A Logic for SDSI's Linked Local Name Spaces

Joseph Y. Halpern
Cornell University
Dept. of Computer Science
Ithaca, NY 14853
halpern@cs.cornell.edu
http://www.cs.cornell.edu/home/halpern

Ron van der Meyden
School of Computer Science and Engineering,
University of New South Wales,
Sydney 2052,
Australia.
meyden@cse.unsw.edu.au
http://www.cse.unsw.edu.au/~meyden

April 11, 2024

Abstract

Abadi has introduced a logic to explicate the meaning of local names in SDSI, the Simple Distributed Security Infrastructure proposed by Rivest and Lampson. Abadi's logic does not correspond precisely to SDSI, however; it draws conclusions about local names that do not follow from SDSI's name resolution algorithm. Moreover, its semantics is somewhat unintuitive. This paper presents the Logic of Local Name Containment, which does not suffer from these deficiencies. It has a clear semantics and provides a tight characterization of SDSI name resolution. The semantics is shown to be closely related to that of logic programs, leading to an approach to the efficient implementation of queries concerning local names. A complete axiomatization of the logic is also provided.

1 Introduction

Rivest and Lampson [RL96] introduced SDSI—a Simple Distributed Security Infrastructure—to facilitate the construction of secure systems.[1] In SDSI, principals (agents) are identified with public keys. In addition to principals, SDSI allows other names, such as `poker-buddies`. Rather than having a global name space, these names are interpreted locally, by each principal. That is, each principal associates with each name a set of principals. Of course, the interpretation of a name such as `poker-buddies` may be different for each agent. However, a principal can "export" his bindings to other principals. Thus, Ron may receive a message from the principal he names Joe describing a set of principals Joe associates with `poker-buddies`. Ron may then refer to the principals Joe associates with `poker-buddies` by the expression `Joe's poker-buddies`.

Rivest and Lampson [RL96] give an operational account of local names; they provide a name-resolution algorithm that, given a principal k and a name n, computes the set of principals associated with n according to k. Abadi [Aba98] has provided a logic that, among other things, gives a more semantic account of local names. According to Abadi, its purpose "is to explain local names in a general, self-contained way, without requiring reference to particular implementations." Abadi shows that the SDSI name-resolution algorithm can be captured in terms of a collection of sound proof rules in his logic.

Abadi's focus is on axioms. He constructs a semantics, not with the goal of capturing the intended meaning of his constructs, but rather, with the goal of showing that certain formulas are not derivable from his axioms. (In particular, he shows that *false* is not derivable, showing that his axioms are consistent.) While adequate for Abadi's restricted goals, his semantics validates some formulas that we certainly would not expect to be valid. One consequence of this is that, while he is able to pinpoint some potential concerns with the logic, the resolution of these concerns is less satisfactory. For example, he observes that adding two seemingly reasonable axioms to his logic allows us to reach quite an unreasonable conclusion. However, it is not obvious from the semantic intuitions provided by Abadi which (if either) of the axioms is unreasonable, or why it is unreasonable. Moreover, while he proves that this particular unreasonable conclusion is not derivable in his framework, as we show, a closely related (and equally unreasonable) conclusion is in fact valid. This means we have no assurance that it or other similar formulas cannot be derived from Abadi's axioms.

We very much subscribe to Abadi's goal of using a logic to give a general account of naming. In this paper, we provide a logic whose syntax is very similar to Abadi's, but whose semantics is quite different and, we believe, captures better the meaning we intend the constructs to have. Nevertheless, all but one of Abadi's name space axioms are sound in our system.

We remark that, in a sense, our task is much easier than Abadi's, since we give the constructs in the logic a somewhat narrower reading than he does. Abadi tends to intertwine and occasionally identify issues of naming with issues of rights and delegation. (Such an identification is also implicitly made to some extent in designs such as PolicyMaker [BFL96].) We believe that it is important to treat these issues separately. Such a separation allows us to both

[1] SDSI now forms the basis for the Simple Public Key Infrastructure (SPKI) standardization work [Gro98]. SPKI simplifies some SDSI features (e.g., it eliminates groups) but adds many others. We focus in this paper on the core naming features of SDSI—there are some minor differences in the way that SPKI has chosen to handle these features, but we believe that our work is equally relevant to the the fragment of SPKI dealing with naming.

give a cleaner semantics for each of the relevant notions and to clarify a number of subtleties. This paper focuses on naming, which we carefully separate from the other issues; a companion paper [HvdMS99] considers authority and delegation.

We believe that our approach has a number of significant advantages:

- We can still simulate the SDSI's name resolution algorithm; Abadi's extra axiom is unnecessary. In fact, our logic captures SDSI's name resolution more accurately than Abadi's. Abadi's logic can draw conclusions that SDSI's name resolution cannot; our logic, in a precise sense, draws exactly the same conclusions as SDSI's name resolution algorithm.

- According to our semantic intuition, one of Abadi's proposed additional axioms is in fact quite unreasonable; it does not hold under our semantics, and it is quite clear why.

- We are able to provide a sound and complete axiomatization of our logic. Thus, unlike Abadi, we have a proof system that corresponds precisely to our semantics. This will allow us to prove stronger results than Abadi's about formulas that cannot be derived in our framework. Our completeness proof also yields a (provably optimal) NP-complete decision procedure for satisfiability of formulas in the logic.

- Our logic is closely related to Logic Programming. This allows us to translate queries about names to Logic Programming queries, and thus use all the well-developed Logic Programming technology to deal with such queries.

- Our approach opens the road to a number of generalizations, which allow us to deal with issues like permission, authority, and delegation [HvdMS99].

The rest of this paper is organized as follows. In Section 2, we review Abadi's logic and, in the process, describe SDSI's naming scheme. We also point out what we see as the problems with Abadi's approach. In Section 3, we give the syntax and semantics of our logic, and present a complete axiomatization. In Section 4 we show that our logic provides a tight characterization of SDSI name resolution. Section 5 deals with the connection between our account of SDSI name resolution and logic programming, and Section 6 concerns `Self`, an additional construct considered by Abadi. Section 7 concludes.

2 SDSI's Name Spaces and Abadi's Logic

In this section, we briefly review SDSI's naming scheme and Abadi's logic, and discuss our criticism of Abadi's logic. Like Abadi, we are basing our discussion on SDSI 1.1 [RL96].

2.1 SDSI's Name Spaces

SDSI has local names and a set of reserved names, which we refer to as global names. Both are associated with sets of principals, but the set of principals associated with a local name depends on the principal owning the local name space, while the set of principals associated with a global name does not. We denote the set of global names by G with generic element g, the set of local names by N with generic element n, and the set of keys (principals) by K with generic element k. We assume that all these sets are pairwise disjoint and that K is nonempty. Global identifiers are either keys or global names.[2]

The elements of $K \cup G \cup N$ are said to be simple names. We form principal expressions from simple names inductively. Simple names are principal expressions, and if p and q are principal expressions, then so is $(p\text{'s } q)$. Abadi's semantics (and ours) makes the latter operation associative, in that $((p\text{'s } q)\text{'s } r)$ and $(p\text{'s } (q\text{'s } r))$ have the same meaning. In light of this, we can ignore parenthesization when writing such expressions. The expression $p_1\text{'s} \ldots p_{m-1}\text{'s } p_m$ is written in SDSI as $(\mathrm{ref}: p_1; \ldots; p_m)$.[3] We remark for future reference that SDSI has a special global name denoted "DNS!!", which represents the root of the DNS (Internet mail) hierarchy; this allows us to express an email address such as bob@fudge.com as DNS!!'s com's fudge's bob.

SDSI allows a principal to issue certificates of the form $n \mapsto p$, signed with its key. If k issues such a certificate, it has the effect of binding local name n in k's name space to the principals denoted by the principal expression p.[4] Notice that only principals issue certificates, and that these certificates bind a local name (not a global name) to some set of principals. In general, a local name may be bound to a unique principal, no principal, or many principals. SDSI allows a principal k to issue certificates $n \mapsto p_1$ and $n \mapsto p_2$. This has the effect of binding n to (at least) the principals denoted by p_1 and p_2.

