

# Stateful Authorization Logic

– Proof Theory and A Case Study

Deepak Garg   Frank Pfenning

Carnegie Mellon University

September 24, 2010

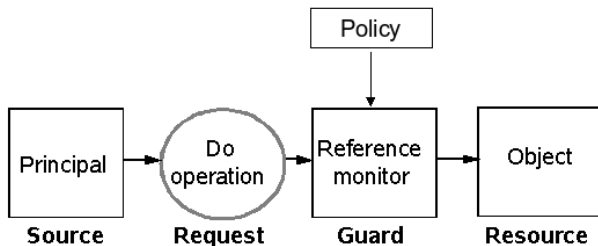
# Outline

- 1 Review
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State
- 4 MetaTheory with State
- 5 Conclusion

# Outline

- 1 **Review**
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State
- 4 MetaTheory with State
- 5 Conclusion

# Access Control



Source: Lampson et al. '92

- Authentication: Who is requesting access?
- Authorization: Does the policy allow access to the requester?

Key question: How is the policy represented and interpreted?

# Authorization Logics

- Represent the authorization policy as a logical theory [ABLP'93,AF'99]
- Represent policy elements and permissions through predicates
- Represent policy as a set of logical formulas
- Allow access iff policy *entails* permission

Logics designed for this purpose are called **authorization logics**

# Authorization Logics

- Represent the authorization policy as a logical theory [ABLP'93,AF'99]
- Represent policy elements and permissions through predicates
- Represent policy as a set of logical formulas
- Allow access iff policy *entails* permission

Logics designed for this purpose are called **authorization logics**

# Authorization Logics

- Represent the authorization policy as a logical theory [ABLP'93,AF'99]
- Represent policy elements and permissions through predicates
- Represent policy as a set of logical formulas
- Allow access iff policy *entails* permission

Logics designed for this purpose are called **authorization logics**

# Authorization Logics

- Represent the authorization policy as a logical theory [ABLP'93,AF'99]
- Represent policy elements and permissions through predicates
- Represent policy as a set of logical formulas
- Allow access iff policy *entails* permission

Logics designed for this purpose are called **authorization logics**

# Authorization Logics

- Represent the authorization policy as a logical theory [ABLP'93,AF'99]
- Represent policy elements and permissions through predicates
- Represent policy as a set of logical formulas
- Allow access iff policy *entails* permission

Logics designed for this purpose are called **authorization logics**

## Example in First-Order Logic

- $\text{may}(k, f, o)$ : Principal  $k$  is permitted to perform operation  $o$  on file  $f$
- $\text{manager}(k)$ : Principal  $k$  is a manager
- $\text{chairman}(k)$ : Principal  $k$  is a chairman
  
- “A chairman or a manager may read `accounts.xls`.”  
 $\forall k. (\text{manager}(k) \vee \text{chairman}(k)) \supset \text{may}(k, \text{accounts.xls}, \text{read})$
- “Alice is a manager”  
 $\text{manager}(\text{Alice})$
  
- Previous two facts entail  $\text{may}(\text{Alice}, \text{accounts.xls}, \text{read})$   
 $\Rightarrow$  Alice should be allowed to read `accounts.xls`

# Key Point 1: Proof Theory is Important

- Access is based on entailment (“Allow access iff policy entails permission”)
- So, entailment **defines** the meaning of authorization policies
- Consequently, formal study of entailment, i.e. **proof theory is critical**

Proof theory is central to authorization logics  
and to this paper

## Key Point 2: *A priori* Evidence?

- Permissions are inferred from policies
- But, what constitutes evidence that a **policy** holds?
- Two conceivable ways to establish a fact a priori:
  - ① A trusted principal *supports* the fact  
Example: “A chairman or a manager may read `accounts.xls`”  
may be stated by an administrator
  - ② The fact can be verified from the **state of the system**  
Example: “Alice is a manager” may be evident from the  
employment database  
*Requires trusted decision procedures for enforcement*
- Logical mechanisms for principal-stated facts well-understood
  - ▶ Connective *k says s* (Principal *k* supports formula *s*) [ABLP'93]
  - ▶ Lot of work on proof theory of *k says s* [GP'06,Aba'06,GA'08]
- Logics with stateful facts not well-studied (esp. proof theory)

