Variations in Access Control Logic

Martín Abadi[1,2]

[1] University of California, Santa Cruz
[2] Microsoft Research, Silicon Valley

Abstract. In this paper we investigate the design space of access control logics. Specifically, we consider several possible axioms for the common operator `says`. Some of the axioms come from modal logic and programming-language theory; others are suggested by ideas from security, such as delegation of authority and the Principle of Least Privilege. We compare these axioms and study their implications.

1 Introduction

While access control appears in various guises in many aspects of computer systems, it is attractive to reduce it, as much as possible, to few central concepts and rules [17]. The development and use of general logics for access control is an ongoing effort in this direction. In this paper, we examine and compare several logics for access control.

The logics that we consider all have the same operators and intended applications, but they differ in their axioms and rules. They all start from propositional logic with the `says` operator, which is central in several theories and systems for access control (e.g., [4, 16, 20, 9, 5, 1, 12, 6, 8, 19, 14]). Moreover, they all allow the definition of a "speaks for" relation [4, 16, 18] from `says` and quantification: A speaks for B if, for every X, if A `says` X then B `says` X. In a formula A `says` s, the symbol A represents a principal and s represents a statement (such as a request or a delegation of authority). Intuitively, A `says` s means that A supports s, whether or not A has uttered s explicitly.

Perhaps because intuitive explanations of `says` are invariably loose and open-ended, the exact properties that `says` should satisfy do not seem obvious. The goal of this paper is to investigate the space of options, exploring the formal consequences and the security interpretations of several possible axiomatizations, and thus to help in identifying logics that are sufficiently strong but not inconsistent, degenerate, or otherwise unreasonable.

Some of the axioms that we study come from modal logic [15], computational lambda calculus [21], and other standard formal systems. Other axioms stem from ideas in security, such as delegations of authority and the Principle of Least Privilege [22]. For instance, we consider the hand-off axiom, which says that if A says that B speaks for A, then B does speak for A [16]. We evaluate these axioms in both classical and intuitionistic contexts.

More specifically, we start with the basic axioms of standard modal logic, in particular that `says` is closed under consequence (if A says s_1 and A says that

s_1 implies s_2, then A says s_2), together with the necessitation rule (if s is valid then A says s). In addition, the axioms that we consider include the following:

1. The hand-off axiom, as described above, and a generalization: if A says that s_1 implies A says s_2, then s_1 does imply A says s_2. In the special case where s_1 is B says s_2, we obtain a hand-off from A to B for s_2.
2. A further axiom that if A can make itself speak for B, then A speaks for B in the first place. This axiom may be seen roughly as a dual to the hand-off axiom.
3. The axiom that s implies A `says` s. This axiom is similar to the necessitation rule but stronger, and has been considered in access control in the past. It is also suggested by the computational lambda calculus. We call it Unit.
4. The other main axiom from the computational lambda calculus, which we call Bind: if s_1 implies A says s_2, then A says s_1 implies A says s_2.
5. The axiom that if A `says` s then s or A `says false`. We call this axiom Escalation, because it means that whenever A `says` s, either s is true or A says anything—-possibly statements intuitively "much falser" than s.
6. An axiom suggested by the Principle of Least Privilege, roughly that if a principal is trusted on a statement then it is also trusted on weaker statements.

We obtain the following results:

– In classical logics, the addition of axioms beyond the basic ones from modal logic quickly leads to strong and surprising properties that may not be desired. Bind is equivalent to Escalation, while Unit implies Escalation. Pictorially, we have:

$$\begin{tikzcd} \text{Unit} \arrow[d, Rightarrow] \\ \text{Escalation} \arrow[d, Leftrightarrow] \\ \text{Bind} \end{tikzcd}$$

  There are systems intermediate between the basic modal logic and Escalation. For instance, one may require the standard axiom C4 from modal logic (if A says A says s then A says s) without obtaining Escalation. However, these intermediate systems appear quite limited in their support of delegation and related concepts.

– In intuitionistic logics, we have a little more freedom. In particular, a system that includes Unit and Bind, which we call CDD [2, Section 8], does not lead to Escalation.
  Pictorially, we have:

$$\begin{tikzcd} \text{CDD} \arrow[d, Rightarrow] \arrow[dr, Rightarrow] & \text{Escalation} \arrow[d, Rightarrow] \\ \text{Unit} & \text{Bind} \end{tikzcd}$$

  Many further refinements become possible, in particular because Escalation and Unit are independent intuitionistically.

– The general form of the hand-off axiom (1) is equivalent to Bind.
– Unit implies axiom (2). This axiom is equivalent to Unit if there is a truth-telling principal.
– Finally, Escalation implies axiom (6). Conversely, this axiom and C4 imply Escalation.