SDSI provides a name-resolution algorithm for computing the set of principals bound to a name. The core of the algorithm consists of a nondeterministic procedure REF2. For ease of exposition, we take REF2 to have four arguments: a principal k, a function c that associates with each principal k' a set of bindings (intuitively, ones that correspond to certificates signed by k'), a function γ which associates with each global name g a set of principals (intuitively, the ones bound to g), and a principal expression p. $\mathrm{REF2}(k, \gamma, c, p)$ returns the principal(s) bound to p in k's name space, given the bindings γ and the certificates c. REF2 is nondeterministic; the set of possible outputs of REF2 is taken to be the set of principals bound to p in k's name space. REF2 is described in Figure 1.[5]

2.2 Abadi's Logic: Syntax, Semantics, and Axiomatization

The formulas in Abadi's logic are formed by starting with a set of primitive propositions and formulas of the form $p \mapsto p'$, where p and p' are principal expressions. More complicated for-

[2] Note that Abadi uses G for global identifier; thus, his G corresponds to our $G \cup K$.

[3] SDSI allows m to be 0, taking (ref:) to be the current principal. In Section 6, we follow Abadi by considering an expression Self that represents (ref:).

[4] SDSI also allows other forms of binding that we do not consider here. Our notation is also a simplification of that used by SDSI.

[5] Our version of REF2 is similar, although not identical, to Abadi's. Like Abadi's, it is simpler than that in [RL96], in that we do not deal with a number of issues, such as quoting or encrypted objects, dealt with by SDSI. Our presentation of REF2 differs from Abadi's mainly in its treatment of global names. Abadi assumes that REF2 takes only two arguments, o and p, where o is either a global identifier (i.e., an element of $G \cup K$) or current principal, denoted cp. Although he does not write c explicitly as an argument, he does assume that there is a set he denotes assumptions(o) that includes bindings corresponding to signed certificates. In addition, it includes bindings for cp. We do not have a distinguished current principal; rather, if the current principal is k, then for uniformity we assume that all of the current principal's bindings are also described by the bindings in $c(k)$. More significantly, if g is a global name, then Abadi's $\mathrm{REF2}(o, g)$ would return g, while ours would return some principal k to which g is bound in γ. Our approach seems more consistent with the SDSI presentation of REF2, but this difference is minor, and all of Abadi's results hold for our presentation of REF2.

```
REF2(k, σ, c, p)
  if p ∈ K then return(p)
  else if p ∈ G
     then if σ(p) = ∅ then fail
          else return(k') for some k' ∈ σ(p)
  else if p is a local name n in N
     then if c(k) = ∅ then fail
          else for some n ↦ q ∈ c(k) return(REF2(k, σ, c, q))
  else if p is of the form q's r
     then return(REF2(REF2(k, σ, c, q), σ, c, r))
```

Figure 1: Procedure REF2

mulas are formed by closing off under conjunction, negation, and formulas of the form p says φ, where φ is a formula.

Abadi views $p \mapsto p'$ as meaning that p is "bound to" p'. He considers two possible interpretations of "bound to". The first is equality; however, he rejects this as being inappropriate. (In particular, it does not satisfy some of his axioms.) The second is that $p \mapsto p'$ means p' "speaks-for" p, in the sense discussed in [ABLP93, LABW92]. Roughly speaking, this says that any message certified by p' should be viewed as also having been certified by p. While the "speaking-for" interpretation is the one favored by Abadi, he does not commit to it. Note that under Abadi's "speaking-for" interpretation, it makes sense to write $p \mapsto p'$ for arbitrary principal expressions p and p'. However, SDSI allows only local (simple) names to be bound to principal expressions. We shall make a similar restriction in our logic (and, indeed, under our semantic interpretation of binding, it would not make sense to allow an arbitrary principal expression to be bound to another one.)

The "speaks-for" interpretation intertwines issues of delegation with those of naming. As we suggested in the introduction, we believe these issues should be separated. We shall give $\mapsto$ a different interpretation that we believe is simpler and more in the spirit of binding. We believe that the "speaks-for" relation of [ABLP93, LABW92] should have quite different semantics than that of binding names to principals. (We hope to return to this issue in future work.)

Abadi interprets p says φ as "the principal denoted by p makes a statement that implies φ". In the case where p is a key (i.e., principal) k, this could mean that k signs a statement saying φ. Under our more restrictive interpretation, this is exactly how we interpret our analogue to says.

In any case, note that Abadi translates SDSI's local name n being bound to p as $n \mapsto p$ and captures k signing a certificate saying n is bound to p by the formula k says $n \mapsto p$. For future reference, it is worth noting that, in order to capture the binding of names to principals, no use is made of primitive propositions.

Abadi interprets formulas in his logic with respect to a tuple $(W; \psi; \sigma; \rho)$. The function σ maps global identifiers $(G \cup K)$ to subsets of W. The function ρ maps $N \times W$ to subsets of W.

Finally, π associates with each world (principal) k and primitive proposition p a truth value $\pi(p,k)$.

Abadi does not provide any intuition for his semantics, but suggests that W should be thought of as a set of possible worlds, as in modal logic. However, he also suggests [private communication, 1999] that his semantics was motivated by the work of Grove and Halpern [GH93], in which the corresponding set contains pairs consisting of a world and an agent. Some of Abadi's definitions make more intuitive sense if we think of W as a set of agents, while others make more sense if we think of W as a set of worlds. We elaborate on this point below.

Given $k \in W$ and $p \in P$, Abadi defines $\llbracket p \rrbracket_k$ inductively, as follows:

$$\llbracket g \rrbracket_k = \rho(g), \text{ for } g \in G \cup K$$

$$\llbracket n \rrbracket_k = \sigma(n,k) \text{ for } n \in N$$

$$\llbracket p_1\text{'s } p_2 \rrbracket_k = \bigcup\{\llbracket p_2 \rrbracket_{k'} : k' \in \llbracket p_1 \rrbracket_k\}$$

Here we have used a notation corresponding to the interpretation of the "worlds" in W as agents. Using this interpretation we may think of $\llbracket p \rrbracket_k$ as the set of principals bound to principal expression p according to k. The clause for $\llbracket p_1\text{'s } p_2 \rrbracket_k$ then says that if k' is one of the principals referred to by k as p_1, then k uses p_1's p_2 to refer to any principal referred to by k' as p_2.

Abadi also defines what it means for a formula φ to be true at world $k \in W$, written $k \models \varphi$, inductively, by

$k \models p$ iff $\pi(p,k) = \mathbf{true}$, if p is a primitive proposition

$k \models \varphi \wedge \psi$ iff $k \models \varphi$ and $k \models \psi$

$k \models \neg\varphi$ iff $k \not\models \varphi$

$k \models p \mapsto p'$ iff $\llbracket p \rrbracket_k \subseteq \llbracket p' \rrbracket_k$

$k \models p \text{ says } \varphi$ iff $k' \models \varphi$ for all $k' \in \llbracket p \rrbracket_k$.

These clauses defining $\models$ are quite intuitive if one interprets W to be a set of worlds and considers $\llbracket p \rrbracket_k$ to be the set of worlds consistent with what principal p has said at world k. In particular, under this interpretation, the clause for says can be read as stating that p says φ if φ holds in all worlds consistent with what p has said. The clause for $\mapsto$ also has quite a plausible reading under the "speaks-for" interpretation of this construct: it states that p' speaks for p if all worlds consistent with what p has said are consistent with what p' has said, i.e., p is constrained to speak consistently with what p' has said. However, it seems rather difficult to extend this intuitive reading to encompass the inductive definition of $\llbracket p \rrbracket_k$. In particular, it is far from clear to us what intuitive understanding to assign to the clause for $\llbracket p_1\text{'s } p_2 \rrbracket_k$ on this reading.

