Permission to Speak: A Logic for Access Control and Conformance☆

Nikhil Dinesh[a], Aravind Joshi[a], Insup Lee[a], Oleg Sokolsky[a]

[a] *Department of Computer Science*
University of Pennsylvania
Philadelphia, PA 19104-6389, USA

Abstract

Formal languages for policy have been developed for access control and conformance checking. In this paper, we describe a formalism that combines features that have been developed for each application. From access control, we adopt the use of a saying operator. From conformance checking, we adopt the use of operators for obligation and permission. The operators are combined using an axiom that permits a principal to speak on behalf of another. The combination yields benefits to both applications. For access control, we overcome the problematic interaction with classical reasoning. For conformance, our formalism accommodates nested obligations and permissions.

The axioms result in a decidable logic, and we characterize its complexity. We integrate the axioms into a logic programming approach, which lets us use quantification in policies while preserving decidability of access control decisions. Conformance checking, in the presence of nested obligations and permissions, is shown to be decidable. Non-interference is characterized using reachability via permitted statements.

1. Introduction

Formal languages for policy have been developed for several applications. In this paper, we consider two applications – (a) access control [1, 2, 3, 4, 5, 6, 7], and (b) conformance checking [8, 9, 10, 11, 12, 13]. We motivate and develop a formalism that combines features that have been examined for each application. The combination yields new features that are useful to both applications. We begin with a brief comparison of prior work on access control and conformance.

Access control is an important problem in trust management systems. Informally, a trust management system involves a set of actors or principals, and

☆This research was supported in part by ONR MURI N00014-07-1-0907, NSF CCF-0429948, and ARO W911NF-05-1-0158.

Email addresses: nikhild@seas.upenn.edu (Nikhil Dinesh), joshi@seas.upenn.edu (Aravind Joshi), lee@seas.upenn.edu (Insup Lee), sokolsky@seas.upenn.edu (Oleg Sokolsky)

a set of controlled or regulated actions, e.g., accessing medical information, or downloading a song. The goal of such a system is to adminstrate requests to perform actions. Trust management systemts are commonly decomposed into two (interacting) components [2]: (a) *authentication* - determining the source of a request, and (b) *access control* - determining whether a request is permitted according to a policy. [1] cast access control as a problem for logic. We assume as given an action (p), which is controlled by a principal (A), and a request to perform p from a principal (B). Access is granted if we can prove that the policy of A *says* that B is *permitted* to perform p. In access control logics, such as [1, 2, 3, 5, 6], *says* is treated as a (modal) operator. However, the concept of *permission* is not explicitly represented.

In conformance, one is interested in checking whether the operations of organizations obey a policy. Example policies include business contracts [8, 10], and health care regulations [9, 11]. Here, we are given a policy and a description of operations (as a state or trace). An organization (A) is conformant if we can prove that for all p, if the policy *says* that A is *required* or *obligated* to do p, then A does p. Conformance is typically assesed using a policy and a state or trace of the organization's operations [11, 12]. In formalisms for conformance, such as [11, 12, 13], there is a distinction between the notions of *obligation* and *permission.* However, the concept of *saying* is not explicitly represented.

A possible explanation for this difference is the kinds of policies that are examined for each application. In access control, we commonly have policies introduced by different principals, and it is important to keep track of what each principal said via her policy. In conformance , there is usually a single policy, and we need to check if the organization does what the policy requires it to do.

In this paper, we describe a formal representation of policy for both access control and conformance. The conceptual motivation is as follows. A policy is created via a *speech-act* of a principal. Policies commonly contain the words *required* and *permitted.* An obvious question is whether the act of speaking is crucial to an understanding of requirement and permission? We hope to convince the reader that it is. A more utilitarian motivation is to examine a class of problems that arise in each application. We propose a formalism that combines *saying* and *permission* to address these problems. Specifically, the problems and our contributions are as follows:

1. For access control, we propose a new decidable axiomatization which accommodates delegation [1, 7] and "speaking for" [1, 3, 6]. Our approach overcomes the problematic interactions with classical reasoning, pointed out by Abadi [2]. "Speaking for" and delegation are obtained as consequences of an axiom that permits a principal to speak on behalf of another.
2. For conformance, the proposed axiomatization is used to reason about nested obligations and permissions. We obtain a partial solution to the problems pointed out by Marcus [14]. Conformance, as the satisfaction of obligations, is shown to be decidable. Previous approaches [11, 12, 13] exclude nested obligations from the syntax of the logic.

In Section 2, we give a detailed motivation for our approach by discussing three constructs that have been examined separately in the literature – (a) representation in access control, under which we include delegation [1, 7] and "speaking for" [1, 2, 6], (b) positive and negative permissions (cf. [15]), and (c) nested obligations and permissions [14], e.g., "required to forbid". Our thesis is that these constructs involve the interaction between *saying* and *permission.*

Section 3 develops a logic in the form of two interacting components. *The inference component* determines what has been said, and involves the choice of appropriate axioms, as in [1, 2, 5]. We introduce two axioms to characterize the interaction between saying and permission. The decidability and complexity of the resulting logic are established. *The saying component* is used to create new utterances, and we adopt a logic programming approach, as in [7, 16, 17]. The modularization lets us integrate the logics of saying with the logic programming approaches, thereby allowing for restricted forms of quantification while preserving decidability of access control and conformance. We also prove a non-interference property which is crucial for the distributed policies that arise in access control.

In Section 4, we discuss our formalism in the context of related work. We consider access control examples, and conformance in the presence of nested obligations and permissions. We also identify some interesting lines for further research. Section 5 concludes.

2. Permission to Speak

In this section, we motivate the explicit use of saying and permission in a formal language for policy. Section 2.1 considers the problem of representation in access control, under which we include delegation [1, 7] and "speaking for" [1, 2, 6]. In Section 2.2, we discuss two kinds of permission – positive and negative (cf. [15]). Positive permission is the only kind that is used in access control. Finally, we examine nested obligations and permissions [14], and their relationship to the two kinds of permission (Section 2.3).

2.1. Representation in Access Control

While there are a wide variety of access control logics, one commonality that stands out is a notion of *saying* [2]. We can express the fact that a principal makes a statement. We use $\mathrm{says}_{l(A)}\varphi$ to denote that principal A says φ in the set of laws $l(A)$. Informally, a law is understood as a single statement in the policy of a principal, e.g., a hospital says "Alice is permitted to access her health information", in its policy. And, the interpretation of a set of laws is the conjunction of the individual laws. These intuitions are formalized in Section 3. Our approach differs from others in that we associate statements to a principal via a set of laws ($\mathrm{says}_{l(A)}\,\varphi$) rather than directly with the principal ($\mathrm{says}_A\,\varphi$). This indirection lets us use *saying* to reason about exceptions to laws, as in [11], and we will discuss an example in Section 3.2.

All access control logics give a principal the ability to let another principal make statements on her behalf. We use the term *representation* to describe such constructs. As an example (from [6]), consider a file access scenario, where an administrator (A) trusts Bob (B) to decide whether a file is to be deleted (del). In this scenario, we say that B represents A on del, and we wish to conclude that if $\mathrm{says}_{l(B)}(\mathrm{del})$, then $\mathrm{says}_{l(A)}(\mathrm{del})$.

A naive approach to representation is to introduce "$\mathrm{says}_{l(B)}(\mathrm{del}) \Rightarrow \mathrm{says}_{l(A)}(\mathrm{del})$" into A's policy (where $\Rightarrow$ is the implication connective of the underlying logic). However, such statements create an access control risk, because "$\mathrm{says}_{l(B)}(\mathrm{del}) \Rightarrow \mathrm{says}_{l(A)}(\mathrm{del})$" could be introduced by B, thereby giving B the ability to decide whether any file is to be deleted.

To address this security risk, a principal A is only allowed to introduce statements of the form $\mathrm{says}_{l(A)}\, \psi$. Additional machinery (usually an axiom) is needed to accommodate representation. Abadi [2] discusses several alternatives, involving variants of the hand-off axiom:

- $\mathrm{says}_{l(A)}(\phi \Rightarrow \mathrm{says}_{l(A)}\, \psi) \Rightarrow (\phi \Rightarrow \mathrm{says}_{l(A)}\, \psi)$

B represents A on del is expressed as:

- $\mathrm{says}_{l(A)}(\mathrm{says}_{l(B)}(\mathrm{del}) \Rightarrow \mathrm{says}_{l(A)}(\mathrm{del}))$

The hand-off axiom lets us conclude that $\mathrm{says}_{l(B)}(\mathrm{del}) \Rightarrow \mathrm{says}_{l(A)}(\mathrm{del})$. However, the hand-off axiom has displeasing consequences in classical logics. For example, $\mathrm{says}_{l(B)}\, \varphi \Rightarrow (\neg\varphi \Rightarrow \mathrm{says}_{l(B)}\, \psi)$ (for all ψ) is provable [2], i.e., if a statement by B fails, then B gives access to all the actions that she controls. The solution to this problem has been to move to an intuitionistic setting, as in [3, 5, 6].

We suggest that the problem is not with classical reasoning, but with the hand-off axiom. The key idea is to reformulate the axiom using the interaction between *saying* and *permission*. We now introduce the reformulated version of the axiom, followed by a discussion of its benefits.

We say that B represents A on del, if A says that B is *permitted* to say del. More formally, the statement $\mathrm{says}_{l(A)}(\mathcal{P}_B(\mathrm{says}_{l(B)}\, \mathrm{del}))$ is added to A's policy, where $\mathcal{P}_B(\mathrm{says}_{l(B)}\, \mathrm{del})$ is read as "B is permitted to say del". The following are equivalent versions of *the axiom of representation*:

- If A says that B is permitted to say φ, then if B says φ, A says φ
- $\mathrm{says}_{l(A)}(\mathcal{P}_B(\mathrm{says}_{l(B)}\varphi)) \Rightarrow (\mathrm{says}_{l(B)}\, \varphi \Rightarrow \mathrm{says}_{l(A)}\, \varphi)$

The axiom of representation is intended for a particular sense of speaking/saying, i.e., *speaking on someone's behalf.* This sense of saying is the usual one in access control. To simplify matters, we do not explicitly represent the principal on behalf of whom a statement is being made.

"Speaking for" [1, 3, 6] is a case of representation when one principal represents another on all statements. If B speaks for A, we wish to conclude $\mathrm{says}_{l(B)}\, \varphi \Rightarrow \mathrm{says}_{l(A)}\, \varphi$ for all φ. "Speaking for" has a compelling definition in our approach. We say that B speaks for A if A permits B to say anything ($\perp$) on her behalf, i.e., $\mathrm{says}_{l(A)}\, \mathcal{P}_B(\mathrm{says}_{l(B)}\, \perp)$.

A novelty in our approach is that "speaking for" and hand-off are both obtained as a consequence of the axiom of representation. In [1, 3, 6], "speaking for" and hand-off are not related, i.e., the former involves an algebra over principals or second-order quantification, and the latter is obtained using an axiom (which implies hand-off). This suggests that the representation axiom is quite different from the hand-off axiom. It is tempting to relate the representation axiom to a restricted version of hand-off:

- $\mathrm{says}_{l(A)}(\mathrm{says}_{l(B)}\,\varphi \Rightarrow \mathrm{says}_{l(A)}\,\varphi) \Rightarrow (\mathrm{says}_{l(B)}\,\varphi \Rightarrow \mathrm{says}_{l(A)}\,\varphi)$

However, even for this restricted case, we do not know of a complete semantics for hand-off, which makes it difficult to show that a statement is not provable (Abadi et al [1] observe similar difficulties). We believe that the representation axiom is a persuasive alternative to hand-off, because it yields a decidable logic with a complete semantics, and more importantly, it has an intuitive interpretation.