In addition to occasional trickiness in proofs, the main difficulties of this work are in identifying and formulating the results summarized above. While some previous work also explores various axiomatizations of access control logics (e.g., [4]), those explorations have focused on classical logics, dealing for instance with properties of compound principals. We previously knew that Unit implies Escalation [1], and that Bind implies the hand-off axiom [2]. All the other results appear to be new.

Section 2 reviews the basic intuitionistic and classical logics that serve as our starting point. Section 3 studies CDD, considering axioms (1), (2), (3), and (4). Section 4 focuses on Escalation (axiom (5)). Section 5 considers axiom (6). Section 6 concludes with a brief discussion.

2 Basic Logics

In this section we briefly review the basic logics on which we build.

2.1 Formulas

Formulas are given by the grammar:

$$s ::= \texttt{true} \mid (s \vee s) \mid (s \wedge s) \mid (s \to s) \mid A \;\texttt{says}\; s \mid X \mid \forall X.\, s$$

where A ranges over elements of a set $\mathcal{P}$ (intuitively the principals), and X ranges over a set of variables. The variable X is bound in $\forall X.\, s$, and subject to renaming.

We write `false` for $\forall X.\, X$. We write $s_1 \equiv s_2$ for $(s_1 \to s_2) \wedge (s_2 \to s_1)$. We write $A \Rightarrow B$ as an abbreviation for

$$\forall X.\, (A \;\texttt{says}\; X \to B \;\texttt{says}\; X)$$

This formula is our representation of "A speaks for B". We write A `controls` s as an abbreviation for $(A \;\texttt{says}\; s) \to s$.

2.2 Basic Axioms and Rules

All of the logics that we consider are based on second-order propositional intuitionistic logic. We review this logic in Appendix A. In addition, we rely on a standard axiom (closure under consequence):

$$\forall X, Y.\, ((A \;\texttt{says}\; (X \to Y)) \to (A \;\texttt{says}\; X) \to (A \;\texttt{says}\; Y))$$

and a standard rule (necessitation):

$$\frac{s}{A\ \texttt{says}\ s}$$

Thus, we obtain a second-order, intuitionistic, multi-modal version of the standard logic K. It is the least system that we consider in this paper.

Sometimes we consider classical variants. In those, we use the following additional principle:

$$[\textit{Excluded-middle}] \quad \forall X.\,(X \vee (X \rightarrow \texttt{false}))$$

Throughout the paper, we also introduce other additional axioms, as explained in the introduction.

3 CDD

CDD arose as a simplified version of the Dependency Core Calculus (DCC) [3], but it is similarly adequate as a logic for access control [2, Section 8]. CDD is related to lax logic [10] and the computational lambda calculus [21]. It has been used for language-based authorization [11], and its central rules also appear in other systems for access control, such as Alpaca [19].

In comparison with DCC, CDD may be seen as straightforward and conservative. For instance, while DCC proves $(A\ \texttt{says}\ B\ \texttt{says}\ s) \rightarrow (B\ \texttt{says}\ A\ \texttt{says}\ s)$, CDD does not. Although we do not discuss DCC in detail, the results of this paper are relevant to DCC as well.

A self-contained definition of CDD is in Appendix B. In the context of the basic intuitionistic logic presented in Section 2.2, however, CDD amounts to adopting the following two additional axioms, Unit and Bind:

$$\begin{array}{rl} [\textit{Unit}] & \forall X.\,(X \rightarrow A\ \texttt{says}\ X) \\ [\textit{Bind}] & \forall X, Y.\,((X \rightarrow A\ \texttt{says}\ Y) \rightarrow (A\ \texttt{says}\ X) \rightarrow (A\ \texttt{says}\ Y)) \end{array}$$

It is easy to show that neither of these axioms is derivable in the logic of Section 2.2, neither intuitionistically nor classically. We prove some stronger results below, in Section 3.1, also considering the axiom C4 mentioned in the introduction. In Sections 3.2 and 3.3, we relate Unit and Bind to formulas motivated by security considerations.

3.1 C4 in CDD

This section is devoted to some simple results on the relation between CDD and the axiom C4:

$$[\textit{C4}] \quad \forall X.\,(A\ \texttt{says}\ A\ \texttt{says}\ X \rightarrow A\ \texttt{says}\ X)$$

We can replace Bind with the simpler C4 when we have Unit:

Proposition 1. *Starting from the basic logic (without Excluded-middle), we have:*

1. *Bind implies C4;*
2. *Unit and C4 (together) imply Bind;*
3. *C4 does not imply Bind;*
4. *Unit does not imply C4 (and a fortiori not Bind).*

Proof. 1. In Bind, take X to be A `says` Y.