On the other hand, note that if we interpret the worlds as agents, then we can think of $k \models \varphi$ as saying that φ is true when local names are interpreted according to agent k. But this reading of the clauses, when combined with the intuitive reading of $\llbracket p \rrbracket_k$ as the set of principals that k refers to using p, also has its difficulties. Intuitively, when n is bound to p in principal

Reflexivity:	$p \mapsto p$
Transitivity:	$(p \mapsto q) \Rightarrow ((q \mapsto r) \Rightarrow (p \mapsto r))$
Left Monotonicity:	$(p \mapsto q) \Rightarrow ((p\text{'s } r) \mapsto (q\text{'s } r))$
Globality:	$(p\text{'s } g) \mapsto g$ if g is a global identifier
Associativity:	$((p\text{'s } q)\text{'s } r) \mapsto (p\text{'s } (q\text{'s } r))$
	$(p\text{'s } (q\text{'s } r)) \mapsto ((p\text{'s } q)\text{'s } r)$
Linking:	$(p \text{ says } (n \mapsto r)) \Rightarrow ((p\text{'s } n) \mapsto (p\text{'s } r))$ if n is a local name
Speaking-for:	$(p \mapsto q) \Rightarrow ((q \text{ says } \varphi) \Rightarrow p \text{ says } \varphi)$

Figure 2: Abadi's axioms for linked local name spaces

k's local name space, the principals that k refers to using p should be a subset of the principals that k refers to using n. Abadi interprets n being bound to p as $n \mapsto p$; this holds with respect to principal k when $[\![p]\!]_k$ is a superset of $[\![n]\!]_k$. This is precisely the opposite of what we would expect. Thus, neither the interpretation of W as a set of worlds nor the interpretation of W as a set of agents gives a fully satisfactory justification for Abadi's semantics. As we shall see, in our semantics, the interpretation of a principal expression p according to an agent will be a set of agents, but we use the reverse of Abadi's containment to represent binding.

Abadi provides an axiom system for his logic, which has three components:

1. The standard axioms and rules of propositional logic.

2. The standard axiom and rule for modal logic for the says operator:

$$(p \text{ says } (\varphi \Rightarrow \psi)) \Rightarrow ((p \text{ says } \varphi) \Rightarrow (p \text{ says } \psi))$$

$$\frac{\varphi}{p \text{ says } \varphi}$$

3. New axioms dealing with linked local name spaces, shown in Figure 2.

He shows that this axiomatization is sound, but conjectures it is not complete.

2.3 Name Resolution in Abadi's Logic

Abadi proves a number of interesting results relating his logic to SDSI. First, he shows that in a precise sense his logic can simulate REF2. He provides a collection of name-resolution rules NR and proves the following results:[6]

Proposition 2.1: Given a collection of c of bindings corresponding to signed certificates and a set of bindings of global names to keys, let E be the conjunction of the formulas k says $n \mapsto q$

[6] The results stated here are a variant of those stated in Abadi's paper, since our version of REF2 differs slightly from his. Nevertheless, the proofs of the results are essentially identical.

for each certificate $n \mapsto q \in c(k)$ and the formulas $g \mapsto k$ for each $k \in$ (g). Then $E \Rightarrow ((k\text{'s } p) \mapsto k_1)$ is provable with the name resolution rules NR if and only if REF2$(k; \; ; c; p)$ yields k_1.

Proposition 2.2: The name resolution rules are sound with respect to the logic. That is, given E as in Proposition 2.1 and any principal expression p, if $E \Rightarrow (p \mapsto k)$ is provable using NR then $E \Rightarrow (p \mapsto k)$ is also provable in the logic.

These results show that any bindings of names to principals that can be deduced using REF2 can also be deduced using Abadi's logic. However, Abadi shows that his logic is actually more powerful than REF2, by giving two examples of conclusions that can be deduced from his logic but not using REF2:

Example 2.3: Using the Globality, Associativity, and Transitivity axioms, if k and k' are keys, we immediately get k's (Lampson's k') $\mapsto k'$. This result does not follow from the REF2 algorithm. That is, REF2$(k; \; ; c;$ Lampson's $k')$ does not necessarily yield k' for arbitrary c and (in particular, it will not do so if Lampson is not bound to anything in c). ∎

Example 2.4: Suppose c consists of the four certificates that correspond to the following formulas: k says (Lampson $\mapsto k_1$), k says (Lampson $\mapsto k_2$), k_1 says (Ron $\mapsto$ Rivest), and k_2 says (Rivest $\mapsto k_3$) (where k, k_1, k_2, and k_3 are keys). Using the Speaking-for axiom, it is not hard to show that we can conclude that k's (Lampson's Ron) $\mapsto k_3$. It is easy to show that REF2 cannot reach this conclusion; that is, REF2$(k; \; ; c;$ Lampson's Ron) does not yield k_3 for any .[7] ∎

In reference to Example 2.3, Abadi [Aba98] says that "it is not clear whether [these conclusions] are harmful, and they might in fact be useful". In general, he views it as a feature of his logic that it allows reasoning about names without knowing their bindings [private communication, 1999]. While we agree that, in general, reasoning about names without knowing their bindings is a powerful feature, we believe it is important to make clear exactly which conclusions are desirable and which are not. This is what a good semantics can provide. Under our semantics, neither of these two conclusions are valid. In fact, our logic draws precisely the same conclusions as REF2. Of course, the conclusions of Examples 2.3 and 2.4 are valid under Abadi's semantics but, as we observed earlier, Abadi's semantics is not really meant to be used as a guide to which conclusions are acceptable (and, indeed, as we shall see, it validates a number of conclusions that do not seem so acceptable).

Abadi also considers the effect of extending his axiom system. In particular, he considers adding the following two axioms:

the converse of Globality: $g \mapsto (p\text{'s } g)$

[7] SPKI certificates and SDSI certificates have a slightly different syntactic form. A SPKI certificate issued by k to bind n to p could be expressed in the logic as k says (k's $n \mapsto p$). Abadi has remarked [private communication 1999], that if we rewrite the example using assertions in this form, the corresponding conclusion of this example would not follow in his logic. We have followed the SDSI format for certificates in this paper, but note that after some minor changes to the definitions, all the results in Sections 3–5 would still apply to SPKI certificates.

- a generalization of Linking: $(p \text{ says } (p_1 \mapsto p_2)) \Rightarrow (p\text{'s } p_1 \mapsto p\text{'s } p_2)$, for an arbitrary principal p_1 (instead of a local name).

The generalization of Linking is in fact sound under Abadi's semantics. The converse of Globality is not, but only because we may have $[\![p]\!]_k = \emptyset$. Note that $[\![p]\!]_k = \emptyset$ iff $k \models p$ says false; thus, the following variant of the converse of Globability is sound under Abadi's semantics: $\neg(p \text{ says false}) \Rightarrow (g \mapsto (p\text{'s } g))$.

This is quite relevant to our purposes because Abadi shows that if we added the two axioms above to his system, then from k says (DNS!! $\mapsto k$), we can conclude DNS!! $\mapsto k$. Thus, just from k saying that DNS!! is bound to k, it follows that DNS!! is indeed bound to k. This is particularly disconcerting under Abadi's "speaks-for" interpretation, where DNS!! $\mapsto k$ becomes "k speaks for DNS!!". We certainly do not want an arbitrary principal to speak for the name server!

Abadi proves a result showing that such conclusions are not derivable from hypotheses of a certain type in his logic (which does not have these two axioms).

Proposition 2.5: [Aba98] Let k and k' be distinct global names; let φ be a formula of the form $(k' \text{ says } (n_1 \mapsto p_1)) \wedge \ldots \wedge (k' \text{ says } (n_k \mapsto p_k))$, where $n_1, \ldots, n_k$ are local names and $p_1, \ldots, p_k$ are principal expressions; let ψ be a formula of the form $(k \text{ says } \psi_1) \wedge \ldots \wedge (k \text{ says } \psi_m)$, where $\psi_1, \ldots, \psi_m$ are arbitrary formulas. Then $\varphi \wedge \psi \Rightarrow (k' \mapsto k)$ is not valid.[8]

While Proposition 2.5 provides some assurance that undesirable formulas are not derivable in the logic, it does not provide much. Indeed, if we allow the φ to include the formula $\neg(k' \text{ says false})$, then the result no longer holds. In fact, it follows from our earlier discussion that the formula

$$(k \text{ says } (\text{DNS!!} \mapsto k)) \wedge \neg(k \text{ says false}) \Rightarrow (\text{DNS!!} \mapsto k)$$

is valid. Moreover, it does not seem so unreasonable to allow conjuncts such as $\neg(k$ says false) as part of φ. We certainly want to be able to use the logic to be able to say that if a principal's statements are not blatantly inconsistent, then certain conclusions follow.