A restricted version of the axiom of representation has been proposed by Becker et al [18], in the context of the authorization language SECPal. In SECPal, representation is restricted to atomic predicates, and hence, "speaking for" cannot be accomodated. Moreover, the relationship between permission and obligation is not explored. Our formalism generalizes SECPal, to accomodate both "speaking for" and obligation. We now discuss further motivation for our approach.

2.2. Positive and Negative Permission

We take, as a starting point, the definition of permission as the dual of obligation, i.e., $\mathcal{P}_A\varphi = \neg\mathcal{O}_A\neg\varphi$ ($\mathcal{O}_A\varphi$ is read a A is obligated to bring about φ). This definition of permissions has been observed to be inadequate, and theories have argued for further distinctions (cf. [15]). The most common distinction is between positive and negative permission. Suppose *a hospital (H) permits a principal (A) to access her health information.* Consider the following questions:

1. Does H permit A to access her information?
2. Does H permit A to listen to music?

The answer to Question 1 would be yes. However, matters are not so clear for Question 2. We follow the analysis of Makinson and van der Torre [19]. In one sense (positive permission), the answer is "no", because H has not explicitly permitted A to listen to music. In another sense (negative permission), the answer is "yes", because H has not forbidden A from doing so. Makinson and van der Torre [19] distinguish between the two senses in the meta-logic, by using operations on the consequences of a sentence.

In our approach, the two kinds of permission are distinguished by varying the scope of negation. The permission given by H is represented as: $\varphi = \mathrm{says}_{l(H)}(\mathcal{P}_A\mathrm{access})$, i.e., H *says* that A is permitted to access her information. Question 2 can be formulated in two ways:

1. Is $\varphi \Rightarrow \text{says}_{l(H)}(\mathcal{P}_A\text{music})$ provable? Where, music denotes that A listens to music. Equivalently, is $\varphi \Rightarrow \text{says}_{l(H)}(\neg\mathcal{O}_A\neg\text{music})$ provable? No. Listening to music is not positively permitted.
2. Is $\varphi \Rightarrow \text{says}_{l(H)}(\mathcal{O}_A\neg\text{music})$ *not* provable? Yes. Listening to music is negatively permitted.

In Section 3, we use the formalism in [11] to reason about provability and its negation. The negation of provability is needed to express *didn't say.* In other words, H didn't say φ iff $\text{says}_{l(H)}\varphi$ is not provable from H's statements. The discussion in [15, 19] suggests to us that the relationship between negative permission and *didn't say* is known, but to our knowledge, an explicit representation of saying has not been carried out in a logic of obligation and permission.

2.3. Nested Obligations and Permissions

Marcus [14] pointed out a problem in formalizing nested obligations and permissions. We relate it to the distinction between the two senses of permission. Consider the following statement: "A should not allow her child (B) to play near the road". Which sense of permission is appropriate here? Using positive permission, we get "A should not explicitly permit B to play", which is inadequate. Negative permission is appropriate here – "A should not *not require B not to play*", i.e., "A should require B not to play". To accomodate such reasoning, the two kinds of permission need to be distinguished in the syntax of the logic. Previous approaches to conformance [11, 12, 13] do not provide for this distinction, and it is no surprise that nested obligations and permissions are excluded. [9] discusses policies where nesting is used frequently, but again, a formal characterization is not provided.

We now relate representation and nested permissions, to emphasize that *saying* is crucial to the analysis. Consider the following statement: "A hospital (H) permits a patient (A) to permit her mother (B) to access her information". We will rephrase the permission as follows: H says that A is *permitted to say* that B is permitted to access her information. Formally, this is represented as $\text{says}_{l(H)}(\mathcal{P}_A(\text{says}_{l(A)}(\mathcal{P}_B\text{access})))$. If A does indeed permit access to her mother ($\text{says}_{l(A)}(\mathcal{P}_B\text{access})$), we will conclude $\text{says}_{l(H)}(\mathcal{P}_B\text{access})$ using the axiom of representation, i.e., H permits access to B. As a result, nested permissions are related to representation, i.e., "H permits A to permit B to do φ" iff "A represents H in permitting B to do φ".

Belnap and Bartha [20] provide a solution to nested obligations and permissions in the context of *seeing to it that* (stit) logic. While stit logics offer a general solution, the concept of stit is an abstract one (to cover all applications). We analyze "A should require B not to play" to mean "A should *say* that B is required not to play", and declare B to be non-conforming if she does play. In stit logic, the sentence is analysed as "A should *stit* B does not play", and it may necessitate a stronger action, e.g., physically preventing B from playing near the road. However, axiomatizing the implications of stit is a challenging problem in practice. We focus on the problem of determining what a principal

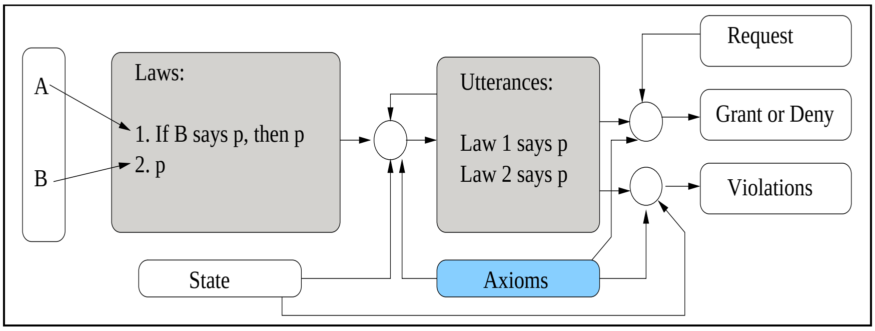

Figure 1: Interaction between the components of the logic

says that she requires or permits via policy, and discuss how stit-like notions can be added to our system in Section 4.2.

3. A Logic for Access Control and Conformance

In this section, we develop a logic in the form of two interacting components – (a) the inference component, which involves the choice of appropriate axioms, and (b) the saying component, which is used to represent policies. Figure 1 shows the interaction between the components of the access control system. There are two kinds of actions of interest – (1) operational acts, e.g., downloading a song, and (2) speech acts. The operational acts are described using a state, which contains the interpretation of predicates, and the speech acts are described using laws.

A principal speaks by introducing laws. In Figure 1, the principals A and B introduce the laws 1 and 2 respectively. The laws are evaluated using the axioms to produce a set of *utterances*, i.e., what the laws say. To determine what a principal says, we look at what her laws say, e.g., B says p iff "Law 2 says p" is provable from the utterances using the axioms. A set of laws can be thought of as a logic program, and utterances as the extensions that result from the program (via a fixed point computation). Once we have the utterances, there are several decision problems of interest. *The access control problem* is to decide whether a request is permitted by the set of utterances. *The conformance problem* is to decide whether operational and speech acts satisfy the obligations imposed by the utterances, and if they do not, violations are reported.

Section 3.1 is an overview of the inference component. We describe (axiomatically) a logic with two modalities – *saying* and *obligation*. In Section 3.2, we adapt a formalism from our prior work [11] for the saying component. We describe the evaluation of polices, using an example. We extend [11] in two ways. First, we prove a non-interference property which is crucial for the distributed policies that arise in access control (Section 3.3). Second, we show that conformance, in the presence of nested obligations and permissions, is decidable (Section 3.4).

3.1. The Inference Component – Axioms

In this section, we develop a predicate logic with two modalities *saying* and *obligation*. We allow formulas with free variables, but no quantifier over objects. The quantification over objects is carried out in the process of saying (Section 3.2), which uses provability in the propositional subset of the language defined here. We begin by defining the syntax:

Definition 1 (Syntax)**.** *Given sets $\Phi_1, ..., \Phi_n$ (of predicate names), countable sets of variables X, object names O, a finite set of identifiers ID, and a function $l : O \to 2^{Id}$, the language $L(\Phi_1, ..., \Phi_n, X, O, l, ID)$, abbreviated as L, is defined as follows:*

$$\begin{aligned} \varphi &::= \alpha \mid \varphi \wedge \varphi \mid \neg\varphi \mid \text{says}_{Id}\, \psi \mid \text{says}_{l(y)}\, \psi \\ \psi &::= \varphi \mid \psi \wedge \psi \mid \neg\psi \mid \mathcal{O}_y\varphi \end{aligned}$$

where, $y \in X \cup O$, and α generates atomic predicates of the form $p(y_1, ..., y_j)$ with $p \in \Phi_j$ and $(y_1, ..., y_j) \in (X \cup O)^j$. We assume that $X \cap O = \emptyset$. The set of formulas generated by each BNF rule are referred to as L_φ and L_ψ respectively, and $L = L_\varphi \cup L_\psi$.

Disjunction $\varphi \vee \psi = \neg(\neg\varphi \wedge \neg\psi)$ and implication $\varphi \Rightarrow \psi = \neg\varphi \vee \psi$ are derived connectives. $\mathcal{O}_y\varphi$ is read as "φ is obligated of y". Permission is defined as the dual of obligation, i.e., $\mathcal{P}_y\varphi = \neg\mathcal{O}_y\neg\varphi$. The saying operators are understood as follows. Principals speak by introducing identified laws (Section 3.2), and we are interested in determining what a set of laws say. $\text{says}_{Id}\, \varphi$ is read as "φ is said in the laws Id", and $\text{says}_{l(y)}\, \varphi$ is read as "y says φ in the laws $l(y)$". $l(A)$ is the set of laws introduced by the principal $A \in O$.

We now mention a peculiarity of Definition 1. The BNF rules ensure the alternation of *obligation* and *saying* modalities, e.g., $\mathcal{O}_y\, \text{says}_{l(y)}\, \mathcal{O}_z\varphi \in L$, but $\mathcal{O}_y\mathcal{O}_z\varphi \notin L$. Following von Wright [21], we understand obligations as applying to actions and their consequences. The language L_φ (obtained from the first BNF rule) is used to describe actions – (a) atomic actions, (b) combinations of actions (using connectives), or (c) *saying*, which is (a consequence of) a speech act. An obligation is an opinion, which is created via a speech act, but is not an act by itself.

The statements in L will be used in *the inference component* of access control, i.e., to determine what has been said. In other words, we will be given a set of utterances U and a question ψ, and we need to determine whether $U \Rightarrow \psi$ is *provable*. We focus on provability for the propositional subset of L, i.e., without variables and function applications. The propositional subset of L has the modalities $\text{says}_{Id}\, \varphi$ (for all $Id \subseteq ID$), and $\mathcal{O}_A(\varphi)$ (for all $A \in O$).

We adopt the axiomatization in Figure 2. **A1** and **R1** give us propositional reasoning. **A2** and **R2** are common to both saying and obligation. **A3** and **A4** are specific to saying and obligation respectively. Finally, **A5** and **A6** describe the interaction between the two modalities. We will now discuss the axioms in the context of related work.

A1 All substitution instances of propositional tautologies.