2. For arbitrary X and Y, assume $(X \rightarrow A\ \texttt{says}\ Y)$ and $A\ \texttt{says}\ X$. We wish to show $A\ \texttt{says}\ Y$.
   By Unit, we have $(X \rightarrow A\ \texttt{says}\ Y) \rightarrow A\ \texttt{says}\ (X \rightarrow A\ \texttt{says}\ Y)$.
   By closure under consequence, $(X \rightarrow A\ \texttt{says}\ Y)$ yields $(A\ \texttt{says}\ X) \rightarrow (A\ \texttt{says}\ A\ \texttt{says}\ Y)$. By C4, we obtain $(A\ \texttt{says}\ X) \rightarrow (A\ \texttt{says}\ Y)$.
   It follows that $A\ \texttt{says}\ Y$.
3. We prove a stronger result in Proposition 2, with Excluded-middle.
4. Mapping the logic to its fragment without `says` (to System F [13, 7], essentially), we interpret $A\ \texttt{says}\ s$ as

$$(X_A \rightarrow s) \vee X_A$$

   where X_A is a distinct type variable used only for this purpose for each principal A. This interpretation satisfies Unit. It does not satisfy C4, because

$$A\ \texttt{says}\ A\ \texttt{says}\ \texttt{false} \rightarrow A\ \texttt{says}\ \texttt{false}$$

   translates to

$$((X_A \rightarrow ((X_A \rightarrow \texttt{false}) \vee X_A)) \vee X_A) \rightarrow ((X_A \rightarrow \texttt{false}) \vee X_A)$$

   The left-hand side of this implication is intuitionistically provable, and the right-hand side is not, so the implication is not. ■

Bind does not imply Unit in the basic logic. We do not state it explicitly in the intuitionistic case (in Proposition 1, above) because it is a corollary from a stronger result in the classical case (Proposition 2, below). Conversely, Bind implies C4 in the classical case, but we do not state explicitly there because it follows from a stronger result in the intuitionistic case.

Proposition 2. *Starting from the basic logic plus Excluded-middle, we have:*

1. *C4 implies neither Bind nor Unit.;*
2. *Unit implies C4 (and therefore Bind);*
3. *Bind does not imply Unit.*

Proof. 1. We consider a Kripke model with two possible worlds w and w', with the accessibility relation $\{w, w'\} \times \{w'\}$ associated with A. This model satisfies C4. It does not satisfy the instance of Bind

$$(X \rightarrow A\ \texttt{says}\ \texttt{false}) \rightarrow (A\ \texttt{says}\ X) \rightarrow (A\ \texttt{says}\ \texttt{false})$$

   for a proposition X that holds in w' but not in w (so $A\ \texttt{says}\ X$ holds in w). It does not satisfy the instance of Unit $X \rightarrow A\ \texttt{says}\ X$ for a proposition X that holds in w but not in w'.

2. In classical logic, we assume A `says` A `says` X in order to prove A `says` X. We proceed by cases on A `says` X, using Excluded-middle. If A `says` X holds, we are done. On the other hand, if (A `says` X) $\rightarrow$ `false` holds, Unit yields A `says` ((A `says` X) $\rightarrow$ `false`), and closure under consequence yields A `says` `false`, and then A `says` X.
   That Unit implies Bind follows from Proposition 1, which says that Unit and C4 together imply Bind.
3. This part follows from Theorem 4. ■

3.2 Hand-off in CDD

In CDD, we obtain the hand-off axiom as a theorem:

$$[\textit{Hand-off}] \quad A \texttt{ controls } (B \Rightarrow A)$$

A slight generalization of the hand-off axiom is also interesting and also a theorem:

$$[\textit{Generalized-hand-off}] \quad \forall X, Y.\ A \texttt{ controls } (X \rightarrow A \texttt{ says } Y)$$

Theorem 1. *Starting from the basic logic: Bind is equivalent to Generalized-hand-off.*

Proof. First we establish that Bind implies Generalized-hand-off. In order to prove that, for all X and Y, we have A `controls` ($X \rightarrow A$ `says` Y), we assume X and A `says` ($X \rightarrow A$ `says` Y) in order to prove A `says` Y. By Bind, we have:

$$\begin{array}{l} ((X \rightarrow A \texttt{ says } Y) \rightarrow A \texttt{ says } Y) \\ \quad\rightarrow \\ A \texttt{ says } (X \rightarrow A \texttt{ says } Y) \rightarrow A \texttt{ says } Y \end{array}$$

Since we have A `says` ($X \rightarrow A$ `says` Y), we obtain:

$$\begin{array}{l} ((X \rightarrow A \texttt{ says } Y) \rightarrow A \texttt{ says } Y) \\ \quad\rightarrow \\ A \texttt{ says } Y \end{array}$$

Since we also have X, and hence ($X \rightarrow A$ `says` Y) $\rightarrow A$ `says` Y, we conclude A `says` Y.