## Key Point 2: *A priori* Evidence?

- Permissions are inferred from policies
- But, what constitutes evidence that a **policy** holds?
- Two conceivable ways to establish a fact a priori:
  - 1 A trusted principal *supports* the fact  
Example: “A chairman or a manager may read `accounts.xls`”  
may be stated by an administrator
  - 2 The fact can be verified from the **state of the system**  
Example: “Alice is a manager” may be evident from the  
employment database  
*Requires trusted decision procedures for enforcement*
- Logical mechanisms for principal-stated facts well-understood
  - ▶ Connective *k says s* (Principal *k* supports formula *s*) [ABLP'93]
  - ▶ Lot of work on proof theory of *k says s* [GP'06,Aba'06,GA'08]
- Logics with stateful facts not well-studied (esp. proof theory)

## Key Point 2: *A priori* Evidence?

- Permissions are inferred from policies
- But, what constitutes evidence that a **policy** holds?
- Two conceivable ways to establish a fact a priori:
  - ① A trusted principal *supports* the fact  
Example: “A chairman or a manager may read `accounts.xls`”  
may be stated by an administrator
  - ② The fact can be verified from the **state of the system**  
Example: “Alice is a manager” may be evident from the employment database  
*Requires trusted decision procedures for enforcement*
- Logical mechanisms for principal-stated facts well-understood
  - ▶ Connective *k says s* (Principal *k* supports formula *s*) [ABLP'93]
  - ▶ Lot of work on proof theory of *k says s* [GP'06,Aba'06,GA'08]
- Logics with stateful facts not well-studied (esp. proof theory)

## Key Point 2: *A priori* Evidence?

- Permissions are inferred from policies
- But, what constitutes evidence that a **policy** holds?
- Two conceivable ways to establish a fact a priori:
  - ① A trusted principal *supports* the fact  
Example: “A chairman or a manager may read `accounts.xls`”  
may be stated by an administrator
  - ② The fact can be verified from the **state of the system**  
Example: “Alice is a manager” may be evident from the  
employment database  
*Requires trusted decision procedures for enforcement*
- Logical mechanisms for principal-stated facts well-understood
  - ▶ Connective ***k* says *s*** (Principal *k* supports formula *s*) [ABLP'93]
  - ▶ Lot of work on proof theory of ***k* says *s*** [GP'06,Aba'06,GA'08]
- Logics with stateful facts not well-studied (esp. proof theory)

## Key Point 2: *A priori* Evidence?

- Permissions are inferred from policies
- But, what constitutes evidence that a **policy** holds?
- Two conceivable ways to establish a fact a priori:
  - ① A trusted principal *supports* the fact  
Example: “A chairman or a manager may read `accounts.xls`”  
may be stated by an administrator
  - ② The fact can be verified from the **state of the system**  
Example: “Alice is a manager” may be evident from the  
employment database  
*Requires trusted decision procedures for enforcement*
- Logical mechanisms for principal-stated facts well-understood
  - ▶ Connective *k* **says** *s* (Principal *k* supports formula *s*) [ABLP'93]
  - ▶ Lot of work on proof theory of *k* **says** *s* [GP'06,Aba'06,GA'08]
- Logics with stateful facts not well-studied (esp. proof theory)

## In This Paper ...

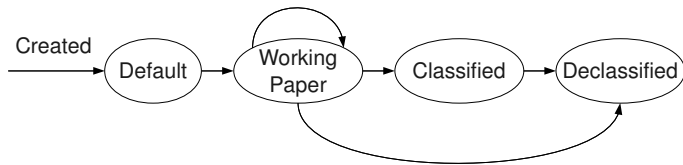
- An authorization logic (BL) with direct support for stateful facts
- Proof theory (sequent calculus) for BL
- Integration of decision procedures for state with proof theory
- Metatheoretic properties: cut-elimination, identity
- Salient interactions between state and other connectives, esp. time (covered very briefly in the talk)
- Illustration by a case study: policies for access to intelligence information in the U.S.

