Permission to Speak: A Novel Formal Foundation for Access Control

Oleg Sokolsky

Nikhil Dinesh, Insup Lee, Aravind Joshi

Outline

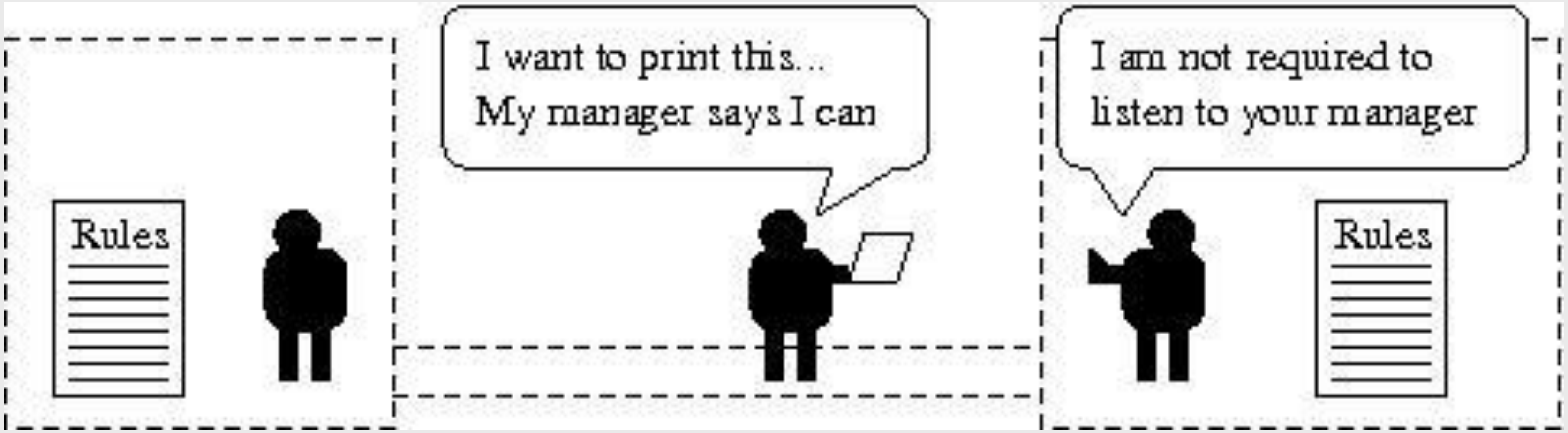
- Motivation
  - Distributed, multi-authority access control
  - Compliance checking and blame assignment
- Formal representation
  - Delegation and obligation
  - Permission as provability
- Access control and conformance checking
  - System architecture
- Summary

Logic for regulation - requirements

- Expressive enough to capture regulatory documents
- Allow systematic translation of regulation into logic formulas
  - Preserving document structure
  - Sentence-by-sentence translation
- Allow efficient compliance checking
  - Decidability
  - Low complexity for common cases

Motivation and problem statement

- Main problem of access control:
  - Should a request for service be granted?
- In a distributed system with multiple authorities:
  - Which policies need to be consulted?
  - Which policies are violated and who is to blame?



Delegation and obligation

- “saying” is a common operator in access control logics
  - Captures both policy and credential introduction
  - Policies are typically obligations and credentials are typically permissions
  - Obligations and permissions are often implicit and must be deduced by the checker
- Explicit permissions and obligations
  - Deontic operators $\mathbf{P}_A\phi$, $\mathbf{O}_A\phi$

L_{PS}: Syntax

- Two-sorted logic enforces alternation of obligations and saying

$$\varphi = \alpha \mid \varphi \wedge \varphi \mid \neg \varphi \mid says_y \psi$$

$$\psi = \varphi \mid \psi \wedge \psi \mid \neg \psi \mid O_y \varphi$$

- Permission is the dual of obligation: $P_y \varphi = \neg O_y \neg \varphi$
- L_{PS} is a decidable logic with complete semantics
- Key formal device: axiom of representation

$$\left(says_{l(A)}\left(P_B says_{l(B)} \varphi\right) \wedge says_{l(B)} \varphi\right) \Rightarrow says_{l(A)} \varphi$$

Policies

- Utterance: ground formula of the form $says_y \psi$
- A policy is a collection of sequents

$$(id)\ \varphi \mapsto \psi$$

  - Preconditions are assertions over world state and proof state (outstanding utterances)
- Evaluation:
  - True preconditions must have true postconditions
  - Postconditions make more preconditions true
    - Create new utterances