For the converse, let us assume that A `controls` ($X \rightarrow A$ `says` Y) in order to prove that ($X \rightarrow A$ `says` Y) $\rightarrow$ (A `says` X) $\rightarrow$ (A `says` Y). So let us assume that $X \rightarrow A$ `says` Y and A `says` X in order to prove A `says` Y. If A `says` X, by closure under consequence we have A `says` (($X \rightarrow A$ `says` Y) $\rightarrow A$ `says` Y) since $X \rightarrow$ (($X \rightarrow A$ `says` Y) $\rightarrow A$ `says` Y) is valid. By Generalized-hand-off, we obtain ($X \rightarrow A$ `says` Y) $\rightarrow A$ `says` Y. Applying this to $X \rightarrow A$ `says` Y, we conclude A `says` Y. ■

3.3 The Limits of Hand-off in CDD

Suppose that a principal A is trusted on whether it speaks for another principal B on every statement. In CDD, it follows that A must speak for B in the first place, whether it says so or not. If A does not wish to speak for B, it should reduce its authority, for instance by adopting an appropriate role [16, Section 6.1]. This result might be seen as a reassuring characterization of who can attribute the right to speak for B; it may also be seen as a dual or a limitation of hand-off in the context of CDD.

More precisely, we define:

$$[\textit{Authority-shortcut}] \quad (\forall X.\ A\ \texttt{controls}\ (A\ \texttt{says}\ X \to B\ \texttt{says}\ X)) \to (A \Rightarrow B)$$

We obtain:

Theorem 2. *Unit implies Authority-shortcut.*

Proof. Suppose that, for all X, A $\texttt{controls}$ $(A\ \texttt{says}\ X \to B\ \texttt{says}\ X)$ and suppose that, for some particular X, we have $A\ \texttt{says}\ X$. We wish to derive $B\ \texttt{says}\ X$.

Because $A\ \texttt{says}\ X$, Unit implies $A\ \texttt{says}\ B\ \texttt{says}\ X$. (Here we apply Unit under $\texttt{says}$.) Then by closure under consequence we have $A\ \texttt{says}\ (A\ \texttt{says}\ X \to B\ \texttt{says}\ X)$.

By our assumption that, for all X, A $\texttt{controls}$ $(A\ \texttt{says}\ X \to B\ \texttt{says}\ X)$, we obtain $A\ \texttt{says}\ X \to B\ \texttt{says}\ X$.

Combining $A\ \texttt{says}\ X \to B\ \texttt{says}\ X$ with $A\ \texttt{says}\ X$, we obtain $B\ \texttt{says}\ X$, as desired.

The proof is peculiar, not least because the hypothesis $A\ \texttt{says}\ X$ is used twice in different roles. ∎

A small variant of the proof of Theorem 2 shows that Unit also implies:

$$\forall X.\ ((A\ \texttt{controls}\ (A\ \texttt{says}\ X \to B\ \texttt{says}\ X)) \to (A\ \texttt{says}\ X \to B\ \texttt{says}\ X))$$

In other words, writing $A \Rightarrow_X B$ for $A\ \texttt{says}\ X \to B\ \texttt{says}\ X$ [18], we have that Unit implies:

$$\forall X.\ ((A\ \texttt{controls}\ (A \Rightarrow_X B)) \to (A \Rightarrow_X B))$$

The converse of Theorem 2 is almost true. Suppose that there is a truth-telling principal A, that is, a principal for which $\forall X. X \equiv (A\ \texttt{says}\ X)$. Applying Authority-shortcut to this principal, we can derive $s \to B\ \texttt{says}\ s$ by propositional reasoning, for every B and s. In other words, given such a truth-teller, we obtain Unit.

Nevertheless, the converse of Theorem 2 is not quite true. All basic axioms plus rules, plus Authority-shortcut, hold when we interpret $A\ \texttt{says}\ s$ as true, for every A and s. Unit does not hold under this interpretation.

In addition, we can prove that Authority-shortcut does not follow from other axioms (such as Bind), even in classical logic. In other words, Authority-shortcut appears to be very close to Unit, and can be avoided by dropping Unit.

4 Escalation

As indicated in the introduction, Escalation is the following axiom:

$$[\textit{Escalation}] \quad \forall X, Y. \, ((A \texttt{ says } X) \rightarrow (X \vee (A \texttt{ says } Y)))$$

Equivalently, Escalation can be formulated as:

$$\forall X, Y. \, ((A \texttt{ says } X) \rightarrow (X \vee (A \texttt{ says } \texttt{false})))$$

Escalation embodies a rather degenerate interpretation of `says`. At the very least, great care is required when Escalation is assumed. For instance, suppose that two principals A and B are trusted on s, and that we express this as $(A \texttt{ controls } s) \wedge (B \texttt{ controls } s)$; with Escalation, if $A \texttt{ says } B \texttt{ says } s$ then s follows. Formally, we can derive:

$$(A \texttt{ controls } s) \wedge (B \texttt{ controls } s) \rightarrow ((A \texttt{ says } B \texttt{ says } s) \rightarrow s)$$

This theorem may be surprising. Its effects may however be avoided: A should not say that B says s unless A wishes to say s. As a result, though, the logic loses flexibility and expressiveness.