# Outline

- 1 Review
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State
- 4 MetaTheory with State
- 5 Conclusion

# Case Study

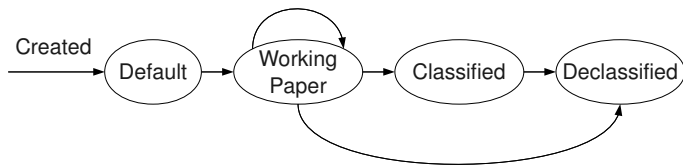
- Policies for access to sensitive intelligence information in the U.S.
- Motivating example, relies on state and almost all other motifs of authorization logics.
- Each sensitive files goes through *stages*



- Allowed access is contingent upon prevailing stage
- Stage changes are not governed by the access policy
- Credentials such as stage are called **stateful facts**

# Case Study

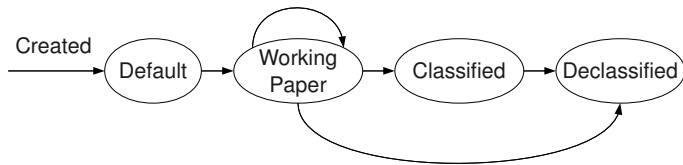
- Policies for access to sensitive intelligence information in the U.S.
- Motivating example, relies on state and almost all other motifs of authorization logics.
- Each sensitive files goes through *stages*



- Allowed access is contingent upon prevailing stage
- Stage changes are not governed by the access policy
- Credentials such as stage are called **stateful facts**

# Case Study

- Policies for access to sensitive intelligence information in the U.S.
- Motivating example, relies on state and almost all other motifs of authorization logics.
- Each sensitive files goes through *stages*



- Allowed access is contingent upon prevailing stage
- Stage changes are not governed by the access policy
- Credentials such as stage are called **stateful facts**

# Formalizing State in Logic: An Introduction

- How to represent stages of a file?
- Natural to write them in an extended attribute (file system metadata)
- How to represent file metadata in logic?
- Introduce a distinct class of predicates called **state predicates** (written in **boldface**), e.g.,
  - ▶ **has\_xattr**( $F, A, V$ ): File  $F$  has attribute  $A$  set to  $V$   
(E.g., file `secret.txt` has attribute “stage” set to “working paper”).
  - ▶ **owner**( $F, K$ ): File  $F$  has owner  $K$
- State predicates can be established (verified) directly by a trusted decision procedure that consults system state

# Formalizing State in Logic: An Introduction

- How to represent stages of a file?
- Natural to write them in an extended attribute (file system metadata)
- How to represent file metadata in logic?
- Introduce a distinct class of predicates called **state predicates** (written in **boldface**), e.g.,
  - ▶ **has\_xattr**( $F, A, V$ ): File  $F$  has attribute  $A$  set to  $V$  (E.g., file `secret.txt` has attribute “stage” set to “working paper”).
  - ▶ **owner**( $F, K$ ): File  $F$  has owner  $K$
- State predicates can be established (verified) directly by a trusted decision procedure that consults system state

# Formalizing State in Logic: An Introduction

- How to represent stages of a file?
- Natural to write them in an extended attribute (file system metadata)
- How to represent file metadata in logic?
- Introduce a distinct class of predicates called **state predicates** (written in **boldface**), e.g.,
  - ▶ **has\_xattr**( $F, A, V$ ): File  $F$  has attribute  $A$  set to  $V$   
(E.g., file `secret.txt` has attribute “stage” set to “working paper”).
  - ▶ **owner**( $F, K$ ): File  $F$  has owner  $K$
- State predicates can be established (verified) directly by a trusted decision procedure that consults system state

