Non-Interference in Constructive Authorization Logic

Deepak Garg* and Frank Pfenning†
Carnegie Mellon University
E-mail: {dg,fp}@cs.cmu.edu

Abstract

We present a constructive authorization logic where the meanings of connectives are defined by their associated inference rules. This ensures that the logical reading of access control policies expressed in the logic and their implementation coincide. We study the proof-theoretic consequences of our design including cut-elimination and two non-interference properties that allow administrators to explore the correctness of their policies by establishing that for a given policy, assertions made by certain principals will not affect the truth of assertions made by others.

1. Introduction

An *authorization logic* is a logic for access control in distributed systems. An access control policy is presented as a logical theory in an authorization logic, and a principal is granted access to a resource if there is a *formal proof* that he or she is authorized to do so according to the present access control policy.

The study of authorization from a logical perspective was initiated by Abadi et al. [4]. A number of different proposals for authorization logics and supporting distributed architectures have been made since then [2]. We are particularly interested in *proof-carrying authorization* (PCA) [5, 6] where a resource is presented with a formally checkable proof object for authorization, expressed in a logical framework. This is combined with a separate mechanism for *authentication* using cryptographic techniques such as digital signatures. In combination, these provide a powerful, flexible, and extensible foundation for access control in distributed systems [8].

In this paper we propose a particular authorization logic and study it with proof-theoretic means. The following features distinguish our logic from previous proposals along similar lines. First, the logic is constructive, as opposed to most previous proposals that are classical. This is done to keep evidence contained in proofs as direct as possible. For example, Abadi [2] remarks that in a classical logic where principals K affirm all true propositions, we have that if K affirms A (written K `says` A), then either A is true or K affirms the truth of every other proposition B (written K `says` $A \supset (A \vee (K$ `says` $B))$). However, it is in general impossible to come up with a proof of *either* A or K `says` B given a proof of K `says` A. In our logic this classical reasoning does not hold.

The second fact that distinguishes our logic from other authorization logics is that we define the meanings of connectives only by the inference rules that introduce them, without relying on any other semantics. This ensures that the intended reading of logical propositions coincides with the available formal proofs. Finally, we want our logic to remain open to extensions with new connectives. This requires that the meanings of connectives be independent of each other. Formally this is documented in the form of a cut elimination theorem and the subformula property for our logic. To our knowledge, this is the first time that an access control logic has been developed constructively, and that the structure of proofs has been delineated precisely through a cut elimination theorem.

Even with a good understanding of the authorization logic it is possible for complex policies to have unintended consequences. We therefore would like to give users and administrators the ability to explore their policies. We propose the use of *non-interference properties* for this purpose. In this context, non-interference properties characterize classes of propositions whose presence or absence can have no effect on the existence of proofs of certain other propositions. For example, without explicit links between principals K and L, no assertions made by K (including even contradictory statements) can influence the truth of assertions made by L. We formalize and prove two such properties for our authorization logic. As far as we know this is the first time that explicit non-interference properties for authorization logics have been formulated and proved.

In summary, our paper makes two major conceptual contributions. First, it introduces a new approach to designing authorization logics based on principles of judgments and

*Supported by grant DAAD19-02-1-0389 of the Army Research Office
†Supported by grant N0014-04-1-0724 of the Office of Naval Research

explicit evidence, in contrast with the prevalent axiomatic approach. We believe that our approach makes it much easier to be confident in the correct interpretation and proper enforcement of complex policies. Second, it introduces the concepts of affirmation flow and non-interference for access control policies and relates them to the proof-theoretic property of strengthening. These take a quite different form than the standard non-interference properties used to characterize information flow.

The paper also makes two significant technical contributions, namely a fully formalized proof of cut elimination for an authorization logic, and a decidable approximation of affirmation flow for automatic policy analysis. If we are to generalize access control policies to admit complex logical specification, then tools to analyze policy specifications to uncover possibly unintended consequences will be a critical component of future security infrastructures.

The remainder of this paper is organized as follows. In section 2 we develop a sequent calculus for our authorization logic and present its meta-theoretic properties. Section 3 describes non-interference properties for reasoning with policies written in our logic. Section 4 discusses related work and section 5 concludes with some directions for future work.

2. Constructive Authorization Logic

Access control in a distributed system decomposes into two distinct, but necessarily interconnected tasks: *authentication* and *authorization*. Authentication requires us to determine reliably *who* is requesting access to a resource. Authorization must answer the question *whether* and, more generally, *why* access should be granted. In this paper we are concerned with authorization; we refer to other papers regarding authentication and the surrounding protocols and infrastructure currently being deployed at Carnegie Mellon University in the Grey project [8].

From the problem statement itself it should be immediately clear that authorization is a question of *logic* since we are trying to *reason about* whether a principal should have access to a resource. But which logic should we use? There have been several proposals in the literature (see [2] for a recent survey), but we are not aware of a systematic development and meta-theoretic analysis of authorization logic from first principles. Such an analysis is necessary if we are to properly understand the meaning of statements in an authorization logic and the access control policies expressed in it. Moreover, if access control decisions in an implemented system are based on an authorization logic, then the security of whole system rests critically on properties of this logic. For example, if a principal makes contradictory statements (which is quite plausible), we do not want the whole logic to become inconsistent because this would give every principal access to every resource. We will isolate some desirable properties, design an authorization logic around them, and then verify that these properties do indeed hold. In fact, most of the properties have been formally verified using the Twelf meta-logical framework [22] and are available on-line at `http://www.cs.cmu.edu/~self/`.

2.1. Judgments and Verifications

In authorization logic, access to a resource is granted precisely when presented with a proof that it should be. To understand the meaning of a proposition in authorization logic therefore requires us to understand its proofs. The idea that the meaning of logical connectives is determined by their proofs goes back to Gentzen [15]. Later, Martin-Löf [18] introduced a distinction between judgments and propositions which is crucial in our setting. We can give here only a very brief sketch as it impacts our development; the interested reader is referred to [21] for further justifications and discussions of the general approach in modal logic. A *judgment* may be *evident* by virtue of a *proof*. Judgments are therefore the subjects of inference rules. The most basic judgment is the truth of a proposition A, written $A\ \mathsf{true}$. The meaning of a proposition A is determined by its *verifications* which are the proofs of $A\ \mathsf{true}$ that proceed entirely by analysis of A. The cut elimination theorem tells us how to construct a verification of A given an arbitrary proof and thereby shows that the meaning of the connectives has been properly defined.

In this paper, we will concentrate on the cut-free sequent calculus, which is a calculus of *verifications*, that is, each proof only analyzes the structure of the given propositions. In other words, it obeys the *subformula property*. We write a *sequent*

$$A_1\ \mathsf{true}, \ldots, A_n\ \mathsf{true} \Longrightarrow C\ \mathsf{true}$$

to express "*Under the hypotheses that $A_1, \ldots, A_n$ are all true, C is true*". We will often omit the explicit judgment true for the sake of brevity. We will abbreviate a collection of hypotheses as Γ and write $\Gamma \Longrightarrow C\ \mathsf{true}$. We freely permit reordering of hypotheses.

