Stateful Authorization Logic
– Proof Theory and a Case Study

Deepak Garg and Frank Pfenning

Carnegie Mellon University
{dg,fp}@cs.cmu.edu

Abstract. Authorization policies can be conveniently represented and reasoned about in logic. Proof theory is important for many such applications of logic. However, so far, there has been no systematic study of proof theory that incorporates system state, upon which access policies often rely. The present paper fills this gap by presenting the design and proof theory of an authorization logic BL that, among other features, includes direct support for external procedures to verify predicates on system state. We discuss design choices in the interaction between state and other features of the logic and validate the logic both foundationally, by proving relevant metatheoretic properties of the logic's proof system, and empirically, through a case study of policies that control access to sensitive intelligence information in the U.S.

Keywords: Authorization logic, proof theory, stateful policies, case study

1 Introduction

Many authorization policies rely on conditions that are controlled by the environment and whose changes are not stipulated by the policies themselves. For example, a sensitive file may be accessible to the public **if the file is marked unclassified**; an employee may enter her office **if it is between 9 AM and 5 PM on a weekday**; a doctor may read any patient's health records **if there is a medical emergency**. Conditions such as "if the file is marked unclassified", written in boldface in the previous sentence, have the following characteristics: (a) They influence consequences of the authorization policy of interest, and (b) The authorization policy itself does not stipulate when or how such conditions change, although, in conjunction with other enforcement mechanisms in the system, it may constrain who may change them (e.g., the list of individuals who may mark a file unclassified may be stipulated by the authorization policy itself). We informally call conditions satisfying these two criteria *conditions of state*, and any authorization policy relying on them a stateful authorization policy.

Many formal proposals for representing, enforcing and reasoning about stateful authorization policies use logic to represent the policies. Central to such use of logic is *proof theory*, which is used both to enforce the authorization policies through proof-carrying authorization or PCA [3,4,5], and to facilitate logical inference to analyze their consequences [11,10]. Yet, despite several papers

on proof theory of authorization logics without state [15,1,12], to the best of our knowledge, there has been no systematic work on proof theory for logics that can represent stateful authorization policies. The main objective of this paper is to fill this gap: we examine in detail the proof theory of a new logic BL in which stateful authorization policies can be represented. We validate the logic's foundations by proving several metatheoretic properties of its proof system including cut-elimination, which is a proof-theoretic statement of the logic's soundness [28,22]. Empirically, we illustrate BL and justify its expressiveness through a case study of policies for access to sensitive U.S. intelligence information. (Owing to its length, we defer the entire case study to a separate technical report [17] and present here only illustrative examples from it.) Further, we discuss subtle design choices in the interaction between state and other components of the logic. Orthogonal to our main objective, we provide a new interpretation for the common access control connective k says s [2], which increases the logic's expressiveness.

At its core, BL is a first-order intuitionistic logic. To that we add the connective k says s, which means that principal k supports statement s, and *state predicates*, a subclass of predicates that can be established through decision procedures external to the logic that may refer to the system state. Finally, in order to represent real time, we include the connective s @ $[u_1, u_2]$ from our prior work with DeYoung [12]. s @ $[u_1, u_2]$ means that formula s holds in the time interval $[u_1, u_2]$, but possibly not outside of it. Through its combination of state predicates, explicit time (the @ connective), and the says connective, BL is a very expressive authorization logic.

There are two main challenges in incorporating state in an authorization logic like BL. The first is to decide the interaction between state predicates and other features of the logic, especially explicit time. For example, if s is a state predicate then depending on the values of u_1 and u_2, s @ $[u_1, u_2]$ may refer to a property of state in the *past* (or *future*) of the authorized access, which may be difficult (resp. impossible) for a reference monitor to enforce. In BL, we make a deliberate decision to eliminate such policies by interpreting s @ $[u_1, u_2]$ as "s at the time of access" irrespective of $[u_1, u_2]$ whenever s is a state predicate. Another central decision is whether or not to treat time as a part of the state (e.g., via a constant localtime that evaluates to the time of access), as in some prior work [4,7]. We argue in Section 3.3 that this choice results in a loss of expressiveness — any policy that refers to more than one point of time cannot be expressed if time is treated as a part of state. Thus, a treatment of time through the @ connective is useful even in the presence of state predicates.

The second challenge in incorporating state is the integration of external procedures for checking state predicates with the inference rules of the logic without breaking metatheoretic properties like cut-elimination. In this regard, our proof theory is guided by, and similar to, prior work on integrating decision procedures for constraint domains into a logic [29,20]. The key idea is to formally represent the external procedure for checking state predicates by an abstract judgment $E \models i$ (any environment which satisfies all state predicates in E, also

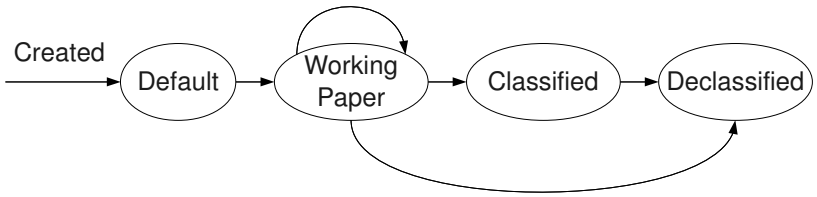

Fig. 1. Stages of a sensitive intelligence file in the U.S.

satisfies state predicate i), and stipulating only a few, reasonable conditions on the judgment. We list these conditions in Section 3.2 and show that they are enough to obtain metatheoretic properties of interest, including cut-elimination.

The rest of this paper is organized as follows. In Section 2, we present our case study, thus motivating the need for stateful authorization policies and also illustrating, by way of example, the syntax and features of BL. In Section 3, we introduce the logic and its proof theory stepwise. We start with a first-order intuitionistic logic with the connective k says s, then add state predicates, and finally add explicit time. Section 4 presents metatheoretic properties of the proof system. Related work is discussed in Section 5 and Section 6 concludes the paper. A related paper [16] presents a file system, PCFS, capable of enforcing policies represented in BL, including those presented in this paper.