On the whole, we consider that Escalation is not a desirable property. Unfortunately, it can follow from the combination of properties that may appear desirable in isolation, as we show.

Theorem 3. *Starting from the basic logic (without Excluded-middle),*

1. *Unit and Bind (together) do not imply Escalation (in other words, Escalation is not a theorem of CDD);*
2. *Escalation implies Bind (and therefore C4).*

Proof. 1. Following Tse and Zdancewic [23], we can interpret CDD in System F [13, 7]. We map $A \texttt{ says } s$ to $X_A \rightarrow s$, where X_A is a distinct type variable used only for this purpose. If s is provable in CDD, then its translation is provable in System F.
The translation of Escalation is:

$$\forall X, Y. \, ((X_A \rightarrow X) \rightarrow (X \vee (X_A \rightarrow Y)))$$

This formula is not provable in System F.

2. Suppose that $X \rightarrow A \texttt{ says } Y$ and that $A \texttt{ says } X$. We wish to prove $A \texttt{ says } Y$.
By Escalation, $A \texttt{ says } X$ implies $X \vee A \texttt{ says } Y$. Combining this with $X \rightarrow A \texttt{ says } Y$, we obtain $A \texttt{ says } Y \vee A \texttt{ says } Y$, that is, $A \texttt{ says } Y$. ∎

Theorem 4. *Starting from the basic logic plus Excluded-middle, we have:*

1. *Unit implies Escalation (and therefore Bind);*
2. *Escalation (and a fortiori Bind) does not imply Unit;*

3. Bind implies Escalation;
4. C4 does not imply Escalation.

Proof. 1. Suppose A `says` X. If X is true, then we are done, as we obtain $X \vee (A$ `says` $Y)$. If X is false, that is, $X \rightarrow$ `false` is true, then Unit yields A `says` $(X \rightarrow$ `false`$)$, and by closure under consequence we obtain A `says` `false` and then A `says` Y for any Y, and then $X \vee (A$ `says` $Y)$.

2. Escalation (and a fortiori Bind) is true in a Kripke model with two possible worlds w and w', in which every principal is mapped to the universal accessibility relation $\{w, w'\} \times \{w, w'\}$. This Kripke model does not satisfy the instance of Unit $X \rightarrow A$ `says` X for a proposition X that holds in w but not in w'.
3. We prove Escalation by cases on whether X is true or not. If it is true, then Bind yields X vacuously, and hence Escalation. If it is false, then that means $X \rightarrow$ `false`, which entails $X \rightarrow A$ `says false`, and applying Bind with `false` for Y we obtain $(A$ `says` $X) \rightarrow (A$ `says false`$)$.
4. The Kripke model described in part 1 of Proposition 2 does not satisfy Escalation: A `says` X at w means that X is true in w', while X may be false in w and A `says false` is false in w. ■

Going further, in classical logic Unit implies that each principal A is either a perfect truth-teller or says `false`. In the former case, A speaks for any other principal; in the latter case, any other principal speaks for A. Formally, we can derive $(A \Rightarrow B) \vee (B \Rightarrow A)$. While this conclusion does not represent a logical contradiction, it severely limits the flexibility and expressiveness of the logic: policies can describe only black-and-white situations. This point is a further illustration of the fact that usefulness degrades even before a logic becomes inconsistent.

5 On the Monotonicity of Controls

The monotonicity of controls means that, if a principal controls a formula X, then it controls every weaker formula Y. Formally, we write:

$$[\textit{Control-monotonicity}] \quad \forall X, Y. \begin{pmatrix} (X \rightarrow Y) \\ \rightarrow \\ ((A \texttt{ controls } X) \rightarrow (A \texttt{ controls } Y)) \end{pmatrix}$$

This monotonicity property may seem attractive. In particular, it may make it easier to comply with the Principle of Least Privilege. This principle says [22]:

> Every program and every user of the system should operate using the least set of privileges necessary to complete the job.

The monotonicity of controls implies that, if A wants to convince a reference monitor of Y, and it can convince it of a stronger property X, then A should be able to state Y directly, rather than the stronger property X. For instance, suppose that Y is the statement that B may access a file f_1, and that X is the statement that B may access both f_1 and another file f_2. When A wishes to allow B to access f_1, it should not have to state also that B may access f_2. The monotonicity of controls allows A to say only that B may access f_1.