The first principle of *identity* pertains to the nature of hypothetical reasoning: if we have a hypothesis $A\ \mathsf{true}$ we should be able to conclude $A\ \mathsf{true}$. We assume this for atomic propositions P as an explicit rule; we have to prove it for compound propositions A (see theorem 2, part 2).

$$\frac{}{\Gamma, P\ \mathsf{true} \Longrightarrow P\ \mathsf{true}}\ \text{INIT}$$

The second principle of *cut* states that if we can prove that A is true, we are justified in assuming it as a hypothesis:

If $\Gamma \Longrightarrow A$ true *and* Γ, A true $\Longrightarrow C$ true *then* $\Gamma \Longrightarrow C$ true.

We have to prove this (see theorem 2, part 4); if it were formulated as an inference rule it would violate the nature of verifications because the proposition A does not appear in the conclusion judgment $\Gamma \Longrightarrow C$ true.

The first logical connective we introduce is implication. Each connective $\circ$ is characterized by *right rules* that show how to verify $A \circ B$ true, and *left rules* that show how to use the hypothesis $A \circ B$ true. Here we have one each, hopefully self-explanatory. They are most easily understood when read bottom-up.

$$\frac{\Gamma, A \text{ true} \Longrightarrow B \text{ true}}{\Gamma \Longrightarrow A \supset B \text{ true}} \supset\text{R}$$

$$\frac{\Gamma \Longrightarrow A \text{ true} \qquad \Gamma, B \text{ true} \Longrightarrow C \text{ true}}{\Gamma, A \supset B \text{ true} \Longrightarrow C \text{ true}} \supset\text{L}$$

We use the convention that the principal proposition of the left rule ($A \supset B$ in the second rule) is still available in all premises, even though it is not explicitly written down again in order to save space and draw attention to the more essential features of the rule.

We can also add the proposition *false* ($\bot$). It has no right rule, because there should be no verification of $\bot$ unless the hypotheses are contradictory, and a left rule that allows us to infer anything.

no $\bot$ *right rule* $$\frac{}{\Gamma, \bot \text{ true} \Longrightarrow C \text{ true}} \bot\text{L}$$

Even though we have included $\bot$ as a connective, we do not advocate its use in access control policies because we believe it is a security risk. $\bot$ can be used to define negation as $\neg A = A \supset \bot$, which expresses the fact that A does not hold. This can be very difficult to verify as it may require looking at all facts in the system, including those on remote sites. If access to a resource depends on $\neg A$, we run the risk of inadvertently granting access simply because we failed to discover A on some site in the system.

2.2. Affirmation

Gentzen and Martin-Löf were mainly concerned with truth of propositions because of its central role in mathematics. However, in authorization logic the principals have to express *intent* or *policy*. So in addition to the truth of a proposition (which is independent of any particular principal), we have a new judgment form K affirms A, stating that principal K affirms proposition A. Note that the logic does not make any particular commitment to a set or language of principals because the logical reasoning does not depend on it.

When should we be able to conclude that K affirms A? The first rule states that if A true then K affirms A.

$$\frac{\Gamma \Longrightarrow A \text{ true}}{\Gamma \Longrightarrow K \text{ affirms } A} \text{ AFF}$$

How do we use the knowledge that K affirms A? We cannot simply assume that A is true because by using the rule above, any principal would then affirm A. However, if we are trying to prove that the same principal K affirms some C, then it is perfectly legitimate to assume that A is true, because we are currently inside K's mind, so to speak.

$$\frac{\Gamma, A \text{ true} \Longrightarrow K \text{ affirms } C}{\Gamma, K \text{ affirms } A \Longrightarrow K \text{ affirms } C}$$

The restriction of this rule to the same principal K is crucial and the basis of non-interference properties. It turns out to be convenient to limit ourselves to hypotheses of the form A true. We can achieve this by only always eliminating K affirms A immediately when such an assumption is introduced. The saysL rule below is an example of this kind.

The new judgment entails a new cut principle that relates the two rules above:

If $\Gamma \Longrightarrow K$ affirms A *and*
Γ, A true $\Longrightarrow K$ affirms C *then*
$\Gamma \Longrightarrow K$ affirms C.

Note the accordance of the principals K in the two given deductions.

K affirms C is a judgment, not a proposition. We therefore cannot use it inside propositions, for example, in an argument to implication. Thus we need to internalize the judgment as a new form of proposition with proper left and right rules to define when it is true. We write this proposition as K says A.

$$\frac{\Gamma \Longrightarrow K \text{ affirms } A}{\Gamma \Longrightarrow (K \text{ says } A) \text{ true}} \text{ saysR}$$

$$\frac{\Gamma, A \text{ true} \Longrightarrow K \text{ affirms } C}{\Gamma, (K \text{ says } A) \text{ true} \Longrightarrow K \text{ affirms } C} \text{ saysL}$$

Note that the premises of the left and right rule match up exactly in the way predicted by the second cut principle. This ensures that the proof of cut elimination goes through in this case and justifies these rules.

The introduction of a new judgment also requires us to generalize the previously introduced left rules to allow a conclusion of the K affirms C. We write γ for a judgment of the form C true or K affirms C. The resulting collection of rules is summarized in Figure 1 at the end of this section.

We can now state and prove some simple theorems in access control logic. Perhaps equally important are properties that cannot be proved parametrically in A, B, K, K_1 and K_2. We use the convention that the says operator binds more tightly than other logical operators, although sometimes we make parentheses explicit for the sake of clarity. We write $\vdash A$ for $\cdot \Longrightarrow A$ true and $\not\vdash A$ if $\cdot \Longrightarrow A$ true is not derivable in the given generality.

$\vdash \quad A \supset (K \texttt{ says } A)$
$\vdash \quad (K \texttt{ says } (A \supset B)) \supset (K \texttt{ says } A) \supset (K \texttt{ says } B)$
$\vdash \quad (K \texttt{ says } (K \texttt{ says } A)) \supset (K \texttt{ says } A)$
$\not\vdash \quad (K \texttt{ says } A) \supset A$
$\not\vdash \quad (K \texttt{ says } \bot) \supset \bot$
$\not\vdash \quad (K_1 \texttt{ says } A) \supset (K_2 \texttt{ says } A)$

The fourth proposition, $(K \texttt{ says } A) \supset A$ is an example of something not true in general where particular instances may either be true (for example, the one right above) or false (for example, the one just below).

It is very easy to show that the last three propositions are not provable: try to construct a sequent derivation using the given rules and the proof attempt will fail after one or two steps. This demonstrates, for example, that even contradictory statements by a principal K do not imply inconsistency of the logic.

2.3. Quantification

First-order quantification can now be added in a standard and straightforward way. We introduce sorts s the extent of which is open-ended, except that we explicitly postulate a sort principal of principals K. Quantification over principals is necessary, for example, to model groups and other, more complex, access control mechanisms. We use c for parameters introduced in a derivation and t for general terms. We track all constants and parameters that may occur in a sequent in a signature Σ which has the form $c_1{:}s_1, \ldots, c_n{:}s_n$, and in which no constant may be declared more than once. If necessary we choose names so as to avoid re-declaration. We write $\Sigma \vdash t : s$ for the standard judgment that term t has sort s. We further assume that all predicate and function symbols prescribe sorts for their arguments, and that their usage respects these declarations. We write $A(x)$ to denote a proposition A with possible occurrences of the variable x and $A(t)$ for the result of replacing all occurrences of x by t.