2 Case Study: Stateful Authorization by Example

As a canonical example of stateful authorization policies, we introduce our case study: U.S. policies for access to sensitive intelligence information.[1] We assume that the unit of sensitive information is a digital file that is already classified or will potentially be classified. It is often necessary to share such files (among intelligence agencies, for example) and, as a result, there are federal policies that mandate who may access such files. For the purpose of this section, there are two interesting points about these policies. First, any sensitive file goes through a life cycle consisting of up to four stages that are shown in Figure 1. Rules to access a file depend on the stage the file is in, as shown in Figure 2. Second, transitions between stages are dictated by non-mechanizable factors such as human intent and beliefs and are, therefore, not prescribed by the authorization policy itself. As far as the authorization policy is concerned, the stages can change unpredictably. Hence, for the purposes of formalization and enforcement, it is easiest to represent the stage of a file as an element of system state. In the specific encoding of policies described here, the stage of a file is represented as

[1] The primary sources of policies in this paper are interviews of intelligence personnel conducted by Brian Witten and Denis Serenyi of Symantec Corporation. Some parts of the policies are based on Executive Orders of the White House [26,25] or Director of Central Intelligence Directives (DCIDs) [24,23]. The policies presented here are themselves unclassified, and may not represent official practices.

Value of extended attribute status on file F	Meaning	Who has access
default	F is in default stage	Owner
working T	F is a working paper, put into that stage at time T	Anyone, at the discretion of owner
classified T T'	F is classified from time T to time T'	Complex rules to decide access
declassified	F is declassified	Everyone

Fig. 2. Formalization of stages and permissions allowed in them

an extended attribute (file meta-data) called status, which is stored in the file system. The policy relies on this extended attribute to determine who may read the file but does not stipulate the conditions under which the attribute may change. Hence, the authorization policy is stateful in the sense mentioned in the introduction.

Formalizing state in BL. System state can be represented in BL through a special class of predicates called *state predicates*, denoted i. State predicates can be established in a proof through an external procedure, which may vary by application. For formalizing the policy at hand, we need two state predicates: (a) (**has_xattr** f a v), which means that file f has extended attribute a set to value v; in particular, a special attribute status determines the stage of a sensitive file, and (b) (**owner** f k), which means that file f is owned by principal k. We do not stipulate what principals are — they may be individuals, agencies, or groups. We assume that a procedure to check both predicates in the file system being used for implementation is available. A brief word on notation before we proceed further: we write state predicates in **boldface** to distinguish them from others and use Curried notation for applying arguments to predicates, e.g., we write (**owner** f k) instead of the more conventional **owner**(f, k).

The attribute status in our formalization can take four possible values corresponding to the four stages in Figure 1. These are listed in Figure 2, together with a description of principals who have access to the file in each stage. Technically, the words default, working, classified, and declassified are uninterpreted function symbols in BL having arities 0, 1, 2, and 0, respectively. The arguments T, T' represent time points (discussed later).

Formalizing policy rules in BL. Authorization is formalized in the logic as a predicate (may k f p), which means that principal k has permission p on file f. may k f p is *not* a state predicate because it must be established by logical deduction starting from the policy rules and valid credentials. Representative examples of formulas from our formalization of the policy are shown below:

admin claims ((may K F read) :-
has_xattr F status default,
owner F K).

admin claims $(((\texttt{may}\ K\ F\ \mathsf{read})\ \text{:-}$
$\mathbf{has\_xattr}\ F\ \mathsf{status}\ (\mathsf{classified}\ T\ T'),$
$\texttt{indi/has-clearances/file}\ K\ F,$
$\mathbf{owner}\ F\ K',$
$K'\ \mathsf{says}\ (\texttt{may}\ K\ F\ \mathsf{read}))\ @\ [T, T']).$

admin claims $(((\texttt{may}\ K\ F\ \mathsf{read})\ \text{:-}$
$\mathbf{has\_xattr}\ F\ \mathsf{status}\ (\mathsf{classified}\ T\ T'))\ @\ [T', +\infty]).$

admin claims $(((\texttt{may}\ K\ F\ \mathsf{read})\ \text{:-}$
$\mathbf{has\_xattr}\ F\ \mathsf{status}\ (\mathsf{working}\ T),$
$\mathbf{owner}\ F\ K',$
$K'\ \mathsf{says}\ (\texttt{may}\ K\ F\ \mathsf{read}),$
$T' = (T + 90d))\ @\ [T, T']).$

The notation s :- $s_1, \ldots, s_n$ means "formula s holds if formulas $s_1, \ldots, s_n$ hold", and is equal to $(s_1 \wedge \ldots \wedge s_n) \supset s$. Uppercase variables like K and F occurring in $s, s_1, \ldots, s_n$ are implicitly assumed to be universally quantified outside this formula. The prefix `admin claims` ... before each rule means that the rule is created by the principal `admin`, who is assumed to be the ultimate authority on access decisions. The prefix is formally discussed in Section 3. Accordingly, the first rule above states that (it is the admin's policy that) principal K may read file F if F is in stage default and K owns F. The latter two facts are determined by looking at the file's meta-data through the external procedure for state predicates.

The second rule illustrates two central, but possibly unfamiliar connectives of authorization logics including BL: k says s and s @ $[u_1, u_2]$. The former has been studied extensively in the context of authorization logics [2,15,1,3] and means that principal k supports formula s or declares that the formula is true. k says s can be established *a priori* in a proof through a digital certificate that contains formula s and is signed by principal k's private key. In fact, this is the *only* way to establish a priori any formula other than a state predicate, constraint (constraints are discussed below), or tautology. s @ $[u_1, u_2]$ captures real time in the logic; it means that formula s holds during the time interval $[u_1, u_2]$, but does not say anything about s outside this interval. The second rule above states that principal K may read file F if F is classified from time T to time T' (**has_xattr** F status (classified T T')), K has the right clearances to read file F (`indi/has-clearances/file` K F), and the owner K' of file F allows K to read the file (K' says (`may` K F read)). The suffix @ $[T, T']$ after the formula means that the entire formula and, hence, its consequence (`may` K F read) apply only in the interval $[T, T']$. Beyond T', the file is effectively declassified and accessible to everyone as captured in the third rule above, which means that during the interval $[T', +\infty]$, any principal K may read a file F that was classified from time T to time T'.

