A Logic for Access Control in Distributed Systems

Example 6a

CSE 791: E-6a

Title: Access Control and Delegation Logic
SML files: none
Objective: Reveal Syntax and Semantics of the Logic

1 Introduction

The purpose of this logic is to reason about authentication and authorization in distributed systems. Authentication is the determination of who has made a request. Authorization or access control is the determination of whether the agent that makes a statement is trusted on that statement.

Essentially, the logic described here is used to answer questions such as “who is speaking?” and “who is trusted?” Answers to these questions involve principals and credentials. Informally, principals are those things (people, machines, groups, channels, etc.) that make statements; credentials are those things that provide the basis for confidence that a principal can be trusted (access control lists, certificates, etc.).

This set of examples most closely follows the ACM Transactions on Computer Systems article, “Authentication in Distributed Systems,” [1]. A more complete description of the actual calculus is found in [3]. Some additional examples as part of the TAOS operating system are developed in [2].

2 Syntax

2.1 Principals

Examples of simple principals

- Names of basic system components
  - People, e.g., Chin
  - Machines, e.g., chin.cat.syr.edu, ps printer on gunter
  - Roles, e.g., faculty, student, dean, OS, authentication server

- Names standing for sets of principals
  - Services, e.g., NFS
- Groups, e.g., CASE formal methods group

- Names in a directory structure
  - Simple names, e.g., root, SU, MIT
  - Path names, e.g., /root/SU/EECS/chin

- Channels – principals that can actually transmit information
  - Wires or I/O ports, e.g., port 5
  - Encrypted channels, e.g., DES encryption with OS key $K_O$,
  - Network address, e.g., 128.230.32.2

Examples of **compound** principals.

- Principals in roles, e.g., Chin as faculty.
- Delegations, e.g., Zhou for Chin.
- Quoting, e.g., Zhou|Chin (Zhou quoting Chin)
- Conjunction, e.g., Zhou ∧ Chin. This captures the notion of co-signers. (Note: $\land$ is overloaded as a binary operator. In this case the signature of $\land$ is $(principal \times principal) \rightarrow principal$. In propositional logic its signature is $(bool \times bool) \rightarrow bool$.)
- Traversing directory a structure, e.g., /A/B/C except D The intent of except is to force the consideration of other points in the directory structure besides the starting point, in this case any point except /A/B/C/D.

We can define well-formed expressions that denote principals (i.e., define the type $pexp$). A BNF description of $pexp$ is as follows:

```
<name> ::= 1*ALPHA *ALPHANUM
<pathname> ::= / / <pathname>/<name>
<role> ::= <name>
<pexp> ::= <name> / <pexp> ∧ <pexp> / <pexp> | <pexp> /
         <pexp> for <pexp> / <pexp> as <role> /
         <pathname> / <pathname> except <name>
```

### 2.2 Statements

Statement are things that spoken by principals. Statements are:

- **simple** statements such as “read personnel file”
- **compound** statements using logical operations \(\neg, \land, \lor, \rightarrow, \equiv\), and
- **principals saying statements** $A$ says $s$
- **principals speaking for principals** $A \Rightarrow B$, $A$ speaks for $B$
3 Semantics

3.1 Statements

The following inference rules are similar to the inference rules of propositional logic. They are used to introduce and derive new sentences or propositions.

Inference rule (S1) states that if \( s \) is a theorem in propositional logic (denoted by \( \vdash s \)) then \( s \) is also a theorem in the logic of authentication, (denoted by \( ^n s \)).

\[
\frac{\vdash s}{\check n s} \quad (S1)
\]

Inference rule (S2) is modus ponens or resolution on assumptions. If \( ^n s \) and \( ^n s \supset s' \) are theorems of the logic then so is \( ^n s' \).

\[
\frac{^n s \quad ^n s \supset s'}{^n s'} \quad (S2)
\]

Inference rule (S3) derives new statements based on conditional statements made by principals. It corresponds to modus ponens on says.

