Deconstructing Trust Management

Ajay Chander*
Computer Science Department
Stanford University
ajayc@cs.stanford.edu

Drew Dean†
Computer Science Laboratory
SRI International
ddean@csl.sri.com

John C. Mitchell*
Computer Science Department
Stanford University
mitchell@cs.stanford.edu

Abstract

We present soundness and completeness theorems for our earlier work on modeling trust management systems. We then investigate the addition of distributed structured naming to the model, and prove that naming and authorization are orthogonal in a precise sense. The orthogonality theorem gives us soundness and completeness theorems for the entire system, in effect yielding a clean decomposition between naming and authorization concerns.

1 Introduction

Trust management [4] combines a distributed naming model along with authorization. One popular model has been the SDSI/SPKI work. Much ink has been spilled examining the naming portion of SDSI [1, 6], and more recently its interaction with SPKI [7]. We believe, though, that the problem should submit to a simple solution, and find it instructive to attempt to understand what is so hard to formalize about SDSI's naming, which was originally given with only an operational definition. Hence, we present here a clean sheet design, with an extremely simple formalization.

Earlier work [3, 2, 10] has combined delegation and naming. While this leads to some interesting properties, and the reuse of delegation to control the naming hierarchy, this time we chose to separate the two. In our earlier work [5], we presented a distributed authorization system without naming. We believe that they can each be studied individually, and compose nicely.

This paper makes three main technical contributions. First, we complete our earlier work with soundness and completeness theorems. Second, we present a naming system as a set of rewrite rules for name resolution, and prove that these rules commute with the logical rules associated with authorization. This commutativity theorem proves that naming is orthogonal to authorization in a precise sense. Lastly, the commutativity result leads to soundness and completeness results for the augmented system, thereby confirming our hypothesis that naming and authorization compose cleanly.

The most closely related work to this is the study of SDSI's naming. Lampson and Rivest proposed SDSI in 1996 [10]. Abadi did the first formal study of SDSI's name resolution algorithm in 1998 [1]. Halpern and van der Meyden followed Abadi's work with two influential papers [6, 7]. Li presented a logic programming formulation of the problem that is much simpler than the other work [8]. Our graphical model of access control is in some ways similar to the Take-Grant model of Lipton and Snyder [9].

*Supported in part by DARPA contract N66001-00-C-8015 and ONR grant N00014-97-1-0505.
†Supported in part by DARPA contract N66001-00-C-8015 and ONR grant N00014-01-1-0837.

	O	R	A	D
Create(o_c, o)	$\cup\{o\}$		$(o, r_e) \mapsto (o_c, 1)$ $(o, r) \mapsto \emptyset$	$(s, r, o) \mapsto \emptyset$
Add(o, r, o_s, d)			$(o, r) \mapsto A((o, r)) \cup \{o_s, d\}$	
Remove(o, r, o_s, d)			$(o, r) \mapsto A((o, r)) - \{o_s, d\}$	
Delegate(o_s, o, r, o_d, d)				$(o_s, r, o) \mapsto D((o_s, r, o)) \cup \{o_d, d\}$
Revoke(o_s, o, r, o_d, d)				$(o_s, r, o) \mapsto D((o_s, r, o)) - \{o_d, d\}$
Delete(o)	$-\{o\}$		$\vert$	$\vert$

Table 1: Trust Management

2 A State-Transition Model for Authorization

In earlier work [5], we presented a model for deciding the access question in distributed trust management systems, based on a labeled-transition modeling comprised of abstract system states, state transitions, and logical deduction of access control judgments. Furthermore, the model was general enough to capture a variety of access control mechanisms, including flavors of access control lists and capabilities, and the relationship between these access control systems was formalized in terms of simulation and bisimulation relations between the corresponding labeled transition systems. For completeness, we recap the basic model as it applies to trust management systems, below. In Section 5, we will show how this modeling composes with our formalization of distributed, structured names to give a sound and complete model for distributed trust management with names.

The general state-transition based approach to the modeling of access control mechanisms in systems specifies the following:

- A *world state*, the part of the system configuration that is relevant to the access control mechanism,
- A set of possible *actions*, each defining a transition mapping from world states to world states.
- An *access judgment*, which states when one object can access another. This may be specified in the form of logical inference, equivalent to some implementable algorithm. Given a world state WS, the judgment that subject s can exercise the right r on object o is written as $WS \vdash s \to (o, r)$.