The main sequent now has the form $\Sigma; \Gamma \Longrightarrow A$ true for a signature Σ, context Γ, and proposition A. Σ is added to all prior rules, but never changes from conclusion to premises. In addition, we have the following right and left rules for universal quantification, $\forall x{:}s.\, A(x)$, which binds the variable x.

$$\frac{\Sigma, c{:}s; \Gamma \Longrightarrow A(c) \text{ true}}{\Sigma; \Gamma \Longrightarrow (\forall x{:}s.\, A(x)) \text{ true}}\ \forall\text{R}$$

$$\frac{\Sigma \vdash t : s \qquad \Sigma; \Gamma, A(t) \text{ true} \Longrightarrow \gamma}{\Sigma; \Gamma, (\forall x{:}s.\, A(x)) \text{ true} \Longrightarrow \gamma}\ \forall\text{L}$$

For $\Sigma, c{:}s$ to be well-formed in the $\forall$R rule, c must be new which is therefore an implicit side condition.

2.4. Meta-Theory

The meta-theory of the logic is relatively straightforward given the careful justification of the rules. We assume here weakening as well as a substitution property for sorts: *If* $\Sigma \vdash t : s$ *and* $\Sigma, c{:}s \vdash t' : s'$ *then* $\Sigma \vdash [t/c]t' : s'$*, where* $[t/c]t'$ *denotes the result of substituting* t *for* c *in* t'.

Theorem 2

1. *(Weakening) If* $\Sigma; \Gamma \Longrightarrow \gamma$ *then* $\Sigma; \Gamma, A \text{ true} \Longrightarrow \gamma$ *and* $\Sigma, c{:}s; \Gamma \Longrightarrow \gamma$.
2. *(Identity)* $\Sigma; \Gamma, A \text{ true} \Longrightarrow A$ true *for any proposition* A.
3. *(Substitution) If* $\Sigma \vdash t : s$ *and* $\Sigma, c{:}s; \Gamma \Longrightarrow \gamma$ *then* $\Sigma; [t/c]\Gamma \Longrightarrow [t/c]\gamma$
4. *(Cut) If* $\Sigma; \Gamma \Longrightarrow A$ true *and* $\Sigma; \Gamma, A \text{ true} \Longrightarrow \gamma$ *then* $\Sigma; \Gamma \Longrightarrow \gamma$.
5. *(Affirmation Cut) If* $\Sigma; \Gamma \Longrightarrow K$ affirms A *and* $\Sigma; \Gamma, A \text{ true} \Longrightarrow K$ affirms C *then* $\Sigma; \Gamma \Longrightarrow K$ affirms C.

Proof. Weakening follows by induction on the structure of the given derivation. Identity follows by induction on the structure of A. Substitution follows by induction on the structure of the second given derivation, using our assumption about the substitution property for sorting as needed.

Cut and affirmation cut follow simultaneously by nested induction, first on the structure of the proposition A, second on the structure of the two given derivations. □

The structure of the proof of the cut principles follows a prior proof by the second author [20] and has been formalized in the Twelf meta-logical framework. The cut elimination theorem proper, namely that if we formulate cut as an inference rule it can be eliminated from any sequent derivation, follows by a straightforward induction from part 4 above, as in [20]. The judgmental methodology entails a formulation where the definitions of the connectives are independent of one another and depend only on the underlying judgments. This means it is straightforward to add other

$$\frac{}{\Sigma;\Gamma, P\ \mathsf{true} \Longrightarrow P\ \mathsf{true}}\ \text{INIT}$$

$$\frac{\Sigma;\Gamma \Longrightarrow A\ \mathsf{true} \qquad \Sigma \vdash K : \mathsf{principal}}{\Sigma;\Gamma \Longrightarrow K\ \mathsf{affirms}\ A}\ \text{AFF}$$

$$\frac{\Sigma;\Gamma, A\ \mathsf{true} \Longrightarrow B\ \mathsf{true}}{\Sigma;\Gamma \Longrightarrow A \supset B\ \mathsf{true}}\ \supset\text{R}$$

$$\frac{\Sigma;\Gamma \Longrightarrow A\ \mathsf{true} \qquad \Sigma;\Gamma, B\ \mathsf{true} \Longrightarrow \gamma}{\Sigma;\Gamma, A \supset B\ \mathsf{true} \Longrightarrow \gamma}\ \supset\text{L}$$

no $\bot$ right rule $\qquad\qquad \dfrac{}{\Sigma;\Gamma, \bot\ \mathsf{true} \Longrightarrow \gamma}\ \bot\text{L}$

$$\frac{\Sigma;\Gamma \Longrightarrow K\ \mathsf{affirms}\ A}{\Sigma;\Gamma \Longrightarrow (K\ \mathtt{says}\ A)\ \mathsf{true}}\ \mathtt{says}\text{R}$$

$$\frac{\Sigma;\Gamma, A\ \mathsf{true} \Longrightarrow K\ \mathsf{affirms}\ C}{\Sigma;\Gamma, (K\ \mathtt{says}\ A)\ \mathsf{true} \Longrightarrow K\ \mathsf{affirms}\ C}\ \mathtt{says}\text{L}$$

$$\frac{\Sigma, c{:}s;\Gamma \Longrightarrow A(c)\ \mathsf{true}}{\Sigma;\Gamma \Longrightarrow (\forall x{:}s.\, A(x))\ \mathsf{true}}\ \forall\text{R}^c$$

$$\frac{\Sigma \vdash t : s \qquad \Sigma;\Gamma, A(t)\ \mathsf{true} \Longrightarrow \gamma}{\Sigma;\Gamma, (\forall x{:}s.\, A(x))\ \mathsf{true} \Longrightarrow \gamma}\ \forall\text{L}$$

Principal propositions of left rules implicitly remain available in all premises. γ stands for K affirms C or C true

Figure 1. Rules for authorization logic

connectives, such as conjunction, disjunction, or existential quantification while preserving theorem 2. We omit the details.

The logic itself is related to lax logic [14] in the formulation of Pfenning and Davies [21], except that we use a sequent calculus here instead of natural deduction. For each principal K, the judgment K affirms A is a judgment of lax truth, with no direct interaction between different principals. Hence, each "K says " forms a strong monad [19] as familiar from functional programming [24]. There, monads provide a means of isolating pure computations from effects; here the monads isolate the intentions of the principals from each other. The Dependency Core Calculus (DCC) [3] uses a family of strong monads indexed by elements of a lattice to model security levels in systems in a generic manner. Non-interference theorems similar to our property 1 (section 3) hold in this calculus also.

Authorization logics have sometimes been characterized as epistemic logics or belief logics, where K says A is interpreted as "K believes A" or "K knows A". We believe that this is not quite the right view. In authorization logics, truth and intent are freely shared. There are no secrets. Instead, the logic serves to relate truth and the intents of different principals. For example, $A \supset (K\ \mathtt{says}\ A)$ is true in our authorization logic and expresses that everyone is prepared to affirm true propositions. On the other hand, in epistemic logics it is *not* the case that $A \supset \Box_K A$: not everyone knows A just because it is true.

3. Non-Interference Properties

Non-interference properties, in our context, characterize classes of propositions whose presence or absence has no effect on proofs of certain other propositions. They can be used by policy administrators and users to explore the consequences and verify the correctness of policies expressed in the logic. In this section, we develop and prove two such properties. These are theorems of the following form: $\Sigma;\Gamma, A \Longrightarrow \gamma$ if and only if $\Sigma;\Gamma \Longrightarrow \gamma$, provided some *non-interference condition* holds on Σ, Γ, A and γ.[1] The two properties differ in the non-interference condition that must hold. In general, there is a trade-off between the expressiveness of the property and the ease of verifying the associated non-interference condition.

