theory in the same language.*

LEMMA A.10. *Let (Δ, Θ) be complete and saturated. Assume $\{x\}\varphi \,\mathbf{says}\, \psi$ is not in Θ. Then*

$$\Delta_0 = \{\alpha | \{x\}\varphi(x) \,\mathbf{says}\, \alpha \in \Delta\}$$

does not prove ψ, otherwise

$$\vdash \bigwedge \alpha_i \to \psi$$

hence

$$\vdash \bigwedge_i \{x\}\varphi(x) \,\mathbf{says}\, \alpha_i \to \{x\}\varphi(x) \,\mathbf{says}\, \psi$$

hence

$$\{x\}\varphi(x) \textbf{ says } \psi \in \Delta$$

Since Δ_0 *does not prove* ψ*, and* $(\Delta_0, \{\psi\})$ *is consistent, we can extend* Δ_0 *to a saturated theory* (Δ', Θ')*. In case the logic is CD,* (Δ', Θ') *will be in the same language. Otherwise we use more variables.*

LEMMA A.11. *Properties of the model* $(S, \subseteq, R_{\{x\}\varphi})$*:*

1. $\Delta_1 \subseteq \Delta_2$ *and* $\Delta_2 R_{\{x\}\varphi}\Theta$ *then* $\Delta_1 R_{\{x\}\varphi}\Theta$

   PROOF. $\Delta_2 R_{\{x\}\varphi}\Theta$ means for every $\{x\}\varphi \textbf{ says } \psi \in \Delta_2$ we have $\psi \in \Theta$. Since $\Delta_1 \subseteq \Delta_2$ we have for every $\{x\}\varphi \textbf{ says } \psi \in \Delta$ we have $\psi \in \Theta$. ■

2. *If we add the axiom* $\forall x(\varphi(x) \leftrightarrow \varphi'(x)) \rightarrow (\{x\}\varphi \textbf{ says } \psi \leftrightarrow \{x\}\varphi' \textbf{ says } \psi)$ *we get the condition*

   $$\Delta \vdash \forall x(\varphi(x) \leftrightarrow \varphi'(x))$$

   implies for all Θ

   $$\Delta R_{\{x\}\varphi}\Theta \leftrightarrow \Delta R_{\{x\}\varphi'}\Theta$$

DEFINITION A.12 (Construction of the model). Take $(S, \subseteq, R_{\{x\}\varphi(x)})$ as defined above. For atomic $P(x_1, \ldots, x_n)$ and $\Delta \in S$, let

$$\Delta \models P \textit{ iff } P \in \Delta$$

The domain of Δ is defined by the variables of Δ. If the logic is CD all Δ will have variables $\mathcal{V}$ as domain, otherwise we will have variable domains.

LEMMA A.13. *For any* ψ, Δ

$$\Delta \models \psi \text{ iff } \psi \in \Delta$$

PROOF. Proof by taking in exam "$\rightarrow$" and "**says**". ■

References

[1] ABADI, M., 'Access Control in a Core Calculus of Dependency', *Electr. Notes Theor. Comput. Sci.*, 172: 5–31, 2007.

[2] ABADI, M., 'Variations in Access Control Logic', in R. van der Meyden, and L. van der Torre, (eds.), *DEON*, vol. 5076 of *LNCS*, Springer, 2008, pp. 96–109.

[3] ABADI, M., M. BURROWS, B. W. LAMPSON, and G. D. PLOTKIN, 'A Calculus for Access Control in Distributed Systems', in *Adcances in Cryptology (CRYPTO)*, vol. 576 of *LNCS*, Springer, 1991, pp. 1–23.

[4] Abadi, M., and T. Wobber, 'A Logical Account of NGSCB', in D. de Frutos-Escrig, and M. Núñez, (eds.), *Formal Techniques for Networked and Distributed Systems (FORTE)*, vol. 3235 of *LNCS*, Springer, 2004, pp. 1–12.

[5] Barker, S., 'The Next 700 Access Control Models or a Unifying Meta-Model?', *ACM Symposium on Access Control Models and Technologies SACMAT 09 (*to appear*)*.

[6] Bauer, L., M. A. Schneider, Edward W. Felten, and A. W. Appel, 'Access Control on the Web Using Proof-carrying Authorization', in *DARPA Information Survivability Conference and Exposition (DISCEX)*, IEEE Computer Society, 2003, pp. 117–119.

[7] Becker, M. Y., Cédric Fournet, and Andrew D. Gordon, 'Design and Semantics of a Decentralized Authorization Language', in *IEEE Computer Security Foundations Symposium (CSF)*, IEEE Computer Society, 2007, pp. 3–15.

