Lecture Notes on Authorization Logic

15-316: Software Foundations of Security & Privacy
Frank Pfenning

Lecture 15
October 29, 2024

1 Introduction

We started the course with an analysis of *safety properties* of computations, and how to counter them statically (via verification conditions in dynamic logic) or dynamically (via sandboxing). Then we moved on to the more complex *information flow properties*. An attacker may discover secrets via particular programs or inputs in a variety of ways, with side-channel attacks (for example, timing attacks) being the most sophisticated. Countermeasures are mostly via information-flow type systems.

With this lecture we are starting a new section of the course, considering *authentication* (you are who you say you are) and *authorization* (you are allowed to perform the actions you are trying to perform). Authentication and authorization are pervasive in today's computing environment, from shared file systems like AFS and cloud services like Github, to shopping and banking services. Today's and the next lecture will focus on *authorization*, followed in later lectures by authentication. In many cases, authorization is just embedded in code. Once authorization becomes complex, this can easily be compromised by bugs since flow of the authorization checks through a program can be difficult to audit. It can also be difficult to even understand what the authorization policy actually is, which then means it is hard to compare it against the code.

In this lecture we take a very general approach, expression policy in an *authorization logic*. Like in other applications we have seen, the logic is the interface between policy and implementation. It expresses, at a high level of abstraction, who is allowed to do what in a complex system. On one side, this serves as a specification one can reason about rigorously, divorced from an implementation. On the other side, which can use it directly to enforce authorization policies in an implementation by using formal proofs of authorization. Abadi [2003] gives a general overview of authorization logics. The architecture where formal proofs are used was pioneered by Bauer [2003] under the name of *proof-carrying authorization* (PCA).

Among the direct applications of PCA are the Grey systems [Bauer et al., 2005] for access to offices in Cylab at CMU, and a *proof-carrying file system* that particularly explored issues of efficiency [Garg, 2009, Garg and Pfenning, 2010]. We ignore here a number of practically relevant issues, such as revocation and temporal aspects of authorization. Once added [DeYoung et al., 2007], authorization logic is expressive enough, for example, to express the rules governing access to classified information in the American intelligence community [Garg et al., 2009].

2 Affirmations

Early on this class, we introduced and reasoning in Boolean logic, with connectives such as conjunction, disjunction, implication, etc. Then we introduced so-called *modal operators*, $[\alpha]Q$, $\langle\alpha\rangle Q$ that speak about programs, and $\Box P$ that guarantees validity of P (that is, truth in all possible states).

In order to express access control policies *logically*, we need a new kind of modal operator that expresses an affirmation, A **says** P (*principal* A *says proposition* P). Principals A, B, C, etc. can stand for user ids like $admin$, fp, or $hemant$ Propositions include atomic propositions, conjunction, implication, quantifiers, and other connectives as we need them.

As an example, we use the Grey [Bauer et al., 2005] system that is used to control access to offices in Cylab at CMU. There is an administrator (principal $admin$) that sets policies, and individual professors and students identified by their Andrew id. There are also resources, like offices and conference rooms. Among the atomic propositions are the following:

- $\mathsf{mayOpen}(A, R)$. Principal A may open room R.
- $\mathsf{owns}(A, R)$. Principal A owns office R.
- $\mathsf{studentOf}(B, A)$. Principal B is a student of A.

Here are some examples. The administrator may say that fp owns $ghc6017$.

$$admin\ \textbf{says}\ \mathsf{owns}(fp, ghc6017)$$

This is a basic affirmation. A general rules could state that the owner of a room may open it.

$$admin\ \textbf{says}\ (\forall A.\, \forall R.\, \mathsf{owns}(A, R) \rightarrow \mathsf{mayOpen}(A, R))$$

An even more complex policy component would be that any student of the owner of an office, may also open the office. The twist here is that the $\mathsf{studentOf}$ relationship should be affirmed by the owner, rather than the administrator.