We use the term *formula* to mean a proposition or an affirmation $(K\ \mathsf{affirms}\ A)$. Before we describe our non-interference properties, we develop the notion of *signs* on formulas. Given a formula A, we can associate a sign $g \in \{+, -\}$ to it. This is written as A^g. The notion of subformula (denoted by $\leq$) extends to signed subformulas and affirmations by the inductive rules given below. We use g, g', g'' to denote arbitrary signs and $\bar{g}$ to denote the sign complementary to g.

$$A^g \leq A^g \qquad \frac{A^g \leq B^{g'} \qquad B^{g'} \leq C^{g''}}{A^g \leq C^{g''}}$$

$$A^{\bar{g}} \leq (A \supset B)^g \qquad B^g \leq (A \supset B)^g$$

$$A(t)^- \leq (\forall x{:}s.A(x))^- \qquad A(c)^+ \leq (\forall x{:}s.A(x))^+$$

$$(K\ \mathsf{affirms}\ A)^g \leq (K\ \mathtt{says}\ A)^g \qquad A^g \leq (K\ \mathsf{affirms}\ A)^g$$

We say that A^g is a signed subformula of a sequent $\Sigma;\Gamma \Longrightarrow \gamma$ if either $A^g \leq \gamma^+$ or for some $B \in \Gamma$, $A^g \leq B^-$. The sign of a subformula of a sequent completely determines whether the subformula will occur as an assumption or a conclusion in a cut-free proof of the sequent. This is formalized by lemma 2.

[1] We drop the judgment true from A true and simply write A because this does not cause any ambiguity.

Lemma 2 (Signed subformula property) *If* $\Sigma'; \Gamma', A \implies \gamma'$ *is a sequent occurring in a cut-free proof of* $\Sigma; \Gamma \implies \gamma$*, then* A^- *and* γ'^+ *are signed subformulas of* $\Sigma; \Gamma \implies \gamma$*.*

Our first non-interference property (shown below) is simple and probably obvious, but it has important consequences in the authorization logic. In particular, it can be used to *formally* show that statements made by principals other than those mentioned explicitly in a policy have no consequence on decisions made using the policy.

Property 1 (First non-interference property) *Suppose* K *does not occur free in* Γ *and* γ*, and the sequent* $\Sigma; \Gamma \implies \gamma$ *does not have any signed subformula of the form* $(\forall x{:}\mathsf{principal}.A(x))^-$*. Then* $\Sigma; \Gamma, K \texttt{ says } B \implies \gamma$ *if and only if* $\Sigma; \Gamma \implies \gamma$*.*

Proof. In the "if" direction by weakening and "only if" direction by induction on the given derivation. □

Example 1. Figure 2 shows the policies of three principals: a company BCL, its parent company BigCo and a service company S that assesses if employees of BigCo work hard.[2] Let $\Sigma_e; \Gamma_e$ represent this policy in our logic, i.e., let $\Sigma_e = \{\mathsf{BCL} : \mathsf{principal},\ \mathsf{BigCo} : \mathsf{principal},\ \mathsf{S} : \mathsf{principal},\ john : \mathsf{person}\}$ and let Γ_e be the set of all five policy statements in figure 2. Then for any principal K distinct from BigCo, BCL and S, and for every x, K does not occur in Γ_e and S says $\mathsf{employee}(x, \mathsf{BigCo})$. Also, there are no negative occurrences of a universal quantifier over principals in $\Sigma_e; \Gamma_e \implies \mathsf{S}$ says $\mathsf{employee}(x, \mathsf{BigCo})$. We can apply property 1 to conclude that $\Sigma_e; \Gamma_e, K$ says $B \implies \mathsf{S}$ says $\mathsf{employee}(x, \mathsf{BigCo})$ if and only if $\Sigma_e; \Gamma_e \implies \mathsf{S}$ says $\mathsf{employee}(x, \mathsf{BigCo})$. Thus, statements made by principals other than those occurring explicitly in the policy have no effect on conclusions drawn by S regarding who the employees of BigCo are. The same analysis works if we replace S by BigCo or BCL.

Property 1 is too weak to let us conclude anything about hypotheses of the form BigCo says B, BCL says B and S says B, because each of these three principals occurs in the policy. The property can be used to show that "irrelevant" principals (those that are not mentioned in a policy) cannot inadvertently or maliciously affect decisions made from the policy.

3.1. Non-Interference Through Affirmation Flow Analysis

In many cases, we can examine a policy to determine pairs of principals (K_1, K_2) such that an affirmation of the truth of some formula by K_1 may result in the affirmation of truth of some other formula by K_2. For example, consider a policy which contains the formulas K_1 says $A \supset K_2$ says B and K_2 says $B' \supset K_3$ says C. Suppose K_1 makes an assertion of the form K_1 says A'. Then, up to a very coarse approximation, it is possible that this assertion will lead to the truth of K_2 says B, which might result in the truth of K_3 says C. In this case we say that *affirmation may flow* from K_1 to K_3. The related analysis is called an *affirmation flow analysis*.

If we can determine that affirmation cannot flow from K_1 to K_3 in some policy, then we can safely say that statements made by K_1 will not affect statements concluded by K_3 using the policy. Such a result has a strong flavor of a non-interference property. In this subsection, we will formalize the idea of affirmation flow analysis and use it to obtain a non-interference property. As it turns out, it is not enough to consider just principals for affirmation flow analysis. We will consider a richer set of symbols over which we perform the analysis.

Throughout this subsection, we impose two restrictions on the logical formulas we consider. First, we assume that there are no positive occurrences of the universal quantifier over principals in our formulas. This may appear to be a big restriction, but we are yet to come across any policy that uses such a quantification in its encoding. The reason for this is that a positively occurring universal quantifier expresses *creation* of fresh principals. Even though the implementation of a policy may require creation of principals, the specification usually does not. Second, we assume that first order function symbols do not generate principals. This also does not appear to be a significant restriction in practice. Together these restrictions imply that if a principal K occurs in a cut-free proof of the sequent $\Sigma; \Gamma \implies \gamma$, then $K{:}\mathsf{principal} \in \Sigma$.

Policy of BCL:
  BCL says $\mathsf{employee}(john, \mathsf{BCL})$

Policy of BigCo:
  BigCo says $\forall x : \mathsf{person}.$
    $(\mathsf{BCL}$ says $\mathsf{employee}(x, \mathsf{BCL}) \supset \mathsf{employee}(x, \mathsf{BCL}))$
  BigCo says $\forall x : \mathsf{person}.$
    $(\mathsf{employee}(x, \mathsf{BCL}) \supset \mathsf{employee}(x, \mathsf{BigCo}))$
  BigCo says $\forall x : \mathsf{person}.$
    $(\mathsf{S}$ says $\mathsf{workshard}(x) \supset \mathsf{workshard}(x))$

Policy of S:
  S says $\forall x : \mathsf{person}.$
    $(\mathsf{BigCo}$ says $\mathsf{employee}(x, \mathsf{BigCo}) \supset \mathsf{employee}(x, \mathsf{BigCo}))$

Figure 2. Policies of principals for example 1

[2] This example is based on an example in [13].