Our fourth rule highlights the need for another integral feature of BL — constraints, such as the atom $T' = (T + 90d)$ in the fourth formula. Constraints are similar to state predicates in that they are decided by an external procedure but different in that they are independent of state. Constraints are useful for reasoning about time. For instance, the fourth rule means that principal K may

read file F if F is a working paper, the owner K' of F allows the access, and less than 90 days have elapsed since the file became a working paper. The last condition enforces a policy mandate that a file can remain a working paper for at most 90 days.

The second, third, and fourth rules above also exemplify an interesting interaction between state and time: they apply over time intervals that depend on time values T obtained through state predicates. There are other, more interesting interactions between state and time that do not show up in our case study but are described in Section 3.3. The remaining case study is devoted to formalizing the predicate (`indi/has-clearances/file` K F) from the second rule above. This predicate relates credentials of an individual K with attributes of a classified file F to determine whether or not K should have access to F. Besides the encoding of state and time, which we have described above, this is the most challenging part of the case study and involves 38 rules. However, these rules do not refer to state, so we defer their details to a technical report [17].

3 The Logic BL: Syntax and Proof Theory

We present the logic BL stepwise. The core of BL is sorted (typed) first-order intuitionistic logic. In the first stage of presentation (Section 3.1), we consider the core with one additional connective, k says s, calling the resulting logic BL_0. Then we add state predicates, calling the logic BL_1 (Section 3.2). Finally, we add explicit time through the connective s @ $[u_1, u_2]$ to obtain the full logic BL (Section 3.3). For brevity, we omit a description of sorts, disjunction, and existential quantification as well as proofs of theorems from this paper. These details may be found in the first author's thesis [14, Chapters 3&4].

3.1 BL_0: The says Connective

The first fragment of BL we consider, BL_0, has the following syntax:

$$
\begin{array}{lrl}
\text{Terms} & t, u, k & ::= \mathsf{Alice} \mid \mathsf{Bob} \mid \mathsf{admin} \mid \ldots \\
\text{Predicates} & P & ::= \mathtt{may} \mid \ldots \\
\text{Atoms} & p & ::= P\ t_1 \ldots t_n \\
\text{Formulas} & r, s & ::= p \mid s_1 \wedge s_2 \mid s_1 \supset s_2 \mid \top \mid \bot \mid \forall x.s \mid k\ \mathsf{says}\ s
\end{array}
$$

Although we do not stipulate the domain of terms, it must include at least principals who are authorized access and who create policies (Alice, Bob, admin, etc.). Formulas are either atomic p, or built from the usual connectives of first-order logic — $\wedge$ (conjunction), $\supset$ (implication), $\top$ (truth), $\bot$ (falsity), $\forall$ (universal quantification), and the access control specific connective k says s. As explained and illustrated in Section 2, k says s means that principal k supports or states formula s, without necessarily implying that s is true. Negation, $\neg s$, may be defined as $s \supset \bot$ if needed to represent a policy.

Proof theory. When using authorization logics in practice, access is granted only if there is a proof which justifies the access. Therefore, to understand the meaning of a proposition in authorization logic, we must understand how it can be proved. This naturally leads us to the proof theory of BL_0, i.e., a systematic study of its formal proofs. We adopt Gentzen's sequent calculus style [18] in describing the proof theory and follow Martin-Löf's judgmental approach [22], which has been used previously to describe other modal logics [28,15]. Briefly, a *judgment* J is an assertion that can be established through proofs. For BL_0 we need two kinds of judgments: s true, meaning that formula s is true, and k claims s, meaning that principal k claims or supports formula s (but s may or may not be true). The latter is needed to define the meaning of the says connective. A sequent has the following form, where Γ abbreviates a multi-set $J_1, \ldots, J_n$ of judgments and Σ is a set containing all first-order variables free in Γ, s, and k.

$$\Sigma; \Gamma \xrightarrow{k} s \ \mathsf{true}$$

The informal meaning of the sequent is: "Parametrically in the variables in Σ, assuming that everything that principal k claims is true, the judgment s true follows logically from the judgments in Γ." The principal k is called the *view* of the sequent, and can be roughly thought of as the principal relative to whose claims we wish to prove the sequent (hence the hypothesis "assuming that everything that principal k claims is true ..." in the meaning of the sequent).

Sequents are established through proofs that are trees of inference rules with axioms at leaves. The following two rules relate the judgments themselves. The first rule, (init), is standard and states that if atom p is assumed as a hypothesis, then it can be concluded. We restrict the rule to atoms, but it can be proved that a generalization to arbitrary formulas holds (we prove a similar theorem for the entire logic BL in Section 4). The second rule, (claims), captures the meaning of the view of a sequent: if k claims s is assumed and the view is k, then s true can also be assumed.

$$\frac{}{\Sigma; \Gamma, p \ \mathsf{true} \xrightarrow{k} p \ \mathsf{true}}\,\mathrm{init} \qquad \frac{\Sigma; \Gamma, k \ \mathsf{claims} \ s, s \ \mathsf{true} \xrightarrow{k} r \ \mathsf{true}}{\Sigma; \Gamma, k \ \mathsf{claims} \ s \xrightarrow{k} r \ \mathsf{true}}\,\mathrm{claims}$$

Other inference rules of the sequent calculus are directed by connectives. We list below rules for the says connective. The notation $\Gamma|$ in the first rule denotes the subset of Γ containing only judgments of the form k' claims s', i.e., $\Gamma| = \{(k' \ \mathsf{claims} \ s') \mid k' \ \mathsf{claims} \ s' \in \Gamma\}$. The first rule, ($\mathsf{says}$R), can be interpreted as follows: we can establish that k says s is true in hypotheses Γ in any view k_0 (conclusion of the rule) if we can prove only from $\Gamma|$ in view k that s is true (premise). Hypotheses of the form s' true are removed in the premise because they may have been introduced in Γ in the view k_0, but may not be claimed or trusted by k. The second rule (saysL) states that the judgment (k says s) true entails the judgment k claims s. In fact, the two judgments are equivalent in BL_0. (Technically, we say that the connective says internalizes the judgment claims into the syntax of formulas.)