Contributions to science

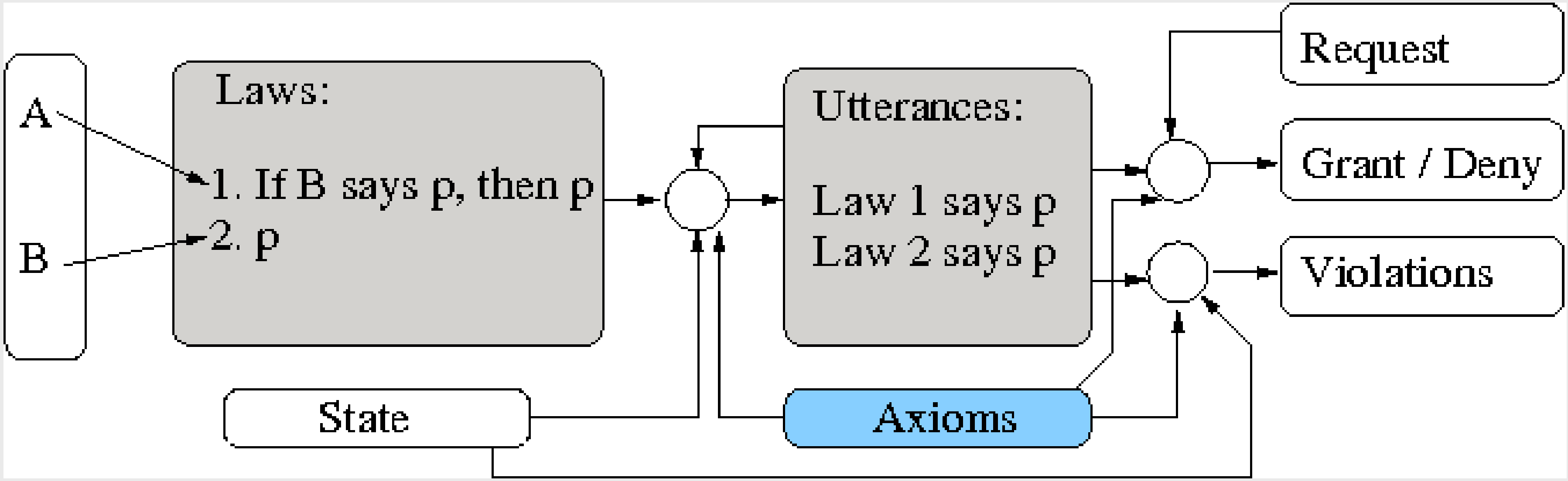
- Uniform treatment of access control and conformance
  - Access control is verification of permissions
  - Conformance is satisfaction of obligations
  - Both are formalized as provability of statements in the logic
- Clarified semantics of deontic modalities
  - Nested permissions and obligations
  - Positive and negative permissions

Nested deontic modalities

- Parents (A) should not let their children (B) play by the road
  - Multiple possible interpretations:
    - A should not give B permission to play (positive permission)
    - A should tell B not to play (negative permission)
    - A should physically prevent B from playing
  - Each interpretation make sense in some context
- Alternation with saying solves the problem
  - “require to allow” becomes “require to make a rule…”
    - $O_A\left(\neg says_{l(A)} P_B\, play_{road}(B)\right)$
    - $O_A\left(says_{l(A)} O_B \neg play_{road}(B)\right)$

System architecture

- Principals introduce laws
- Logic programming engine computes utterances, ground saying terms
- Request is granted if utterances contain a permission for it



On-going work and new results

- Translation of regulatory documents
  - NLP parser design
  - Hand-annotated sentences
- Improving checking efficiency
  - L_{PS} fragment with poly-time complexity
- Non-interference theorem
  - Which laws need to be considered?
  - Unrelated statements should not affect outcomes

Restricted logic: chain formulas

- Strict alternation between saying and obligation
- No negation

$$\varphi = \bot \mid \alpha \mid says_{l(y)} \psi$$

$$\psi = \bot \mid \alpha \mid P_y \varphi \mid O_y \varphi$$

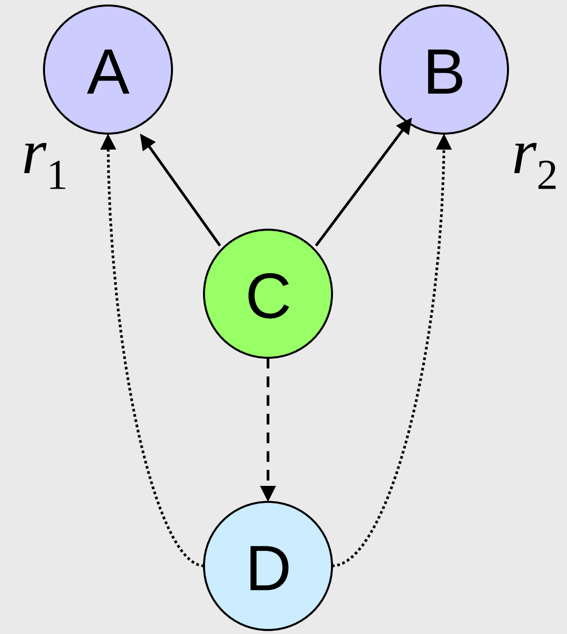
- Conjunctions can be accommodated for saying and obligations
  - Conjunction under permissions as well as negation are open problems
- Chain formulas have poly-time decision procedure

Expressive power of chain formulas

- Chain formulas are generalizations of SECPAL expressions
- Prohibitions cannot always be expressed
  - (6) A bloodbank must not ship a donation, if it tests positive for HIV
  - Gives rise to utterances: $says_{\{6\}} O_B \neg ship(d)$
  - Does not generalize to complex statements, such as “A much not prevent B from doing x”

Non-interference

- Principal C delegates to D access to resources r_1 and r_2, controlled by A and B, resp.:
  - (1) $says_A P_D\, access(D, r_1)$?
  - (2) $says_B P_D\, access(D, r_2)$?



- Computed utterances:

$$(u1)\quad says_A P_C access(C, r_1) \qquad (u3)\quad says_C P_D access(D, r_1)$$
$$(u2)\quad says_B P_C says_C P_D access(D, r_2) \qquad (u4)\quad says_C P_D access(D, r_2)$$

- For (1), need to check only (u1) [not provable]
- For (2), need to check only (u1), (u3) [provable]

Non-interference theorem

- For a set of utterances and formula $says_B \varphi$, the set of reachable utterances U^*_B contains
  - If $says_B \psi \in U$ then $says_B \psi \in U^*_B$
  - If $says_C \psi \in U^*_B$ and $says_A \psi'$ is a subformula of ψ, then $says_A \psi' \in U^*_B$

- Theorem:
  $says_B \varphi$ is provable from U if and only if it is provable from U^*_{B}