For the case of trust management without local name spaces, the world state consists of a set O of objects, a set R of rights and two maps A (bounded root ACL) and D (bounded Delegate):

$$\begin{aligned} A &: O \times R \to \mathcal{P}(O \times \mathbb{N}) \\ D &: O \times R \times O \to \mathcal{P}(O \times \mathbb{N}) \end{aligned}$$

The maps A and D capture the two ways through which access privileges are propagated in this framework—either they are granted directed by the host object via the root ACL map A, or they are delegated through other objects that hold the access right, through the delegation map D. Each of these methods provides for a delegation depth which bounds further granting privileges, and we capture this by the set $\mathbb{N} = \{0, 1, \ldots\}$ in the domain of the two functions. We call this system TM.

An action is specified by how it changes the components of the world state w in which it is invoked. Table 1 summarizes the actions for the trust management model.

We specify the access judgment as a logical judgment, given the following four inference rules. The component maps of the world state can be interpreted as set-membership predicates; $\mathsf{ACL}(s, o, r, d)$ is true iff $(s, d) \in A((o, r))$, and $\mathsf{Del}(s, o, r, r_s, d)$ is true iff $(r_s, d) \in D(s, o, r)$. Subject s can access the (o, r) pair iff it can produce a proof of $\mathsf{Access}(s, o, r, d)$, for some d, from the predicate equivalent of the world state and the following four inference rules:

$$\begin{array}{lr} (\textit{RootACL}) & \mathsf{ACL}(A, B, r, d) \vdash \mathsf{Access}(A, B, r, d) \\ (\textit{Delegation}) & \mathsf{Access}(A, B, r, d+1), \mathsf{Del}(A, B, r, C, d) \\ & \vdash \mathsf{Access}(C, B, r, d) \\ (\textit{Ord1}) & \mathsf{Access}(A, B, r, d+1) \vdash \mathsf{Access}(A, B, r, d) \\ (\textit{Ord2}) & \mathsf{Del}(A, B, r, C, d+1) \vdash \mathsf{Del}(A, B, r, C, d) \end{array}$$

The first two rules capture the root ACL and delegation chain mechanisms of obtaining access, and the last two capture the "downward closure" property of delegation depths.

3 Theorems

We define a model $\mathcal{M}$ as the 4-tuple $O^{\mathcal{M}}, R^{\mathcal{M}}, A^{\mathcal{M}}, D^{\mathcal{M}}$ where $O^{\mathcal{M}}$ and $R^{\mathcal{M}}$ are non-empty sets and $A^{\mathcal{M}}, D^{\mathcal{M}}$ are maps with the following signatures:

$$\begin{array}{rcl} A^{\mathcal{M}} & : & O^{\mathcal{M}} \times R^{\mathcal{M}} \to P(O^{\mathcal{M}} \times \mathbb{N}) \\ D^{\mathcal{M}} & : & O^{\mathcal{M}} \times R^{\mathcal{M}} \times O^{\mathcal{M}} \to P(O^{\mathcal{M}} \times \mathbb{N}) \end{array}$$

such that $\forall o^{\mathcal{M}}, s^{\mathcal{M}}, {r_s}^{\mathcal{M}} \in O^{\mathcal{M}}, r^{\mathcal{M}} \in R^{\mathcal{M}}. \forall n \in \mathbb{N}:$

$$\begin{array}{rcl} ({r_s}^{\mathcal{M}}, n+1) \in D^{\mathcal{M}}(s^{\mathcal{M}}, r^{\mathcal{M}}, o^{\mathcal{M}}) & \Rightarrow & ({r_s}^{\mathcal{M}}, n) \in D^{\mathcal{M}}(s^{\mathcal{M}}, r^{\mathcal{M}}, o^{\mathcal{M}}) \\ (s^{\mathcal{M}}, n+1) \in A^{\mathcal{M}}(o^{\mathcal{M}}, r^{\mathcal{M}}) & \Rightarrow & (s^{\mathcal{M}}, n) \in A^{\mathcal{M}}(o^{\mathcal{M}}, r^{\mathcal{M}}) \end{array}$$

In other words, both maps A and D respect the downward-closure of delegation depths. Given a world state WS, we say that a model $\mathcal{M}$ satisfies WS, written $\mathcal{M} \models WS$, if the following conditions hold:

1. For each element $o \in O, r \in R$, there exist unique elements $o^{\mathcal{M}} \in O^{\mathcal{M}}, r^{\mathcal{M}} \in R^{\mathcal{M}}$ respectively.
2. The maps $A^{\mathcal{M}}$ and $D^{\mathcal{M}}$ are faithful to the maps A and D respectively. Formally, if

$$(([\![o]\!], s), d) \in A(([\![o]\!], \text{SELF}), r) \Rightarrow (([\![o]\!]^{\mathcal{M}}, s^{\mathcal{M}}), d) \in A^{\mathcal{M}}(([\![o]\!]^{\mathcal{M}}, \text{SELF}), r^{\mathcal{M}})$$
$$(([\![s]\!], r_s), d) \in D(([\![s]\!], \text{SELF}), r, ([\![s]\!], o)) \Rightarrow (([\![s]\!]^{\mathcal{M}}, {r_s}^{\mathcal{M}}), d) \in D^{\mathcal{M}}(([\![s]\!], \text{SELF}), r^{\mathcal{M}}, ([\![s]\!]^{\mathcal{M}}, o))$$

A model $\mathcal{M}$ can be thought of as an equivalent directed labeled access graph $\{O^{\mathcal{M}}, E\}$ where

$$\begin{array}{rcl} (s,d) \in A^{\mathcal{M}}(o,r) & \iff & (s,(o,r,d),o) \in E \\ (r_s,d) \in D^{\mathcal{M}}(s,r,o) & \iff & (r_s,(o,r,d),s) \in E \end{array}$$

In words, the set of vertices of the graph associated with model $\mathcal{M}$ is the set of elements of $O^{\mathcal{M}}$, and there is an edge between vertices u and v iff u can access a right on vertex v, or if v delegates a right (on some other vertex) to u. The edge itself is labeled with a tuple (o,r,d) specifying the object/right pair and further delegation powers d. A subject, s, can access right r on object o, written $\mathcal{M} \models s \to (o,r)$, iff there exists a directed *access path* from s to o marked with tuples of the sort (o,r,d) and for every prefix of the path (including the entire path), the number of edges in the prefix is $\leq 1+$ the delegation bound, (d_i), in the last edge of the prefix.

$$o \xleftarrow{(o,r,d_1)} s_1 \leftarrow \ldots \leftarrow s_i \xleftarrow{(o,r,d_{i+1})} s_{i+1}$$

We say that $\mathcal{M} \models \mathsf{Access}(s,o,r,d)$ iff the first edge of some access path from s to o is marked with (o,r,d).

THEOREM 3.1 *Given an arbitrary world state WS, subject $s \in O$, object $o \in O$ and right $r \in R$, $WS \vdash \mathsf{Access}(s,o,r,d) \Rightarrow \forall \mathcal{M}. \mathcal{M} \models WS \Rightarrow \mathcal{M} \models \mathsf{Access}(s,o,r,d')$ for all $d' \leq d$*

Proof sketch. We proceed by induction on the length of the proof of $WS \vdash \mathsf{Access}(s,o,r,d)$; for each proof rule, we combine the access path of the antecedent (provided by the induction hypothesis) with the fact that $\mathcal{M} \models WS$ to obtain the access path of the consequent.

COROLLARY 3.2 (Soundness) *Given an arbitrary world state WS, subject $s \in O$, object $o \in O$ and right $r \in R$, $WS \vdash s \to (o,r) \Rightarrow \forall \mathcal{M}. \mathcal{M} \models WS \Rightarrow \mathcal{M} \models s \to (o,r)$*

THEOREM 3.3 (Completeness) *Let WS be an arbitrary world state and let $s,o \in O, r \in R$. Then, $[\forall \mathcal{M}.(\mathcal{M} \models WS) \Rightarrow (\mathcal{M} \models s \to (o,r))] \Rightarrow WS \vdash s \to (o,r)$*

Proof sketch. Each model $\mathcal{M}$ that satisfies WS contains the sets $O^{\mathcal{M}}$ and $R^{\mathcal{M}}$, and has edges in the graph corresponding to the maps A and D of the world state. Since $\mathcal{M} \models s \to (o,r)$, there exists a valid access path in the graph, whose edges correspond to instances of the ACL and Del predicates that hold in the world state. These can be combined in a straight-forward manner using the rules $\vdash$ to construct a proof of $WS \vdash s \to (o,r)$.

4 Structured Names

It is desirable to express access control policies with respect to names, not keys, for two major reasons:

1. Policies are often written in terms of people, and humans think about people by name, not key.
2. A person's key is likely to change over time. Keys get both inadvertently lost (recoverable by no one) and maliciously compromised by attackers. The process of a person getting a new key should not require policy changes.

These reasons all argue for policies to be expressed in terms of names, not keys. However, the actual access control decisions get made in terms of keys, hence we need a mapping between the surface syntax of the policy language and the keys used to decide access control.