[8] Bertolissi, C., M. Fernández, and S. Barker, 'Dynamic Event-Based Access Control as Term Rewriting', in S. Barker, and G.-J. Ahn, (eds.), *Data and Applications Security (DBSec)*, vol. 4602 of *LNCS*, Springer, 2007, pp. 195–210.

[9] Ceri, S., Georg Gottlob, and Letizia Tanca, 'What you Always Wanted to Know About Datalog (And Never Dared to Ask)', *IEEE Trans. Knowl. Data Eng.*, 1 (1): 146–166, 1989.

[10] Dekker, M. A. C., and Sandro Etalle, 'Audit-Based Access Control for Electronic Health Records', *Electr. Notes Theor. Comput. Sci.*, 168: 221–236, 2007.

[11] Ellison, C., B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, 'SPKI certificate theory', *IETF RFC 2693*, (2009).

[12] Gabbay, D. M., 'Labelled Deductive Systems: Vol. 1', *Oxford University Press*, (1996).

[13] Gabbay, D. M., 'Fibring Logics', *Oxford University Press*, (1999).

[14] Garg, D., and M. Abadi, 'A Modal Deconstruction of Access Control Logics', in *Foundations of Software Science and Computational Structures (FoSSaCS)*, vol. 4962 of *LNCS*, Springer, 2008, pp. 216–230.

[15] Garg, D., L. Bauer, Kevin D. Bowers, F. Pfenning, and M. K. Reiter, 'A Linear Logic of Authorization and Knowledge', in *European Symposium on Research in Computer Security (ESORICS)*, vol. 4189 of *LNCS*, Springer, 2006, pp. 297–312.

[16] Giuri, L., and P. Iglio, 'Role Templates for Content-based Access Control', in *ACM Workshop on Role-Based Access Control*, 1997, pp. 153–159.

[17] Gurevich, Y., and I. Neeman, 'DKAL: Distributed-Knowledge Authorization Language', in *IEEE Computer Security Foundations Symposium (CSF)*, IEEE Computer Society, 2008, pp. 149–162.

[18] Halpern, J. Y., and V. Weissman, 'Using First-Order Logic to Reason about Policies', *ACM Trans. Inf. Syst. Secur.*, 11 (4), 2008.

[19] Kosiyatrakul, T., S. Older, and S.-K. Chin, 'A Modal Logic for Role-Based Access Control', in V. Gorodetsky, I. V. Kotenko, and V. A. Skormin, (eds.), *MMM-ACNS*, vol. 3685 of *LNCS*, Springer, 2005, pp. 179–193.

[20] Lampson, B. W., 'Computer Security in the Real World', *IEEE Computer*, 37 (6): 37–46, 2004.

[21] Lampson, B. W., M. Abadi, M. Burrows, and E. Wobber, 'Authentication in Distributed Systems: Theory and Practice', *ACM Trans. Comput. Syst.*, 10 (4): 265–310, 1992.

[22] Li, N., B. N. Grosof, and J. Feigenbaum, 'Delegation logic: A Logic-based Approach to Distributed Authorization', *ACM Trans. Inf. Syst. Secur.*, 6 (1): 128–171, 2003.

[23] Li, N., and J. C. Mitchell, 'DATALOG with Constraints: A Foundation for Trust Management Languages', in V. Dahl, and P. Wadler, (eds.), *PADL*, vol. 2562 of *LNCS*, Springer, 2003, pp. 58–73.

[24] Lupu, E., and M. Sloman, 'Reconciling Role Based Management and Role Based Access Control', in *ACM Workshop on Role-Based Access Control*, 1997, pp. 135–141.

[25] Wobber, E., M. Abadi, and M. Burrows, 'Authentication in the Taos Operating System', *ACM Trans. Comput. Syst.*, 12 (1): 3–32, 1994.

Guido Boella
Dept. of Computer Science
Università di Torino
C.so Svizzera, 185 - 10149 Torino, Italy
guido@di.unito.it

Dov M. Gabbay
Dept. of Computer Science
King's College London
The Strand, London, WC2A 2LS, UK
and
Dept. of Computer Science
Bar Ilan University
Ramat Gan
Israel
dov.gabbay@kcl.ac.uk

Valerio Genovese
Dept. of Computer Science
Università di Torino
C.So Svizzera, 185 - 10149 Torino, Italy
valerio.click@gmail.com

Leendert van der Torre
Faculty of Sciences, Technology and Communication (FSTC)
University of Luxembourg
6, rue Richard Coudenhove - Kalergi
L-1359 Luxembourg
leon.vandertorre@uni.lu