A2 $\mathcal{Q}(\varphi \Rightarrow \psi) \Rightarrow (\mathcal{Q}(\varphi) \Rightarrow \mathcal{Q}(\psi))$ (for all modalities $\mathcal{Q}$)

A3 $\text{says}_{Id}\, \varphi \Rightarrow \text{says}_{Id'}\, \varphi$ (for all $Id \subseteq Id'$)

A4 $\mathcal{O}_A\varphi \Rightarrow \mathcal{P}_A\varphi$ (for all $A \in O$)

A5 $\text{says}_{Id_A}(\mathcal{P}_B\, \text{says}_{Id_B}\, \varphi) \Rightarrow (\text{says}_{Id_B}\, \varphi \Rightarrow \text{says}_{Id_A}\, \varphi)$ (for all $\{A, B\} \subseteq O$, $Id_A \subseteq l(A)$, and $Id_B \subseteq l(B)$)

A6 $\text{says}_{Id_A}(\mathcal{P}_B\, \text{says}_{Id_A}\, \varphi) \Rightarrow \text{says}_{Id_A}\, \varphi$ (for all $\{A, B\} \subseteq O$, and $Id_A \subseteq l(A)$)

R1 From $\vdash \varphi \Rightarrow \psi$ and $\vdash \varphi$, infer $\vdash \psi$

R2 From $\vdash \varphi$, infer $\vdash \mathcal{Q}(\varphi)$ (for all modalities $\mathcal{Q}$)

Figure 2: Axiomatization of the propositional fragment of L. The set of modalities $\mathcal{Q}$ consists of $\text{says}_{Id}\, \varphi$ (for all $Id \subseteq ID$) and $\mathcal{O}_A\varphi$ (for all $A \in O$).

Axioms for Saying: The axioms **A1** and **A2**, together with the rules **R1** and **R2**, gives us the modal logic **K**. The **K** axiomatization was used by Abadi et al [1] as a basis for all (classical) access control logics. Further motivation comes from our prior work [11]. In Section 3.2, following [11], we describe policies by evaluating $\text{says}_{Id}\, \varphi$ using provability. Given a set of formulas U, $\text{says}_{Id}\, \varphi$ is true w.r.t. U iff $\bigwedge U \Rightarrow \text{says}_{Id}\, \varphi$ is provable. The **K** axiomatization is sound w.r.t. this definition.

A3 says that if φ is said by the statements (Id), then φ also holds according to a larger set of statements (Id'). This axiom is also sound w.r.t. [11]. **A3** is also sound w.r.t. the principal algebra of [1, 6]. The formula "$A \vee B$ says φ" in [1, 6] corresponds to $\text{says}_{l(A)\, \cup\, l(B)}\, \varphi$ here. A complete characterization of the algebra needs additional axioms, e.g., $(\text{says}_{Id_1}\, \varphi \wedge \text{says}_{Id_2}\, \varphi) \Rightarrow \text{says}_{Id_1 \cap Id_2}\, \varphi$ (the converse is provable using **A3**). We omit these axioms because they are not sound w.r.t. [11]. From a technical standpoint, the axioms can be easily accommodated in the system here.

Several other axiomatizations have been proposed in the literature (c.f. [2]). The **K** axiomatization is a minimal set, which is common to all (classical) systems. We discuss the adaptation of our results to other systems of saying in Section 5.

Obligation and Its Interaction with Saying: The **K** axiomatization, together with **A4**, gives us the the modal logic **KD**. This axiomatization is common to many systems, giving it the name Standard Deontic Logic (c.f. [13]).

The main focus of this work is on the interaction between *saying* and *permission.* We characterize the interaction with two axioms. *The representation axiom*, **A5**, is read as "If A says that B is permitted to say φ, and B says φ, then A says φ". We remind the reader that a principal speaks by introducing identified laws. As we discussed in Section 2.1, **A5** is needed to accommodate notions of representation in access control. *The self-respecting axiom*, **A6**, is read as "If A permits B to say φ using A's laws, then A says φ". **A6** ensures that statements in $l(B)$ do not (inadvertently) interfere with the consequences of statements in $l(A)$. We dicuss an example in Proposition 1 (items 4 and 5).

Provability: The process of saying (Section 3.2) relies on provability in the language L. We say that φ is provable (denoted $\vdash \varphi$), if φ is an instance of the axioms **A1-A6** or follows from the axioms using the rules **R1** and **R2**. In Appendix A, we provide a Kripke semantics for which the axiomatization is sound and complete (Appendix B). As in [6], semantics is used to show that a statement is not provable. The following are interesting provable and non-provable statements:

Proposition 1. *The following are provable/not provable:*

1. $\vdash \mathsf{says}_{l(A)}(\mathcal{O}_B \, \mathsf{says}_{l(B)} \, \varphi) \Rightarrow (\mathsf{says}_{l(B)} \, \varphi \Rightarrow \mathsf{says}_{l(A)} \, \varphi)$
2. $\vdash \mathsf{says}_{l(A)}(\mathcal{P}_B \, \mathsf{says}_{l(B)} \perp) \Rightarrow (\mathsf{says}_{l(B)} \, \varphi \Rightarrow \mathsf{says}_{(A)} \, \varphi)$
3. $\nvdash \mathsf{says}_{l(A)}(\mathcal{P}_B \, \mathsf{says}_{l(B)} \, p(o_1)) \Rightarrow (\mathsf{says}_{l(B)} \perp \Rightarrow \mathsf{says}_{l(A)} \perp)$*, if* $l(B) \not\subseteq l(A)$
4. $\vdash \mathsf{says}_{l(A)}(p(o_1) \wedge \mathcal{P}_B \, \mathsf{says}_{\emptyset} \neg p(o_1)) \Rightarrow \mathsf{says}_{l(A)} \perp$
5. $\vdash \mathsf{says}_{l(A)}(p(o_1) \wedge \mathcal{P}_B \, \mathsf{says}_{\emptyset} \neg p(o_1)) \Rightarrow (\mathsf{says}_{l(B)} \neg p(o_1) \Rightarrow \mathsf{says}_{l(A)} \perp)$

PROOF. Items 1 and 2 follow easily from **A5**, **A2** and **A4**. Item 1 gives representation via obligation. Item 2 gives us *speaking for*, i.e., B speaks for A. Item 3 ensures that if one is conservative about the permissions that are granted, the principal receiving the permission cannot take advantage. Non-provability is shown by constructing a model which satisfies the negation.

Items 4 and 5 are consequences of **A6**. It can be shown that without **A6**, item 4 is not provable, but item 5 *is still provable.* This lets one use the empty set in a dangerous way. $\mathsf{says}_{l(A)}(p(o_1) \wedge \mathcal{P}_B \, \mathsf{says}_{\emptyset} \neg p(o_1))$ does not mention $l(B)$ explicitly, but taken together with $\mathsf{says}_{l(B)} \neg p(o_1)$, we can ascribe anything to $l(A)$. The question is which formula do we blame for this danger? **A6** places the blame on $\mathsf{says}_{l(A)}(p(o_1) \wedge \mathcal{P}_B \, \mathsf{says}_{\emptyset} \neg p(o_1))$, by making item 4 provable. □

For subsequent development, we will need the decidability of the provability question:

Theorem 1 (Decidability)**.** *Given $\varphi \in L$ which is propositional: $\vdash \varphi$ is decidable*

PROOF. In Appendix C, decidability is established via the finite model property. We briefly discuss the complexity (proofs are omitted). The complexity of satisfiability testing is NEXPTIME-complete, i.e., complete for non-deterministic exponential time. The axioms **A5** and **A6** introduce dependencies between sibling states in the Kripke structures (**C5** and **C6** in Appendix A) which leads to an increase from the usual PSPACE bound. The following modified versions of **A5** and **A6** yield a PSPACE-complete logic:

A5' $\text{says}_{Id_B}\,\varphi \Rightarrow \text{says}_{Id_A}((\mathcal{P}_B\,\text{says}_{Id_B}\,\varphi) \Rightarrow \varphi)$

A6' $\text{says}_{Id_A}((\mathcal{P}_B\,\text{says}_{Id_A}\,\varphi) \Rightarrow \varphi)$

We say that φ is provable with the new axioms (denoted $\vdash_1 \varphi$)) iff it follows from the axioms **A1-A4** together with **A5'** and **A6'** and the rules **R1** and **R2**. It is easy to show that if $\vdash \varphi$, then $\vdash_1 \varphi$, i.e., **A5** and **A6** are derived. However, the converse is not true. Consider, for example, $\varphi = \text{says}_{Id_A}(q \Rightarrow (\mathcal{P}_A\,\text{says}_{Id_A}\,\neg q)) \Rightarrow \text{says}_{Id_A}\,\neg q$. We can show that $\vdash_1 \varphi$ and $\not\vdash \varphi$. The question of interest is whether the additional validities introduced by $\vdash_1$ are appropriate for access control. We do not have motivation against it, and the computational benefits seem worthwhile. □

In the following section, we will use provability (and its negation) to describe the process of saying.

3.2. The Saying Component - Policies

In this section, we describe the representation and evaluation of policies or regulations. The result of evaluating regulation is a set of utterances, which forms the basis for access control and conformance. The formalism developed here is an extension of our prior work [11], and is a generalized form of logic programming. Logic programs are popular in representing regulatory texts [22, 23, 13], and access control policies [7, 16, 17]. We begin by defining the syntax of regulations:

Definition 2 (Syntax of Regulation)**.** *Given a finite set of identifiers ID, a body of regulation Reg is a set of statements such that for each $id \in ID$, there exist $\varphi \in L_\varphi$ and $\psi \in L_\psi$ such that:* (id) $\varphi \mapsto \psi \in Reg$

(id) $\varphi \mapsto \psi$ is read as: "the precondition φ leads to the postcondition ψ".
Example: We will describe the evaluation of regulation using an example from [11], which is from the Food and Drug Administration's Code of Federal Regulations (FDA CFR). The CFR governs the operations of American bloodbanks. Bloodbanks are organizations which collect, test and ultimately distribute donations of blood to their end recepients. Given a description of a bloodbank's operations as an abstract state, we will check if these operations conform to the CFR. Consider the following statements, which are based on CFR 610.40:

(1) Except as specified in (2), every donation of blood or blood component must be tested for evidence of infection due to Hepatitis B.

(2) You are not required to test donations of source plasma for evidence of infection due to Hepatitis B.

Statement (1) conveys an obligation to test donations of blood or blood component for Hepatitis B, and (2) conveys a permission not to test a donation of source plasma (a blood component) for Hepatitis B. To assess an organization's conformance to (1) and (2), it suffices to check whether "All non-source plasma donations are tested for Hepatitis B". In other words, (1) and (2) imply the following obligation:

(3) Every non-source plasma donation must be tested for Hepatitis B.

Sentences in regulation can have several exceptions, and manually creating derived obligations is difficult.In [11], we developed a logic in which sentences refer to others, thereby letting us represent statements (1) and (2), and recovering the derived obligation during evaluation. We note that the presence of obligation makes this example different from those that are typically considered in access control. However, we use it to facilitate comparison with [11], and to motivate the definition of conformance. We will consider access control examples in Section 4.1.

Representing Regulations and Organizations: The statements (1) and (2) are represented as follows:

- (1) $bb(u) \wedge d(x,u) \wedge \neg \, \text{says}_{\{2\}}(\neg \mathcal{O}_u test(x,u)) \mapsto \mathcal{O}_u test(x,u)$, and
- (2) $bb(z) \wedge d(y,z) \wedge sp(y) \mapsto \neg \mathcal{O}_z test(y,z)$

The predicates are understood as follows. $bb(u)$ is true iff u is a bloodbank, $d(x,u)$ is true iff x is a donation collected by u, $sp(y)$ is true iff y consists of source plasma, and $test(x,u)$ is true iff x is tested for Hepatitis B by u. In the obligation, the subformula $\text{says}_{\{2\}}(\neg \mathcal{O}_u test(x,u))$ is understood as "u is not obligated to test x according to statement (2)".