Syntax We denote the set of keys by $\mathcal{K}$. Our modeling assumes keys are globally unique identifiers, used to authenticate principals in a distributed system. The intention is that every name in the system eventually resolves to a key. We denote the set of names by $\mathcal{N}$, which is a union of the set of local names $\mathcal{N}_{\mathcal{L}}$, and the set of global names $\mathcal{N}_{\mathcal{G}}$. The set of names $\mathcal{N}$ is distinguished from the set of keys $\mathcal{K}$, *i.e.* $\mathcal{K} \cap \mathcal{N} = \emptyset$.

We assume the existence of a set of universally recognized global names $\mathcal{N}_{\mathcal{G}}$. Such mechanisms exist in other naming schemes, *e.g.*, SDSI provides for distinguished global names. A global name g is meant to have context independent semantics, *i.e.*, if $[\![n]\!]$ denotes the meaning of name n, then $\forall n_1, n_2 \in \mathcal{N} : g \in \mathcal{N}_{\mathcal{G}} \Rightarrow [\![n_1.g]\!] = [\![n_2.g]\!]$. We identify names with evaluation environments below; as a corollary of the condition above we get that the meaning of a global name in the empty environment, $[\![g]\!]$, is well- defined.

The set of local names consists of *base names* chosen from an appropriate alphabet, and *compound names* created through linking. A local name l is therefore defined by the grammar: $l ::= b|l.b$. Note that the grammar is left recursive, and indicates the natural operational semantics of parsing a linked local name. To avoid confusion with global names, we assume that the alphabet for base names excludes the set $\mathcal{N}_{\mathcal{G}}$, *i.e.*, $b \in \Sigma^{\star} - \mathcal{N}_{\mathcal{G}}$.

Name Semantics We model the collection of information relevant to the name resolution process held by an entity, by its *name space*. Informally, the name space of a name is the structure pointed to by the name, which may contain as its fields, local name definitions in terms of keys or pointers to local names in other name spaces, and the identity (key) behind the name. In particular, we assume that every name space has a distinguished element, SELF, which resolves to the current key of the owner of the name space. We call this structure a NameObject.

A local name definition $b = n$ defines the local name b to be n, where n is a NameObject or a key. We denote the name space of a name $n \in \mathcal{N}$ by $[\![n]\!]$. We have that, for each $n \in \mathcal{N}$, $[\![n]\!]$: NameObject and $[\![n.\mathrm{SELF}]\!] : \mathcal{K}$. We use $n\downarrow$ as shorthand for $[\![n]\!].\mathrm{SELF}$.

We now have the machinery to define fully qualified names (FQNs): a FQN is a $([\![N]\!], N.\vec{n}) \in \mathsf{NameObject} \times \mathcal{N}$ pair. A FQN is simply a local name which has the value of the starting NameObject provided to resolve the name appropriately. The following facts directly follow from definitions:

FACT 4.1 *A FQN evaluates to the same key in any environment (name space).*

FACT 4.2 *Global names and compound names containing global names are FQNs. For a global name g and compound name $\vec{b}.g.n$, the corresponding FQN is $([\![g]\!], n)$.*

The interpretation of a name definition depends on the environment that it appears in. We abuse notation slightly, and define the name-definition interpretation function $[\![]\!]$ inductively as follows:

$$\begin{array}{rcll} [\![n]\!][\![b \stackrel{\mathrm{def}}{=} k]\!] & = & [\![[\![n]\!].b]\!].\mathrm{SELF} \leftarrow k & (\mathbf{Key}) \\ [\![n]\!][\![b \stackrel{\mathrm{def}}{=} n_l]\!] & = & [\![[\![n]\!].b]\!] \leftarrow [\![[\![n]\!].n_l]\!] & (\mathbf{Link}), \text{where} \end{array}$$

$$[\![n]\!][\![n_1.n_2 \ldots n_k]\!] = [\![\ldots [\![[\![n]\!].n_1]\!].n_2 \ldots]\!].n_k]\!].$$

Notice that name definitions are not defined without the name space that provides the environment, and thus we avoid scope extrusion problems that in conjunction with primitives like "speaks-for" have led to logical inconsistencies [1].

We may now define the *name resolution* process ($\leadsto$) operationally. Given a fully qualified name ($[\![n]\!], n.n_1.n_2 \ldots n_k$), we say that

$$[\![n]\!][n_1.n_2 \ldots n_k] \leadsto [\![\ldots[\![[\![n]\!].n_1]\!].n_2 \ldots]\!].n_k]\!].\mathrm{SELF}$$

where the evaluation of the key is done inside out, corresponding to a left-to-right traversal of NameObject pointers for $n_1.n_2 \ldots n_k$. For completeness, we may assume the existence of a special key $\perp_K$ to be used in case this process encounters dangling pointers.