3 The Logic of Local Name Containment

In this section we propose the Logic of Local Name Containment (henceforth LLNC) as an alternative to Abadi's logic. LLNC interprets local names as sets of principals and interprets SDSI certificates as stating containment relationships between these sets. We define the syntax in Section 3.1. In Section 3.2 we describe two distinct semantics for the logic. Section 3.3 presents a complete axiomatization.

3.1 Syntax

LLNC has syntactic elements that are closely related to the syntactic elements of Abadi's logic. However, our notation differs slightly from Abadi's to help emphasize some of the differences in intuition.

[8] Abadi's result actually says "$\varphi \wedge \psi \Rightarrow (k' \mapsto k)$ is not derivable"; since his axiomatization is sound, but not necessarily complete, the claim that it is not valid is stronger, and that is what Abadi's proof shows.

Again, we start with keys K, global names G, and local names N, and form principal expressions from them. The formulas of our language are formed as follows:

- If p and q are principal expressions then $p \mapsto q$ is a formula.
- If $k \in K$ and φ is a formula then k cert φ is a formula.[9]
- If φ_1 and φ_2 are formulas, then so are: $\neg\varphi_1$ and $\varphi_1 \wedge \varphi_2$. As usual, $\varphi_1 \vee \varphi_2$ is an abbreviation for $\neg(\neg\varphi_1 \wedge \neg\varphi_2)$ and $\varphi_1 \Rightarrow \varphi_2$ is an abbreviation for $\neg\varphi_1 \vee \varphi_2$.

We write $\mathcal{L}$ for the set of all formulas. (For simplicity, we omit primitive propositions, although we could easily add them. They play no role in Abadi's account of SDSI names, nor will they in ours.)

We read the expression $p \mapsto q$ as "p contains q"; we intend for it to capture the fact that all the keys bound to q are also bound to p. However, our intuitions about the meaning of $p \mapsto q$ are quite different from Abadi's. In particular, we do not wish to interpret $p \mapsto q$ as "q speaks for p." We consider the "speaks for" relation as being about rights and delegation, which requires a more sophisticated semantics than we wish to consider here. (See [HvdMS99] for a logic for reasoning about rights and delegation.) The expression $p \mapsto q$ should be understood as simply asserting a containment relationship between the denotations of principal expressions p and q; this is exactly what our semantics will enforce.

We read the expression k cert φ as "k has certified that φ." This corresponds roughly to Abadi's k says φ. There are two significant differences, however. For one thing, we do not allow arbitrary principal expressions on the left-hand side; only keys may certify a formula φ. For another, our interpretation of cert is more restrictive than Abadi's says, in that cert is treated quite syntactically; it refers to an actual certificate issued by a principal, while says considers logical consequences of such certificates. As a consequence, whereas says satisfies standard properties of modal operators (e.g., closure under logical consequence), cert does not.

3.2 Semantics

Our semantics is designed to model the SDSI principle that principals bind names in their local name space to values by issuing certificates. The interpretation of a local name depends on the principal and the certificates that have been issued. As the principal may rely on others for its interpretation of local names, the certificates issued by other principals also play a role. The interpretation of global names and keys will be independent of both the principal and the certificates that have been issued.

A world is a pair $w = (\sigma, c)$, where $\sigma : G \to \mathcal{P}(K)$ and $c : K \to \mathcal{P}(\mathcal{L})$ (where $\mathcal{P}(X)$ denotes the set of subsets of X) and $\cup_{k \in K} c(k)$ is finite. Intuitively, the function σ interprets global (or fixed) names as sets of keys. The intended interpretation of the function c is that it associates

[9] For our account of SDSI naming, it would suffice to restrict this clause to formulas of the form k cert $n \mapsto p$ where $n \in N$ and $p \in P$: our semantics will treat more general certificates as irrelevant to the meaning of principal expressions. We allow the more general form for purposes of discussion and because we envisage generalizations of the logic in which other types of certificates will be required.

with every key k the set of formulas $c(k)$ that have been certified using this key. That is, if $\varphi \in c(k)$ then, intuitively, a certificate asserting φ has been signed using k.[10]

Formulas of the logic will be interpreted in a world with respect to a key. Intuitively, this key indicates the principal from whose perspective we interpret principal expressions.

To interpret local names, we introduce an additional semantic construct. A *local name assignment* will be a function $l : K \times N \to \mathcal{P}(K)$ associating each key and local name with a set of keys. Intuitively, $l(k, n)$ is the set of keys represented by principal k's local name n. We write LNA for the set of all local name assignments.

Given a world $w = (\sigma, c)$, a local name assignment l, and a key k, we may assign to each principal expression p an interpretation $[\![p]\!]_{w,l,k}$, a set of keys. The definition is much like that of Abadi's $[\![p]\!]_k$:

$[\![k']\!]_{w,l,k} = \{k'\}$, if $k' \in K$ is a key,

$[\![g]\!]_{w,l,k} = \sigma(g)$, if $g \in G$ is a global name,

$[\![n]\!]_{w,l,k} = l(k, n)$, if $n \in N$ is a local name,

$[\![p\text{'s } q]\!]_{w,l,k} = \bigcup \{[\![q]\!]_{w,l,k'} \mid k' \in [\![p]\!]_{w,l,k}\}$, for principal expressions $p, q \in P$.

Our intuitions for $[\![p]\!]_{w,l,k}$ are essentially the same as for the "agent-based" reading of Abadi's logic, discussed above. That is, $[\![p]\!]_{w,l,k}$ is the set of keys associated with the expression p in k's local name space, when local names are interpreted according to l. With respect to principal k, the expression p's q denotes the set of principals that principals referred to by k as p refer to as q.

We now define what it means for a formula φ to be true at a world $w = (\sigma, c)$ with respect to a local name assignment l and key k, written $w, l, k \models \varphi$, by induction on the structure of φ.[11]

$w, l, k \models p \mapsto q$ if $[\![p]\!]_{w,l,k} \supseteq [\![q]\!]_{w,l,k}$

$w, l, k \models k' \text{ cert } \varphi$ if $\varphi \in c(k')$

$w, l, k \models \neg\varphi_1$ if not $w, l, k \models \varphi_1$

$w, l, k \models \varphi_1 \wedge \varphi_2$ if $w, l, k \models \varphi_1$ and $w, l, k \models \varphi_2$.

Note that the semantics of cert reinforces its syntactic nature. To determine if k' cert φ is true at (w, l, k), we check whether a certificate has been issued in world w by k' certifying φ. Moreover, as we shall see, while we allow any formula to be certified by k, the only formulas whose certification has a nontrivial semantic impact are those of the form $n \mapsto p$, where n is a local name. We return to this issue below.

[10] We make the simplifying assumption that certificates do not have expiration dates. It is not difficult to extend the logic to take into account certificate expiration; see [HvdM99]. The assumption that $\bigcup_{k \in K} c(k)$ is finite is meant to enforce the intuition that only finitely many certificates are issued. None of our later results depend on this assumption, but it seems reasonable given the intended application of the logic.

[11] Note that our semantics is thus in the spirit of that of Grove and Halpern [GH93], in that the truth of a formula depends on both an agent and some features of the world (captured by w and l).

We do not consider all pairs w, l as being appropriate on the left-hand side of $\models$. If $w = (\sigma, c)$, we expect the local name assignment l to respect the certificates that have been issued in c. That is, if $c(k)$ includes the binding $n \mapsto p$, we would expect that $l(k, n)$ would include all the keys bound to p in k's name space. The question is whether there can be other keys bound to n in k's name space beyond those forced by the certificates. How we answer this question depends on our intuitions for c. For example, we could view c as the set of certificates received by one of the principals. This would be particularly appropriate if we wanted to reason about the knowledge and belief of the agents, an extension we plan to explore in future work. With this viewpoint, we could view l as consisting of all the bindings, including ones that the principal does not know about. Thus, l would at least have all the bindings forced by c, but perhaps others as well. Alternatively, we could view c as consisting of all the certificates that have been issued. In this case, we would want l to be in some sense minimal, and have no bindings beyond those forced by the certificates in c. We now present two different semantics, which reflect each of these two intuitions. We then show that, as far as validity is concerned, the semantics are equivalent; that is, they have the same proof theory.