Nevertheless, the monotonicity of controls has questionable consequences.

Proposition 3. *Starting from the basic logic (without Excluded-middle), Control-monotonicity implies:*

$$A \texttt{ controls } s_1 \rightarrow A \texttt{ says } s_2 \rightarrow (s_1 \vee s_2)$$

Proof. We obtain $A \texttt{ controls } s_1 \rightarrow A \texttt{ says } s_2 \rightarrow (s_1 \vee s_2)$ from Control-monotonicity, as follows: Let X be s_1 and Y be $s_1 \vee s_2$. We have $X \rightarrow Y$. Suppose that $A \texttt{ controls } s_1$. Control-monotonicity yields $A \texttt{ controls } (s_1 \vee s_2)$. If $A \texttt{ says } s_2$, then we obtain $A \texttt{ says } (s_1 \vee s_2)$, and hence $s_1 \vee s_2$. ■

In Proposition 3, the formulas s_1 and s_2 may be completely unrelated. For instance, suppose that A controls whether B may access a file f_1, and A says that B may access another file f_2; curiously, we obtain that B may access f_1 or B may access f_2, by Proposition 3.

In fact, the monotonicity of `controls` is equivalent to Escalation in the presence of C4. (Intuitionistically, C4 is strictly required for this equivalence.)

Theorem 5. *Starting from the basic logic (without Excluded-middle), the following are equivalent:*

- *Escalation,*
- *C4 and Control-monotonicity.*

However, neither Control-monotonicity nor C4 implies the other, not even in combination with Unit.

Proof. – Escalation implies C4, by Theorem 3.

- Escalation implies Control-monotonicity:
  Suppose $X \rightarrow Y$ and $A \texttt{ controls } X$. We wish to prove $A \texttt{ controls } Y$, so we assume $A \texttt{ says } Y$ in order to derive Y.
  By Escalation, we obtain $Y \vee A \texttt{ says } X$. Since $A \texttt{ controls } X$, it follows that $Y \vee X$. Since $X \rightarrow Y$, it follows that Y, as desired.
- C4 and Control-monotonicity together imply Escalation:
  We have $A \texttt{ controls } A \texttt{ says false}$ by C4, and $(A \texttt{ says false}) \rightarrow (Y \vee A \texttt{ says false})$ by propositional reasoning, so Control-monotonicity yields $A \texttt{ controls } (Y \vee A \texttt{ says false})$.
  Since $A \texttt{ says } Y$ implies $A \texttt{ says } (Y \vee A \texttt{ says false})$ by propositional reasoning and closure under consequence, we obtain that $A \texttt{ says } Y$ implies $Y \vee A \texttt{ says false}$.
- C4 does not imply Control-monotonicity, even in combination with Unit, by Proposition 1 (which says that Bind implies C4) and Theorem 3 (which says that Unit and Bind do not imply Escalation).

– Starting from the basic logic (without Excluded-middle), Control-monotonicity and Unit (together) do not imply C4, and therefore not Bind nor Escalation.

  We construct an interpretation of the logic that satisfies the basic axioms, Unit, and Control-monotonicity, but not C4.

  In this interpretation, each formula is mapped to an open set in the Sierpinski space, that is, to one of the sets $\emptyset$, $\{1\}$, and $\{0,1\}$. These open sets form a Heyting algebra with the usual inclusion ordering, so they provide a model for intuitionistic logic. In this model, $\emptyset$ corresponds to `false`. Importantly $\{1\} \rightarrow$ `false` is $\emptyset$ (and not $\{0\}$, since this is not an open set). Quantification works as a finite conjunction. For every A, we let the meaning of A `says` s be $\{1\}$ if the meaning of s is $\emptyset$, and $\{0,1\}$ otherwise.

  This interpretation satisfies Unit, since the meaning of s is always contained in the meaning of A `says` s. A fortiori, it also satisfies necessitation.

  Since `says` is monotonic, we have that A `says` $((X \rightarrow Y) \wedge X) \rightarrow A$ `says` Y. Moreover, since the inclusion ordering is linear, monotonicity implies that `says` distributes over conjunctions, so $((A$ `says` $(X \rightarrow Y)) \wedge (A$ `says` $X)) \rightarrow A$ `says` $((X \rightarrow Y) \wedge X)$. Closure under consequence follows.

  These definitions also imply that the meaning of A `controls` s is the same as the meaning of s:

  - for $s = \emptyset$, A `controls` s is $\{1\} \rightarrow \emptyset$, that is, $\emptyset$;
  - for $s = \{1\}$, A `controls` s is $\{0,1\} \rightarrow \{1\}$, that is, $\{1\}$;
  - for $s = \{0,1\}$, A `controls` s is $\{0,1\} \rightarrow \{0,1\}$, that is, $\{0,1\}$.

