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
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 $P_A \phi$, $O_A \phi$
**L_{PS}: logic and policies**

- $L_{PS}$ is a decidable logic with complete semantics.
- Key formal device: axiom of representation

\[
\left( \text{says } l(A) \left( P_B \text{ says } l(B) \varphi \right) \right) \land \text{says } l(B) \varphi \Rightarrow \text{says } l(A) \varphi
\]

- A policy is a collection of sequents

\[
(id) \varphi \Rightarrow \psi
\]

- True preconditions must have true postconditions
- Postconditions make more preconditions true
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…”
    • $O_A \left( \neg \text{says}_{l(A)} P_B \text{play}_{\text{road}} (B) \right)$
    • $O_A \left( \text{says}_{l(A)} O_B \neg \text{play}_{\text{road}} (B) \right)$
System architecture

• Principals introduce laws
• Logic programming engine computes *utterances*, ground saying terms
• Request is granted if utterances contain a permission for it
Future work: quantitative evaluation

- $L_{PS}$ can be used as an alternative to Keynote in the QuanTM architecture
- A tighter integration with the reputation manager will be more efficient
- Quantitative semantics for $L_{PS}$ will combine TDG construction and evaluation
  - Supported by the logic programming framework of $L_{PS}$
  - Similar to probabilistic Datalog semantics