$$admin \textbf{ says } (\forall A. \forall B. \forall R. \mathsf{owns}(A, R) \land fp \textbf{ says } \mathsf{studentOf}(B, A) \to \mathsf{mayOpen}(B, R))$$

The scope of fp **says** here is intended to be only the studentOf proposition. When the scope is larger, we enclose it in parentheses. We can combine this policy with the affirmation by fp that $hemant$ is his student:

$$fp \textbf{ says } \mathsf{studentOf}(hemant, fp)$$

Under this policy $hemant$ should be able to access $ghc6017$. We can formulate the question whether this is allowed as a sequent in authorization logic, with the policy as antecedents (assumptions) and the query as the succedent. In this particular situation, we would try to prove the sequent

$$\begin{array}{l}
(1) : admin \textbf{ says } (\forall A. \forall R. \mathsf{owns}(A, R) \to \mathsf{mayOpen}(A, R)), \\
(2) : admin \textbf{ says } (\forall A. \forall B. \forall R. \mathsf{owns}(A, R) \land fp \textbf{ says } \mathsf{studentOf}(B, A) \to \mathsf{mayOpen}(B, R)), \\
(3) : admin \textbf{ says } \mathsf{owns}(fp, ghc6017), \\
(4) : fp \textbf{ says } \mathsf{studentOf}(hemant, fp) \\
\vdash \\
admin \textbf{ says } \mathsf{mayOpen}(hemant, ghc6017)
\end{array}$$

Here, we have labeled the assumptions so we can reference them in the proof. Reasoning entirely intuitively (to be formalized later in the lecture) we can deduce:

$$\begin{array}{ll}
(5) : admin \textbf{ says } \mathsf{mayOpen}(fp, ghc6017) & \text{from } (1) \text{ and } (4) \\
(6) : admin \textbf{ says } \mathsf{mayOpen}(hemant, ghc6017) & \text{from } (2), (3), \text{ and } (4)
\end{array}$$

The last line (6) is exactly what we are trying to prove. So we have confirmed that $hemant$ may open my office, according to the policy.

3 Constructive Logic

The Boolean logic of the earlier lectures (including propositional logic and dynamic logic) are based on a semantics where every formula has one of two truth values: $\top$ or $\perp$. This is not a good match for authorization logic, as remarked by Abadi [2003]. The first issue that we may be "agnostic" about a proposition. Any proposition that is affirmed should be seen as *extending* what we can deduce, but not be in conflict with what we know so far. Another example is given by

$$A \textbf{ says } P \to (P \lor A \textbf{ says } Q)$$

We can read this: if A affirms P then either P must be true, or A affirms any proposition (including $\perp$). If we worked with just two truth values, why would this be valid? Let's assume A **says** P. Now we distinguish two cases for P. If P is true, then $P \lor A$ **says** Q. If P is false, then $\neg P$ is true. A would affirm any true proposition, so A **says** $\neg P$. But if A **says** P and A **says** $\neg P$, A is mired in a state of internal contradiction and would have to affirm anything. Not good.

In order to avoid these kind of paradoxes, we use an *intutionistic logic* as the basis for reasoning. Intuitionistic logic does not allow us to reason with he law of excluded middle, or to prove P by assuming $\neg P$ and deriving a contradiction (using an indirect proof). Instead, we want the reason access to a resource is granted to be as clear and direct as possible, which is what intuitionistic logic provides. Actually, we go a step further and rule out negation $\neg P$ and falsehood $\bot$ entirely, because it is easy to make intuitively meaningful statements that are wrong. For example, principal *fp* does **not** want $hemant$ to access his office. Stating *fp* **says** $\neg\mathsf{mayOpen}(hemant, ghc6017)$ turns out to be the wrong way to say this. Because if the rest of the policy says that $\mathsf{mayOpen}(hemant, ghc6017)$ then suddenly *fp* affirms *everything*, including that every principal in the system may access his office—clearly not the desired effect.