  Therefore, `controls` is monotonic.

  The meaning of A `says` `false` is $\{1\}$. The meaning of A `says` A `says` `false` is $\{0,1\}$. So we do not have C4. ∎

Although Control-monotonicity does not imply C4 in intuitionistic logic, it does in classical logic, as the following theorem implies:

Theorem 6. *Starting from the basic logic plus Excluded-middle, the following are equivalent:*

– *Escalation,*
– *Control-monotonicity.*

Proof. By Theorem 5, Escalation implies Control-Monotonicity. Conversely, we instantiate Control-monotonicity in the case the stronger propositions (X) is `false`; we obtain a formula that is classically equivalent to Escalation. ∎

The theorems of this section should not be construed as a criticism of the Principle of Least Privilege. Formulations weaker than Control-monotonicity might be viable and less problematic.

6 Discussion

Overall, the results of this paper indicate that, while in a classical setting we may want to stay close to basic modal logic, in an intuitionistic setting we may adopt CDD. This move may be attractive, in particular, because CDD supports hand-off. These results also suggest that a great deal of caution should be applied in selecting axioms, considering both their formal properties and their security implications.

We do not argue that the use of a particular set of axioms is required for writing good security policies. It is possible that reasonable security policies and other assertions can be formulated in many different systems, with different underlying logics. However, understanding the properties and consequences of these logics is essential for writing appropriate formulas reliably.

The literature contains models for some of these axioms (e.g., [4]), and we are currently developing others (in collaboration with Deepak Garg). Semantics can be helpful in providing a different perspective on axiomatizations. In this paper, we employ semantics as a tool in some of the proofs; more extensive uses of semantics remain attractive but a subject for further research.

Acknowledgments

This work benefited from conversations with Butler Lampson.

References

1. Martín Abadi. Logic in access control. In *Proceedings of the Eighteenth Annual IEEE Symposium on Logic in Computer Science*, pages 228–233, 2003.
2. Martín Abadi. Access control in a core calculus of dependency. *Electronic Notes in Theoretical Computer Science*, 172:5–31, April 2007. *Computation, Meaning, and Logic: Articles dedicated to Gordon Plotkin*.
3. Martín Abadi, Anindya Banerjee, Nevin Heintze, and Jon G. Riecke. A core calculus of dependency. In *Proceedings of the 26th ACM Symposium on Principles of Programming Languages*, pages 147–160, January 1999.
4. Martín Abadi, Michael Burrows, Butler Lampson, and Gordon Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, October 1993.
5. Lujo Bauer, Scott Garriss, and Michael K. Reiter. Distributed proving in access-control systems. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 81–95, May 2005.
6. Moritz Y. Becker, Cédric Fournet, and Andrew D. Gordon. Design and semantics of a decentralized authorization language. In *20th IEEE Computer Security Foundations Symposium*, pages 3–15, 2007.
7. Luca Cardelli. Type systems. In Allen B. Tucker, editor, *The Computer Science and Engineering Handbook*, chapter 103, pages 2208–2236. CRC Press, Boca Raton, FL, 1997.
8. Andrew Cirillo, Radha Jagadeesan, Corin Pitcher, and James Riely. Do as I SaY! programmatic access control with explicit identities. In *20th IEEE Computer Security Foundations Symposium*, pages 16–30, July 2007.

9. John DeTreville. Binder, a logic-based security language. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 105–113, May 2002.
10. Matt Fairtlough and Michael Mendler. Propositional lax logic. *Information and Computation*, 137(1):1–33, 1997.
11. Cédric Fournet, Andrew D. Gordon, and Sergio Maffeis. A type discipline for authorization in distributed systems. In *20th IEEE Computer Security Foundations Symposium*, pages 31–45, 2007.
12. Deepak Garg and Frank Pfenning. Non-interference in constructive authorization logic. In *19th IEEE Computer Security Foundations Workshop*, pages 283–296, 2006.
13. Jean-Yves Girard. *Interprétation Fonctionnelle et Elimination des Coupures de l'Arithmétique d'Ordre Supérieur.* Thèse de doctorat d'état, Université Paris VII, June 1972.
14. Yuri Gurevich and Itay Neeman. DKAL: Distributed-knowledge authorization language. Technical Report MSR-TR-2007-116, Microsoft Research, August 2007.
15. G. E. Hughes and M. J. Cresswell. *An Introduction to Modal Logic.* Methuen Inc., New York, 1968.
16. Butler Lampson, Martín Abadi, Michael Burrows, and Edward Wobber. Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems*, 10(4):265–310, November 1992.
17. Butler W. Lampson. Protection. In *Proceedings of the 5th Princeton Conference on Information Sciences and Systems*, pages 437–443, 1971.
18. Butler W. Lampson. Computer security in the real world. *IEEE Computer*, 37(6):37–46, June 2004.
19. Christopher Lesniewski-Laas, Bryan Ford, Jacob Strauss, M. Frans Kaashoek, and Robert Morris. Alpaca: extensible authorization for distributed services. In *14th ACM Conference on Computer and Communications Security*, pages 432–444, 2007.
20. Ninghui Li, Benjamin N. Grosof, and Feigenbaum. Delegation logic: A logic-based approach to distributed authorization. *ACM Transactions on Information and System Security*, 6(1):128–171, February 2003.
21. Eugenio Moggi. Notions of computation and monads. *Information and Control*, 93(1):55–92, 1991.
22. Jerome H. Saltzer and Michael D. Schroeder. The protection of information in computer system. *Proceedings of the IEEE*, 63(9):1278–1308, September 1975.
23. Stephen Tse and Steve Zdancewic. Translating dependency into parametricity. *Journal of Functional Programming*. To appear.