A local name assignment l is consistent with a world $w = (\sigma, c)$ if, for all keys k, local names n, and principal expressions p, if the formula $n \mapsto p$ is in $c(k)$, then $w, l, k \models n \mapsto p$. Intuitively, assignments that are not consistent with a world provide an inappropriate basis for the interpretation of local names, since the certificates issued by principals are not necessarily reflected in their local bindings. We obtain our first semantics, called the open semantics, by restricting to consistent local name assignments. We write $w, l, k \models_o \varphi$ if $w, l, k \models \varphi$ and l is consistent with w. The formula φ is o-satisfiable if there exists a triple w, l, k such that $w, l, k \models_o \varphi$ and φ is o-valid, denoted $\models_o \varphi$, if there does not exist a triple w, l, k such that $w, l, k \models_o \neg\varphi$.

Although our syntax allows k to certify arbitrary formulas, it is easy to see that, according to the semantics just introduced (as well as the one we are about to introduce), only the certification of formulas of the form $n \mapsto p$ has any impact on consistency; all other formulas certified by k are ignored. There is a good reason for this restriction. We are implicitly assuming that when k' certifies $n \mapsto p$, that very act causes all the keys bound to p to also be bound to n in k's name space. Thus, if $n \mapsto p \in c(k)$, then we want $n \mapsto p$ to be true in (w, l, k). But if k certifies a formula like k_1's $n \mapsto k_3$ where $k_1 \neq k$, then we cannot conclude that this formula is true in (w, l, k) unless we are prepared to make additional assumptions about k's truthfulness. We feel that if such assumptions are to be made, then they should be modeled explicitly in the logic, not hidden in the semantics.

It does seem reasonable to extend the notion of l being consistent with w to require that if k certifies a formula φ which is a Boolean combination of formulas of the form $n \mapsto p$ then $(w, l, k) \models \varphi$. However, once we allow more general Boolean combinations (in particular, once we allow disjunctions), there will be problems making sense out of the intuition of our next semantics, that there are "no bindings beyond those forced by the certificates in c". We consider this issue next.

According to the open semantics, it is possible for a local name n of principal k_1 to be bound to a key k_2 even when no certificate concerning n has been issued. Arguably, this is not in accordance with the intentions of SDSI. To better capture these intentions, we define a second semantics, that restricts the name bindings to those forced by the certificates issued.

To do so, we first establish that the open semantics satisfies a kind of "minimal model" result. Define the ordering $\sqsubseteq$ on the space LNA of local name assignments by $l_1 \sqsubseteq l_2$ if $l_1(k;n) \subseteq l_2(k;n)$ for all $k \in K$ and $n \in N$. It is readily seen that LNA is given the structure of a complete lattice [Bir67] by this relation. Say that a local name assignment l is minimal in a set of local name assignments L if $l \in L$ and $l \sqsubseteq l'$ for all $l' \in L$.

Theorem 3.1: Given a world w, there exists a unique local name assignment l_w minimal in the set of all local name assignments consistent with w. Moreover, if p is a principal expression and k_1 and k_2 are keys, then $w, l_w, k_1 \models_o p \mapsto k_2$ iff, for all local name assignments l consistent with w, we have $w, l, k_1 \models_o p \mapsto k_2$.

The proof of this result (which, like that of all the technical results in this paper, is deferred to the appendix) uses standard techniques from the theory of fixed points.

We now define our second semantics, called the closed semantics. It attempts to capture the intuition that the only bindings in l should be those required by the certificates in c, using the minimal assignment promised by Theorem 3.1. We write $w, k \models_c \varphi$ if $w, l_w, k \models \varphi$. We say that φ is c-satisfiable if there exists a world w and key k such that $w, k \models_c \varphi$ and that φ is c-valid, denoted $\models_c \varphi$, if $w, k \models_c \varphi$ for all worlds w and principals k. Note that by Theorem 3.1, the assignment l_w is consistent with w, so c-satisfiability implies o-satisfiability. Thus, if $\models_o \varphi$ then $\models_c \varphi$. As we shall soon see (Theorem 3.5), somewhat surprisingly, the converse holds as well.

3.3 A Complete Axiomatization

We start this section by presenting a sound and complete axiomatization for LLNC with respect to the open semantics. We then prove that the open and closed semantics are characterized by the same valid formulas, so that the axiomatization is also sound and complete with respect to the closed semantics.

The axiomatization depends in part on whether the set K of keys is finite or infinite. Figure 3 describes the axiom system AX_{inf} for the case where K is infinite.

It is interesting to compare the axioms in AX_{inf} to Abadi's axioms. Although we interpret $\mapsto$ as superset and he interprets it as subset, Reflexivity, Transitivity, Left-Monotonicity, and Associativity, hold in both cases, for essentially the same reasons. The switch from subset to superset means that the Converse of Globality holds in our case. Globality does not hold in general because the denotation of p's g may be empty if the denotation of p is empty (as we observed, this is also why the Converse of Globality does not hold in general for Abadi). In fact, for our logic, p's $g \mapsto g$ holds whenever the interpretation of p is nonempty. We use p's $k \mapsto k$ as a canonical way of denoting that the interpretation of p is nonempty. This explains the form of the Globality axiom. Since the interpretation of a key is always nonempty, we also get Key Globality.

Key Linking is our analogue of Abadi's Linking axiom. Of course, we use cert whereas Abadi uses says; in addition, only keys can certify formulas for us. While this axiom shows that there are some similarities between cert and says, there are some significant differences. We have no analogue of Abadi's Speaking-for axiom and, unlike says, cert does not satisfy the standard axiom and rule of modal logic: $(k \text{ cert } (\varphi \Rightarrow \psi)) \wedge (k \text{ cert } \varphi)$ does not imply k cert

Propositional Logic:	All instances of propositional tautologies
Reflexivity:	$p \mapsto p$
Transitivity:	$(p \mapsto q) \Rightarrow ((q \mapsto r) \Rightarrow (p \mapsto r))$
Left Monotonicity:	$(p \mapsto q) \Rightarrow ((p\text{'s } r) \mapsto (q\text{'s } r))$
Associativity:	$((p\text{'s } q)\text{'s } r) \mapsto (p\text{'s } (q\text{'s } r))$
	$(p\text{'s } (q\text{'s } r)) \mapsto ((p\text{'s } q)\text{'s } r)$
Key Globality:	$(k\text{'s } g) \mapsto g$ if $k \in K$ and $g \in G \cup K$
Globality:	$(p\text{'s } k \mapsto k) \Rightarrow (p\text{'s } g \mapsto g)$ if $k \in K$; $g \in G \cup K$
Converse of Globality:	$g \mapsto (p\text{'s } g)$ if $g \in K \cup G$
Key Linking:	$(k \text{ cert } (n \mapsto r)) \Rightarrow ((k\text{'s } n) \mapsto (k\text{'s } r))$ if n is a local name
Nonemptiness:	(a) $p \mapsto k_1 \Rightarrow p\text{'s } k \mapsto k$
	(b) $\neg(p \mapsto q) \Rightarrow q\text{'s } k \mapsto k$
	(c) $p\text{'s } q \mapsto k_1 \Rightarrow p\text{'s } k \mapsto k$
	(d) $(p\text{'s } k \mapsto k \wedge k' \mapsto p) \Rightarrow (p \mapsto k')$
Key Distinctness:	$\neg(k_1 \mapsto k_2)$ if k_1 and k_2 are distinct keys
Modus Ponens:	From φ and $\varphi \Rightarrow \psi$ infer ψ.