This change in perspective, from two-valued Boolean logic (also called *classical logic*) to a richer intuitionstic logic has two consequences, one semantic and one syntactic. Semantically, we need to generalize to a so-called *Kripke semantics* where we consider multiple *worlds* in which different propositions may be true. This is a good match for a logic of authorization since at the very least each principal defines a world, and different principals will affirm different propositions. For such a semantics, see, for example, Garg [2008]. Syntactically, it is surprisingly simple: we restrict the succedent of a sequent to be exactly one formula. We therefore write $\Gamma \vdash \delta$. That this actually works was one of Gentzen's [1935] profound insights.

We actually make the lack of a mathematical semantics a philosophical principle. We think of the meanings of formulas as given by their possible derivations in the sequent calculus. In other words, the right and left rules of the sequent calculus themselves define the meaning of each logical constant, connective, and modality. In the context of authorization, this can be justified since it is ultimately a formal proof that is used to claim and check authorization—we don't appeal to an external semantics. For a fuller development of this viewpoint, see, for example, Dummett [1991] and Martin-Löf [1983].

As we will in particular see in the next lecture, the properties of the logic change in fundamental ways when we restrict the succedents to be just a single formula.

The rules we get otherwise just reflect the earlier ones.

$$\frac{}{\Gamma, P \vdash P}\ \mathsf{id}$$

$$\frac{\Gamma, P \vdash Q}{\Gamma \vdash P \to Q}\ {\to}R \qquad \frac{\Gamma \vdash P \quad \Gamma, Q \vdash \delta}{\Gamma, P \to Q \vdash \delta}\ {\to}L$$

$$\frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ {\wedge}R \qquad \frac{\Gamma, P, Q \vdash \delta}{\Gamma, P \wedge Q \vdash \delta}\ {\wedge}L$$

$$\frac{\Gamma \vdash P(y) \quad y \notin \Gamma, P(x)}{\Gamma \vdash \forall x.\, P(x)}\ \forall R^y \qquad \frac{\Gamma, P(c) \vdash \delta}{\Gamma, \forall x.\, P(x) \vdash \delta}\ \forall L$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ {\vee}R_1 \quad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ {\vee}R_2 \qquad \frac{\Gamma, P \vdash \delta \quad \Gamma, Q \vdash \delta}{\Gamma, P \vee Q \vdash \delta}\ {\vee}L$$

For the quantifiers, recall our convention of writing $P(x)$ and then $P(c)$ for the result of substituting c for x in $P(x)$. Quantification was previously over integers, but here we think of principals, rooms, etc. c could also be a variable y that was introduced by a $\forall R$ rule.

The most immediate effect of reasoning intuitionistically is perhaps on disjunction. We can no longer prove, for example $p \vee (p \to q)$ because we have to decide between one of the disjuncts instead of carrying both as succedents.

In these rules there is a main formula among the antecedents to which a left rule is applied. This formula may be needed again so it can be kept among the antecedents if so desired. This is particularly useful for $\forall L$. To anticipate a bit the next lecture, it is no longer the case that all rules are invertible, they way it is in Boolean propositional logic. So proof search significantly more difficult than in Boolean logic, and not just because of the quantifiers.

4 Affirmations

We haven't yet discussed A **says** P, the central modality of authorization logic. The key insight is that when we try to prove $\Gamma \vdash A$ **says** P we need to proceed with our reasoning from the perspective of principal A. But what does this mean? First, principal A should be willing to affirm any proposition that is *true*. Second, anything that A says is also available to reason with. On the other hand, if B **says** Q is in Γ for some $B \neq A$, the proposition Q is not available to A: this is something that B affirms, but not necessarily A.