We now formalize our non-interference property based on affirmation flow analysis. Let P denote predicate symbols. First we define the class of *symbols* L.

$$L ::= P \mid \bot \mid K.L$$

During our analysis, we use sets of symbols to abstract over formulas. Briefly, the non-interference property works as follows. We define a function $\mathtt{ps}$, indexed by signatures Σ, that maps a formula A to a set of symbols $\mathtt{ps}_\Sigma(A)$. Informally speaking, this set contains every possible symbol $K_1 \ldots K_n.P$, such that the truth of A may imply the truth of K_1 says $\ldots K_n$ says P. Next, given a sequent $\Sigma; \Gamma \implies \gamma$, we construct an *affirmation flow pre-order* ($\ll$) between symbols based on this sequent. The non-interference property says that if for some $A \in \Gamma$, it is the case that there *do not* exist $L_1 \in \mathtt{ps}_\Sigma(A)$ and $L_2 \in \mathtt{ps}_\Sigma(\gamma)$ such that $L_1 \ll L_2$, then A has no effect on the conclusion γ, i.e., $\Sigma; \Gamma \implies \gamma$ if and only if $\Sigma; \Gamma \backslash A \implies \gamma$.

Let $\mathcal{L}$ denote a set of symbols. If $\mathcal{L} = \{L_1, \ldots, L_n\}$, we define $K.\mathcal{L}$ to be the set $\{K.L_1, \ldots, K.L_n\}$. Now we define the Σ-indexed map $\mathtt{ps}$ from formulas to sets of symbols as follows.

$$\begin{array}{lcll}
\mathtt{ps}_\Sigma(\bot) & = & \{\bot\} & \\
\mathtt{ps}_\Sigma(P\, t_1 \ldots t_n) & = & \{P\} & \\
\mathtt{ps}_\Sigma(A \supset B) & = & \mathtt{ps}_\Sigma(B) & \\
\mathtt{ps}_\Sigma(K \text{ says } A) & = & K.\mathtt{ps}_\Sigma(A) & \\
\mathtt{ps}_\Sigma(\forall x{:}s.A(x)) & = & \mathtt{ps}_\Sigma(A(x)) & (s \neq \mathsf{principal}) \\
\mathtt{ps}_\Sigma(\forall x{:}\mathsf{principal}.A(x)) & = & \bigcup_{(K:\mathsf{principal} \in \Sigma)} \mathtt{ps}_\Sigma(A(K)) & \\
\mathtt{ps}_\Sigma(K \text{ affirms } A) & = & K.\mathtt{ps}_\Sigma(A) &
\end{array}$$

Next, we develop a sequent calculus that lets us reason with affirmation flow between symbols. This calculus is the basis of our affirmation flow analysis. The calculus works with ordering formulas F, defined by the following grammar.

$$F ::= L_1 \preceq L_2 \mid K.F$$

$L_1 \preceq L_2$ is an assertion that affirmation may flow from the symbol L_1 to the symbol L_2. $K.F$ is an assertion of F, which can be used only when we are reasoning about statements made by principal K. We use Φ to denote multisets of ordering formulas. Figure 3 describes the sequent calculus for reasoning with ordering formulas. It uses only one judgment: $\Phi \Rightarrow L_1 \preceq L_2$. This judgment should be read as follows: given that the assumptions in Φ hold, we can conclude that affirmation may flow from L_1 to L_2.

Theorem 3 *The sequent calculus of figure 3 satisfies the following properties.*

1. *(Weakening) If* $\Phi \Rightarrow L_1 \preceq L_2$*, then* $\Phi, F \Rightarrow L_1 \preceq L_2$.
2. *(Contraction) If* $\Phi, F, F \Rightarrow L_1 \preceq L_2$*, then* $\Phi, F \Rightarrow L_1 \preceq L_2$.

$$\frac{}{\Phi \Rightarrow P \preceq P} \qquad \frac{}{\Phi \Rightarrow \bot \preceq L} \qquad \frac{\Phi \Rightarrow L \preceq L'}{\Phi \Rightarrow L \preceq K.L'}$$

$$\frac{\Phi \Rightarrow L \preceq K.L'}{\Phi \Rightarrow K.L \preceq K.L'} \qquad \frac{\Phi, K.F, F \Rightarrow L \preceq K.L'}{\Phi, K.F \Rightarrow L \preceq K.L'}$$

$$\frac{\Phi, L_1 \preceq L_2 \Rightarrow L_3 \preceq L_1 \qquad \Phi, L_1 \preceq L_2 \Rightarrow L_2 \preceq L_4}{\Phi, L_1 \preceq L_2 \Rightarrow L_3 \preceq L_4}$$

Figure 3. Sequent calculus for reasoning with ordering formulas

3. *(Transitivity) If* $\Phi \Rightarrow L_1 \preceq L_2$ *and* $\Phi \Rightarrow L_2 \preceq L_3$*, then* $\Phi \Rightarrow L_1 \preceq L_3$.
4. *(Admissibility of cut) If* $\Phi \Rightarrow L_1 \preceq L_2$ *and* $\Phi, L_1 \preceq L_2 \Rightarrow L \preceq L'$*, then* $\Phi \Rightarrow L \preceq L'$.
5. *(Reflexivity)* $\Phi \Rightarrow L \preceq L$.
6. *(Identity)* $\Phi, L \preceq L' \Rightarrow L \preceq L'$.

Proof. Weakening and contraction follow by induction on the given derivations. Transitivity can be proved by a simultaneous induction on both given derivations. Admissibility of cut follows by induction on the second given derivation, using transitivity as needed. Reflexivity can be proved by induction on L and identity follows from reflexivity using one application of the last rule in figure 3. □

The sequent calculus of figure 3 is designed to make it easy to prove the above theorem and property 2. However, it is not immediate that this calculus is decidable. This can be proved by constructing an alternate sequent calculus equivalent to this one, as shown in Appendix A.

Theorem 4 *The sequent calculus of figure 3 is decidable, i.e., given any* Φ*,* L_1 *and* L_2*, it is decidable whether* $\Phi \Rightarrow L_1 \preceq L_2$ *or not.*

Proof. See Appendix A. □

If $\Phi = \{F_1, \ldots, F_n\}$, we define $K.\Phi = \{K.F_1, \ldots, K.F_n\}$. Next we define a function that obtains the affirmation flow information of a signed formula. Formally, this is a Σ-indexed function that maps a signed formula to a set of ordering formulas Φ. We call this function $\mathcal{AR}_\Sigma$. It is described below. Note that there is no rule defining $\mathcal{AR}_\Sigma((\forall x{:}\mathsf{principal}.A(x))^+)$ because we have assumed that there are no positive occurrences of the universal quantifier over principals.

$$\mathcal{AR}_\Sigma(\bot^g) = \{\}$$
$$\mathcal{AR}_\Sigma((P\ t_1 \ldots t_n)^g) = \{\}$$
$$\mathcal{AR}_\Sigma((A \supset B)^+) = \mathcal{AR}_\Sigma(A^-) \cup \mathcal{AR}_\Sigma(B^+)$$
$$\mathcal{AR}_\Sigma((A \supset B)^-) = \mathcal{AR}_\Sigma(A^+) \cup \mathcal{AR}_\Sigma(B^-) \cup \{L_1 \preceq L_2 \mid L_1 \in \mathtt{ps}_\Sigma(A), L_2 \in \mathtt{ps}_\Sigma(B)\}$$
$$\mathcal{AR}_\Sigma((K \text{ says } A)^g) = K.\mathcal{AR}_\Sigma(A^g)$$
$$\mathcal{AR}_\Sigma((\forall x{:}s.A(x))^g) = \mathcal{AR}_\Sigma(A(x)^g) \quad s \neq \mathsf{principal}$$
$$\mathcal{AR}_\Sigma((\forall x{:}\mathsf{principal}.A(x))^-) = \bigcup_{(K:\mathsf{principal}\in\Sigma)} \mathcal{AR}_\Sigma(A(K)^-)$$
$$\mathcal{AR}_\Sigma((K \text{ affirms } A)^+) = K.\mathcal{AR}_\Sigma(A^+)$$