Appendix

A Second-Order Propositional Intuitionistic Logic

The axioms are:

- $\texttt{true}$
- $s_1 \to (s_2 \to s_1)$
- $(s_1 \to (s_2 \to s_3)) \to ((s_1 \to s_2) \to (s_1 \to s_3))$
- $(s_1 \wedge s_2) \to s_1$

- $(s_1 \wedge s_2) \to s_2$
- $s_1 \to s_2 \to (s_1 \wedge s_2)$
- $s_1 \to (s_1 \vee s_2)$
- $s_2 \to (s_1 \vee s_2)$
- $(s_1 \to s_3) \to ((s_2 \to s_3) \to ((s_1 \vee s_2) \to s_3))$
- $(\forall X.\, s) \to s[t/X]$
- $(\forall X.\, (s_1 \to s_2)) \to (s_1 \to \forall X.\, s_2)$ where X is not free in s_1

The rules are modus ponens and universal generalization:

$$\frac{s_1 \qquad s_1 \to s_2}{s_2} \qquad\qquad \frac{s}{\forall X.\, s}$$

B CDD

The rules of CDD are:

$[\mathit{Var}]$ $\quad \Gamma, s, \Gamma' \vdash s$ $\qquad$ $[\mathit{Unit}]$ $\quad \Gamma \vdash \texttt{true}$

$$[\mathit{Lam}]\quad \frac{\Gamma, s_1 \vdash s_2}{\Gamma \vdash (s_1 \to s_2)} \qquad [\mathit{App}]\quad \frac{\Gamma \vdash (s_1 \to s_2) \quad \Gamma \vdash s_1}{\Gamma \vdash s_2}$$

$$[\mathit{Pair}]\quad \frac{\Gamma \vdash s_1 \qquad \Gamma \vdash s_2}{\Gamma \vdash (s_1 \wedge s_2)}$$

$$[\mathit{Proj\ 1}]\quad \frac{\Gamma \vdash (s_1 \wedge s_2)}{\Gamma \vdash s_1} \qquad [\mathit{Proj\ 2}]\quad \frac{\Gamma \vdash (s_1 \wedge s_2)}{\Gamma \vdash s_2}$$

$$[\mathit{Inj\ 1}]\quad \frac{\Gamma \vdash s_1}{\Gamma \vdash (s_1 \vee s_2)} \qquad [\mathit{Inj\ 2}]\quad \frac{\Gamma \vdash s_2}{\Gamma \vdash (s_1 \vee s_2)}$$

$$[\mathit{Case}]\quad \frac{\Gamma \vdash (s_1 \vee s_2) \qquad \Gamma, s_1 \vdash s \qquad \Gamma, s_2 \vdash s}{\Gamma \vdash s}$$

$$[\mathit{TLam}]\quad \frac{\Gamma \vdash s}{\Gamma \vdash \forall X.\, s}\ (X \text{ not free in } \Gamma) \qquad [\mathit{TApp}]\quad \frac{\Gamma \vdash \forall X.\, s}{\Gamma \vdash s[t/X]}$$

$$[\mathit{UnitM}]\quad \frac{\Gamma \vdash s}{\Gamma \vdash A\ \textsf{says}\ s}$$

$$[\mathit{BindM}]\quad \frac{\Gamma \vdash A\ \textsf{says}\ s \qquad \Gamma, s \vdash A\ \textsf{says}\ t}{\Gamma \vdash A\ \textsf{says}\ t}$$

As is typical for type systems, the rules are presented in a sequent-calculus format, rather than as a Hilbert system. In this definition, though, we simply omit all the terms, as well as declarations for variables. An environment Γ denotes a list of formulas. In the case where Γ is empty, we write $\vdash s$, and say that s is a theorem, when $\vdash s$ is derivable by these rules.