Formalizing this reasoning is not entirely straightforward, and there are different approaches. The one we choose is due to Garg and Pfenning [2006], primarily due to it simplicity. We distinguish a succedent expressing that a proposition is true (written P true) and another that A affirms a proposition P (written A **aff** P). We often omit the "true" notation, but always write A **aff** P to use an affirmation. Formally:

$$\text{Succedent} \quad \delta \quad ::= \quad P\ \mathsf{true} \mid A\ \mathbf{aff}\ P$$

The first rule says that in order to prove that A **says** P is true, we need to prove that A affirms P.

$$\frac{\Gamma \vdash A\ \mathbf{aff}\ P}{\Gamma \vdash (A\ \mathbf{says}\ P)\ \mathsf{true}}\ \mathbf{says}R$$

This may seem redundant, but it exposes the principal A and the proposition P. It is similar to the rule $\rightarrow R$, where the implication $P \rightarrow Q$ is turned into $P \vdash Q$, opening up P and Q to further inferences.

The next rule expresses that if P is true, then A is willing to affirm that.

$$\frac{\Gamma \vdash P\ \mathsf{true}}{\Gamma \vdash A\ \mathbf{aff}\ P}\ \mathbf{aff}$$

This rule is like "peeling an onion", moving entirely now into A's head, reasoning from their perspective.

The left rule for A **says** P jumps directly to P, both of which are propositions and therefore can appear among the antecedents. But we can make this transition only if we are currently reasoning from A's perspective, that is, if the succedent is A **aff** Q for some Q.

$$\frac{\Gamma, P \vdash A\ \mathbf{aff}\ Q}{\Gamma, A\ \mathbf{says}\ P \vdash A\ \mathbf{aff}\ Q}\ \mathbf{says}L$$

5 Some Axioms

Before we go back to our motivating example, we can analyze some of the properties of affirmations.

$\vdash P \rightarrow A\ \mathbf{says}\ P$

As we have said, any principal (here A) is willing to affirm any true proposition (here P), so we should be able to prove this. As usual, we build the proof bottom-up; we show here only the result.

$$\cfrac{\cfrac{\cfrac{\cfrac{}{P \vdash P}\ \mathsf{id}}{P \vdash A\ \mathbf{aff}\ P}\ \mathbf{aff}}{P \vdash A\ \mathbf{says}\ P}\ \mathbf{says}R}{\cdot \vdash P \rightarrow A\ \mathbf{says}\ P}\ \rightarrow R$$

The implication in the other direction should not be valid: just because A says P that doesn't mean P is actually true. We show that it is not derivable.

$\nvdash (A \textbf{ says } p) \to p$

$$\dfrac{\dfrac{\text{XXX}}{A \textbf{ says } p \vdash p}}{\cdot \vdash (A \textbf{ says } p) \to p}\;{\to}R$$

As a first step, only $\to R$ is applicable; for the second no rule is.

We can also "distribute" A **says** over an implication. This captures that if A affirms $P \to Q$ and P, then it also affirms Q.

$\vdash A \textbf{ says } (P \to Q) \to (A \textbf{ says } P \to A \textbf{ says } Q)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P \vdash P}\,\mathsf{id} \quad \dfrac{}{Q \vdash Q}\,\mathsf{id}}{P \to Q, P \vdash Q}\,{\to}L}{P \to Q, P \vdash A \textbf{ aff } Q}\,\textbf{aff}}{A \textbf{ says } (P \to Q), A \textbf{ says } P \vdash A \textbf{ aff } Q}\,\textbf{says}L \times 2}{A \textbf{ says } (P \to Q), A \textbf{ says } P \vdash A \textbf{ says } Q}\,\textbf{says}R}{\vdash A \textbf{ says } (P \to Q) \to (A \textbf{ says } P \to A \textbf{ says } Q)}\,{\to}R \times 2$$

Iterating A **says** twice is the same as just affirming just once.