Regulatory statements are evaluated w.r.t. states (representing and organization) and assignments. If the precondition of a statement is true, the postcondition is *uttered* (under substitution).

Objs	Predicates	Utterances
A	$bb(A)$, $d(o_1,A)$, $d(o_2,A)$	
o_1	$sp(o_1)$, $test(o_1,A)$	$\text{says}_{\{2\}}(\neg \mathcal{O}_A test(o_1,A))$
o_2	$\neg sp(o_2)$, $\neg test(o_2,A)$	$\text{says}_{\{1\}}(\mathcal{O}_A test(o_2,A))$

Table 1: A state and its utterances

Table 1 shows a state of a bloodbank augmented with utterances. There are three objects – A is a bloodbank, o_1 is a donation of source plasma, and o_2 is a non-source plasma donation. Note that A does not conform with the regulation, since o_2 is not tested (we will define conformance at the end of this section). We begin by defining states and assignments:

Definition 3 (States and Assignments)**.** *Given countable sets O of object names, and predicate names $\Phi_1, \ldots, \Phi_n$, a state $S(O, \Phi_1, ..., \Phi_n)$, abbreviated S, is the tuple $(I_{\Phi_1}, \ldots, I_{\Phi_n})$ where $I_{\Phi_j} : \Phi_j \to 2^{O^j}$ is the interpretation of predicates of arity j. Given $p \in \Phi_j$, we will say that $p(o_1, ..., o_j)$ is true at state iff $(o_1, ..., o_j) \in I_{\Phi_j}(p)$.*

Given a set of variables X, an assignment is a function $v : X \to O$. The set of all assignments is denoted by $V(X, O)$, abbreviated V.

Given $v \in V$ and $\varphi \in L$, $v(\varphi)$ is the formula obtained by replacing all variables x occuring in φ with $v(x)$, and laws $l(x)$ with $l(v(x))$. We assume that all variables are free. Note that $v(\varphi)$ is a propositional formula in L.

Evaluating the Example: We now describe the evaluation of our example statements. Given the state S (in Table 1), first we consider the permission:

- (2) $bb(z) \wedge d(y,z) \wedge sp(y) \mapsto \neg\mathcal{O}_z test(y,z)$
- If the precondition $bb(z) \wedge d(y,z) \wedge sp(y)$ is true w.r.t. an assignment $v \in V$, then the postcondition $v(\text{says}_{\{2\}}(\neg\mathcal{O}_z test(y,z)))$ is *uttered*.
- Otherwise, there is no utterance.

Since the precondition of statement (2) is true for the assignment of z to A and y to o_1, we have the utterance $\text{says}_{\{2\}}(\neg\mathcal{O}_A test(o_1, A))$. However, since o_2 is not a donation of source plasma, there is no correponding utterance.

Now consider the formula $\varphi = \text{says}_{\{2\}}(\neg\mathcal{O}_u test(x,u))$ in the precondition of (1). This is evaluated as follows. First, we evaluate the permission (as describe above) at S w.r.t. all assignments. Let U be the set of utterances obtained. Then, φ is *true* at S w.r.t. an assignment v iff $U \vdash v(\varphi)$. We say that U entails $v(\varphi)$ (denoted $U \vdash v(\varphi)$) iff there is a finite subset $U' \subseteq U$ such that $\vdash \bigwedge U' \Rightarrow v(\varphi)$. We note that the propositional fragment of L is compact, and so provability and finite provability are identical.

Returning to Table 1:

- $\text{says}_{\{2\}}(\neg\mathcal{O}_A test(o_1, A))$ is uttered, and
- $\{\text{says}_{\{2\}}(\neg\mathcal{O}_A test(o_1, A))\} \vdash \text{says}_{\{2\}} \neg\mathcal{O}_A test(o_1, A)$, since $\varphi \Rightarrow \varphi$ is a propositional tautology. So $\text{says}_{\{2\}} \neg\mathcal{O}_u test(x,u)$ is true at S w.r.t the assignment v, when $v(u) = A$ and $v(x) = o_1$.

Statement (1) is evaluated by uttering $v(\text{says}_{\{1\}}\, \mathcal{O}_u test(x))$ if the precondition holds. In Table 1, this results in the utterance $\text{says}_{\{1\}}(\mathcal{O}_A test(o_2, A))$. The utterance lets a law which depends on (1) draw the correct inference.

Formal Definitions of Evaluation: The evaluation is formalized using the Kripke-Kleene-Fitting semantics for logic programs (c.f. [11]). We briefly review the definitions, and refer the reader to [11] for a detailed discussion. The semantic evaluation outlined above works only when there are no cyclic dependencies, since an order of evaluation needs to be defined. To handle cycles, a three-valued interpretation is needed, where the third (middle) value stands for ungrounded. Initially, all statements are ungrounded, and there are no utterances. At each step we assign truth values and utterances, using truth values and utterances from the previous step, until we reach a fixed point. The values are denoted by $\mathcal{B}^3 = \{\top, \bot, ?\}$. We now define utterances (called annotations in [11]):

Definition 4 (Utterances)**.** *Given a state S, assignment $v \in V$, and regulation Reg, an utterance is a statement $v(\text{says}_{\{id\}}\, \psi, S)$ such that $id \in ID$ and* (id) $\varphi \mapsto \psi \in Reg$*. The set of all utterances is denoted by $\mathcal{U}(S, V, Reg)$, abbreviated $\mathcal{U}$.*

To evaluate statements, we use two sets of utterances U and U' such that $U \subseteq U'$. Informally, U is the set of utterances obtained from laws with true preconditions, while U' is set of utterances from laws with true or ungrounded preconditions. The truth of $\text{says}_{Id}\, \varphi$ is determined using U, and falsity is determined using U'. We now define the function **tv** which assigns truth values to preconditions:

Definition 5 (Evaluating Preconditions)**.** *Given utterances U and U' such that $U \subseteq U'$, the function $\mathbf{tv}_{(U,U')} : L_\varphi \times \mathcal{S} \times V \to \mathcal{B}^3$ is defined as follows:*

Predicates are evaluated to true or false. Conjunction and negation are handled using the Kleene semantics.

$$\mathbf{tv}_{(U,U')}(\text{says}_{Id}\, \psi, S, v) = \begin{cases} \top \textit{ if } U \vdash v(\text{says}_{Id}\, \psi) \\ \bot \textit{ if } U' \not\vdash v(\text{says}_{Id}\, \psi) \\ ? \textit{ otherwise} \end{cases}$$

In the spirit of Jorgensen and input-output logic (cf. [24]), we do not assign truth values to obligations and permissions. An obligation or permission can only be *uttered* at a state. In other words, we can evaluate whether "φ is obligated by a set of laws", but "φ is obligated" has no truth value.

We now define consistency for the pair of utterances (U, U'), used in Definition 5. We need to ensure that U (resply. U') corresponds to laws with true (resply. true or ungrounded) preconditions:

Definition 6 (Consistent Utterances)**.** *Given a body of regulation Reg and a state S, the utterence pair (U, U') is consistent iff for all* (id) $\varphi \mapsto \psi \in$ *Reg and $v \in V$:*

If $v(\text{says}_{\{id\}}\, \psi, S) \in U$, $\mathbf{tv}_{(U,U')}(\varphi, S, v) = \top$

If $v(\text{says}_{\{id\}}\, \psi, S) \notin U'$, $\mathbf{tv}_{(U,U')}(\varphi, S, v) = \bot$

In addition, we require that $U \subseteq U'$.

Given a consistent evaluation (U_1, U_1'), evaluating the regulation gives us a way to define a new utterance pair (U_2, U_2'). U_2 (resply. U_2') corresponds to the laws whose preconditions become true (resply. do not become false) by evaluating w.r.t. (U_1, U_1'). In [11], we show that $U_1 \subseteq U_2$ and $U_1' \supseteq U_2'$, i.e., more statements become grounded (true or false). We are interested in fixed point utterance pairs:

Definition 7 (Fixed Point)**.** *Given a body of regulation Reg, and state S, the consistent utterance pair (U, U') is a fixed point iff for all* (id) $\varphi \mapsto \psi \in$ *Reg and $v \in V$:*

If $\mathbf{tv}_{(U,U')}(\varphi, S, v) = \top$, $v(\text{says}_{\{id\}}\, \psi) \in U$

If $\mathbf{tv}_{(U,U')}(\varphi, S, v) = \bot$, $v(\text{says}_{\{id\}}\, \psi) \notin U'$

We say that $(U_1, U_1') \leq (U_2, U_2')$ if $U_1 \subseteq U_2$ and $U_1' \supseteq U_2'$. We now review some results from [11]. The partially ordered set of consistent utterances has a least fixed point and one or more maximal fixed points. Distinct fixed points arise if there are circular references. The converse is not necessarily true, i.e., there may be circular references and a unique fixed point. There is a smallest element in the set of consistent utterances (U_0, U'_0) such that $U_0 = \emptyset$ and U'_0 contains all utterances. The least fixed point can be obtained iteratively using (U_0, U'_0). All the examples in this paper have a unique (least) fixed point. We note that if (U, U') is the unique fixed point, then $U = U'$ and all statements are evaluated to true or false.

A state S together with a consistent utterance pair (U, U') forms the basis for all decision problems. Given S, (U, U') and a propositional $\varphi \in L$, we say that φ is valid at S w.r.t. (U, U') (denoted $S \models_{(U,U')} \varphi$) iff $\mathbf{tv}_{(U,U')}(\varphi, S, v) = \top$ for all $v \in V$. We use this notion of validity at a state to formalize access control and conformance decisions, in Sections 3.3 and 3.4 respectively.

3.3. Non-interference in Access Control

An access control decision is made when a principal A requests the performance of action p which is controlled by B. Given a state S and fixed point (U, U') resulting from the evaluation of policy, the decision problem is whether $S \models_{(U,U')} \mathrm{says}_{l(B)}\, \mathcal{P}_A(p)$, i.e., does B say that A is permitted to perform p.

A problem with this definition is that the policies in access control are usually distributed. It is unreasonable to expect (U, U') to reside on a single system. Given that we wish to evaluate $\mathrm{says}_{l(B)}\, \mathcal{P}_A(p)$, the question is whether a smaller set of utterances suffice to answer this question. In other words, the evaluation should be carried out locally by B or a designated evaluator for B, as in [25].