Figure 3: The axiom system AX_{inf}

and k cert φ is not valid even if φ is valid. Interestingly, Abadi does not use these properties of Speaking-for in proving that his name resolution rules NR, used to capture REF2, are sound. As a result, (with very minor changes) we can show that the name resolution rules are also sound for LLNC, and hence we can prove analogues of Propositions 2.1 and 2.2. However, we can actually prove a much stronger result: whereas Abadi's logic is able to draw conclusions about bindings that do not follow from REF2, LLNC captures REF2 exactly (see Theorem 4.1).

AX_{inf} has two axioms that do not appear in Abadi's axiomatization: Key Distinctness and Nonemptiness. Key Distinctness just captures the fact that we interpret keys as themselves. The first three parts of Nonemptiness capture various ways that an expression can be seen to be nonempty. For example, part (a) says that if p is bound to (i.e., is a superset of) a key, then its interpretation must be nonempty and part (b) says that if p is not a superset of q, then q must be nonempty. Part (d) of Nonemptiness says that if p is nonempty and k' is bound to p, then p is bound to k', i.e., p and k' have exactly the same interpretation.

If K is finite we need to add two further axioms to AX_{inf}. Let AX_{fin} consist of all the axioms and rule in AX_{inf} together with:

Witnesses:	$\neg(p \mapsto q) \Rightarrow \vee_{k \in K} (\neg(p \mapsto k) \wedge (q \mapsto k))$
	$(p\text{'s } q) \mapsto k_1 \Rightarrow \vee_{k \in K} ((p \mapsto k) \wedge (k\text{'s } q \mapsto k_1))$
Current Principal:	$\vee_{k \in K} (n_k \mapsto l_k \Leftrightarrow k\text{'s } n_k \mapsto l_k)$
	where $n_k \in N$ and $l_k \in K$ for each $k \in K$.

The two axioms that make up Witnesses essentially capture our interpretation of $\mapsto$ as containment. They tell us that facts about containment of principal expressions can be reduced to facts about keys. For example, the first one says that if p does not contain q, then there is a key bound to q that is not bound to p. Current Principal captures the fact that some key in K must be the current principal; if k is the current principal, then for all local names n and keys k', $n \mapsto k' \Leftrightarrow k$'s $n \mapsto k'$ holds. (This is actually true not just for local names, but for all principal expressions; it suffices to state the axiom just for local names.)

While the properties captured by these two axioms continue to hold even if K is infinite, they can no longer be expressed in the logic, since we cannot take a disjunction over all the elements in K. Interestingly, we can drop Nonemptiness and Globality as axioms in AX_n. These properties already follow from the other properties in the presence of Witnesses.

As the following result shows, these axiom systems completely characterize validity in the logic with respect to the open semantics.

Theorem 3.2: AX_{inf} (resp., AX_n) is a sound and complete axiomatization of LLNC with respect to the open semantics if K is infinite (resp., K is finite).

In the course of proving Theorem 3.2, we also prove a "finite model" result, which we cull out here. Let $|\varphi|$, the length of φ, be the total number of symbols appearing in φ. This result holds both when K is finite and when K is infinite.

Proposition 3.3: Let K_φ be the keys that appear in φ and let $C_\varphi(k)$ consist of all bindings $n \mapsto p$ such that k cert $n \mapsto p$ is a subformula of φ. If φ is satisfiable with respect to the open semantics, then for all sets K' of keys such that $K_\varphi \subseteq K'$ and $|K'| \le \min(|K|, 2|\varphi|^2)$, there is a world $w = (\pi, c)$, local name assignment l, and principal $k \in K'$ such that $w, l, k \models_o \varphi$ and (a) $l(k', n) \subseteq K'$ for all $k' \in K$ and $n \in N$, (b) $l(k', n) = \emptyset$ if $k' \notin K'$, (c) $\pi(g) \subseteq K'$ for all $g \in G$, (d) $\pi(g) = \emptyset$ if g does not occur in φ, and (e) $c(k) \subseteq C_\varphi(k)$ for all keys k.

Corollary 3.4: The problem of deciding if a formula $\varphi \in$ LLNC is satisfiable with respect to the open semantics is NP-complete (whether K is finite or infinite).

Proof: The lower bound is immediate from the fact that we can trivially embed satisfiability for propositional logic into satisfiability for LLNC. For the upper bound, given φ, choose K' such that $|K'| = \min(|K|, 2|\varphi|^2)$ and $K' \supseteq K_\varphi$. Then guess w, l, k as in Proposition 3.3 and check whether $w, l, k \models_o \varphi$. Proposition 3.3 says that the guess is only polynomial in $|\varphi|$; it is clear that checking whether $w, l, k \models_o \varphi$ can also be done in time polynomial in φ. Note that for $|\varphi| \ll |K|$ (which is likely to include all cases of practical interest, given that K will typically be a very large set), the polynomial does not depend on $|K|$. ∎

As we suggested earlier, the closed semantics and the open semantics are characterized by exactly the same axioms.

Theorem 3.5: The same formulas are c-valid and o-valid; i.e., for all formulas φ, we have $\models_o \varphi$ iff $\models_c \varphi$.

We remark that this result is sensitive to the language under consideration. It may no longer hold if we move to a more expressive language.

Corollary 3.6: AX_{inf} (resp., AX_{fin}) is a sound and complete axiomatization of LLNC with respect to the closed semantics when K is infinite (rep., finite).

Corollary 3.7: The problem of deciding if a formula $\varphi \in$ LLNC is satisfiable with respect to the closed semantics is NP-complete (whether K is finite or infinite).

Let us now return to the contentious axioms discussed by Abadi. Converse of Globality is valid in LLNC, as we observed earlier. The generalization of Linking considered by Abadi, restricted to be syntactically well formed, amounts to

$$(k \text{ cert } (p_1 \mapsto p_2)) \Rightarrow (k\text{'s } p_1 \mapsto k\text{'s } p_2).$$

In general, this is not valid, since our semantics ignores certificates stating $p_1 \mapsto p_2$ when p_1 is not a local name. Thus, we avoid the "unreasonable" conclusions that can be drawn from these axioms. In particular, it does not follow in our logic that $(k \text{ cert } (DNS!! \mapsto k)) \Rightarrow DNS!! \mapsto k$. However, the reason it does not follow in LLNC is quite different from the reason it does not follow in Abadi's logic: since DNS!! is a global name, a certificate such as $k \text{ cert } (DNS!! \mapsto k)$ has no impact on the interpretation of global names. This captures the intuition that k should not be trusted when making assertions about bindings not under its control. If we were willing to trust k on everything, then concluding that k is bound to DNS!! after k certifies that it is would not seem so unreasonable.

The following formula is also not valid in LLNC:

$$(\neg(k \text{ cert false}) \wedge (k \text{ cert } (DNS!! \mapsto k))) \Rightarrow DNS!! \mapsto k.$$

(This formula corresponds to the one that we noted earlier is valid in Abadi's logic.) Failure to issue a certificate stating false has no more impact on global names than does any other behavior of k. Nor would a precondition asserting that the interpretation of k is non-empty validate the formula, since this is true in every world. We can in fact prove the following generalization of Abadi's Proposition 2.5, which provides a stronger statement of the safety of our logic than Abadi's result.

Proposition 3.8: Let φ be any c-satisfiable boolean combination of formulas of the form $k \text{ cert } \psi$, and let ψ' be any boolean combination of formulas of the form $p \mapsto q$ where neither p nor q contains a local name. Then $\models_c \varphi \Rightarrow \psi'$ iff $\models_c \psi'$.

Informally, Proposition 3.8 says that facts about global names are completely independent of facts about certificates; issuing certificates can have no impact on the global name assignment. As we observed earlier, the analogous result does not hold for Abadi's logic.

4 Name Resolution in LLNC

In this section, we show that LLNC captures REF2 exactly. Indeed, we show that it does so for several distinct semantic interpretations. Define the order $\le$ on worlds by $(\pi', c') \le (\pi, c)$ if

1. $\pi'(g) \subseteq \pi(g)$ for all global names g, and

2. $c'(k) \supseteq c(k)$ for all keys k.