$$\frac{\Sigma; \Gamma| \xrightarrow{k} s\ \mathsf{true}}{\Sigma; \Gamma \xrightarrow{k_0} k\ \mathsf{says}\ s\ \mathsf{true}}\mathsf{saysR} \qquad \frac{\Sigma; \Gamma, k\ \mathsf{says}\ s\ \mathsf{true}, k\ \mathsf{claims}\ s \xrightarrow{k_0} r\ \mathsf{true}}{\Sigma; \Gamma, k\ \mathsf{says}\ s\ \mathsf{true} \xrightarrow{k_0} r\ \mathsf{true}}\mathsf{saysL}$$

Rules for connectives of first-order logic ($\wedge$, $\supset$, $\top$, $\bot$, $\forall$) are standard so we refer the reader to existing work for details [28]. In all these rules, the view passes unchanged from the premises to the conclusion. Several standard metatheoretic properties including admissibility of cut and consistency hold of BL_0's proof theory but we refrain from presenting them here because we present similar properties for the larger logic BL in Section 4.

BL_0's says connective. The says connective in BL_0 has a different interpretation from that than in prior authorization logics containing says [15,1,2]. For the benefit of readers, we list below a rule and three axioms that completely characterize the says connective of BL_0 (in terms of the sequent calculus, $\vdash s$ means that $\Sigma; \cdot \xrightarrow{k} s\ \mathsf{true}$ for any k and appropriate Σ):

From $\vdash s$ infer $\vdash k\ \mathsf{says}\ s$
$\vdash (k\ \mathsf{says}\ (s_1 \supset s_2)) \supset ((k\ \mathsf{says}\ s_1) \supset (k\ \mathsf{says}\ s_2))$
$\vdash (k\ \mathsf{says}\ s) \supset k'\ \mathsf{says}\ k\ \mathsf{says}\ s$
$\vdash k\ \mathsf{says}\ ((k\ \mathsf{says}\ s) \supset s)$

The reader may ask why we are considering a new interpretation for says, when there exist others that also have simpler proof theories, e.g., [15,1,2]. The answer lies in striking a delicate balance between having a good proof theory and having a usable meaning for the says connective, which we believe BL_0 achieves. As a case in the point for other interpretations of says with simpler proof theories, consider recent work that treats $k\ \mathsf{says}\ \cdot$ as a family of lax modalities [15,1]. Even though these logics have a simpler proof theory, their ability to express delegation is limited. Consider the formula $k\ \mathsf{says}\ ((k'\ \mathsf{says}\ s') \supset s)$, which may intuitively mean that k supports s if k' supports s'. Interpreting $k\ \mathsf{says}\ s$ as a lax modality, it (counterintuitively) suffices for k, *not* k', to state s' in order to derive $k\ \mathsf{says}\ s$ from this rule. Technically, this happens because with a lax interpretation of $k\ \mathsf{says}\ s$, the axiom $s \supset (k\ \mathsf{says}\ s)$ is admissible, which is not the case in BL_0. Such delegations arise in practical examples, as in the following rule from our case study.

$$\begin{aligned}\mathsf{admin\ claims}\ (((&\texttt{indi/has-background}\ K\ \mathsf{topsecret})\ \texttt{:-}\\ &BA\ \mathsf{says}\ (\texttt{indi/has-ssbi}\ K\ T),\\ &T' = (T + 5y))\ @\ [T, T']).\end{aligned}$$

In words, the principal admin agrees that principal K has topsecret clearance if a person certified to check others' background, BA, says that K has passed a SSBI (single scope background investigation). Although the predicate (`indi/has-ssbi` K T) has been delegated to BA by admin, the latter has no authority over this predicate. Comparison to the lax interpretation of says is merely an example, but the general point here is that obtaining an authorization logic that has a sound proof theory and high expressiveness is difficult. We justify BL_0 on both these counts — on its proof theory through metatheorems of Section 4 and on

practical usability through our case study. Of course, this does not preclude the possibility of other logics that are also good on both counts.

Using $\mathbf{BL_0}$ in practice. Like other authorization logics with a says connective, BL_0 can be used to authorize access in practice in a standard way. A fixed predicate such as may from Section 2 is used to represent permissions, and a fixed principal, e.g., admin, is assumed to be the ultimate authority in making access decisions. Access is allowed if the prevailing policy, represented as hypotheses Γ, entails admin says (may $\boldsymbol{t}$) true for appropriate arguments $\boldsymbol{t}$, i.e., if there is a proof of $\cdot; \Gamma \xrightarrow{k}$ (admin says (may $\boldsymbol{t}$)) true (the view k is irrelevant at the top-level; it can be a fresh constant). Evidence for the assumed policy Γ comes in the form of signed certificates: a digital certificate (e.g., in X.509 format [19]) containing s signed by k is taken as a priori evidence of the judgment k claims s. Of course, not all parts of a policy can be established by certificates. For example, state predicates must be checked directly on system state. Reconciling such predicates with proof theory is the main objective of this paper, to which we turn next.

3.2 $\mathbf{BL_1}$: State Predicates

To represent stateful policies, examples of which were shown in Section 2, we extend BL_0 with a special class of atomic formulas called stateful atoms, denoted i, and add a new form of hypotheses — a set of stateful atoms, E — to sequents. The resulting logic, BL_1, has sequents of the form $\Sigma; E; \Gamma \xrightarrow{k} s$ true, which informally mean that: "Parametrically in the variables in Σ, assuming that everything that principal k claims is true, the judgment s true follows logically from the judgments in Γ *in any environment that validates all stateful atoms in* E." In practice, a proof of authorization can be constructed as explained at the end of Section 3.1, except that now the stateful atoms E are also available in the proof. Assumptions in E may be discharged by an external procedure that has access to the environment or system state.