# An Example in BL

Policy: A file in stage “default” can be read by its owner

Formalization:

$$\text{admin says } (\forall K, F. (\mathbf{has\_xattr}(F, \text{stage}, \text{default}) \wedge \mathbf{owner}(F, K)) \supset \text{may}(K, F, \text{read}))$$

- Prefix `admin says ...` means that this policy is evident by virtue of the administrator `admin` saying it.
- Note the use of state predicates.

# An Example in BL

Policy: A file in stage “default” can be read by its owner

Formalization:

$$\text{admin says } (\forall K, F. (\mathbf{has\_xattr}(F, \text{stage}, \text{default}) \wedge \mathbf{owner}(F, K)) \supset \text{may}(K, F, \text{read}))$$

- Prefix **admin says** ... means that this policy is evident by virtue of the administrator admin saying it.
- Note the use of state predicates.

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read with proper clearances and permission of the owner

Formalization:

admin **says** ( $\forall F, K, K', T, T'$ .  
((**has\_xattr**( $F$ , stage, classified( $T, T'$ ))  $\wedge$   
indi/has-clearances/file( $K, F$ )  $\wedge$   
**owner**( $F, K'$ )  $\wedge$   
 $K'$  **says** may( $K, F$ , read))  
 $\supset$  may( $K, F$ , read))  $\textcircled{c}$  [ $T, T'$ ])

- Connective  $s \textcircled{c}$  [ $T, T'$ ]: fact  $s$  holds from time  $T$  to time  $T'$  [DGP'08]

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read with proper clearances and permission of the owner

Formalization:

$$\begin{aligned} \text{admin } \text{says } (\forall F, K, K', T, T'. \\ & ((\text{has\_xattr}(F, \text{stage}, \text{classified}(T, T')) \wedge \\ & \text{indi}/\text{has-clearances}/\text{file}(K, F) \wedge \\ & \text{owner}(F, K') \wedge \\ & K' \text{ says } \text{may}(K, F, \text{read})) \\ & \supset \text{may}(K, F, \text{read})) \textcircled{c} [T, T']) \end{aligned}$$

- Connective  $s \textcircled{c} [T, T']$ : fact  $s$  holds from time  $T$  to time  $T'$  [DGP'08]

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read with proper clearances and permission of the owner

Formalization:

$$\begin{aligned} \text{admin } \text{says } (\forall F, K, K', T, T'. \\ & ((\text{has\_xattr}(F, \text{stage, classified}(T, T')) \wedge \\ & \text{indi/has-clearances/file}(K, F) \wedge \\ & \text{owner}(F, K') \wedge \\ & K' \text{ says } \text{may}(K, F, \text{read})) \\ & \supset \text{may}(K, F, \text{read})) \textcircled{[T, T']}) \end{aligned}$$

- Interaction between state and time

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read with proper clearances and permission of the owner

Formalization:

$$\begin{aligned} \text{admin } \text{says } & (\forall F, K, K', T, T'. \\ & ((\text{has\_xattr}(F, \text{stage}, \text{classified}(T, T')) \wedge \\ & \text{indi}/\text{has-clearances}/\text{file}(K, F) \wedge \\ & \text{owner}(F, K') \wedge \\ & K' \text{ says } \text{may}(K, F, \text{read})) \\ & \supset \text{may}(K, F, \text{read})) \textcircled{[T, T']} \end{aligned}$$

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read **with proper clearances** and permission of the owner