That is, $w' \sqsupseteq w$ when w' contains more certificates than w and the bindings to global names in w are a subset of those in w'. If E is a set of formulas and φ is a formula, we write $E \models_o \varphi$ if for all worlds w, local name assigments l consistent with w and all keys k, if $w, l, k \models_o \psi$ for all ψ in E then $w, l, k \models_o \varphi$. Similarly, $E \models_o \varphi$ if for all worlds w and all keys k, if $w, k \models_c \psi$ for all ψ in E then $w, k \models_c \varphi$.

Theorem 4.1: Suppose k_1, k_2 are principals, $w = (\sigma, c)$ is a world, and p is a principal expression. Let E_w be the set of all formulas $g \mapsto k$ for all global names g and keys $k \in \sigma(g)$ and the formulas k cert φ for all keys k and formulas $\varphi \in c(k)$. The following are equivalent:

1. $k_1 \in REF2(k_2, \sigma, c, p)$,
2. $w, k_2 \models_c p \mapsto k_1$,
3. $w', k_2 \models_c p \mapsto k_1$ for all worlds $w' \sqsupseteq w$,
4. $E_w \models_c k_2$'s $p \mapsto k_1$,
5. $E_w \models_o k_2$'s $p \mapsto k_1$.

This theorem gives a number of perspectives on name resolution in LLNC. The equivalence between (1) and (2) in this theorem tells us that REF2 is sound and complete with respect to key binding, according to the semantics of LLNC. That is, $REF2(k, \sigma, c, p)$ yields k' iff $p \mapsto k'$ is forced to be true by the bindings of global names in σ and the certificates in c. Thus, viewed as a specification of the meaning of SDSI names, the closed semantics and REF2 are equivalent.

Informally, we have viewed REF2 as a procedure that is run by an omniscient agent with complete information about the interpretation of global names and the certificates that have been issued. It is also possible to understand REF2 as performing a computation based on the limited information available to a particular principal. Suppose that the world w expresses the limited information this principal has about the binding of global names and the certificates that have been issued. Suppose that w' describes the actual bindings of global names and the certificates that have been issued. Assuming that all of the principal's information is correct, then $w \sqsubseteq w'$. Thus, the set of $w' \sqsupseteq w$ is the set of all worlds w' that are consistent with the information available to the principal. (We could formalize this using the Kripke semantics for the logic of knowledge in a distributed system [HM90].) The equivalence between (2) and (3) essentially shows that it doesn't matter whether we view the principal as having total or partial information.

The implication from (1) to (4) in Theorem 4.1 is analogous to Abadi's soundness result, Proposition 2.2. Of course, the converse implication gives us completeness, which, as Abadi himself observed, does not hold for Abadi's logic (since it validates conclusions that do not follow from REF2). Interestingly, although, as we have seen, there are significant differences between LLNC and Abadi's logic, an examination of Abadi's soundness proof reveals that it does not use the Speaking-for rule, the unrestricted form of Globality, or the standard axiom and rule for the modal operator says, which are the main points of difference with our logic.

This observation says that the proof of the implication from (1) to (4) is essentially the same for LLNC and for Abadi's logic.

It is instructive to understand why the formulas considered in Examples 2.3 and 2.4, which give conclusions in Abadi's logic beyond those derivable by REF2, are not valid in LLNC. It is easy to see why the formula k's (Lampson's k^0) $\mapsto k^0$ from Example 2.3 (which, by Associativity and Transitivity, is equivalent to (k's Lampson)'s $k^0 \mapsto k^0$) is not valid in LLNC. This is simply because the antecedent of (our version of) Globality does not always hold. Now consider the formula in Example 2.4. The proof that this is valid in Abadi's logic uses the Speaking-for axiom, which does not hold for us (if we replace says by cert). To see that it is not valid in LLNC, consider a world $w = (\sigma, c)$ containing only the certificates forced by the formulas (i.e., $c(k) = \{\text{Lampson} \mapsto k_1, \text{Lampson} \mapsto k_2\}$, $c(k_1) = \{\text{Ron} \mapsto \text{Rivest}\}$, $c(k_2) = \{\text{Rivest} \mapsto k_3\}$). Then it is easy to see that $w, k \not\models k$'s (Lampson's Ron) $\mapsto k_3$, since $[\![k\text{'s (Lampson's Ron)}]\!]_{w,l_w,k} = \emptyset$ whereas $[\![k_3]\!]_{w,l_w,k} = \{k_3\}$.

5 Logic Programming Implementations of Name Resolution Queries

The reader familiar with the theory of logic programming may have noted a close resemblance of the results and constructions of the preceding sections to the (now standard) fixpoint semantics for logic programs developed originally by van Emden and Kowalski [EK76]. Indeed, it is possible to translate our semantics into the framework of logic programming. In fact, we provide a translation that does not require the use of function symbols and thus produces a Datalog program, a restricted type of logic program that has significant computational advantages over unrestricted logic programs. Our translation allows us to take advantage of the significant body of research on the optimization of Datalog programs [Ull88, Ull89].

The idea is to translate queries to formulas in a first-order language over a vocabulary V which consists of a constant symbol for each element in $K \cup G \cup N$ and a ternary predicate symbol name. Intuitively, name(x, y, z) says that, in the local name space of key x, the basic principal expression (i.e., key, global name or local name) y is bound to key z.

Using name, for each principal expression p and pair of variables x, y, we define a first-order formula $\varphi_{x,y}(p)$ that, intuitively, corresponds to the assertion "$y \in [\![p]\!]_x$," by induction on the structure of p:

1. $\varphi_{x,y}(p) = \text{name}(x, p, y)$ when $p \in K \cup G \cup N$.

2. $\varphi_{x,y}(q\text{'s } r) = \exists z(\varphi_{x,z}(q) \wedge \varphi_{z,y}(r))$, where $z \neq x, y$.

Recall that a Herbrand structure over the vocabulary V is a first-order structure that has as its domain the set of constant symbols $K \cup G \cup N$ in V and interprets each constant sybol as itself. Such a structure may be represented as a set of tuples of the form name(x, y, z), where $x, y, z \in K \cup G \cup N$. The subset relation on such sets partially orders the Herbrand structures.

We say that a Herbrand structure M over V represents a world $w = (\sigma, c)$ and local name assignment l if, for all $x, y, z \in K \cup G \cup N$, we have name$(x, y, z) \in M$ iff either

1. $x, y, z \in K$ and $z = y$, or

2. $x \in K$, $y \in G$ and $z \in \ (y)$, or

3. $x \in K$, $y \in N$ and $z \in l(x;y)$.

Intuitively, M represents w and l if it encodes all the interpretations of basic principal expressions given by w and l. The following result, whose straightforward proof is left to the reader, shows that in this case M also captures the interpretation of all other principal expressions, and expresses the correctness of our translation of principal expressions.

Proposition 5.1: If M represents w and l then, for all principal expressions p and $x;y \in K \cup G \cup N$, we have $M \models \ _{x,y}(p)$ iff $x;y \in K$ and $w;l;x \models p \mapsto y$.

We now show how a logic program can be used to capture the relationsip between w and l_w. For each world $w = (\ ;c)$, we define a theory (set of sentences) $\ _w$ that characterizes w; $\ _w$ consists of the following sentences:

1. a sentence $\mathrm{name}(k_1;k_2;k_2)$, for each pair of keys $k_1;k_2 \in K$, and

2. the sentence $\mathrm{name}(k_1;g;k_2)$, for each pair of keys $k_1;k_2 \in K$ and global name $g \in G$ such that $k_2 \in \ (g)$,

3. the sentence $\forall y(\ _{k,y}(q) \Rightarrow \mathrm{name}(k;n;y))$, for each key k and binding $n \mapsto q$ in $c(k)$.

After some equivalence-preserving syntactic transformations (moving the existentials in the body of these sentences to the front), the theory $\ _w$ is a definite Horn theory, i.e., it consists of formulas of the form $\forall x(B \Rightarrow H)$, where B is a (possibly empty) conjunction of atoms (that is, formulas of the form $\mathrm{name}(x;y;z)$ or $y = z$) and H is an atom. Well-known results from the theory of logic programming show that such a theory has a Herbrand model M minimal with respect to the containment ordering on Herbrand structures. Moreover, this minimal Herbrand model captures the minimal name assignments for w.