5 Authorization and Naming

We now extend the syntax of the basic trust management model TM to allow for names. This effectively replaces the occurrences of subjects with NameObjects that contain their identifying keys. We assume that objects are referenced through subjects that own them.

The world state WS_N of the trust management system TM_N is defined to be the tuple

$$WS_\mathsf{N} = (K, L, G, B, R, NS, A_\mathsf{N}, D_\mathsf{N})$$

where K is the set of keys, L is the set of local names, with G as the set of global names and B as the set of base names. The set of name spaces, NS, consists of records, each of which has a SELF field, which takes on a value in the set K, and a set of mappings $B \to NS$ from base names to name spaces. The maps A_N and D_N are counterparts of the maps A and D of the system TM, and capture root ACL and delegation mechanisms of granting access respectively. (Here F is the set of fully qualified names; $F = NS \times L$.)

$$\begin{aligned} A_\mathsf{N} &: F \times R \to P(F \times \mathbb{N}) \\ D_\mathsf{N} &: F \times R \times F \to P(F \times \mathbb{N}) \end{aligned}$$

We note the change in the domain and co-domain sets from the set of objects O(in the system TM) to F. The context independent semantics of fully qualified names ensures that these global maps are meaningful.

The actions of TM_N can reference local names and are therefore interpreted in the context of a name space. We illustrate how they may be compiled into actions of the system TM, by resolving names to keys.

$$\begin{aligned} [\![s]\!][\![\mathrm{Add}_\mathsf{N}(s, r, n_s, d)]\!] &= \mathrm{Add}(s\downarrow, r, [\![s]\!][n_s], d) \\ [\![s]\!][\![\mathrm{Remove}_\mathsf{N}(s, r, n_s, d)]\!] &= \mathrm{Remove}(s\downarrow, r, [\![s]\!][n_s], d) \\ [\![s]\!][\![\mathrm{Del}_\mathsf{N}(s, o, r, n_d, d)]\!] &= \mathrm{Del}(s\downarrow, o, r, [\![s]\!][n_d], d) \\ [\![s]\!][\![\mathrm{Revoke}_\mathsf{N}(s, o, r, n_d, d)]\!] &= \mathrm{Revoke}(s\downarrow, o, r, [\![s]\!][n_d], d) \\ [\![s_c]\!][\![\mathrm{Create}_\mathsf{N}(s_c, s)]\!] &= \mathrm{Create}(s_c\downarrow, [\![s_c]\!][s]) \\ [\![s]\!][\![\mathrm{Delete}_\mathsf{N}(([\![o]\!], \mathrm{SELF}))]\!] &= \mathrm{Delete}(o\downarrow) \end{aligned}$$

If WS_N is a world space of the system TM_N, $WS_\mathsf{N}\downarrow$ stands for the world space of the system TM where NameObjects have been replaced by keys, and the $A_\mathsf{N}, D_\mathsf{N}$ maps compiled like the actions above.

The access judgment rules in this system are derived from the rules of the base system TM (Section 2), with additional support for names. The world state WS_N is transformed into its (signed) predicate equivalent: $\mathsf{ACL}_\mathsf{N}([\![B]\!].A, [\![B]\!], r, d)_{B\downarrow}$ holds iff $(([\![B]\!], A), d) \in A_\mathsf{N}(([\![B]\!], \mathrm{SELF}), r)$ and $\mathsf{Del}_\mathsf{N}([\![A]\!], o, r, [\![A]\!].C, d)_{A\downarrow}$ holds iff $(([\![A]\!], C), d) \in D_\mathsf{N}(([\![A]\!], \mathrm{SELF}), r, ([\![A]\!], o))$.