[5, 3] advocate the use of non-interference properties to obtain such results, and to demonstrate that the logic protects the rights of each principal. In our case, the access control decision is of the form $S \models_{(U,U')} \mathrm{says}_{Id}\, \psi$, and this holds iff $U \vdash \mathrm{says}_{Id}\, \psi$. The goal is to identify a subset of utterances ($U^* \subseteq U$), such that $U \vdash \mathrm{says}_{Id}\, \psi$ iff $U^* \vdash \mathrm{says}_{Id}\, \psi$. We begin by defining this subset:

Definition 8 (Reachable Utterances)**.** *Given a set of utterances U and a formula $\mathrm{says}_{Id}\, \psi$, U^*_{Id} is the smallest set such that:*

- *If $id \in Id$ and $\mathrm{says}_{\{id\}}\, \varphi \in U$, $\mathrm{says}_{\{id\}}\, \varphi \in U^*_{Id}$*
- *If $\mathrm{says}_{\{id\}}\, \varphi \in U^*_{Id}$ and $\mathrm{says}_{Id'}\, \psi'$ is a subformula of φ, then $U^*_{Id'} \subseteq U^*_{Id}$*

If we think of formulas $\mathrm{says}_{Id}\, \psi$ as *pointing* to utterances in U (labeled Id), then U^*_{Id} is the set of utterances that are reachable. We believe that it is reasonable to restrict to the set of reachable utterances. Given the question $\mathrm{says}_{l(B)}\, \psi$, $U^*_{l(B)}$ is determined by B and the principals that she delegates to. We can now show the following:

Theorem 2 (Non-interference)**.** *Given a set of utterances U, for all $\mathrm{says}_{Id}\, \psi \in L$:*

$U \vdash \mathrm{says}_{Id}\, \psi$ iff $U^*_{Id} \vdash \mathrm{says}_{Id}\, \psi$

PROOF. The proof relies on properties of the canonical Kripke structure (Appendix D). As an example, consider $U = \{\mathsf{says}_{l(A)}(P_B\,\mathsf{says}_{l(B)}\,\varphi), \mathsf{says}_{l(B)}\,\psi\}$. To decide whether $U \vdash \mathsf{says}_{l(B)}(P_A\,\mathsf{says}_{l(A)}\,\psi')$ it suffices to consider $U^*_{l(B)} = \{\mathsf{says}_{\{B\}}\,\psi\}$, provided that ψ does not refer to $l(A)$. In other words, the utterances in $l(A)$ can affect the inferences from $l(B)$ iff $l(B)$ refers to $l(A)$, and $l(A)$ referring to $l(B)$ is irrelevant while inferring $\mathsf{says}_{l(B)}\,\phi$. □

We note that the distinction between the inference component and the saying component allows us to restrict attention to inferences of the form $U \vdash \mathsf{says}_{Id}\,\varphi$, where U only has formulas of the form $\mathsf{says}_{Id'}\,\psi$. If the set U could contain arbitrary formulas, non-interference would have a more complex characterization, as in [5]. For example, if we allowed formulas of the form $\neg\,\mathsf{says}_{Id'}\,\psi$ in U, then any principal can render U inconsistent.

3.4. Conformance

We now turn to the definition of conformance. While the definition of conformace has some variation between formalisms [11, 12, 13], all of them require a principal to satisfy the obligations that are imposed on her. [12] gives the option of defining a mitigating action such as "paying a fine" if an obligation is not satisfied. Such mitigating actions are easily accommodated here, e.g., using disjunction. We define conformance as a relation between a principal and a set of laws:

Definition 9 (Conformance)**.** *Given a state S with a set of objects O, and a body of regulation Reg with identifiers ID, we say that $A \in O$ conforms to the laws $Id \subseteq ID$ w.r.t. the fixed point (U, U') iff for all propositional $\varphi \in L$:*
If $S \models_{(U,U')} \mathsf{says}_{Id}\,\mathcal{O}_A\varphi$, then $S \models_{(U,U')} \varphi$

In other words, conformance is the satisfaction of all obligations. Returning to our example in Table 1, we consider the conformance of the bloodbank (A) to the laws $\{1, 2\}$. At the unique fixed point (U, U'), $U = U'$ consisting of the utterances in Table 1. Furthermore, $S \models_{(U,U')} \mathsf{says}_{\{1,2\}}\,\mathcal{O}_A test(o_2, A)$, but $S \not\models_{(U,U')} test(o_2, A)$. As a result, A does not conform to the laws $\{1, 2\}$. We will examine the definition of conformance in the presence of nested obligations and permissions in Section 4.2.

We briefly discuss the proof of decidability of conformance:

Theorem 3 (Decidability of Conformance)**.** *Given a state S, a fixed point (U, U'), identifiers $Id \subseteq ID$ and a principal $A \in O$, there is a procedure to decide whether A conforms to Id.*

PROOF. First, we observe that $S \models_{(U,U')} \mathsf{says}_{Id}\,\mathcal{O}_A\varphi$ iff $U \vdash \mathsf{says}_{Id}\,\mathcal{O}_A\varphi$ (by definition). So, it suffices to check that for all φ, if $U \vdash \mathsf{says}_{Id}\,\mathcal{O}_A\varphi$, then $S \models_{(U,U')} \varphi$

The key idea is to show that there is a formula $\psi_{(U,A,Id)}$ such that: (a) $U \vdash \mathsf{says}_{Id}\,\mathcal{O}_A\psi_{(U,A,Id)}$, and (b) for all $\varphi \in L$ such that $U \vdash \mathsf{says}_{Id}\,\mathcal{O}_A\varphi$, we have $\vdash \psi_{(U,A,Id)} \Rightarrow \varphi$. The proof relies on the finite (canonical) model property (Appendix D).

It is easy to see from the definition of evaluation that for all ϕ, if $\vdash \phi$, then $S \models_{(U,U')} \phi$. In other words, the axioms of L are sound w.r.t. the evaluation of regulations. As a result, it suffices to check if $S \models_{(U,U')} \psi_{(U,A,Id)}$. □

We now turn to a discussion of examples, in the context of related work.

4. Discussion

In this section, we discuss how various constructs from the literature are expressed in our framework. In Section 4.1, we discuss access control examples. Section 4.2 discusses conformance in the presence of nested obligations and permissions [14]. We then discuss other relationships to prior work, in Section 4.3.

4.1. Access Control

We discuss two access control examples in this section. The first example highlights an important restriction of the policies in Section 3.2, i.e., a policy lets us conclude what has been said, but not what actually happens. The second example illustrates how the delegation operator of [7] can be defined in our framework.

Example 1: We begin with an example from [6]. Consider a file-access scenario with an administrating principal (A), a user (B), a file (file1), and the following policy:

1. If A says that file1 should be deleted, then this must be the case.
2. A trusts B to decided whether file1 should be deleted.
3. B wants to delete file1.

We introduce a new principal F for the file system. The following are the utterances (U) obtained at the fixed point:

1. $\text{says}_{l(F)}\, \mathcal{P}_A\, \text{says}_{l(A)}\, \mathcal{O}_F(\text{delfile1})$
2. $\text{says}_{l(A)}\, \mathcal{P}_B\, \text{says}_{l(B)}\, \mathcal{O}_F(\text{delfile1})$
3. $\text{says}_{l(B)}\, \mathcal{O}_F(\text{delfile1})$

The first utterance is read as follows: The file system F says that A is permitted to require it (F) to delete file1. The second utterance is the delegation from A to B, and the third utterance is B's wish to delete file1. Using **A5**, we will conclude that $U \vdash \text{says}_{l(F)}\, \mathcal{O}_F(\text{delfile1})$. In other words, we conclude that the system requires itself to delete file1.

Our analysis differs in an important way from [6]. We do not conclude that file1 is actually deleted, i.e., $U \nvdash \text{delfile1}$. In fact, we can show that there is no policy (as defined in Section 3.2) that lets us make this conclusion. delfile1 is true at a state where F conforms to $l(F)$, as per Defintion 9. In some cases, it may be warranted to assume/axiomatize self-conformance, i.e., $(\text{says}_{l(F)}\, \mathcal{O}_F(\varphi)) \Rightarrow \varphi$. However, conflicting self-imposed requirements would make U inconsistent.

Example 2: The delegation operator of [7] has a compelling definition in our framework. The syntax (in [7]) for delegation is "x delegates $(\varphi)^d$ to y", where d is the depth of delegation. We define the schema $\mathrm{ps}(\varphi, x, d)$, where x is used to generate variable names, and $d \in N$:

- $\mathrm{ps}(\varphi, x, 1) = \mathcal{P}_{x_1}\ \mathrm{says}_{l(x_1)}\ \varphi$
- $\mathrm{ps}(\varphi, x, d) = \mathcal{P}_{x_d}\ \mathrm{says}_{l(x_d)}(\varphi \wedge \mathrm{ps}(\varphi, x, d-1))$, for $d > 1$

The statement "A delegates $(\mathrm{delfile1})^2$ to B" is interpreted as follows: A says delfile1 if B says it or anyone that B trusts says it. Suppose, in addition, that B delegates $(\mathrm{delfile1})^1$ to C, and C says delfile1. We express this with the following rules:

- (1) $(x_2 = B) \mapsto \mathrm{ps}(\mathrm{delfile1}, x, 2)$
- (2) $(y_1 = C) \mapsto \mathrm{ps}(\mathrm{delfile1}, y, 1)$
- (3) $\top \mapsto \mathrm{delfile1}$

We assume that $1 \in l(A)$, $2 \in l(B)$ and $3 \in l(C)$. At the fixed point, we will have $U \vdash \mathrm{says}_{l(A)}\ \mathrm{delfile1}$, i.e., A says delfile1. Further redelegations by C (by modifying statement 3) will not be attributed to A.

In [7], a representation statement is used to grant permission to speak *without consuming delegation depth.* If C represents B on delfile1, then C should be permitted to at most one redelegation. Statement 2 is modified as follows:

- (2) $(y_2 = C) \mapsto \mathrm{ps}(\mathrm{delfile1}, y, 2)$

With this modification, a delegation by C will be attributed to A. The reader may have noticed the similarity between statement 1 and the modified version of statement 2. In our approach, delegation is just a special kind of representation. A delegates $(\varphi)^d$ to B iff B represents A on "delegating $(\varphi)^{d-1}$ to anyone". If C represents B on "delegating $(\varphi)^{d-1}$ to anyone", then she represents A as well.

As [7] points out, in the presence of representation, delegation depth does not have much meaning. For example, A may not wish to trust C to the same extent as B. There are a few options to address this issue by modifying the representation axiom. One way is to keep tract of the delegation depth in the axiom, as in [18]. Yet another way is to keep track of the principal on belhalf of whom a statement in made. We avoid these modifications, to simplify presentation.

4.2. Nested Obligations and Permissions

We discuss two examples of conformance in the presence of nested obligations and permissions. The first example illustrates how several fine-grained notions of conformance can be captured. The second example points out an important practical difficulty.

Example 1: We consider an example (based on one in [14]):

(4) The owners of parking lots ought to forbid parking near the entrance.

What does it mean to conform to (4)? We analyze this sentence as follows: "The owners of parking lots ought to (introduce laws that) forbid parking near the entrance.". In other words, (4) is an obligation to introduce a prohibition. If the owner introduces such a law, then the person parking is viewed as non-conformant, but it is the owner that needs to conform to (4). We can represent (4) in logic as follows:

- (4) $\mathrm{own}(x) \wedge \mathrm{p}(y) \mapsto \mathcal{O}_x\, \mathrm{says}_{l(x)}\, \mathcal{O}_y \neg \mathrm{pk}(y, x)$

Here $\mathrm{own}(x)$ is true iff x is the owner of a parking lot, $\mathrm{p}(y)$ is true iff y is a person, and $\mathrm{pk}(y, x)$ is true iff y parks near the entrance of the lot owned by x. $l(x)$ refers to the laws that are introduced by x.

Let us assume a state $S = (I_{\Phi_1}, ..., I_{\Phi_n})$ in which A is the owner of a parking lot, and B parks near the entrances of A's lot. The true predications are: $\{\mathrm{own}(A),\ \mathrm{p}(B),\ \mathrm{pk}(B, A)\}$. In addition, A is assigned the identifier 5, i.e., $l(A) = \{5\}$. We will now consider two scenarios – (a) A does not introduce any laws, and (b) A introduces a law forbidding parking near the entrance. We are interested in the conformance (Definition 9) of the owner A and the driver B.
Scenario 1: Suppose that A does not introduce any laws:

- The fixed point is: $U = U' = \{\mathrm{says}_{\{4\}}\, \mathcal{O}_A\, \mathrm{says}_{\{5\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A)\}$
- $U \vdash \mathrm{says}_{\{4\}}\, \mathcal{O}_A\, \mathrm{says}_{\{5\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A)$, but $S \not\models_{(U,U')} \mathrm{says}_{\{5\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A)$.
- Hence A does not conform to $\{4\}$.