Finally we define the *affirmation flow pre-order* ($\ll$) of a sequent $\Sigma; \Gamma \implies \gamma$. Let $\Gamma = A_1, \ldots, A_n$ and $\Phi = \mathcal{AR}_\Sigma(A_1^-) \cup \ldots \cup \mathcal{AR}_\Sigma(A_n^-) \cup \mathcal{AR}_\Sigma(\gamma^+)$. Then the affirmation flow pre-order of this sequent is the binary relation $\ll$ between symbols such that $L_1 \ll L_2$ if and only if $\Phi \Rightarrow L_1 \preceq L_2$. Note that $\ll$ is a pre-order because of the reflexivity and transitivity properties from theorem 3.

Property 2 (Second non-interference property) *Let $\ll$ be the affirmation flow relation for the sequent $\Sigma; \Gamma, A \implies \gamma$, and suppose that there* do not *exist $L_1 \in \mathtt{ps}_\Sigma(A)$ and $L_2 \in \mathtt{ps}_\Sigma(\gamma)$ such that $L_1 \ll L_2$. Then $\Sigma; \Gamma, A \implies \gamma$ if and only if $\Sigma; \Gamma \implies \gamma$.*

Proof. By weakening in the "if" direction and by induction on the structure of the derivation $\Sigma; \Gamma, A \implies \gamma$ in the other direction. □

Example 2. Let $\Sigma_e; \Gamma_e$ represent the policy in figure 2. Consider the sequent $\Sigma_e; \Gamma_e, \mathsf{BigCo}\ \mathtt{says}\ \mathsf{employee}(x, y) \implies \mathsf{BCL}\ \mathtt{says}\ \mathsf{employee}(z, u)$ for any x, y, z, u. Then the affirmation flow pre-order of this sequent is $L_1 \ll L_2$ if and only if $\Phi \Rightarrow L_1 \preceq L_2$, where

$$\begin{aligned}\Phi = \{&\mathsf{BigCo}.(\mathsf{BCL}.\mathsf{employee} \preceq \mathsf{employee}),\\ &\mathsf{BigCo}.(\mathsf{employee} \preceq \mathsf{employee}),\\ &\mathsf{BigCo}.(\mathsf{S}.\mathsf{workshard} \preceq \mathsf{workshard})\\ &\mathsf{S}.(\mathsf{BigCo}.\mathsf{employee} \preceq \mathsf{employee})\}\end{aligned}$$

By definition $\mathtt{ps}_{\Sigma_e}(\mathsf{BigCo}\ \mathtt{says}\ \mathsf{employee}(x, y)) = \{\mathsf{BigCo}.\mathsf{employee}\}$ and $\mathtt{ps}_{\Sigma_e}(\mathsf{BCL}\ \mathtt{says}\ \mathsf{employee}(z, u)) = \{\mathsf{BCL}.\mathsf{employee}\}$. It is very easy to show using the rules of figure 3 that $\Phi \not\Rightarrow \mathsf{BigCo}.\mathsf{employee} \preceq \mathsf{BCL}.\mathsf{employee}$. Consequently by property 2, we can conclude that $\Sigma_e; \Gamma_e, \mathsf{BigCo}\ \mathtt{says}\ \mathsf{employee}(x, y) \implies \mathsf{BCL}\ \mathtt{says}\ \mathsf{employee}(z, u)$ if and only if $\Sigma_e; \Gamma_e \implies \mathsf{BCL}\ \mathtt{says}\ \mathsf{employee}(z, u)$, which is to say that statements regarding the predicate employee by BigCo do not influence the truth of similar statements by BCL.

Property 2 applies in a large number of cases, but verifying its associated non-interference condition requires reasoning with the sequent calculus in figure 3, which can be cumbersome in some cases, if done manually. We see this property as a valuable tool for reasoning about policies expressed in our logic using an automated decision procedure for the sequent calculus in figure 3, which is decidable (theorem 4). Efficient implementation of such a decision procedure is a subject of immediate future work. The next example describes a more complicated policy where property 2 can be used.

Policy of K_a:
K_a says $\mathtt{isHospital}(K_c)$

K_a says $\mathtt{isHospital}(K_d)$

K_a says $\forall x, y : \mathsf{person}.$
$\mathtt{isPhysicianOf}(x, y) \supset \mathtt{readMedRec}(x, y)$

K_a says $\forall x, y : \mathsf{person}.\ \forall k : \mathsf{principal}.$
$\mathtt{isHospital}(k) \supset$
$(k$ says $\mathtt{isPhysicianOf}(x, y)) \supset$
$\mathtt{isPhysicianOf}(x, y)$

K_a says $\forall k_1, k_2, k : \mathsf{principal}.$
$\mathtt{isHospital}(k_1) \supset \mathtt{isHospital}(k_2) \supset$
$(k_1$ says $\mathtt{isHospital}(k)) \supset (k_2$ says $\mathtt{isHospital}(k)) \supset$
$\mathtt{isHospital}(k)$

Policy of K_b:
K_b says $\mathtt{isPhysicianOf}(alice, peter)$

Policy of K_c:
K_c says $\mathtt{isHospital}(K_b)$

Policy of K_d:
K_d says $\mathtt{isHospital}(K_b)$

Figure 4. Policies of four hospitals

Example 3. The policies of four hospitals K_a, K_b, K_c and K_d are shown in figure 4.[3] The policies govern physicians' access to medical records of patients. The proposition $\mathtt{readMedRec}(x, y)$ means that x can read the medical records of y. K_a's policy can be summarized as follows. K_a believes that the principals K_c and K_d are hospitals. K_a grants x access to y's medical records if it believes that x is y's physician. K_a trusts all hospitals' statements regarding physician-patient relationships. Finally, if two hospitals affirm that a principal is a hospital, K_a is willing to believe this claim. It is quite easy to show that K_a says $\mathtt{readMedRec}(alice, peter)$ is provable with this policy.

According to K_a's policy, it believes other hospitals' statements regarding the predicates `isHospital` and `isPhysicianOf` only. It does not directly trust other hospitals' statements regarding the predicate `readMedRec`. In particular, if K_b says $\mathtt{readMedRec}(x, y)$ for any persons x, y, then this should not influence the provability

[3] This example is based on an example in [23].

of K_a says $\texttt{readMedRec}(x, y)$. Formally, let $\Sigma_e = \{K_a : \mathsf{principal}, K_b : \mathsf{principal}, K_c : \mathsf{principal}, K_d : \mathsf{principal}, alice : \mathsf{person}, peter : \mathsf{person}\}$ be the set of all terms under consideration and let Γ_e be the set of all policy statements in figure 4. Then we want to show that for any x, y, $\Sigma_e; \Gamma_e, K_b$ says $\texttt{readMedRec}(x, y) \Longrightarrow K_a$ says $\texttt{readMedRec}(x, y)$ only if $\Sigma_e; \Gamma_e \Longrightarrow K_a$ says $\texttt{readMedRec}(x, y)$.