$$\begin{array}{ll} (\mathit{RootACL}_\mathsf{N}) & \mathsf{ACL}_\mathsf{N}([\![B]\!].A, [\![B]\!], r, d)_{B\downarrow} \vdash_\mathsf{N} \mathsf{Access}_\mathsf{N}([\![B]\!][A], B\downarrow, r, d) \\ (\mathit{Delegation}_\mathsf{N}) & \mathsf{Access}_\mathsf{N}(A\downarrow, o, r, d+1), \mathsf{Del}_\mathsf{N}([\![A]\!], o, r, [\![A]\!].C, d)_{A\downarrow} \\ & \vdash_\mathsf{N} \mathsf{Access}_\mathsf{N}([\![A]\!][C], o, r, d) \\ (\mathit{Ord1}_\mathsf{N}) & \mathsf{Access}_\mathsf{N}(A, B, r, d+1) \vdash_\mathsf{N} \mathsf{Access}_\mathsf{N}(A, B, r, d) \\ (\mathit{Ord2}_\mathsf{N}) & \mathsf{Del}_\mathsf{N}(A\downarrow, o, r, [\![A]\!].C, d+1)_{A\downarrow} \vdash_\mathsf{N} \mathsf{Del}_\mathsf{N}(A\downarrow, o, r, [\![A]\!].C, d)_{A\downarrow} \end{array}$$

The signatures on the predicates provide integrity assurances by matching the signing key with the identity of the originating name space.

Theorem 5.1 (Commutativity) *For any world state WS_N of the system* TM_N*, and access formula Q_N, the following commutative diagram holds ($\leadsto$ represents name resolution):*

$$
\begin{tikzcd}
WS_N \arrow[r, "\vdash_N"] \arrow[d, "\leadsto"'] & Q_N \arrow[d, "\leadsto"] \\
WS_N\downarrow \arrow[r, "\vdash"'] & Q_N\downarrow
\end{tikzcd}
$$

Proof sketch. Let p be a hybrid proof combining rules of all systems. We proceed by induction on the number of pairs of proof steps that are not in the desired order, considering each possible pair separately, and show that it may be rearranged as desired.

Semantics We define a model M as the tuple $(K^{\mathcal{M}}, R^{\mathcal{M}}, L^{\mathcal{M}}, G^{\mathcal{M}}, B^{\mathcal{M}}, NS^{\mathcal{M}}, A_{\mathsf{N}}{}^{\mathcal{M}}, D_{\mathsf{N}}{}^{\mathcal{M}})$ where $K^{\mathcal{M}}, R^{\mathcal{M}}, L^{\mathcal{M}}, G^{\mathcal{M}}, B^{\mathcal{M}}$ and $NS^{\mathcal{M}}$ are non-empty sets and $A_{\mathsf{N}}{}^{\mathcal{M}}$, $D_{\mathsf{N}}{}^{\mathcal{M}}$ are maps with the following signatures ($F^{\mathcal{M}} = NS^{\mathcal{M}} \times L^{\mathcal{M}}$):

$$
\begin{array}{lcl}
A_{\mathsf{N}}{}^{\mathcal{M}} & : & F^{\mathcal{M}} \times R^{\mathcal{M}} \rightarrow P(F^{\mathcal{M}} \times \mathbb{N}) \\
D_{\mathsf{N}}{}^{\mathcal{M}} & : & F^{\mathcal{M}} \times R \times F^{\mathcal{M}} \rightarrow P(F^{\mathcal{M}} \times \mathbb{N}), \text{ such that}
\end{array}
$$

$$
\begin{array}{c}
(([\![o']\!], s), d) \in A_{\mathsf{N}}{}^{\mathcal{M}}(([\![o]\!], l), r) \Rightarrow o' = o \wedge l = \text{SELF} \\
(([\![s']\!], r_s), d) \in D_{\mathsf{N}}{}^{\mathcal{M}}(([\![s]\!], l), r, ([\![s'']\!], o)) \Rightarrow (s' = s'' = s) \wedge l = \text{SELF}
\end{array}
$$

In other words, the root ACL map $A_{\mathsf{N}}{}^{\mathcal{M}}$ must consistently name the object and subject in question as belonging to the name space of the owner of the object. The delegate map $D_{\mathsf{N}}{}^{\mathcal{M}}$ should name the delegating subject s, object o, and delegatee subject r_s as belonging to the name space of the subject s.

In addition, the maps $A^{\mathcal{M}}$ and $D^{\mathcal{M}}$ satisfy the downward-closure property for delegation depths, *i.e.*, $\forall [\![s]\!] \in NS^{\mathcal{M}}, o, r_s \in L^{\mathcal{M}}, n \in \mathbb{N}$:

$$
\begin{array}{c}
(([\![o]\!], s), n+1) \in A^{\mathcal{M}}(([\![o]\!], \text{SELF}), r) \Rightarrow (([\![o]\!], s), n) \in A^{\mathcal{M}}(([\![o]\!], \text{SELF}), r) \\
(([\![s]\!], r_s), n+1) \in D^{\mathcal{M}}(([\![s]\!], \text{SELF}), r, ([\![s]\!], o)) \Rightarrow (([\![s]\!], r_s), n) \in D^{\mathcal{M}}(([\![s]\!], \text{SELF}), r, ([\![s]\!], o))
\end{array}
$$

Given a world state WS_{N}, we say that a model $\mathcal{M}$ satisfies WS_{N}, written $\mathcal{M} \models WS_{\mathsf{N}}$, if the following conditions hold:

1. For each element $k \in K, r \in R$, there exist unique elements $k^{\mathcal{M}} \in K^{\mathcal{M}}, r^{\mathcal{M}} \in R^{\mathcal{M}}$ respectively.