However, it can be shown that B conforms to $\{4\}$.
Scenario 2: Now suppose that A introduces the law:

(5) $\mathrm{p}(y) \mapsto \mathcal{O}_y \neg \mathrm{pk}(y, A)$

The fixed point utterance pair is:
$U = U' = \{\mathrm{says}_{\{4\}}\, \mathcal{O}_A\, \mathrm{says}_{\{5\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A),\ \mathrm{says}_{\{5\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A)\}$

It can be shown that A conforms to $\{4\}$. What about B? It is clear that B does not conform to $\{5\}$, but what about $\{4\}$? Observe that $U \vdash \mathrm{says}_{\{4\}}\, \mathcal{O}_B \neg \mathrm{pk}(B, A)$ (using the transfer axiom **A5**), but $S \not\models_{(U,U')} \neg \mathrm{pk}(B, A)$. Hence, B does not conform to $\{4\}$. In other words, the statement (4) conveys an obligation to A and if A conforms, the embedded obligation is conveyed to B. As we noted in Section 2.1, we are formalizing the notion of *speaking on someone's behalf*, i.e., the obligation (5) issued by A is understood as being on behalf of the issuer of (4). Some applications may need a distinction between the different senses of saying.
Example 2: As we mentioned, our approach provides only a partial analysis of nested modalities. Consider the following example:

(6) You are required to allow a patient to see his records.

By our analysis, (6) is an obligation on the hospital to provide a permission. Suppose that a hospital introduces such a permission in its policy. Has it conformed to (6)? The problem arises in distinguishing between *claimed permission*, and *actual permission.* A hospital claims that it permits a patient to see his records, by making an appropriate rule. On the other hand, a hospital actually permits a patient to see his records, by taking an action, e.g., sending the records via mail.

We suggest that a formalization of actual permission needs notions of *bringing about* or *seeing to it that* (e.g., [20, 26]). If a principal A *says* that she permits an action p, we need to check if she prevents p either by some other action or non-action. We can capture such notions by introducing laws that require facilitation. For example, (id) $\mathrm{says}_{l(A)}\, \mathcal{P}_B(p) \mapsto \mathcal{O}_A(\varphi)$, where φ is understood as a prerequisite for p, which is in the control of A. Thus, actual permission can be determined during conformance checking. However, listing the prerequistes for all the actions is quite difficult in practice, and notions of control and responsibility could lead to a more elegant solution.

4.3. Related Work

We have discussed several relationships to prior work, in Sections 2, 4.1, and 4.2. In this section, we discuss other relationships, to identify interesting lines for further research.

Logic programming has been popular in access control [7, 16, 17]. The formalism that we adopted (Section 3.2) provides a way to integrate the logic programming approaches with the logics of saying [1, 2, 3, 5, 6], i.e., by evaluating saying using provability. The negation of provability gives a good interepretation to *didn't say*, thereby establishing a connection between saying and non-monotonic reasoning. Non-monotonic reasoning plays a useful role in formalizing exceptions to laws [11, 13]. In addition, the distributed proving techniques in [25] are directly applicable here.

However, the provability tests $U \vdash \mathrm{says}_{l(A)}\, \psi$ can be expensive, if U is large. Logic programs restrict the heads of rules to be atomic (as in [7, 16, 17]). This restriction to atomic formulas lets one decide provability in polynomial time. An important question is whether similar restrictions can be applied here to get polytime fragments. Disjunction is the main culprit in the exponential worst-case complexities here (Section 3.1). Even if we exclude disjunction syntactically, the representation axiom is defined using implication, and introduces disjunction indirectly. This leads to the following question: *Is there a fragment of the logic that accommodates representation, and yields a polytime decision procedure?*

Due to the problematic interactions between hand-off and classical logic (Section 2.1), intuitionistic approaches have been developed [3, 5, 6]. While we have focussed on the classical setting here, the representation axiom can be adapted to the intuitionistic setting. However, as [5] points out, a notion of constructivism is also desirable in an intuitionistic logic. Constructivism requires the meaning of an operator to be independent of others, and as a result, axioms which describe interaction between operators (such as the representation axiom) are excluded. While constructivism is important in programming languages (see [5, 3]), interaction axioms have also proved useful. For example, [27] discusses 48 systems of knowledge and time. This leads to our next question: *Is there a more constructive form of the representation axiom, that yields a useful programming language?*

Finally, notions of time have been used in conformance checking [12, 28, 10]. The policies are used to synthesize monitors that are used to detect violations at runtime [12, 28]. Since the saying component (Section 3.2) uses the formalism in [11, 28], notions of linear time can be easily added here, and the monitor synthesis in [28] can be used directly. Once notions of time are available, we can place constraints on how a policy changes. This leads to our final question: *Are there useful interactions between saying and time, to characterize how a policy is updated?*

5. Conclusions

We have motivated and described a logic for access control and conformance. The focus was on the interaction between *saying* and *permission*, as needed for these applications. We proposed two axioms to characterize their interaction (Section 3.1), and showed how these axioms could be incorporated into a logic programming approach (Section 3.2).

A combined analysis of saying and permission yielded benefits to both applications. For access control, we find a way to avoid the problematic interaction between hand-off and classical reasoning. Our axioms yield a decidable logic with a complete semantics (Section 3.1), and we hope that they have intuitive appeal to the reader. For conformance, we are able to accommodate nested obligations and permissions. We showed, in Section 3.4, that conformance checking remains decidable.

We believe that the joint study of access control and conformance is a rich area for research. In Section 4.3, we identified several avenues for further inquiry.

References

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

[2] M. Abadi, Logic in access control, in: Proceedings of the Symposium on Logic in Computer Science, 2003.

[3] M. Abadi, Access control in a core calculus of dependency, Electronic notes in Theoretical Computer Science 172 (2007) 5–31.

[4] A. Cirillo, R. Jagadeesan, C. Pitcher, J. Riely, Do as I SaY! Programmatic access control with explicit identities, in: 20th IEEE Computer Security Foundations Symposium, 2007.

[5] D. Garg, F. Pfenning, Non-interference in constructive authorization logic, in: 19th IEEE Computer Security Foundations Workshop, 2006.

[6] D. Garg, M. Abadi, A modal deconstruction of access control logics, in: Proceedings of the 11th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS), 2008.

[7] N. Li, B. N. Grosof, J. Feigenbaum, Delegation logic: a logic-based approach to distributed authorization, ACM Transactions on Information and System Security 6 (1) (2003) 128–171.

[8] A. Abrahams, Developing and executing electronic commerce applications with occurrences, Ph.D. thesis, Univeristy of Cambridge (2002).

[9] T. D. Breaux, M. W. Vail, A. I. Anton, Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations, in: Proceedings of the 14th IEEE International Requirements Engineering Conference, 2006.

[10] C. Giblin, A. Liu, S. Muller, B. Pfitzmann, X. Zhou, Regulations Expressed as Logical Models (REALM), in: M.-F. Moens, P. Spyns (Eds.), Legal Knowledge and Information Systems, 2005.

[11] N. Dinesh, A. Joshi, I. Lee, O. Sokolsky, Reasoning about conditions and exceptions to laws in regulatory conformance checking, in: Proceedings of the Conference on Deontic Logic in Computer Science, 2008.

[12] M. Kyas, C. Prisacariu, G. Schneider, Run-time monitoring of electronic contracts, in: 6th International Symposium on Automated Technology for Verification and Analysis (ATVA'08), 2008.

[13] G. Governatori, A. Rotolo, Bio logical agents: Norms, beliefs, intentions in defeasible logic, Autonomous Agents and Multi-Agent Systems 17 (1) (2008) 36–69.

[14] R. B. Marcus, Iterated deontic modalities, Mind 75 (300).

[15] G. Boella, L. van der Torre, Permissions and obligations in hierarchical normative systems, in: Proceedings of the 9th international conference on AI and law, 2003.

[16] J. Crampton, G. Loizou, G. O. Shea, A logic of access control, The Computer Journal 44 (1) (2001) 137–149.

[17] E. Bertino, B. Catania, E. Ferrari, P. Perlasca, A logical framework for reasoning about access control models, ACM Transactions on Information Systems Security 6 (1) (2003) 71–127.

[18] M. Y. Becker, C. Fournet, A. D. Gordon, Design and semantics of a decentralized authorization language, in: Computer Security Foundations Symposium, 2007.

[19] D. Makinson, L. van der Torre, Permissions from an input/output perspective, Journal of Philosophical Logic 32 (4).

[20] N. D. Belnap, P. Bartha, Marcus and the Problem of Nested Deontic Modalities, in: W. Sinnot-Armstrong, D. Raffman, N. Asher (Eds.), Morality and Belief: Festschrift in Honour of Ruth Barcan Marcus, 1995.

[21] G. H. von Wright, Deontic logic, Mind 60 (1951) 1–15.

[22] M. Sergot, F.Sadri, R. Kowalski, F.Kriwaczek, P.Hammond, H. Cory, The british nationality act as a logic program, Communications of the ACM 29 (5) (1986) 370–86.

[23] L. T. McCarty, A language for legal discourse - i. basic features, in: Proceedings of ICAIL, 1989.

[24] D. Makinson, L. van der Torre, Input/output logics, Journal of Philosophical Logic 29 (2000) 383–408.

[25] L. Bauer, S. Garriss, M. K. Reiter, Distributed proving in access control systems, in: 20th IEEE Computer Security Foundation Symposium, 2007.

[26] J. F. Horty, N. D. Belnap, The Deliberative Stit: A Study of Action, Omission, Ability, and Obligation, Journal of Philosophical Logic 29 (1995) 109–136.

[27] J. Y. Halpern, R. van der Meyden, M. Y. Vardi, Complete axiomatizations for reasoning about knowledge and time, SIAM Journal of Computing 33 (3) (2004) 674–703.

[28] N. Dinesh, A. Joshi, I. Lee, O. Sokolsky, Checking traces for regulatory conformance, in: Proceedings of the Workshop on Runtime Verification, 2008.

[29] J. Y. Halpern, Y. Moses, A guide to completeness and complexity for modal logics of knowledge and belief, Artif. Intell. 54 (3) (1992) 319–379.