Syntax. The syntax of BL_1 formulas is shown below. The meta-variables p and t inherit their syntax from BL_0. State predicates I are assumed to be distinct from regular predicates P.

$$\begin{array}{lll} \text{State predicates} & I & ::= \textbf{has\_xattr} \mid \textbf{owner} \mid \ \ldots \\ \text{Stateful atoms} & i & ::= I\ t_1 \ldots t_n \\ \text{Formulas} & r, s & ::= p \mid i \mid s_1 \wedge s_2 \mid s_1 \supset s_2 \mid \top \mid \bot \mid \forall x.s \mid k \text{ says } s \end{array}$$

Proof theory. We incorporate relations between stateful atoms into the proof theory through an abstract judgment $\Sigma; E \models i$, which means that "for all ground instances of variables in Σ, any environment that satisfies all stateful atoms in E also satisfies atom i". We do not stipulate any rules to establish this judgment since they may vary from environment to environment. For instance, in an environment where some constraint forces that files a.txt and

b.txt always have the same value for attribute status, it may be the case that Σ; **has_xattr** a.txt status $v \models$ **has_xattr** b.txt status v, but this may not be case in other environments. In the simplest instance, the judgment may hold if and only if $i \in E$. Our metatheoretic results (Section 4) assume only the following properties of this judgment, all of which follow from its intuitive explanation.

$\Sigma; E, i \models i$ (Identity)
$\Sigma; E \models i$ implies both $\Sigma, x; E \models i$ and $\Sigma; E, E' \models i$ (Weakening)
$\Sigma; E \models i$ and $\Sigma; E, i \models i'$ imply $\Sigma; E \models i'$ (Cut)
$\Sigma, x; E \models i$ implies $\Sigma; E[t/x] \models i[t/x]$ if $\mathtt{fv}(t) \subseteq \Sigma$ (Substitution)

As explained earlier, BL_1 sequents have the form $\Sigma; E; \Gamma \xrightarrow{k} s$ true. BL_1 inherits all inference rules of BL_0 with the proviso that the new context E passes unchanged from the conclusion to premises in all rules. We do not reiterate these rules. Two new rules for reasoning about stateful atoms are added. The first rule states that the judgment i true holds if $E \models i$ for the assumed state E. The second rule means that a hypothesis i true implies that the stateful atom i holds. Together, the two rules imply that the judgment i true is equivalent to the atom i holding in the prevailing environment, which closely couples the stateful *formula* i to its intended interpretation.

$$\frac{\Sigma; E \models i}{\Sigma; E; \Gamma \xrightarrow{k} i \text{ true}}\text{stateR} \qquad \frac{\Sigma; E, i; \Gamma, i \text{ true} \xrightarrow{k} s \text{ true}}{\Sigma; E; \Gamma, i \text{ true} \xrightarrow{k} s \text{ true}}\text{stateL}$$

We list below admissible and inadmissible statements relating to stateful atoms and the says connective. The second and third statements mean that a false stateful atom signed by a principal does not contaminate the entire logic.

$\vdash i \supset k$ says i
$\not\vdash (k \text{ says } i) \supset i$
$\not\vdash (k \text{ says } i) \supset (k' \text{ says } i)$ if $k \neq k'$

3.3 BL: Explicit Time and The @ Connective

In our final increment to the logic, we add explicit time by including the connective s @ $[u_1, u_2]$. This treatment of time is very similar to that in our prior work with DeYoung for a different logic η [12] and, as a result, we describe the proof theory of the final extension only briefly. The reason for considering this extension is two-fold. First, explicit time is needed to correctly represent policy rules that have a pre-determined expiration, as well as other rules that limit the temporal validity of formulas (e.g., the second, third, and fourth rules of Section 2). Second, there are important design decisions in the interaction between state and time that we wish to highlight.

Since s @ $[u_1, u_2]$ means that s holds throughout the interval $[u_1, u_2]$, it also seems reasonable that s @ $[u_1, u_2]$ imply s @ $[u_1', u_2']$ if $u_1 \leq u_1'$ and $u_2' \leq u_2$. To make such properties admissible in the logic, we need a theory of the total order $u_1 \leq u_2$ on time points and, for expressing certain policies (e.g., the fourth rule in Section 2), we also need a theory of arithmetic over time points. We include

both by adding a single *constraint domain* of time points to the logic. From the perspective of proof theory, constraints are similar to state. However, the external procedure for solving constraints does not depend on state.

Syntax. Time points are integers or the elements $\{-\infty, +\infty\}$. The numbers represent time elapsed in seconds from a fixed point of reference. In the concrete syntax we often write time points in the format YYYY:MM:DD:hh:mm:ss. We also include the function symbol $+$ of arity 2. A new syntactic class of constraints, c, is also added. Constraints are predicates of one of two forms: $u_1 \leq u_2$ or $u_1 = u_2$.

$$
\begin{array}{lrl}
\text{Terms} & t, u, k ::= & \mathsf{Alice} \mid \mathsf{Bob} \mid \text{YYYY:MM:DD:hh:mm:ss} \mid -\infty \mid +\infty \mid \\
& & u_1 + u_2 \mid \ldots \\
\text{Constraints} & c ::= & u_1 \leq u_2 \mid u_1 = u_2 \\
\text{Formulas} & r, s ::= & p \mid i \mid c \mid s_1 \wedge s_2 \mid s_1 \supset s_2 \mid \top \mid \bot \mid \forall x.s \mid k \ \mathsf{says} \ s \mid \\
& & s \ @ \ [u_1, u_2]
\end{array}
$$

Proof theory. The addition of time requires a significant change to the logic's judgments [12]. Instead of the judgments s true and k claims s, we use refined judgments $s \circ [u_1, u_2]$ (s is true throughout the interval $[u_1, u_2]$) and k claims $s \circ [u_1, u_2]$ (k claims that s is true throughout the interval $[u_1, u_2]$). Sequents in BL have the form $\Sigma; \Psi; E; \Gamma \xrightarrow{k,u_1,u_2} s \circ [u_1', u_2']$. Here, Ψ is a set of constraints, much like E is a set of stateful atoms. The meaning of the sequent is: "Parametrically in the variables in Σ, assuming that everything that *principal* k *claims about intervals containing* $[u_1, u_2]$ is true, the judgment $s \circ [u_1', u_2']$ follows logically from the judgments in Γ in any environment that validates all stateful atoms in E, *if all constraints in* Ψ *hold*." Besides the addition of constraints as hypotheses, another change is the addition of an interval of time to the view. This is not particularly important here since we could also have constructed a logic without time intervals in views (for details of the trade-offs involved in making this choice, see [14, Section 4.4]).