$\vdash A \textbf{ says } (A \textbf{ says } P) \to A \textbf{ says } P$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P \vdash P}\,\mathsf{id}}{P \vdash A \textbf{ aff } P}\,\textbf{aff}}{A \textbf{ says } P \vdash A \textbf{ aff } P}\,\textbf{says}L}{A \textbf{ says } (A \textbf{ says } P) \vdash A \textbf{ aff } P}\,\textbf{says}L}{A \textbf{ says } (A \textbf{ says } P) \vdash A \textbf{ says } P}\,\textbf{says}R}{\vdash A \textbf{ says } (A \textbf{ says } P) \to A \textbf{ says } P}\,{\to}R$$

The other direction of this implication is an instance of the first axiom. On the other hand, for different principals A and B we cannot prove

$$\nvdash A \textbf{ says } p \to B \textbf{ says } p$$

These axioms (together with those for intuitionistic logic) are complete for implication and affirmation and identify this as a generalization of *lax logic* [Fairtlough and Mendler, 1997]. Instead of a single modality $\bigcirc P$ we have a whole family of such modalities, indexed by principals. From the perspective of functional programming, each modality "A **says** $-$" is a strong monad [Moggi, 1989, Wadler, 1992]. Here, it relativizes reasoning to each principal; in functional programming monads can isolate the pure part of the language from effects. The connection to modal logics is further explored by Pfenning and Davies [2001]. A summary of the axioms is in Figure 1.

$$\begin{array}{l} \vdash P \to A \;\textbf{says}\; P \\ \vdash A \;\textbf{says}\; (P \to Q) \to (A \;\textbf{says}\; P \to A \;\textbf{says}\; Q) \\ \vdash A \;\textbf{says}\; (A \;\textbf{says}\; P) \to A \;\textbf{says}\; P \\ \\ \nvdash (A \;\textbf{says}\; p) \to p \end{array}$$

Figure 1: Axioms for Authorization Logic

6 An Example of Authorization

We return to our motivating example, where we have labeled each of the antecedents as before.

$$\begin{array}{l} (1) : \mathit{admin}\;\textbf{says}\;(\forall A.\,\forall R.\,\mathsf{owns}(A,R) \to \mathsf{mayOpen}(A,R)), \\ (2) : \mathit{admin}\;\textbf{says}\;(\forall A.\,\forall B.\,\forall R.\,\mathsf{owns}(A,R) \wedge \mathit{fp}\;\textbf{says}\;\mathsf{studentOf}(B,A) \to \mathsf{mayOpen}(B,R)), \\ (3) : \mathit{admin}\;\textbf{says}\;\mathsf{owns}(\mathit{fp},\mathit{ghc6017}), \\ (4) : \mathit{fp}\;\textbf{says}\;\mathsf{studentOf}(\mathit{hemant},\mathit{fp}) \\ \vdash \\ \mathit{admin}\;\textbf{says}\;\mathsf{mayOpen}(\mathit{hemant},\mathit{ghc6017}) \end{array}$$

We highlighted in blue the focus of the next inference. Using **says**R, we reduce this in one step to proving the sequent

$$\begin{array}{l} (1) : \mathit{admin}\;\textbf{says}\;(\forall A.\,\forall R.\,\mathsf{owns}(A,R) \to \mathsf{mayOpen}(A,R)), \\ (2) : \mathit{admin}\;\textbf{says}\;(\forall A.\,\forall B.\,\forall R.\,\mathsf{owns}(A,R) \wedge \mathit{fp}\;\textbf{says}\;\mathsf{studentOf}(B,A) \to \mathsf{mayOpen}(B,R)), \\ (3) : \mathit{admin}\;\textbf{says}\;\mathsf{owns}(\mathit{fp},\mathit{ghc6017}), \\ (4) : \mathit{fp}\;\textbf{says}\;\mathsf{studentOf}(\mathit{hemant},\mathit{fp}) \\ \vdash \\ \mathit{admin}\;\textbf{aff}\;\mathsf{mayOpen}(\mathit{hemant},\mathit{ghc6017}) \end{array}$$

This unlocks the affirmations by admin among the antecedents, so applying **says**L three times we get