A. Semantics

We begin by defining models (Kripke structures):

Definition 10 (Models)**.** *Given countable sets O of object names, $\Phi_1, ..., \Phi_n$ (where Φ_j is a set of predicate names of arity j), function names F, and identifiers for rules ID, a model $M(O, \Phi_1, ..., \Phi_n, ID)$, abbreviated as M, is the tuple $(S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$ where:*

- *S is a set of states*
- $I_{\Phi_j} : \Phi_j \times S \to 2^{O^j}$ *is the interpretation of predicates of arity j. Given $p \in \Phi_j$, we will say that $p(o_1, ..., o_j)$ is true at state s iff $(o_1, ..., o_j) \in I_{\Phi_j}(p, s)$.*
- $\delta_{\mathcal{L}} : S \times 2^{ID} \to 2^S$. *$\delta_{\mathcal{L}}(s, \mathrm{Id})$ corresponds to a description of s according to the laws labeled with identifiers in Id (taken conjunctively).*
- $\delta_{\mathcal{O}} : \mathcal{S} \times O \to 2^S$. *$\delta_{\mathcal{O}}(s, A)$ corresponds to an idealization of s, for which the principal A is held responsible.*

For the axioms **A3**-**A6** *we need the following constraints* **C3**-**C6** *(resply). For all $s \in \mathcal{S}$:*

C3 $\delta_{\mathcal{L}}(s, \mathrm{Id}) \supseteq \delta_{\mathcal{L}}(s, \mathrm{Id}')$ *for all $Id \subseteq Id'$*

C4 $\delta_{\mathcal{O}}(s, A) \neq \emptyset$ *for all $A \in O$*

C5 *For all $\{A, B\} \subseteq O$, $Id_A \subseteq l(A)$, $Id_B \subseteq l(B)$, and $s' \in \delta_{\mathcal{L}}(s, Id_A)$:*

1. *$s' \in \delta_{\mathcal{L}}(s, Id_B)$, or*
2. *There exists $s_1 \in \delta_{\mathcal{L}}(s, Id_A)$ such that for all $s_2 \in \delta_{\mathcal{O}}(s_1, B)$, $s' \in \delta_{\mathcal{L}}(s_2, Id_B)$*

C6 *For all $\{A, B\} \subseteq O$, $Id_A \subseteq l(A)$, and $s' \in \delta_{\mathcal{L}}(s, Id_A)$:*
There exists $s_1 \in \delta_{\mathcal{L}}(s, Id_A)$ such that for all $s_2 \in \delta_{\mathcal{O}}(s_1, B)$, $s' \in \delta_{\mathcal{L}}(s_2, Id_B)$

C5 and **C6** can be understood in the context of soundness (Lemma 1). Given the object names O, predicate names $(\Phi_1, ..., \Phi_n)$ and identifiers ID, the space of models is denoted by $\mathcal{M}(O, \Phi_1, ..., \Phi_n, ID)$, abbreviated as $\mathcal{M}$. We can now define satisfaction and validity, and we restrict attention to the propositional fragment of L:

Definition 11 (Semantics)**.** *Given a model $M = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$, $s \in S$ and a propositional $\varphi \in L$, the relation $(M, s) \models \varphi$ is defined inductively as follows:*

- *$(M, s) \models p(o_1, ..., o_j)$ iff $(o_1, ..., o_j) \in I_{\Phi_j}(p, s)$.*
- *The semantics of conjunction and negation is defined in the usual way.*
- *$(M, s) \models \mathrm{says}_{Id}\, \varphi$ iff $(M, s') \models \varphi$, for all $s' \in \delta_{\mathcal{L}}(s, Id)$.*
- *$(M, s \models \mathcal{O}_A \varphi$ iff $(M, s') \models \varphi$, for all $s' \in \delta_{\mathcal{O}}(s', A)$.*

We can now define validity:

- *φ is valid in a model M ($M \models \varphi$) iff for all $s \in S$, $(M, s) \models \varphi$*
- *φ is valid ($\models \varphi$) iff for all $M \in \mathcal{M}$, $M \models \varphi$*

B. Soundness and Completeness

Theorem 4 (Soundness and Completeness)**.** *Given a propositional $\varphi \in L$, $\vdash \varphi$ iff $\models \varphi$*

Lemma 1 (Soundness)**.** *Given a propositional $\varphi \in L$, if $\vdash \varphi$, then $\models \varphi$*

PROOF. We need to show that the axioms are valid, and that the rules preserve validity. It is well-known that the axioms **A1** and **A2** are valid, and that **R1** and **R2** preserve validity in all Kripke structures. The validity of **A3** and **A4** can easily be shown using **C3** and **C4**. We discuss the case for **A5**.

Suppose **A5** is not valid. There exists M, s, φ, A, B, Id_A and Id_B such that:

- $(M, s) \models \mathrm{says}_{Id_A}(\mathcal{P}_B\, \mathrm{says}_{Id_B}\, \varphi)$
- $(M, s) \models \mathrm{says}_{Id_B}\, \varphi$, and
- $(M, s) \not\models \mathrm{says}_{Id_A}\, \varphi$

Since $(M, s) \not\models \mathrm{says}_{Id_A}\, \varphi$, there exists $s' \in \delta_{\mathcal{L}}(s, Id_A)$ such that $(M, s') \not\models \varphi$. Since **C5** holds, there are two cases to consider:

1. If $s' \in \delta_{\mathcal{L}}(s, Id_B)$, then $(M, s) \not\models \mathrm{says}_{Id_B}\, \varphi$ giving us a contradiction.
2. If there exists $s_1 \in \delta_{\mathcal{L}}(s, Id_A)$ such that for all $s_2 \in \delta_{\mathcal{O}}(s_1, B)$, $s' \in \delta_{\mathcal{L}}(s_2, Id_B)$, then:
   - $(M, s_1) \models \mathcal{O}_B \neg\, \mathrm{says}_{Id_B}\, \varphi$
   - $(M, s) \not\models \mathrm{says}_{Id_A}(\neg \mathcal{O}_B \neg\, \mathrm{says}_{Id_B}\, \varphi)$

   Hence, $(M, s) \not\models \mathrm{says}_{Id_A}(\mathcal{P}_B\, \mathrm{says}_{Id_B}\, \varphi)$ (since $\mathcal{P}_B\varphi = \neg \mathcal{O}_B \neg \varphi$), giving us a contradiction.

Hence, **A5** is valid. The proof for **A6** is similar. □

Lemma 2 (Completeness)**.** *Given a propositional $\varphi \in L$, if $\models \varphi$, then $\vdash \varphi$*

The rest of this section gives the proof. We will use a canonical model argument (c.f. [29]). We show the contrapositive, i.e., if $\not\vdash \varphi$, then $\not\models \varphi$. In other words, if $\not\vdash \varphi$ then there exist M and s such that $(M, s) \models \neg\varphi$. We begin with some terminology.

We say that φ is *consistent* if $\neg\varphi$ is not provable ($\not\vdash \neg\varphi$). A finite set of formulas $\{\varphi_1, ..., \varphi_n\}$ is consistent if $\varphi_1 \wedge ... \wedge \varphi_n$ is consistent. An infinite set of formulas is consistent if every finite subset is consistent. A set of formulas Δ is *maximal consistent* if for all $\varphi \in L - \Delta$, $\Delta \cup \{\varphi\}$ is inconsistent. The following are properties of maximal consistent sets:

Proposition 2. *Given a maximal consistent set Δ:*

1. *For all $\varphi \in L$, exactly one of $\varphi \in \Delta$ or $\neg\varphi \in \Delta$*
2. *If $\vdash \varphi \Rightarrow \psi$ and $\varphi \in \Delta$, then $\psi \in \Delta$*

3. *If $\vdash \varphi$, then $\varphi \in \Delta$ and $\mathcal{Q}\varphi \in \Delta$ (for all modalities $\mathcal{Q}$)*

The proof is straightforward. We now define *the canonical model*, in which every consistent formula is true at some state:

Definition 12 (Canonical Model)**.** *The canonical model $M = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$ is such that:*

- *S is the set of all maximal consistent sets*
- *$(o_1, ..., o_j) \in I_{\Phi_j}(p, \Delta)$ iff $p(o_1, ..., o_j) \in \Delta$*
- *$\Delta' \in \delta_{\mathcal{L}}(\Delta, Id)$ iff for all φ, if $\text{says}_{Id}\,\varphi \in \Delta$, then $\varphi \in \Delta'$*
- *$\Delta' \in \delta_{\mathcal{O}}(\Delta, B)$ iff for all φ, if $\mathcal{O}_B\varphi \in \Delta$, then $\varphi \in \Delta'$*

We now show that the canonical model satisfies the frame constraints:

Proposition 3. *The canonical model satisfies the frame constraints* **C3**-**C6**

Proof. The proof that **C3** and **C4** hold are left to the reader. We discuss the case for **C5**. Given the canonical model $M = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$, $\Delta \in S$, and suppose for the purpose of contradiction that there exists $\Delta' \in \delta_{\mathcal{L}}(\Delta, Id_A)$ such that:

- $\Delta' \notin \delta_{\mathcal{L}}(\Delta, Id_B)$. By Proposition 4 item 1 (below), there exists $\text{says}_{Id_B}\,\psi \in \Delta$ such that $\neg\psi \in \Delta'$.
- For all $\Delta_1 \in \delta_{\mathcal{L}}(\Delta, Id_A)$, there exists $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, $\Delta' \notin \delta_{\mathcal{L}}(\Delta_2, Id_B)$. By Proposition 4 item 2 (below), there exists $\text{says}_{Id_A}\,\mathcal{P}_B\,\text{says}_{Id_B}\,\varphi \in \Delta$ such that $\neg\varphi \in \Delta'$.

Using Proposition 2, $\text{says}_{Id_B}(\varphi \vee \psi) \in \Delta$ and $\text{says}_{Id_A}\,\mathcal{P}_B\,\text{says}_{Id_B}(\varphi \vee \psi) \in \Delta$. So, $\text{says}_{Id_A}(\varphi \vee \psi) \in \Delta$, and hence $\varphi \vee \psi \in \Delta'$. That is $\varphi \in \Delta'$ or $\psi \in \Delta'$, which contradicts the fact that $\neg\varphi \in \Delta'$ and $\neg\psi \in \Delta'$. The proof of **C6** is similar. □

Proposition 4. *Given the canonical model $M = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$, for all $\Delta \in S$ $\{A, B\} \subseteq O$, $Id_A \subseteq l(A)$, and $Id_B \subseteq l(B)$, if $\Delta' \in \delta_{\mathcal{L}}(\Delta, Id_A)$:*

1. *If $\Delta' \notin \delta_{\mathcal{L}}(\Delta, Id_B)$, then there exists $\text{says}_{Id_B}\,\psi \in \Delta$ such that $\neg\psi \in \Delta'$*
2. *If for all $\Delta_1 \in \delta_{\mathcal{L}}(\Delta, Id_A)$, there exists $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, $\Delta' \notin \delta_{\mathcal{L}}(\Delta_2, Id_B)$, then there exists $\text{says}_{Id_A}\,\mathcal{P}_B\,\text{says}_{Id_B}\,\varphi \in \Delta$ and $\neg\varphi \in \Delta'$*

Proof. The first item is immediate from the definition of the canonical model. For the second item, we proceed by contradiction. Suppose for all $\varphi \in L$, if $\text{says}_{Id_A}\,\mathcal{P}_B\,\text{says}_{Id_B}\,\varphi \in \Delta$, then $\varphi \in \Delta'$. Let F be the smallest set such that:

- If $\text{says}_{Id_A}\,\varphi \in \Delta$, then $\varphi \in F$, and
- If $\neg\psi \in \Delta'$, then $\mathcal{O}_B\neg\,\text{says}_{Id_B}\,\psi \in F$.