Formalization:

$$\begin{aligned} \text{admin } \text{says } (\forall F, K, K', T, T'. \\ & ((\text{has\_xattr}(F, \text{stage}, \text{classified}(T, T')) \wedge \\ & \text{indi/has-clearances/file}(K, F) \wedge \\ & \text{owner}(F, K') \wedge \\ & K' \text{ says } \text{may}(K, F, \text{read})) \\ & \supset \text{may}(K, F, \text{read})) \text{ @ } [T, T']) \end{aligned}$$

- Most effort of the case study is in formalizing `indi/has-clearances/file(K, F)`, but no new insights about state; see TR for details

## Another Example in BL

Policy: A file  $F$  classified from time  $T$  to time  $T'$  can be read with proper clearances and **permission of the owner**

Formalization:

$$\begin{aligned} \text{admin } \text{says } (\forall F, K, K', T, T'. \\ & ((\text{has\_xattr}(F, \text{stage}, \text{classified}(T, T')) \wedge \\ & \text{indi/has-clearances/file}(K, F) \wedge \\ & \text{owner}(F, K') \wedge \\ & K' \text{ says } \text{may}(K, F, \text{read})) \\ & \supset \text{may}(K, F, \text{read})) \textcircled{t} [T, T']) \end{aligned}$$

- Example of delegation of authority from principal  $K$  to  $K'$ .

# Summary of BL and Case Study

- Authorization logic designed support stateful authorization policies
- State represented through designated state predicates (written in **boldface**).
- Connective  $k$  **says**  $s$  to represent policies made by principals
- Connective  $s @ [T, T']$  to represent time explicitly
- Interaction between state and **says**, and state and **@**
  
- Implementation of BL-based authorization in a file system, PCFS, in earlier work [Oakland'10]
  
- Case study facts:
  - ▶ 50 fixed rules (+ additional credentials for each access)
  - ▶ 2 state predicates
  - ▶ 36 non-state predicates, including  $\text{may}(k, f, o)$

# Outline

- 1 Review
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State**
- 4 MetaTheory with State
- 5 Conclusion

# A Fragment of BL with State

- Fragment of BL for this talk: First-order logic + state predicates (Paper, TR have details of  $k$  **says**  $s$ ,  $s @ [T, T']$ , and other connectives)

- Syntax of first-order (intuitionistic) logic formulas:

$$p ::= \text{may}(k, f, o) \mid \text{indi/has-clearances/file}(k, f) \mid \dots$$
$$r, s ::= p \mid r \supset s \mid r \wedge s \mid \forall x. s \mid \perp$$

- Add state predicates, denoted  $i$  in the syntax:

$i$	$::=$	$\text{has\_xattr}(f, a, v) \mid \text{owner}(f, k) \mid \dots$
$r, s$	$::=$	$p \mid r \supset s \mid r \wedge s \mid \forall x. s \mid \perp \mid i$

# A Fragment of BL with State

- Fragment of BL for this talk: First-order logic + state predicates (Paper, TR have details of  $k$  **says**  $s$ ,  $s @ [T, T']$ , and other connectives)

- Syntax of first-order (intuitionistic) logic formulas:

$$p ::= \text{may}(k, f, o) \mid \text{indi/has-clearances/file}(k, f) \mid \dots$$
$$r, s ::= p \mid r \supset s \mid r \wedge s \mid \forall x. s \mid \perp$$

- Add state predicates, denoted  $i$  in the syntax:

$$i ::= \text{has\_xattr}(f, a, v) \mid \text{owner}(f, k) \mid \dots$$
$$r, s ::= p \mid r \supset s \mid r \wedge s \mid \forall x. s \mid \perp \mid i$$

# A Fragment of BL with State

- Fragment of BL for this talk: First-order logic + state predicates (Paper, TR have details of  $k$  **says**  $s$ ,  $s @ [T, T']$ , and other connectives)

- Syntax of first-order (intuitionistic) logic formulas:

$$p ::= \text{may}(k, f, o) \mid \text{indi/has-clearances/file}(k, f) \mid \dots$$
$$r, s ::= p \mid r \supset s \mid r \wedge s \mid \forall x.s \mid \perp$$