\[
^n(\check A \text{ says } s \land \check A \text{ says } (s \supset s')) \supset \check A \text{ says } s' \quad (S3)
\]

Inference rule (S4) says that if \( ^n s \) is a theorem, then any principal \( A \) saying \( s \) is also a theorem.

\[
\frac{^n s}{\check A \text{ says } s} \quad (S4)
\]

Theorem (TS5) (derivable from (S1) through (S4)) states that says distributes over \( \land \).

\[
^n \check A \text{ says } (s \land s') \equiv (\check A \text{ says } s \land \check A \text{ says } s') \quad (TS5)
\]

3.2 Principals

There are two basic operations on principals, \( \land \) (and) and \( | \) (quoting). The set of principals is closed under these operations. Axioms (P1) and (P2) define meaning of \( \land \) and \( | \).

If \( A \land B \) make a statement \( s \), then that is the same as \( A \) and \( B \) both making the same statement \( s \).
\[
\eta(A \land B) \text{ says } s \equiv (A \text{ says } s) \land (B \text{ says } s)
\]  
(P1)

A quoting B is the same as A saying B says s.

\[
\eta(A \mid B) \text{ says } s \equiv A \text{ says } B \text{ says } s
\]  
(P2)

If A and B are identical, then they say the same things.

\[
\eta(A = B) \supset ((A \text{ says } s) \equiv (B \text{ says } s))
\]  
(P3)

With respect to says and the other operators operating on principals, \(\land\) is associative, commu-
tative, and idempotent. This is derivable, and is stated in theorem (TP4).

\[
''[(A \land B) \land C = A \land (B \land C)] \land [A \land B = B \land A] \land [A \land A = A]
\]  
(TP4)

(P5) states axiomatically that quoting | is associative.

\[
\eta(A \mid B) \mid C = A \mid (B \mid C)
\]  
(P5)

(P6) states axiomatically that quoting distributes over conjunction of principals.

\[
\eta[A \mid (B \land C) = (A \mid B) \land (A \mid C)] \land [(A \land B) \mid C = (A \mid C) \land (B \mid C)]
\]  
(P6)

(P7) defines \(\text{speaks for } (\Rightarrow)\) in terms of conjunction.

\[
\eta(A \Rightarrow B) \equiv (A = A \land B)
\]  
(P7)

(TP8) is a property of \(\Rightarrow\) derivable from (P7) and (P3).

\[
''(A \Rightarrow B) \supset ((A \text{ says } s) \supset (B \text{ says } s))
\]  
(TP8)

(TP9) is a property of =.

\[
''(A = B) \equiv ((A \Rightarrow B) \land (B \Rightarrow A))
\]  
(TP9)

(P10) is the **handoff axiom**.
(TP11) is the handoff theorem or handoff rule.

\[''((\mathcal{A}' \Rightarrow \mathcal{A}) \wedge \mathcal{A}' \text{ says } (\mathcal{B} \Rightarrow \mathcal{A})) \supset (\mathcal{B} \Rightarrow \mathcal{A})\]  

(TP12) is the joint authority theorem.

\[''((\mathcal{A}' \wedge \mathcal{B} \Rightarrow \mathcal{A}) \wedge (\mathcal{B} \Rightarrow \mathcal{A}')) \supset (\mathcal{B} \Rightarrow \mathcal{A})\]  

(P13) is the delegation axiom.

\[''(\mathcal{A} \text{ says } ((\mathcal{B} | \mathcal{A}) \Rightarrow (\mathcal{B} \text{ for } \mathcal{A}))) \supset ((\mathcal{B} | \mathcal{A}) \Rightarrow (\mathcal{B} \text{ for } \mathcal{A}))\]  

### 3.3 Path Names

\[\pi P \text{ except } M \Rightarrow P\]  

(N1)

(N2) allows us to move “down” in the hierarchy.

\[\pi M \neq N \supset (P \text{ except } M) | N \Rightarrow P/N \text{ except } '..'\]  