2. For each element $g \in G$, there exists a unique element $\mathsf{GNO}(g) \in G^{\mathcal{M}}$. We assume the existence of a global name oracle, GNO, which, semantically speaking, provides a central point for resolution of global names so that they may mean the same in any name space.

3. For each element $b \in B, l \in L$, there exist unique elements $b^{\mathcal{M}} \in B^{\mathcal{M}}, l^{\mathcal{M}} \in L^{\mathcal{M}}$ respectively.

4. The set $NS^{\mathcal{M}}$ is a set of records, each of which has a SELF field, and a name resolution function E that associates a name space with a base name. For each element $ns \in NS$, there exists a unique element $ns^{\mathcal{M}} \in NS^{\mathcal{M}}$ such that

$$
\begin{array}{rcl}
ns.\text{SELF} = k & \Longleftrightarrow & ns^{\mathcal{M}}.\text{SELF} = k^{\mathcal{M}} \\
ns.b = ns' & \Longleftrightarrow & ns^{\mathcal{M}}.E(b^{\mathcal{M}}) = ns'^{\mathcal{M}}
\end{array}
$$

5. The maps $A_{\mathsf{N}}{}^{\mathcal{M}}$ and $D_{\mathsf{N}}{}^{\mathcal{M}}$ are faithful to the maps A_{N} and D_{N} respectively. Formally, if

$$
\begin{array}{c}
(([\![o]\!], s), d) \in A_{\mathsf{N}}(([\![o]\!], \text{SELF}), r) \Rightarrow (([\![o]\!]^{\mathcal{M}}, s^{\mathcal{M}}), d) \in A_{\mathsf{N}}{}^{\mathcal{M}}(([\![o]\!]^{\mathcal{M}}, \text{SELF}), r^{\mathcal{M}}) \\
(([\![s]\!], r_s), d) \in D_{\mathsf{N}}(([\![s]\!], \text{SELF}), r, ([\![s]\!], o)) \Rightarrow (([\![s]\!]^{\mathcal{M}}, r_s{}^{\mathcal{M}}), d) \in D_{\mathsf{N}}{}^{\mathcal{M}}(([\![s]\!], \text{SELF}), r^{\mathcal{M}}, ([\![s]\!]^{\mathcal{M}}, o))
\end{array}
$$

We can formulate an equivalent description of the model as a directed labeled access graph, with NameObjects as vertices and edges marked with tuples of the sort $([\![o]\!], r, d)$ derived from the maps $A_{\mathsf{N}}, D_{\mathsf{N}}$. A subject s can access right r on an object o in the model, written $M \models_{\mathsf{N}} s \to (o, r)$, iff there exists a directed path from $[\![s]\!]$ to $[\![o]\!]$ in the access graph marked with tuples of the sort $([\![o]\!], r, d)$ and for every prefix of the path (including the entire path), the number of edges in the prefix is $\leq 1+$ the delegation bound, (d_i), in the last edge of the prefix.

$$[\![o]\!] \xleftarrow{([\![o]\!],r,d_1)} [\![s_1]\!] \leftarrow \ldots \leftarrow [\![s_i]\!] \xleftarrow{([\![o]\!],r,d_{i+1})} [\![s_{i+1}]\!]$$

LEMMA 5.2 (Model Compilation) *For any world state WS_N of TM_N and model M,*

$$\exists M'.[M \models WS_N \iff M' \models WS_N\!\downarrow] \wedge [M \models_N s \to (o,r) \iff M' \models s\!\downarrow \to (o\!\downarrow, r)]$$

Proof. Construct the access graph of the model M' from the access graph of the model M by replacing the vertices with the keys corresponding to the SELF fields, and the edges $([\![o]\!], r, d)$ by $(o\!\downarrow, r, d)$. Since the world state $WS_{\mathsf{N}}\!\downarrow$ is a replacement of the names in WS_{N} by the keys they resolve to, the lemma follows immediately.