We can prove this result using property 2. Consider the affirmation flow pre-order of the sequent $\Sigma_e; \Gamma_e, K_b$ says $\texttt{readMedRec}(x, y) \Longrightarrow K_a$ says $\texttt{readMedRec}(x, y)$. This pre-order is $L_1 \ll L_2$ if and only if $\Phi \Rightarrow L_1 \preceq L_2$, where

$$\begin{aligned}\Phi = \quad & \{K_a.(\texttt{isPhysicianOf} \preceq \texttt{readMedRec}),\\ & K_a.(\texttt{isHospital} \preceq \texttt{isPhysicianOf}),\\ & K_a.(K_a.\texttt{isHospital} \preceq \texttt{isPhysicianOf}),\\ & K_a.(K_b.\texttt{isHospital} \preceq \texttt{isPhysicianOf}),\\ & K_a.(K_c.\texttt{isHospital} \preceq \texttt{isPhysicianOf}),\\ & K_a.(K_d.\texttt{isHospital} \preceq \texttt{isPhysicianOf}),\\ & K_a.(\texttt{isHospital} \preceq \texttt{isHospital}),\\ & K_a.(K_a.\texttt{isHospital} \preceq \texttt{isHospital}),\\ & K_a.(K_b.\texttt{isHospital} \preceq \texttt{isHospital}),\\ & K_a.(K_c.\texttt{isHospital} \preceq \texttt{isHospital}),\\ & K_a.(K_d.\texttt{isHospital} \preceq \texttt{isHospital})\}\end{aligned}$$

We observe that $\texttt{ps}_{\Sigma_e}(K_a$ says $\texttt{readMedRec}(x, y)) = \{K_a.\texttt{readMedRec}\}$ and $\texttt{ps}_{\Sigma_e}(K_b$ says $\texttt{readMedRec}(x, y)) = \{K_b.\texttt{readMedRec}\}$. Thus, if we can show that $\Phi \not\Rightarrow K_b.\texttt{readMedRec} \preceq K_a.\texttt{readMedRec}$, we can use property 2 to conclude that $\Sigma_e; \Gamma_e, K_b$ says $\texttt{readMedRec}(x, y) \Longrightarrow K_a$ says $\texttt{readMedRec}(x, y)$ if and only if $\Sigma_e; \Gamma_e \Longrightarrow K_a$ says $\texttt{readMedRec}(x, y)$. The former can be shown using the sequent calculus in figure 3, but this is a tedious exercise to perform manually. Using an automated decision procedure for this calculus would enable verification of this fact easily.

4. Further Related Work

We only touch here upon some of the most closely related work. Further pointers to the literature can be found in a survey by Abadi [2].

The study of access control with logical means was initiated by Abadi et al. [4]. Their system was a classical propositional logic with a rich algebra of principals. They presented a Kripke semantics for which the axioms and inference rules were sound. However, they were complete only for a fragment. The system was axiom-based and no proof-theoretic analysis or non-interference properties were given. Here we avoid a complex calculus of principals by allowing first-order quantification over principals to express concepts such as groups. This comes at the cost of a priori undecidability, which, however, has not been a problem in the setting of proof-carrying authorization [5, 6] (PCA), where distributed theorem proving procedures appear to be efficient enough in practice [9]. We believe that it is possible to extend our logic with further judgments or proposition to deal with more complex principals where a mapping to first-order quantification may be undesirable (as perhaps in [1]).

Another concept studied by Abadi et al. is delegation of rights among principals. Delegation of rights related to specific predicates can be readily encoded in the logic we have presented. There is also a more general form of delegation where a principal delegates another principal the right to make *any* statements on its behalf. While there is a method to systematically extend our logic to include such delegation, we do not present it here, because we believe that such delegation is a security risk for policies. The problem with general delegation is that it extends to all predicates, including those that the delegating principal may not be aware of.

The logic underlying proof-carrying authorization [5, 6] has been explicitly designed as a (classical) higher-order logic. This makes a meta-theoretic analysis of the logic and policies expressed in it extremely difficult because proof calculi will not obey the subformula property. Besides an argument for consistency, no analysis of non-interference properties of this logic has been provided. We advocate a more tractable logic such as the one presented here since the logical calculus is part of the trusted computing base of a PCA architecture implementation such as the Grey system [8] and should therefore be easy to understand and analyze, ideally with formal verification tools as we have done.

A logic that more closely resembles ours is the one underlying Binder [13]. Even though Binder is first-order, queries are decidable in polynomial time because it is derived from the Datalog fragment of first-order logic. The Binder logic appears to be intuitionistic because on Horn formulas intuitionistic and classical logic coincide. However, the logical status of the modal operators, their non-interference, or other properties of proofs are not analyzed. We conjecture that a similarly decidable Datalog-inspired restriction exists for our logic and plan to investigate this in future work. There are several other proposals [10, 17, 16, 12] for authorization logics based on Datalog and its extensions, but like Binder, they do not analyze the logical status of their operators, or non-interference.

Rueß and Shankar's Cyberlogic [23] is a more expressive logic in that one can reason about more than just authorization. It is constructed along similar lines as the Binder logic in that interesting fragments permit a logic programming interpretation and that it is intuitionistic, just like our logic. However, no analysis of *attestation* (the analogue of *affirmation*) in terms of standard modal logic concepts or proof theory is provided, and some of the axioms such as distribution of attestation over disjunction and implication are ques-

tionable from our point of view. Proving non-interference properties for Cyberlogic appears to be difficult.

5. Conclusion

We have presented a new constructive authorization logic developed from judgmental principles, which yields a clean proof theory, an analysis of the meaning of the connectives from their proof rules, and is inherently open-ended and extensible. We have shown that the logic itself satisfies some non-interference properties between principals and provided some high-level tools for analyzing policies expressed in the authorization logic. We believe it is a promising logic for reasoning about authorization in general, and also a good foundation for the implementation of a proof-carrying authorization infrastructure where policy enforcement is directly based on proof objects.

There are several avenues for future work besides those mentioned with the related work. One is to consider whether one should integrate certificate revocation in a more logical manner than standard techniques using revocation lists or short-lived certificates. Another is whether explicit reasoning about time should be integrated into the logic via temporal operators. Either of these might change the character of the logic significantly and complicate its analysis and the interpretation of the statements made in it.

Another promising direction is the integration of *linear reasoning* which can support consumable credentials and therefore electronic transactions. Intuitionistic linear logic is fully compatible with lax logic [11, 25] and therefore with the family of monads indexed by principals we developed here. The logic design based on judgmental principles means that such an extension will be conservative over the logic presented here, so that the present non-interference properties will continue to hold. The difficulty then appears not to be the logical reasoning, but the design of an enforcement mechanism to support consumable credentials at the right level of atomicity that is consistent with the logic. See [7] for some initial developments in this direction.

Acknowledgments. We would like to thank Lujo Bauer, Kevin Bowers, Mike Reiter, and Kevin Watkins for discussions related to the topic of this paper.

References

[1] M. Abadi. On SDSI's linked local name spaces. *Journal of Computer Security*, 6(1–2):3–21, Oct. 1998.

[2] M. Abadi. Logic in access control. In *Proceedings of the 18th Annual Symposium on Logic in Computer Science (LICS'03)*, pages 228–233, Ottawa, Canada, June 2003. IEEE Computer Society Press.