Relations between constraints are incorporated into the logic through an abstract judgment $\Sigma; \Psi \models c$, which is similar to $\Sigma; E \models i$. As for the latter judgment, our metatheoretic properties rely only on basic properties of $\Sigma; \Psi \models c$, which we borrow from prior work [12]. In particular, we require that $u_1 \leq u_2$ be reflexive and transitive. Inference rules of the sequent calculus for BL are derived from those of BL_1, taking into account carefully the interaction between time and the different connectives. Although this interaction is non-trivial in most cases, it is similar to that in prior work. Accordingly, we describe here rules for only the @ connective and state predicates (the latter reflect a key design choice), describe the interaction between @ and the remaining connectives through properties, and refer the reader to the first author's thesis for remaining details of the proof theory [14, Chapter 4].

The @ connective. In BL, $s \mathbin{@} [u_1, u_2]$ internalizes the judgment $s \circ [u_1, u_2]$ into the syntax of formulas. Because $s \mathbin{@} [u_1, u_2]$ means that s holds throughout $[u_1, u_2]$, a further qualification by adding $\circ [u_1', u_2']$ as in $s \mathbin{@} [u_1, u_2] \circ [u_1', u_2']$ does not add anything to the meaning, so the judgments $s \circ [u_1, u_2]$ and $s \mathbin{@} [u_1, u_2] \circ [u_1', u_2']$ are equivalent. This results in the following two rules for the @ connective. (ν denotes an arbitrary view k, u_1, u_2.)

$$\frac{\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} s \circ [u_1, u_2]}{\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} s \mathbin{@} [u_1, u_2] \circ [u_1', u_2']} @\mathrm{R}$$

$$\frac{\Sigma; \Psi; E; \Gamma, s \mathbin{@} [u_1, u_2] \circ [u_1', u_2'], s \circ [u_1, u_2] \xrightarrow{\nu} r \circ [u_1'', u_2'']}{\Sigma; \Psi; E; \Gamma, s \mathbin{@} [u_1, u_2] \circ [u_1', u_2'] \xrightarrow{\nu} r \circ [u_1'', u_2'']} @\mathrm{L}$$

State predicates and time. If i is a stateful atom, what should $i \circ [u_1, u_2]$ mean? One possibility (which we don't use in BL) is to apply the usual meaning of $s \circ [u_1, u_2]$, implying that $i \circ [u_1, u_2]$ mean that the stateful atom i holds throughout the time interval $[u_1, u_2]$. Although intuitive, this interpretation can result in policies that are impossible to enforce. Consider, for example, the policy $((T' = (T + 5)) \wedge (i \mathbin{@} [T, T'])) \supset ((\texttt{may}\ K\ F\ \mathsf{read}) \mathbin{@} [T, T])$. Intuitively, the policy says that a principal K may read file F at time T if i holds in the interval $[T, T + 5]$. Thus, permission to access file F at time T refers to state at later points of time, which is, of course, impossible to enforce in a reference monitor.

To avoid such non-enforceable policies, we make a substantial design decision in BL: we assume that all stateful atoms are interpreted at exactly one point of time and $i \circ [u_1, u_2]$ simply means that i holds in the environment at this point of time (independent of u_1 and u_2). The logic does not stipulate what that point of time is, but it seems practical to use the time at which the access happens. In that interpretation, $i \circ [u_1, u_2]$ means that i holds at the time of access. Following this decision, the following rules for stateful atoms are self-explanatory:

$$\frac{\Sigma; E \models i}{\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} i \circ [u_1, u_2]} \mathrm{stateR} \qquad \frac{\Sigma; \Psi; E, i; \Gamma, i \circ [u_1, u_2] \xrightarrow{\nu} r \circ [u_1', u_2']}{\Sigma; \Psi; E; \Gamma, i \circ [u_1, u_2] \xrightarrow{\nu} r \circ [u_1', u_2']} \mathrm{stateL}$$

Seemingly, we are limiting the logic's expressiveness because we are eliminating (enforceable) policies that refer to stateful atoms in intervals prior to access. However, this is not a significant limitation because such policies can still be encoded by requiring evidence of the stateful atom(s) having been true in the past (e.g., a trusted observer's certificate) to exist at the time of access. As a result, we consider this design decision reasonable.

Time as a special case of state? A different possibility for including time is to treat it as a part of state without explicitly including the connective $s \mathbin{@} [u_1, u_2]$, as in some prior work [7,4]. The idea is to have an interpreted constant, e.g., $\mathsf{localtime}$, that evaluates to the time of access. Although this choice avoids the need for the @ connective, it also results in a loss of expressiveness: since there is

no way to state that a formula holds at a time other than the time of access, we can only represent policies all of whose subformulas need to hold at the time of access. In particular, a policy like $((u' = (u + 5)) \wedge (p \mathbin{@} [u, u])) \supset (p' \mathbin{@} [u', u'])$ (if predicate p holds at time u, then p' holds at $u + 5$) is impossible to represent in such a setup. Thus, a representation of explicit time through the @ connective is useful even when state is included in the logic.

Other connectives and time. The following list of admissible and inadmissible properties highlights salient points of the interaction between @ and other connectives of BL. Notably, $(s_1 \supset s_2) \mathbin{@} [u_1, u_2]$ is equivalent to having a single proof of $(s_1 \mathbin{@} [x_1, x_2]) \supset (s_2 \mathbin{@} [x_1, x_2])$ for every subinterval $[x_1, x_2]$ of $[u_1, u_2]$ (property 8 below). In the following, $s \equiv r$ denotes $(s \supset r) \wedge (r \supset s)$, $\vdash s$ means that $\Sigma; \cdot; \cdot; \cdot \xrightarrow{\nu} s \circ [u_1, u_2]$ for all u_1, u_2, ν and appropriate Σ, and $\nvdash s$ means that the latter is not true for s in the stated generality.

