Permission to Speak: A Novel Formal Foundation for Access Control

Oleg Sokolsky
Nikhil Dinesh, Insup Lee, Aravind Joshi
<table>
<thead>
<tr>
<th>4. TITLE AND SUBTITLE</th>
<th>5. a. CONTRACT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permission to Speak: A Novel Formal Foundation for Access Control</td>
<td></td>
</tr>
<tr>
<td>6. AUTHOR(S)</td>
<td>5b. GRANT NUMBER</td>
</tr>
<tr>
<td>University of Pennsylvania, Computer and Information Science, Philadelphia, PA, 19104</td>
<td>5c. PROGRAM ELEMENT NUMBER</td>
</tr>
<tr>
<td>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</td>
<td>5d. PROJECT NUMBER</td>
</tr>
<tr>
<td>University of Pennsylvania, Computer and Information Science, Philadelphia, PA, 19104</td>
<td>5e. TASK NUMBER</td>
</tr>
<tr>
<td>8. PERFORMING ORGANIZATION REPORT NUMBER</td>
<td>5f. WORK UNIT NUMBER</td>
</tr>
<tr>
<td>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</td>
<td>10. SPONSOR/MONITOR’S ACRONYM(S)</td>
</tr>
<tr>
<td>10. SPONSOR/MONITOR’S REPORT NUMBER(S)</td>
<td></td>
</tr>
<tr>
<td>12. DISTRIBUTION/AVAILABILITY STATEMENT</td>
<td>Approved for public release; distribution unlimited</td>
</tr>
<tr>
<td>13. SUPPLEMENTARY NOTES</td>
<td></td>
</tr>
<tr>
<td>14. ABSTRACT</td>
<td></td>
</tr>
<tr>
<td>15. SUBJECT TERMS</td>
<td></td>
</tr>
<tr>
<td>16. SECURITY CLASSIFICATION OF:</td>
<td></td>
</tr>
<tr>
<td>a. REPORT unclassified</td>
<td>b. ABSTRACT unclassified</td>
</tr>
<tr>
<td>c. THIS PAGE unclassified</td>
<td></td>
</tr>
<tr>
<td>17. LIMITATION OF ABSTRACT Same as Report (SAR)</td>
<td>18. NUMBER OF PAGES 9</td>
</tr>
<tr>
<td>19a. NAME OF RESPONSIBLE PERSON</td>
<td></td>
</tr>
</tbody>
</table>
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

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

- 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…”
  • \( O_A \left( \neg says_{l(A)} P_B \text{play}_{\text{road}}(B) \right) \)
  • \( O_A \left( says_{l(A)} O_B \neg \text{play}_{\text{road}}(B) \right) \)
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