[3] M. Abadi, A. Banerjee, N. Heintze, and J. G. Riecke. A core calculus of dependency. In *Conference Record of the 26th Sympoisum on Principles Of Programming Languages (POPL'99)*, pages 147–160, San Antonio, Texas, Jan. 1999. ACM Press.

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

[5] A. W. Appel and E. W. Felten. Proof-carrying authentication. In G. Tsudik, editor, *Proceedings of the 6th Conference on Computer and Communications Security*, pages 52–62, Singapore, Nov. 1999. ACM Press.

[6] L. Bauer. *Access Control for the Web via Proof-Carrying Authorization*. PhD thesis, Princeton University, Nov. 2003.

[7] L. Bauer, K. D. Bowers, F. Pfenning, and M. K. Reiter. Consumable credentials in logic-based access control. Technical Report CMU-CYLAB-06-002, Carnegie Mellon University, Feb. 2006.

[8] L. Bauer, S. Garriss, J. M. McCune, M. K. Reiter, J. Rouse, and P. Rutenbar. Device-enabled authorization in the Grey system. In *Proceedings of the 8th Information Security Conference (ISC'05)*, pages 431–445, Singapore, Sept. 2005. Springer Verlag LNCS 3650.

[9] L. Bauer, S. Garriss, and M. K. Reiter. Distributed proving in access-control systems. In V.Paxon and M.Waidner, editors, *Proceedings of the 2005 Symposium on Security and Privacy (S&P'05)*, pages 81–95, Oakland, California, May 2005. IEEE Computer Society Press.

[10] E. Bertino, B. Catania, E. Ferrari, and P. Perlasca. A logical framework for reasoning about access control models. *ACM Trans. Inf. Syst. Secur.*, 6(1):71–127, 2003.

[11] B.-Y. E. Chang, K. Chaudhuri, and F. Pfenning. A judgmental analysis of linear logic. Submitted. Extended version available as Technical Report CMU-CS-03-131R, Dec. 2003.

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

[13] J. DeTreville. Binder, a logic-based security language. In M.Abadi and S.Bellovin, editors, *Proceedings of the 2002 Symposium on Security and Privacy (S&P'02)*, pages 105–113, Berkeley, California, May 2002. IEEE Computer Society Press.

[14] M. Fairtlough and M. Mendler. Propositional lax logic. *Information and Computation*, 137(1):1–33, Aug. 1997.

[15] G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935. English translation in M. E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131, North-Holland, 1969.

[16] N. Li, 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.

[17] N. Li and J. C. Mitchell. Datalog with constraints: A foundation for trust management languages. In *PADL '03: Proceedings of the 5th International Symposium on Practical Aspects of Declarative Languages*, pages 58–73. Springer-Verlag, 2003.

$$\frac{}{\Phi \Rightarrow_D P \preceq P} \qquad \frac{}{\Phi \Rightarrow_D \bot \preceq L} \qquad \frac{\Phi \Rightarrow_D L \preceq L'}{\Phi \Rightarrow_D L \preceq K.L'}$$

$$\frac{\Phi \Rightarrow_D L \preceq K.L'}{\Phi \Rightarrow_D K.L \preceq K.L'} \qquad \frac{\Phi, F \Rightarrow_D L \preceq K.L'}{\Phi, K.F \Rightarrow_D L \preceq K.L'}$$

$$\frac{\Phi \Rightarrow_D L_3 \preceq L_1 \qquad \Phi \Rightarrow_D L_2 \preceq L_4}{\Phi, L_1 \preceq L_2 \Rightarrow_D L_3 \preceq L_4}$$

Figure 5. Modified sequent calculus for reasoning with ordering formulas

[18] P. Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60, 1996.

[19] E. Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, 1991.

[20] F. Pfenning. Structural cut elimination I. Intuitionistic and classical logic. *Information and Computation*, 157(1/2):84–141, Mar. 2000.

[21] F. Pfenning and R. Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001. Notes to an invited talk at the *Workshop on Intuitionistic Modal Logics and Applications* (IMLA'99), Trento, Italy, July 1999.

[22] F. Pfenning and C. Schürmann. System description: Twelf — a meta-logical framework for deductive systems. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction (CADE-16)*, pages 202–206, Trento, Italy, July 1999. Springer-Verlag LNAI 1632.

[23] H. Rueß and N. Shankar. Introducing Cyberlogic. In *Proceedings of the 3rd Annual High Confidence Software and Systems Conference*, Baltimore, Maryland, Apr. 2003.

[24] P. Wadler. The essence of functional programming. In *Conference Record of the 19th Symposium on Principles of Programming Languages*, pages 1–14, Albuquerque, Jan. 1992. ACM Press.

[25] K. Watkins, I. Cervesato, F. Pfenning, and D. Walker. A concurrent logical framework I: Judgments and properties. Technical Report CMU-CS-02-101, Department of Computer Science, Carnegie Mellon University, 2002. Revised May 2003.

A. Decidability of Ordering Formulas

This appendix shows that the sequent calculus for ordering formulas given in figure 3 is decidable. This is done in two steps. First we construct another sequent calculus for reasoning with ordering formulas and show that it is equivalent to the one given in figure 3. In the second step, we show that the new calculus is decidable.

The second calculus we consider is shown in figure 5. It differs from the first calculus only in the premises of the last two rules. We denote its sequents using the arrow $\Rightarrow_D$.

We first show that the calculi in figure 3 and 5 are equivalent in the sense that $\Phi \Rightarrow L_1 \preceq L_2$ if and only if $\Phi \Rightarrow_D L_1 \preceq L_2$. We need a few lemmas about the two calculi.

Lemma 3 (Properties of calculus in figure 3) *1. If* $\Phi, F, K.F \Rightarrow L_1 \preceq L_2$*, then* $\Phi, F \Rightarrow L_1 \preceq L_2$.

2. If $\Phi, L_1 \preceq L_2 \Rightarrow L_3 \preceq L_1$*, then* $\Phi \Rightarrow L_3 \preceq L_1$.

3. If $\Phi, L_1 \preceq L_2 \Rightarrow L_2 \preceq L_3$*, then* $\Phi \Rightarrow L_2 \preceq L_3$.

Proof. In each case by induction on the given derivation. □

Lemma 4 (Weakening for calculus in figure 5) *The calculus in figure 5 satisfies weakening: If* $\Phi \Rightarrow_D L_1 \preceq L_2$*, then* $\Phi, F \Rightarrow_D L_1 \preceq L_2$.

Lemma 5 (Equivalence) $\Phi \Rightarrow L_1 \preceq L_2$ *if and only if* $\Phi \Rightarrow_D L_1 \preceq L_2$.

Proof. In each direction by induction on the given derivation. The proof in the "only if" direction requires use of lemma 3. □

Lemma 6 *Given any* Φ, L_1, L_2*, it can be decided whether* $\Phi \Rightarrow_D L_1 \preceq L_2$ *or not.*

Proof. In the calculus in figure 5, the sequents in the premise of each rule have a strictly smaller size than the sequent of the conclusion. Further, given any sequent, there are only a finite number of rules that could have been used to conclude it. As a result, if we reason backward, then the calculus is decidable. □

Theorem 5 *Given any* Φ, L_1, L_2*, it can be decided whether* $\Phi \Rightarrow L_1 \preceq L_2$ *or not.*

Proof. Follows immediately from lemmas 5 and 6. □