(N2)

(N3) allows us to move “up” in the hierarchy.

\[\pi M \neq '..' \supset (P/N \text{ except } M) | '..' \Rightarrow P \text{ except } N\]  

(N3)

### 3.4 Roles and Programs

\[R \in \text{Roles} \quad \pi \mathcal{A} \text{ as } R = \mathcal{A} | R\]  

(R1)

\[R \in \text{Roles} \quad \pi \mathcal{A} \Rightarrow \mathcal{A} \text{ as } R\]  

(R2)
\[ P \in \text{Roles} \quad R \in \text{Roles} \]
\[ \overline{[A \text{ as } P \text{ as } R = A \text{ as } R \text{ as } P]} \wedge [A \text{ as } R \text{ as } R = A \text{ as } R] \quad (R3) \]

### 3.5 Delegation

\[ \overline{(A \wedge B \mid A)} \Rightarrow B \text{ for } A \quad \text{(D1)} \]

\[ \overline{B \Rightarrow B'} \quad \overline{A \Rightarrow A'} \]
\[ \overline{(B \text{ for } A) \Rightarrow (B' \text{ for } A')} \wedge [A \text{ for } (B \wedge C) = (A \text{ for } B) \wedge (A \text{ for } C)] \quad (D2) \]

### 3.6 Derivable Properties

(TP14) is the monotonicity of as.

\[ \overline{(B \Rightarrow B' \wedge A \Rightarrow A')} \supset ((B \text{ as } A) \Rightarrow (B' \text{ as } A')) \quad \text{(TP14)} \]

(TP15) is the monotonicity of and.

\[ \overline{(B \Rightarrow B' \wedge A \Rightarrow A')} \supset ((B \wedge A) \Rightarrow (B' \wedge A')) \quad \text{(TP15)} \]

(TP16) is the monotonicity of |.

\[ \overline{(B \Rightarrow B' \wedge A \Rightarrow A')} \supset ((B \mid A) \Rightarrow (B' \mid A')) \quad \text{(TP16)} \]

(TP17) is the reflexivity of \( \Rightarrow \).

\[ \overline{A \Rightarrow A} \quad \text{(TP17)} \]

(TP18) is the transitivity of \( \Rightarrow \).

\[ \overline{(A \Rightarrow B \wedge B \Rightarrow C) \Rightarrow (A \Rightarrow C)} \quad \text{(TP18)} \]

(TP19) is the antisymmetry of \( \Rightarrow \).
References


A Proof of \((\mathcal{A} \Rightarrow \mathcal{B} \land \mathcal{B} \Rightarrow \mathcal{C}) \supset (\mathcal{A} \Rightarrow \mathcal{C})\)

<table>
<thead>
<tr>
<th>A1. (A \Rightarrow B)</th>
<th>(\mathcal{A} \Rightarrow \mathcal{B} \land \mathcal{B} \Rightarrow \mathcal{C}) (\supset (\mathcal{A} \Rightarrow \mathcal{C}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2. (B \Rightarrow C)</td>
<td>(</td>
</tr>
<tr>
<td>G1. (A \Rightarrow C)</td>
<td>By P7</td>
</tr>
<tr>
<td>A3. (A = A \land B)</td>
<td>By A3, A4 EQ-LR</td>
</tr>
<tr>
<td>A4. (B = B \land C)</td>
<td>By P7</td>
</tr>
<tr>
<td>A5. (A = A \land (B \land C))</td>
<td>By A5, P4</td>
</tr>
<tr>
<td>A6. (A = (A \land B) \land C)</td>
<td>By A5, P7</td>
</tr>
<tr>
<td>A7. (A = A \land C)</td>
<td>By A7, P7</td>
</tr>
<tr>
<td>A8. (A \Rightarrow C)</td>
<td>A8, G1 resolution</td>
</tr>
<tr>
<td>G2. (true)</td>
<td></td>
</tr>
</tbody>
</table>