- Add state predicates, denoted **i** in the syntax:

$\mathbf{i} ::= \mathbf{has\_xattr}(f, a, v) \mid \mathbf{owner}(f, k) \mid \dots$
$r, s ::= p \mid r \supset s \mid r \wedge s \mid \forall x.s \mid \perp \mid \mathbf{i}$

# Proof Theory (Sequent Calculus)

Proof theory of BL establishes **sequents**

First-order logic sequents:  $\Sigma; \Gamma \rightarrow s$

- $\Gamma = s_1, \dots, s_n$ , the set of assumptions (hypotheses)
- $\Sigma = x_1, \dots, x_m$ , the set of variables mentioned in the remaining sequent
- Meaning of sequent: Parametrically in variables of  $\Sigma$ , the hypotheses  $\Gamma$  entail the formula  $s$

# Proof Theory (Sequent Calculus)

Proof theory of BL establishes **sequents**

First-order logic sequents:  $\Sigma; \Gamma \rightarrow s$

- $\Gamma = s_1, \dots, s_n$ , the set of assumptions (hypotheses)
- $\Sigma = x_1, \dots, x_m$ , the set of variables mentioned in the remaining sequent
- Meaning of sequent: Parametrically in variables of  $\Sigma$ , the hypotheses  $\Gamma$  entail the formula  $s$

# Proof Theory (Sequent Calculus)

Proof theory of BL establishes sequents

Sequents with state:  $\Sigma; E; \Gamma \rightarrow s$

- $\Gamma = s_1, \dots, s_n$ , the set of assumptions (hypotheses)
- $\Sigma = x_1, \dots, x_m$ , the set of variables mentioned in the remaining sequent
- $E = i_1, \dots, i_k$ , the set of assumed state predicates
- Meaning of sequent: Parametrically in variables of  $\Sigma$ , in any environment that satisfies all state predicates in  $E$ , the hypotheses  $\Gamma$  entail the formula  $s$
- Assumptions in  $E$  can be discharged by trusted decision procedures

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x. s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x. s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules

Follow Per Martin-Löf's philosophy: Formulas are *defined* by their inference rules. In general,

- Every connective has right and left rules
- Right rules define how a connective is verified
- Left rules define how a connective, once established, can be used
- The two must accord, as established in cut-elimination and identity (later)

Examples of standard connectives:

Implication:

$$\frac{\Sigma; E; \Gamma, r \rightarrow s}{\Sigma; E; \Gamma \rightarrow r \supset s} \supset R$$

$$\frac{\Sigma; E; \Gamma \rightarrow r \quad \Sigma; E; \Gamma, s \rightarrow s'}{\Sigma; E; \Gamma, r \supset s \rightarrow s'} \supset L$$

Universal quantification:

$$\frac{\Sigma, x; E; \Gamma \rightarrow s}{\Sigma; E; \Gamma \rightarrow \forall x.s} \forall R$$

$$\frac{\Sigma; E; \Gamma, s[t/x] \rightarrow s'}{\Sigma; E; \Gamma, \forall x.s \rightarrow s'} \forall L$$

# Inference Rules for State Predicates

Right rule: How can we establish that state predicate  $i$  holds?

$$\frac{\Sigma; E \models i}{\Sigma; E; \Gamma \rightarrow i} \text{stateR}$$

- $\Sigma; E \models i$  is a new judgment (key insight)
- Meaning: For every substitution for  $\Sigma$ , every environment that satisfies each atom in  $E$  also satisfies  $i$
- Formalizes domain-specific assumptions in the logic
- No rules stipulated for establishing the judgment, since its definition may vary from domain to domain.