1. $\vdash ((u_1 \leq u_1') \wedge (u_2' \leq u_2)) \supset ((s \mathbin{@} [u_1, u_2]) \supset (s \mathbin{@} [u_1', u_2']))$
2. $\vdash ((s \mathbin{@} [u_1, u_2]) \mathbin{@} [u_1', u_2']) \equiv (s \mathbin{@} [u_1, u_2])$
3. $\vdash ((s_1 \wedge s_2) \mathbin{@} [u_1, u_2]) \equiv ((s_1 \mathbin{@} [u_1, u_2]) \wedge (s_2 \mathbin{@} [u_1, u_2]))$
4. $\vdash ((\forall x.s) \mathbin{@} [u_1, u_2]) \equiv (\forall x.(s \mathbin{@} [u_1, u_2]))$ $\quad (x \notin u_1, u_2)$
5. $\vdash \top \mathbin{@} [u_1, u_2]$
6. $\vdash (\bot \mathbin{@} [u_1, u_2]) \supset (s \mathbin{@} [u_1', u_2'])$
7. There is no interval $[u_1, u_2]$ such that $\vdash \bot \mathbin{@} [u_1, u_2]$
8. $\vdash ((s_1 \supset s_2) \mathbin{@} [u_1, u_2]) \equiv (\forall x_1.\forall x_2.(((u_1 \leq x_1) \wedge (x_2 \leq u_2) \wedge (s_1 \mathbin{@} [x_1, x_2])) \supset (s_2 \mathbin{@} [x_1, x_2])))$
9. $\vdash ((k \mathsf{\ says\ } s) \mathbin{@} [u_1, u_2]) \supset (k \mathsf{\ says\ } (s \mathbin{@} [u_1, u_2]))$
10. $\nvdash (k \mathsf{\ says\ } (s \mathbin{@} [u_1, u_2])) \supset ((k \mathsf{\ says\ } s) \mathbin{@} [u_1, u_2])$

4 Metatheory of BL

We prove several important metatheoretic properties of BL. The first lemma below states that proofs respect substitution of stateful atoms, which, in a sense, means that the proof theory preserves the meaning of the judgment $\Sigma; E \models i$. A similar property holds for constraints, but we do not state it explicitly.

Lemma 1. *$\Sigma; E \models i$ and $\Sigma; \Psi; E, i; \Gamma \xrightarrow{\nu} r \circ [u_1, u_2]$ imply $\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} r \circ [u_1, u_2]$.*

Our main metatheoretic results are admissibility of cut — the proof of a judgment can be used to discharge the same judgment used as a hypothesis in another proof — and identity — any judgment assumed as hypothesis can be concluded. Admissibility of cut is a proof-theoretic statement of soundness of a logic. Dually, identity is a proof-theoretic statement of completeness of the logic's inference rules. Together, the proofs of the two theorems show that the rules of the logic fit well with each other [28].

Theorem 1 (Admissibility of cut). *$\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} s \circ [u_1, u_2]$ and $\Sigma; \Psi; E; \Gamma, s \circ [u_1, u_2] \xrightarrow{\nu} s' \circ [u_1', u_2']$ imply $\Sigma; \Psi; E; \Gamma \xrightarrow{\nu} s' \circ [u_1', u_2']$.*

Proof. By simultaneous induction, first on the structure of s, and then on the depths of the two given derivations, as in prior work [27].

Theorem 2 (Identity). $\Sigma; \Psi; E; \Gamma, s \circ [u_1, u_2] \xrightarrow{\nu} s \circ [u_1, u_2]$.

Proof. By induction on s.

By an analysis of inference rules, it also follows that the logic is proof-theoretically consistent, i.e., $\bot$ cannot be proved a priori. Similarly, k says $\bot$ cannot be proved a priori.

Theorem 3 (Consistency). *(1)* $\Sigma; \cdot; \cdot; \cdot \overset{\nu}{\not\to} \bot \circ [u_1, u_2]$, *and (2)* $\Sigma; \cdot; \cdot; \cdot \overset{\nu}{\not\to} (k \text{ says } \bot) \circ [u_1, u_2]$.

5 Related Work

Several formal frameworks for authorization policies allow for representation of state, but no prior proposal has considered an integration of state and logic from a proof-theoretic perspective. Perhaps closest to BL's treatment of stateful atoms is the Nexus Authorization Logic (NAL) [30] that is used for authorizing access in several components of the Nexus operating system. NAL includes support for state predicates in a manner similar to that stipulated in Section 3.2, i.e., the reference monitor verifies certain predicates using trusted decision procedures that may refer to the system state. Several other logic-based frameworks for representing authorization policies [7,4,9,21] do not make a distinction between constraints and state predicates, and consequently support system state implicitly as part of their support for constraints. However, we believe that maintaining this distinction is important from the perspective of both implementation and reasoning about policies expressed in logic.

There has also been some work on declarative languages and logics in which authorization policies and *state transitions* can be represented and reasoned about simultaneously [6,8,13]. In contrast, BL's state predicates are meant to model situations where rules for state transitions are not specified. Some recent programming languages, e.g., [11,10], use type systems to enforce state-dependent authorization policies that are represented in first-order logic. Stateful atoms are not distinguished from others in the proof theory used in these languages.

The connective k says s has been included in several past proposals for writing access policies, starting with the work of Abadi et al [2]. The BL connective s @ $[u_1, u_2]$ is based on our prior work with DeYoung [12], and our treatment of constraints goes further back to work on reconciling constraint domains and proof theory of linear logic [29,20]. Study of proof theory for authorization logics was initiated in our prior work [15]. The present paper incorporates many ideas from that work, especially the use of intuitionistic first-order logic as a foundation for authorization policies.

6 Conclusion

A proof-theoretic treatment of state in an authorization logic requires careful design. Part of the complication arises due to the well-understood difficulty of reconciling decision procedures with proof theory, but most of the design choices arise in the interaction between state predicates and other features of authorization logic, in particular, explicit time. The logic BL strikes a good balance in this design space, as evident from its strong metatheoretic foundations and validation through a realistic case study.