THEOREM 5.3 (Soundness) *Given an arbitrary world state WS_{N}, fully-qualified subject $s \in F$, object $o \in O$ and right $r \in R$, $WS_N \vdash s \to (o,r) \Rightarrow \forall \mathcal{M}.\mathcal{M} \models WS_N \Rightarrow \mathcal{M} \models s \to (o,r)$.*

Proof. Given $WS_{\mathsf{N}} \vdash s \to (o,r)$, it follows from Theorem 5.1 that $WS_{\mathsf{N}}\!\downarrow \vdash s\!\downarrow \to (o\!\downarrow, r)$. Suppose $\mathcal{M} \models WS_{\mathsf{N}}$, and let $\mathcal{M}'$ be the model given by Lemma 5.2. Then, $\mathcal{M}' \models WS_{\mathsf{N}}\!\downarrow$, and by soundness of $\vdash$ (Corollary 3.2), we get that $\mathcal{M}' \models s\!\downarrow \to (o\!\downarrow, r)$. The result now follows from Lemma 5.2.

THEOREM 5.4 (Completeness) *Let WS_N be an arbitrary world state and let $s \in F, o \in O$ and $r \in R$. Then, $[\forall \mathcal{M}.(\mathcal{M} \models WS_N) \Rightarrow (\mathcal{M} \models s \to (o,r))] \Rightarrow WS_N \vdash s \to (o,r)$.*

Proof. Given any $\mathcal{M} \models WS_{\mathsf{N}}$, it follows from Lemma 5.2 and the hypothesis of the theorem that $\exists \mathcal{M}'.\mathcal{M}' \models WS_{\mathsf{N}}\!\downarrow \wedge \mathcal{M}' \models s\!\downarrow \to (o\!\downarrow, r)$. From the completeness of the rules $\vdash$, we get that $WS_{\mathsf{N}}\!\downarrow \vdash s\!\downarrow \to (o\!\downarrow, r)$. Theorem 5.1 gives us $WS_{\mathsf{N}} \vdash s \to (o,r)$.

Notice that our soundness and completeness theorems make no assumptions on the size of world states or models. The induction hypothesis uses the correspondence between sub-paths and sub-proof trees, and doesn't constrain their size. However, in practice, if we implement a decision procedure which *computes* the path given a proof (or vice versa), then the input to this decision procedure would necessarily be finite and thus the corresponding path (or proof) would be finite as well. The argument extends to names as well: any path in the model with names will refer to a finite number of names, and thus the corresponding path in the compiled model will also be finite.

6 Conclusion

We have presented and analyzed a framework for trust management which incorporates distributed naming and authorization as separate, independent primitives. The separation of naming and authorization leads to a very clean and simple formalization, and allows us to avoid primitives like "speaks-for" that have proven troublesome in the past [1, 6]. The commutativity theorem expresses this orthogonality precisely, and is used to derive soundness and completeness theorems for the full system. We view the simplicity of those proofs as compelling evidence in favor of separating naming and authorization concerns in trust management systems, and analyzing the full system by composing their semantics.

References

[1] Martin Abadi. On SDSI's linked local name spaces. *Journal of Computer Security*, 6:3–21, 1998.

[2] Dirk Balfanz. *Access Control for Ad-hoc Collaboration*. PhD thesis, Princeton University, January 2001.

[3] Dirk Balfanz, Drew Dean, and Mike Spreitzer. Security infrastructure for distributed Java applications. In *Proceedings of the 2000 IEEE Symposium on Research in Security and Privacy*.

[4] Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized trust management. In *Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy*.

[5] Ajay Chander, Drew Dean, and John C. Mitchell. A state-transition model of trust management and access control. *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, pages 27–43, May 2001.

[6] J. Halpern and R. van der Meyden. A logic for SDSI's linked local name spaces. *Journal of Computer Security*, 9(1,2):75–104, 2001.

[7] J. Halpern and R. van der Meyden. A logical reconstruction of SPKI. *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, pages 59–70, 2001.

[8] Ninghui Li. Local names in SPKI/SDSI. *Proceedings of the 13th Computer Security Foundations Workshop*, pages 2–15, July 2000.

[9] Richard J. Lipton and Lawrence Snyder. A linear time algorithm for deciding subject security. *Journal of the ACM*, 24(3):455–464, 1977.

[10] Ron Rivest and Butler Lampson. SDSI - A Simple Distributed Security Infrastructure. `http://theory.lcs.mit.edu/~cis/sdsi/sdsi11.html`, October 1996.