Left rule:

$$\frac{\Sigma; E, i; \Gamma \rightarrow s}{\Sigma; E; \Gamma, i \rightarrow s} \text{stateL}$$

# Inference Rules for State Predicates

Right rule: How can we establish that state predicate  $i$  holds?

$$\frac{\Sigma; E \models i}{\Sigma; E; \Gamma \rightarrow i} \text{stateR}$$

- $\Sigma; E \models i$  is a new judgment (key insight)
- Meaning: For every substitution for  $\Sigma$ , every environment that satisfies each atom in  $E$  also satisfies  $i$
- Formalizes domain-specific assumptions in the logic
- No rules stipulated for establishing the judgment, since its definition may vary from domain to domain.

Left rule:

$$\frac{\Sigma; E, i; \Gamma \rightarrow s}{\Sigma; E; \Gamma, i \rightarrow s} \text{stateL}$$

# Inference Rules for State Predicates

Right rule: How can we establish that state predicate  $i$  holds?

$$\frac{\Sigma; E \models i}{\Sigma; E; \Gamma \rightarrow i} \text{stateR}$$

- $\Sigma; E \models i$  is a new judgment (key insight)
- Meaning: For every substitution for  $\Sigma$ , every environment that satisfies each atom in  $E$  also satisfies  $i$
- Formalizes domain-specific assumptions in the logic
- No rules stipulated for establishing the judgment, since its definition may vary from domain to domain.

Left rule:

$$\frac{\Sigma; E, i; \Gamma \rightarrow s}{\Sigma; E; \Gamma, i \rightarrow s} \text{stateL}$$

# Inference Rules for State Predicates

Right rule: How can we establish that state predicate  $i$  holds?

$$\frac{\Sigma; E \models i}{\Sigma; E; \Gamma \rightarrow i} \text{stateR}$$

- $\Sigma; E \models i$  is a new judgment (key insight)
- Meaning: For every substitution for  $\Sigma$ , every environment that satisfies each atom in  $E$  also satisfies  $i$
- Formalizes domain-specific assumptions in the logic
- No rules stipulated for establishing the judgment, since its definition may vary from domain to domain.

Left rule:

$$\frac{\Sigma; E, i; \Gamma \rightarrow s}{\Sigma; E; \Gamma, i \rightarrow s} \text{stateL}$$

# Proof Theory of State: Summary

- Explicit stateful assumptions  $E$ ; may be established a priori through decision procedures
- Judgment  $\Sigma; E \models \mathbf{i}$  formalizes domain-specific assumptions
- Explicit stateful assumptions and domain-specific assumptions internalized into proof system in the left and right rules for state predicates.

# Outline

- 1 Review
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State
- 4 MetaTheory with State**
- 5 Conclusion

# Metatheorems

How do we check that the proof rules are coherent?

Prove meaningful metatheorems:

## Theorem (State substitution)

*If  $\Sigma; E \models \mathbf{i}$  and  $\Sigma; E, \mathbf{i}; \Gamma \rightarrow s$ , then  $\Sigma; E; \Gamma \rightarrow s$*

## Theorem (Cut)

*If  $\Sigma; E; \Gamma \rightarrow r$  and  $\Sigma; E; \Gamma, r \rightarrow s$ , then  $\Sigma; E; \Gamma \rightarrow s$*

## Theorem (Identity)

$\Sigma; E; \Gamma, s \rightarrow s$

Our proofs of cut and identity are syntactic; together imply that the rules accord with each other

[Pfe00,PD01]

# Outline

- 1 Review
  - Authorization Logic
  - Stateful Authorization Policies
- 2 Case Study and Informal Introduction to BL
- 3 Proof Theory with State
- 4 MetaTheory with State
- 5 Conclusion

# Conclusion

## Summary:

- BL designed to represent and interpret stateful authorization policies
- Reconciles decision procedures, domain-specific assumptions, and proof theory
- Metatheorems justify inference rules
- Validation by case study

## Also in the paper:

- Other connectives:  $k \text{ says } s, s @ [T, T']$
- Design choices in the interaction of state and time