Acknowledgments. This research was supported in part by the AFRL under grant no. FA87500720028, and the iCAST project sponsored by the National Science Council, Taiwan under grant no. NSC97-2745-P-001-001. The first author was also supported by the AFOSR MURI "Collaborative Policies and Assured Information Sharing." We thank Denis Serenyi and Brian Witten for providing textual descriptions of policies for the case study and for subsequent discussions on them, and anonymous referees for their helpful comments on this paper.

References

1. Abadi, M.: Access control in a core calculus of dependency. Electronic Notes in Theoretical Computer Science 172, 5–31 (2007), *Computation, Meaning, and Logic: Articles dedicated to Gordon Plotkin*
2. Abadi, M., Burrows, M., Lampson, B., Plotkin, G.: A calculus for access control in distributed systems. ACM Transactions on Programming Languages and Systems 15(4), 706–734 (1993)
3. Appel, A.W., Felten, E.W.: Proof-carrying authentication. In: 6th ACM Conference on Computer and Communications Security (CCS). pp. 52–62 (1999)
4. Bauer, L.: Access Control for the Web via Proof-Carrying Authorization. Ph.D. thesis, Princeton University (2003)
5. Bauer, L., Garriss, S., McCune, J.M., Reiter, M.K., Rouse, J., Rutenbar, P.: Device-enabled authorization in the Grey system. In: 8th Information Security Conference (ISC). pp. 431–445 (2005)
6. Becker, M.Y.: Specification and analysis of dynamic authorisation policies. In: 22nd IEEE Computer Security Foundations Symposium (CSF). pp. 203–217 (2009)
7. Becker, M.Y., Fournet, C., Gordon, A.D.: Design and semantics of a decentralized authorization language. In: 20th IEEE Computer Security Foundations Symposium. pp. 3–15 (2007)
8. Becker, M.Y., Nanz, S.: A logic for state-modifying authorization policies. In: 12th European Symposium on Research in Computer Security (ESORICS). pp. 203–218 (2008)
9. Becker, M.Y., Sewell, P.: Cassandra: Flexible trust management applied to health records. In: 17th IEEE Computer Security Foundations Workshop (CSFW). pp. 139–154 (2004)
10. Borgstrm, J., Gordon, A.D., Pucella, R.: Roles, stacks, histories: A triple for Hoare. Tech. Rep. MSR-TR-2009-97, Microsoft Research (2009)
11. Broberg, N., Sands, D.: Paralocks: Role-based information flow control and beyond. SIGPLAN Notices 45(1), 431–444 (2010)

12. DeYoung, H., Garg, D., Pfenning, F.: An authorization logic with explicit time. In: 21st IEEE Computer Security Foundations Symposium (CSF). pp. 133–145 (2008), extended version available as Carnegie Mellon University Technical Report CMU-CS-07-166.
13. DeYoung, H., Pfenning, F.: Reasoning about the consequences of authorization policies in a linear epistemic logic (2009), Workshop on Foundations of Computer Security (FCS), http://www.cs.cmu.edu/~hdeyoung/papers/fcs09.pdf
14. Garg, D.: Proof Theory for Authorization Logic and Its Application to a Practical File System. Ph.D. thesis, Carnegie Mellon University (2009), available as Technical Report CMU-CS-09-168
15. Garg, D., Pfenning, F.: Non-interference in constructive authorization logic. In: 19th Computer Security Foundations Workshop (CSFW). pp. 283–293 (2006)
16. Garg, D., Pfenning, F.: A proof-carrying file system. In: Proceedings of the 31st IEEE Symposium on Security and Privacy (Oakland). pp. 349–364 (2010)
17. Garg, D., Pfenning, F., Serenyi, D., Witten, B.: A logical representation of common rules for controlling access to classified information. Tech. Rep. CMU-CS-09-139, Carnegie Mellon University (2009)
18. Gentzen, G.: 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
19. Housley, R., Ford, W., Polk, W., Solo, D.: Internet X.509 public key infrastructure (1999), http://www.ietf.org/rfc/rfc2459.txt
20. Jia, L.: Linear Logic and Imperative Programming. Ph.D. thesis, Department of Computer Science, Princeton University (2008)
21. Li, N., Mitchell, J.C., Winsborough, W.: Design of a role-based trust-management framework. In: 23rd IEEE Symposium on Security and Privacy (Oakland). pp. 114–130 (2002)
22. Martin-Löf, P.: On the meanings of the logical constants and the justifications of the logical laws. Nordic Journal of Philosophical Logic 1(1), 11–60 (1996)
23. Office of the Director of Central Intelligence: DCID 1/19: Security policy for sensitive compartmented information and security policy manual (1995), http://www.fas.org/irp/offdocs/dcid1-19.html
24. Office of the Director of Central Intelligence: DCID 1/7: Security controls on the dissemination of intelligence information (1998), http://www.fas.org/irp/offdocs/dcid1-7.html
25. Office of the Press Secretary of the White House: Executive order 12958: Classified national security information (1995), http://www.fas.org/sgp/clinton/eo12958.html
26. Office of the Press Secretary of the White House: Executive order 13292: Further amendment to executive order 12958, as amended, classified national security information (2003), http://nodis3.gsfc.nasa.gov/displayEO.cfm?id=EO_13292_
27. Pfenning, F.: Structural cut elimination I. Intuitionistic and classical logic. Information and Computation 157(1/2), 84–141 (2000)
28. Pfenning, F., Davies, R.: A judgmental reconstruction of modal logic. Mathematical Structures in Computer Science 11, 511–540 (2001)
29. Saranli, U., Pfenning, F.: Using constrained intuitionistic linear logic for hybrid robotic planning problems. In: International Conference on Robotics and Automation (ICRA). pp. 3705–3710 (2007)
30. Schneider, F.B., Walsh, K., Sirer, E.G.: Nexus Authorization Logic (NAL): Design rationale and applications. Tech. rep., Cornell University (2009), http://ecommons.library.cornell.edu/handle/1813/13679