$(1)'$: $\forall A.\, \forall R.\, \mathsf{owns}(A, R) \rightarrow \mathsf{mayOpen}(A, R)$,
$(2)'$: $\forall A.\, \forall B.\, \forall R.\, \mathsf{owns}(A, R) \wedge \mathit{fp}$ **says** $\mathsf{studentOf}(B, A) \rightarrow \mathsf{mayOpen}(B, R)$,
$(3)'$: $\mathsf{owns}(\mathit{fp}, \mathit{ghc6017})$,
(4) : fp **says** $\mathsf{studentOf}(\mathit{hemant}, \mathit{fp})$

$\vdash$

admin **aff** $\mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$

Note that fp's affirmation cannot be unlocked, since $\mathit{admin} \neq \mathit{fp}$. Now we can apply $\forall L$ three times, instantiation A, B, and C with fp, hemant, and $\mathit{ghc6017}$, respectively.

$(1)'$: $\forall A.\, \forall R.\, \mathsf{owns}(A, R) \rightarrow \mathsf{mayOpen}(A, R)$,
$(2)''$: $\mathsf{owns}(\mathit{fp}, \mathit{ghc6017}) \wedge \mathit{fp}$ **says** $\mathsf{studentOf}(\mathit{hemant}, \mathit{fp}) \rightarrow \mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$,
$(3)'$: $\mathsf{owns}(\mathit{fp}, \mathit{ghc6017})$,
(4) : fp **says** $\mathsf{studentOf}(\mathit{hemant}, \mathit{fp})$

$\vdash$

admin **aff** $\mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$

At this point we can apply $\rightarrow L$ to the antecedent $(2)''$, proving the conjunction by $(3)'$ and (4) and obtaining the new line (5).

$(1)'$: $\forall A.\, \forall R.\, \mathsf{owns}(A, R) \rightarrow \mathsf{mayOpen}(A, R)$,
$(2)''$: $\mathsf{owns}(\mathit{fp}, \mathit{ghc6017}) \wedge \mathit{fp}$ **says** $\mathsf{studentOf}(\mathit{hemant}, \mathit{fp}) \rightarrow \mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$,
$(3)'$: $\mathsf{owns}(\mathit{fp}, \mathit{ghc6017})$,
(4) : fp **says** $\mathsf{studentOf}(\mathit{hemant}, \mathit{fp})$
(5) : $\mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$

$\vdash$

admin **aff** $\mathsf{mayOpen}(\mathit{hemant}, \mathit{ghc6017})$

Now we can apply the rule of affirmation, followed by the identity to complete the proof.

7 Summary

Principals	A, B, C		
Formulas	P, Q	$::=$	$p \mid P \wedge Q \mid P \rightarrow Q \mid P \vee Q \mid \forall x.\, P(x) \mid A$ **says** P
Succedents	δ	$::=$	P true $\mid A$ **aff** P

$$\frac{}{\Gamma, P \vdash P}\ \mathsf{id}$$

$$\frac{\Gamma, P \vdash Q}{\Gamma \vdash P \to Q} \to R \qquad \frac{\Gamma \vdash P \quad \Gamma, Q \vdash \delta}{\Gamma, P \to Q \vdash \delta} \to L$$

$$\frac{\Gamma \vdash P\ \mathsf{true} \quad \Gamma \vdash Q\ \mathsf{true}}{\Gamma \vdash P \wedge Q\ \mathsf{true}} \wedge R \qquad \frac{\Gamma, P, Q \vdash \delta}{\Gamma, P \wedge Q \vdash \delta} \wedge L$$

$$\frac{\Gamma \vdash P(y)\ \mathsf{true} \quad y \notin \Gamma, P(x)}{\Gamma \vdash \forall x.\, P(x)\ \mathsf{true}} \forall R^y \qquad \frac{\Gamma, P(c) \vdash \delta}{\Gamma, \forall x.\, P(x) \vdash \delta} \forall L$$