We claim that F is consistent. Suppose not:

- There exists $\{\varphi_1, ..., \varphi_n, \psi_1, ..., \psi_m\}$ such that $\vdash \neg(\varphi_1 \wedge ... \wedge \varphi_n \wedge \mathcal{O}_B \neg\, \mathrm{says}_{Id_B}\, \psi_1 \wedge ... \wedge \mathcal{O}_B \neg\, \mathrm{says}_{Id_B}\, \psi_m)$
- It follows that $\vdash \varphi_1 \wedge ... \wedge \varphi_n \Rightarrow \mathcal{P}_B\, \mathrm{says}_{Id_B}(\psi_1 \vee ... \vee \psi_m)$
- Using **R2**, it follows that $\mathrm{says}_{Id_A}(\varphi_1 \wedge ... \wedge \varphi_n \Rightarrow \mathcal{P}_B\, \mathrm{says}_{Id_B}(\psi_1 \vee ... \vee \psi_m)) \in \Delta$
- And using **A2**, $\mathrm{says}_{Id_A}\, \mathcal{P}_B\, \mathrm{says}_{Id_B}(\psi_1 \vee ... \vee \psi_m) \in \Delta$. As a result, $\psi_1 \vee ... \vee \psi_m \in \Delta'$, and there exists $\psi_i \in \Delta'$ where $1 \leq i \leq m$.
- By construction, $\neg\psi_i \in \Delta'$ for all $1 \leq i \leq m$, which gives us a contradiction.

We can extend F into a maximal consistent set Δ_1 such that $\Delta_1 \in \delta_{\mathcal{L}}(\Delta, Id_A)$. $\mathcal{P}_B\, \mathrm{says}_{Id_B}\, \varphi \in \Delta_1$ iff $\varphi \in \Delta'$. So, for all $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, if $\mathrm{says}_{Id_B}\, \varphi \in \Delta_2$, then $\varphi \in \Delta'$. This suffices to conclude that $\Delta' \in \delta_{\mathcal{L}}(\Delta_2, Id_B)$ for all $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, giving us a contradiction. □

The completeness proof is now finished in the usual way. Given the canonical model M and a state Δ, it is easy to show that for all $\varphi \in L$, $(M, \Delta) \models \varphi$ iff $\varphi \in \Delta$. Furthermore given a consistent φ, we can construct a maximal consistent set Δ such that $\varphi \in \Delta$. As a result, for every consistent φ, there exists a state Δ in the canonical model such that $(M, \Delta) \models \varphi$. Hence, if $\not\vdash \varphi$, then $\not\models \varphi$.

C. Decidability

In this section, we adapt the completeness proof to show *the bounded-model property*, i.e., if ϕ is satisfiable, then it is satisfiable in a model of bounded size (exponential in the size of ϕ). We assume the existence of an object $A* \in O$ such that there is no subformula $\mathcal{O}_{A*}\varphi$ in ϕ. Given φ, the set of subformulas $sub(\varphi)$ is the set such that:

- $\top \in sub(\phi)$ and $\mathcal{P}_{A*}\top \in sub(\phi)$
- If $\varphi \in sub(\phi)$, then $\neg\varphi \in sub(\phi)$ ($\neg\neg\varphi$ is identified with φ)
- If $\phi \wedge \psi \in sub(\phi)$, then $\phi \in sub(\phi)$ and $\psi \in sub(\varphi)$
- If $\mathcal{O}_B\psi \in sub(\phi)$ or $\mathrm{says}_{Id}\, \psi \in sub(\phi)$, then $\psi \in sub(\varphi)$
- If $\mathrm{says}_{Id}\, \psi \in sub(\phi)$, then $\mathrm{says}_{\emptyset}\, \psi \in sub(\phi)$
- If $\{\mathrm{says}_{Id_B}\, \psi_1, \mathcal{O}_B\psi_2\} \subseteq sub(\phi)$ and $Id_B \subseteq l(B)$, then $\mathcal{P}_B\, \mathrm{says}_{Id_B}\, \psi_1 \in sub(\phi)$

Given ϕ, we will consider maximal consistent sets w.r.t. $sub(\phi)$. A set $\Delta \subseteq sub(\phi)$ is said to be maximal consistent iff Δ is consistent and for all $\psi \in sub(\phi) - \Delta$, $\Delta \cup \{\psi\}$ is inconsistent. We write $\Delta \vdash \varphi$ to denote $\vdash \bigwedge \Delta \Rightarrow \varphi$. The definition of the canonical model needs a few changes:

Definition 13 (Canonical Model of ϕ)**.** *The canonical model of ϕ, denoted $M_\phi = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$, is such that:*

- *S is the set of all maximal consistent sets w.r.t. $sub(\phi)$*
- *$(o_1, ..., o_j) \in I_{\Phi_j}(p, \Delta)$ iff $p(o_1, ..., o_j) \in \Delta$*
- *$\Delta' \in \delta_{\mathcal{L}}(\Delta, Id)$ iff for all $\psi \in sub(\phi)$ and $Id' \subseteq Id$, if $\text{says}_{Id'}\, \psi \in \Delta$, then $\psi \in \Delta'$*
- *$\Delta' \in \delta_{\mathcal{O}}(\Delta, B)$ iff for all $\psi \in sub(\varphi)$, if $\mathcal{O}_B\psi \in \Delta$, then $\psi \in \Delta'$. We assume that for all $B \in O$, if there is no $\psi \in L$ such that $\mathcal{O}_B\psi \in sub(\phi)$, then $B = A*$.*

The proofs of Propositions 3 and 4 can be adapted to show that the frame constraints **C3-C6** hold in the canonical model of φ. We can now show the following:

Theorem 5 (Bounded-model property)**.** *ϕ is satisfiable in M_ϕ iff ϕ is satisfiable*

Proof. One direction is trivial, i.e., if ϕ is satisfiable in M_ϕ, then ϕ is satisfiable (by definition). For the other direction, we can use a standard filtration argument, to show that M_ϕ can be obtained from the canonical model (Definition 12). □

D. Non-interference and Conformance

Proof (**Proof of Theorem 2**)**.** Suppose $U \vdash \text{says}_{Id}\, \psi$, and for the purposes of contradiction, $U^*_{Id} \nvdash \text{says}_{Id}\, \psi$. So, $\phi = U^*_{Id} \wedge \neg\, \text{says}_{Id}\, \psi$ is satisfiable. Given $Id_1 \subseteq ID$, let Id^*_1 be the set such that $id \in Id^*_1$ iff $id \in Id$ or there exists $\text{says}_{\{id\}}\, \varphi \in U^*_{Id}$. Let $M_\phi = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$ be the canonical model of ϕ. Hence, $(M_\phi, \Delta_\phi) \models \phi$ for some $\Delta_\phi \in S$. We construct a new model $M' = (S', I'_{\Phi_1}, ..., I'_{\Phi_n}, \delta'_{\mathcal{L}}, \delta'_{\mathcal{O}})$ as follows. M' is identical to M_ϕ except for a new state $s^* \in S'$ at which the interpretation of predicates and $\delta'_{\mathcal{O}}$ is identical to Δ_ϕ and:

- For all $Id_1 \subseteq Id$, if $Id_1 = Id^*_1$, $\delta'_{\mathcal{L}}(s^*, Id_1) = \delta_{\mathcal{L}}(\Delta_\phi, Id_1)$. Otherwise, $\delta'_{\mathcal{L}}(s^*, Id_1) = \emptyset$

We need to verify that the accessibility constraints hold in M'. The only difficulty is in showing that **C5** holds at s^* in M'. Fix A, B, Id_A, Id_B, and $\Delta' \in \delta'_{\mathcal{L}}(s^*, Id_A)$. The only difficult case is when $Id_A = Id^*_A$ and $Id_B \neq Id^*_B$. In this case, Clause (1) of **C5** (Definition 10) is violated and we need to show that Clause (2) holds . **C6** comes to the rescue:

1. $\delta_{\mathcal{L}}(\Delta, Id_B) = \delta_{\mathcal{L}}(\Delta, \emptyset)$, for all $\Delta \in S$ (by construction of M_ϕ)
2. Since $\Delta' \in \delta'_{\mathcal{L}}(s^*, Id_A)$, $\Delta' \in \delta_{\mathcal{L}}(\Delta_\phi, Id_B)$ (by construction of M')
3. Using **C6** in M_ϕ, there exists $\Delta_1 \in \delta_{\mathcal{L}}(\Delta_\phi, Id_A)$ such that for all $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, $\Delta' \in \delta_{\mathcal{L}}(\Delta_2, Id_A)$.
4. For each $\Delta_2 \in \delta_{\mathcal{O}}(\Delta_1, B)$, it follows using **C3** that $\Delta' \in \delta_{\mathcal{L}}(\Delta_2, \emptyset)$, and using item 1, $\Delta' \in \delta_{\mathcal{L}}(\Delta_2, Id_B)$
5. $\Delta_1 \in \delta'_{\mathcal{L}}(s^*, Id_A)$ (by construction of M')

The existence of such a Δ_1 for each $\Delta' \in \delta'_{\mathcal{L}}(s^*, Id_A)$ suffices to enforce **C5**. It is easy to show that $(M', s^*) \models U \wedge \neg \mathrm{says}_{Id}\, \psi$, contradicting the fact that $U \vdash \mathrm{says}_{Id}\, \psi$. The other direction follows easily using propositional reasoning, i.e., if $U^*_{Id} \vdash \mathrm{says}_{Id}\, \psi$, then $U \vdash \mathrm{says}_{Id}\, \psi$, since $U^*_{Id} \subseteq U$. □

Proof (**Proof of Theorem 3).** Given U, Id and A, there is a formula $\psi_{(U,Id,A)}$ such that (1) $U \vdash \mathrm{says}_{Id}\, \mathcal{O}_A \psi_{(U,Id,o)}$, and (2) for all $\varphi \in L$ such that $U \vdash \mathrm{says}_{Id}\, \mathcal{O}_A \varphi$, we have $\vdash \psi_{(\Delta,Id,A)} \Rightarrow \varphi$.

Let $M_U = (S, I_{\Phi_1}, ..., I_{\Phi_n}, \delta_{\mathcal{L}}, \delta_{\mathcal{O}})$ be the canonical model for U. Let $S_U = \{\Delta_1 | \Delta_1 \in S \textit{ and } U \subseteq \Delta_1\}$. Observe that for all $\Delta_1 \in S_U$, $(M, \Delta_1) \models \bigwedge U$. As a result, for all $\Delta_1 \in S_U$ and $\varphi \in L$, if $U \vdash \mathrm{says}_{Id}\, \mathcal{O}_A \varphi$, then $(M, \Delta_1) \models \mathrm{says}_{Id}\, \mathcal{O}_A \varphi$.

We will now construct a formula $\psi_{(\Delta_1,Id,A)}$ for each $\Delta_1 \in S_\Delta$, and define the desired formula as their disjunction. Given $\Delta_1 \in S_\Delta$ and $\Delta' \in \delta_{\mathcal{L}}(\Delta_1, Id)$, let $\Gamma_{(\Delta',A)} = \{\psi | \mathcal{O}_A \psi \in \Delta'\}$. Observe that for all $\varphi \in L$ such that $\Delta \vdash \mathrm{says}_{Id}\, \mathcal{O}_A \varphi$, we have $\Gamma_{(\Delta',A)} \vdash \varphi$. Let $\psi_{(\Delta_1,Id,A)} = \bigvee_{\Delta' \in \delta_{\mathcal{L}}(\Delta_1,Id)} \bigwedge \Gamma_{(\Delta',A)}$. By propositional reasoning, $\Delta_1 \vdash \mathrm{says}_{Id}\, \mathcal{O}_A \psi_{(\Delta_1,Id,A)}$, and $\vdash \psi_{(\Delta_1,Id,A)} \Rightarrow \varphi$.

As we mentioned before, we define $\psi_{(U,Id,A)} = \bigvee_{\Delta_1 \in S_\Delta} \psi_{(\Delta_1,Id,A)}$. The desired properties follow using propositional reasoning, i.e., if $\Delta_1 \vdash \phi$ for all $\Delta_1 \in S_U$, then $U \vdash \phi$. □