$$\frac{\Gamma \vdash P\ \mathsf{true}}{\Gamma \vdash P \vee Q\ \mathsf{true}} \vee R_1 \quad \frac{\Gamma \vdash Q\ \mathsf{true}}{\Gamma \vdash P \vee Q\ \mathsf{true}} \vee R_2 \qquad \frac{\Gamma, P \vdash \delta \quad \Gamma, Q \vdash \delta}{\Gamma, P \vee Q \vdash \delta} \vee L$$

$$\frac{\Gamma \vdash A\ \mathbf{aff}\ P}{\Gamma \vdash (A\ \mathbf{says}\ P)\ \mathsf{true}}\ \mathbf{says}R \qquad \frac{\Gamma, P \vdash A\ \mathbf{aff}\ Q}{\Gamma, A\ \mathbf{says}\ P \vdash A\ \mathbf{aff}\ Q}\ \mathbf{says}L$$

$$\frac{\Gamma \vdash P\ \mathsf{true}}{\Gamma \vdash A\ \mathbf{aff}\ P}\ \mathbf{aff}$$

Figure 2: Affirmation Logic

References

Martín 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.

Lujo Bauer. *Access Control for the Web via Proof-Carrying Authorization*. PhD thesis, Princeton University, November 2003.

Lujo Bauer, Scott Garriss, Jonathan M. McCune, Michael K. Reiter, Jason Rouse, and Peter Rutenbar. Device-enabled authorization in the Grey system. In *Proceedings of the 8th Information Security Conference (ISC'05)*, pages 431–445, Singapore, September 2005. Springer Verlag LNCS 3650.

Henry DeYoung, Deepak Garg, and Frank Pfenning. An authorization logic with explicit time. Technical Report CMU-CS-07-166, Carnegie Mellon University, Department of Computer Science, December 2007. Revised February 2008.

Michael Dummett. *The Logical Basis of Metaphysics*. Harvard University Press, Cambridge, Massachusetts, 1991. The William James Lectures, 1976.

M. Fairtlough and M.V. Mendler. Propositional lax logic. *Information and Computation*, 137(1):1–33, August 1997.

Deepak Garg. Principal-centric reasoning in constructive authorization logic. In *Workshop on Intuitionistic Modal Logic and Applications (IMLA'08)*, July 2008. Extended and revised version available as Technical Report CMU-CS-09-120, April 2009.

Deepak Garg. *Proof Theory for Authorization Logic and Its Application to a Practical File System*. PhD thesis, Carnegie Mellon University, December 2009. Available as Technical Report CMU-CS-09-168.

Deepak Garg and Frank Pfenning. Non-interference in constructive authorization logic. In J. Guttman, editor, *Proceedings of the 19th Computer Security Foundations Workshop (CSFW'06)*, pages 283–293, Venice, Italy, July 2006. IEEE Computer Society Press.

Deepak Garg and Frank Pfenning. A proof-carrying file system. In D.Evans and G.Vigna, editors, *Proceedings of the 31st Symposium on Security and Privacy (Oakland 2010)*, pages 349–364, Berkeley, California, May 2010. IEEE. Extended version available as Technical Report CMU-CS-09-123, June 2009.

Deepak Garg, Frank Pfenning, Denis Serenyi, and Brian Witten. A logical representation of common rule for controlling access to classified information. Technical Report CMU-CS-09-139, Carnegie Mellon University, June 2009.

Gerhard 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.

Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. Notes for three lectures given in Siena, Italy. Published in *Nordic Journal of Philosophical Logic*, 1(1):11-60, 1996, April 1983. URL `http://www.hf.uio.no/ifikk/forskning/publikasjoner/tidsskrifter/njpl/vol1no1/meaning.pdf`.

Eugenio Moggi. Computational lambda calculus and monads. In *Proceedings of the Fourth Symposium on Logic in Computer Science*, pages 14–23, Asilomar, California, June 1989. IEEE Computer Society Press.

Frank Pfenning and Rowan 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.

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