Theorem 5.2: The minimal Herbrand model M_w of $\ _w$ represents w and l_w.

Using Proposition 5.1, we immediately obtain the following corollary.

Corollary 5.3: For all $x;y \in K \cup G \cup N$ and principal expressions p, we have $M_w \models \ _{x,y}(p)$ iff $x;y \in K$ and $w;x \models_c p \mapsto y$.

Because $\ _w$ is a definite Horn theory, it corresponds to a logic program. Moreover, for existential queries, i.e., queries that are sentences formed from atomic formulas using only conjunction, disjunction and existential quantification (but not negation), we have that entails iff $M \models$. This enables us to exploit logic programming technology to obtain efficient implementations of several types of queries, corresponding to different choices of bound and free variables in the predicate "name". We may even form complex queries not corresponding in any direct way to the capacities of the procedure REF2. Examples of this include the following:

1. the query $\mathrm{name}(k_1;n;k_2)$ returns "yes" if k_2 is bound to the local name n according to k_1;

2. the query $\mathit{name}(X, n, k)$ returns the set of keys X such that k is in n according to X;

3. the query $\mathit{name}(k_1, X, k_2)$ returns the set of global and local names X containing k_2 according to k_1.

4. the query $\mathit{name}(k_1, n, X) \wedge \mathit{name}(k_2, n, X)$ returns the set of keys X that k_1 and k_2 agree to be associated with local name n.

Many more possibilities clearly exist. These observations show the advantage of viewing name resolution in a logic programming framework.

6 Self

Abadi considers an extension of his logic obtained by adding a special basic principal expression Self, intended to represent SDSI's expression (ref:). (We remark that Self is essentially the same as I in the logic of naming considered in [GH93].) Intuitively, Self denotes the current principal. The semantics given to Self by Abadi extends the definition of the set of principals associated with a principal expression by taking $[\![\mathsf{Self}]\!]_a = \{a\}$ for each $a \in W$. This suffices to validate the following axiom.

$$
\begin{array}{lll}
\text{Identity:} & \mathsf{Self}\text{'s}\, p \mapsto p & p \mapsto \mathsf{Self}\text{'s}\, p \\
 & p\text{'s}\, \mathsf{Self} \mapsto p & p \mapsto p\text{'s}\, \mathsf{Self}
\end{array}
$$

These axioms very reasonably capture the intuitions that Self refers to the current principal.

However, not all consequences of this semantics for Self are so reasonable. For example, the following is valid under Abadi's semantics:

$$
\begin{array}{l}
(k_P \text{ says } US \mapsto \mathsf{Self}) \wedge (k_P \text{ says } US \mapsto k_{VP}) \\
\Rightarrow k_P \text{ says } ((US \text{ says } \mathit{false}) \vee (\mathsf{Self} \mapsto k_{VP}))
\end{array}
\tag{1}
$$

Interpreting k_P as the key of the president of the US and k_{VP} the key of the vice-president, this is clearly unreasonable. It should not follow from the fact the the president says that both he and the vice-president speak for the US that according to the president, either the US speaks nonsense or the vice president speaks for the president.

Abadi's suggested semantics for Self works much better in the context of the logic LLNC. Suppose we extend this logic to include Self, and like Abadi, define $[\![\mathsf{Self}]\!]_k = \{k\}$ for keys $k \in K$. This again validates the Identity axioms above. To get completeness, we just need to add one axiom in addition to Identity, which basically says that Self acts like a key (cf. Nonemptiness (d)):

$$
\text{Self-is-key} \quad \mathsf{Self} \mapsto p \wedge p\text{'s}\, k \mapsto k \Rightarrow p \mapsto \mathsf{Self}.
$$

Let AX^{self}_{inf} (resp., AX^{self}_{fin}) be the result of adding Identity and Self-is-key to AX_{inf} (resp., AX_{fin}). Let $LLNC^s$ be the language that results when we add Self to the syntax.

Theorem 6.1: AX^{self}_{inf} (resp., AX^{self}_{fin}) is a sound and complete axiomatization of $LLNC^s$ with respect to the open semantics if K is infinite (resp., K is finite).

Propositions 3.3 and Theorem 3.5 hold with essentially no change in proof for $LLNC^s$; it follows that AX_n^{self} (resp., AX_{inf}^{self}) is also complete with respect to the closed semantics and the satisfiability problem is NP-complete.

Interestingly, the proof of completeness shows that once we add Identity and Self-is-key to the axioms, we no longer need Current Principal as an axiom in the finite case. Here is a sketch of the argument: From Identity we get that Self's $k \mapsto k$ is provable for any key k. Now applying Witnesses, we get that $\vee_{k \in K}$ Self $\mapsto k$ is provable. Together with Self-is-key, this says that Self is one of the keys in K. Identity (together with Transitivity) tells us that for that key k that is Self, $n \mapsto k' \Leftrightarrow k$'s $n \mapsto k'$ holds, giving us Current Principal.

Note that with our semantics for Self, the counterintuitive conclusion (1) does not follow. From k_P cert US $\mapsto$ Self and k_P cert US $\mapsto k_{VP}$ it follows that $[\![US]\!]_{k_P} \subseteq \{k_P, k_{VP}\}$. Thus, we have neither $[\![US]\!]_{k_P} = \emptyset$ nor $\{k_{VP}\} \subseteq [\![US]\!]_{k_P}$, which would be required to get a conclusion similar to that drawn by Abadi's logic.

7 Conclusions

We have introduced a logic LLNC for reasoning about SDSI's local name spaces and have argued that it has some significant advantages over Abadi's logic. Among other things, it provides a complete characterization of SDSI's REF2, has an elegant complete axiomatization, and its connections with Logic Programming lead to efficient implementations of many queries of interest.

We believe that some of the dimensions in which Abadi's logic differs from SDSI warrant further investigation. For example, under some sensible interpretations, the conclusions reached by Abadi's logic in Example 2.4 are quite reasonable. One such interpretation is that while local names may be bound to more than one key, they are intended to denote a single individual. If k knows that k_1 and k_2 are two keys used by the one individual Lampson, and Lampson uses k_1 to certify that his local name Ron is bound to the name Rivest, and also uses his key k_2 to certify that his local name Rivest is bound to k_3, then it is very reasonable to conclude that k's Lampson's Ron is bound to k_3. Another interpretation supporting this conclusion would be that says aggregates the certificates issued using a number of distinct keys (possibly belonging to distinct individuals) much in the way that the notion of distributed knowledge [FHMV95] from the literature on reasoning about knowledge aggregates the knowledge of a collection of agents. We believe that our semantic framework, which, unlike Abadi's, makes the set of certificates issued explicit, provides an appropriate basis for the study of such issues.

Our semantic framework also lends itself to a number of generalizations, which we are currently exploring. These include reasoning about the beliefs of principals and reasoning about permission, authority, and delegation. We hope to report on this work shortly.

A Proofs

In this appendix, we prove all the technical results stated in the main text. For ease of exposition, we repeat the statements of the results here.

Theorem 3.1: Given a world w, there exists a unique local name assignment l_w minimal in the set of all local name assignments consistent with w. Moreover, if p is a principal expression and k_1 and k_2 are keys, then $w, l_w, k_1 \models_o p \mapsto k_2$ iff, for all local name assignments l consistent with w, we have $w, l, k_1 \models_o p \mapsto k_2$.

Proof: This result can be established using standard results from the theory of fixed points. Suppose $(X; \le)$ is a complete partial order. Denote the least upper bound of a set $Y \subseteq X$ by $\sqcup Y$. A mapping $T : X \to X$ is said to be monotonic if for all $x \le y$ in X we have $T(x) \le T(y)$. Such a mapping T is said to be continuous if for all infinite increasing sequences $x_0 \le x_1 \le \ldots$ in X we have $T(\sqcup\{x_i : i \in N\}) = \sqcup\{T(x_i) : i \in N\}$. Note that continuity implies monotonicity. To establish continuity of a monotonic mapping T, it suffices to show that $T(\sqcup\{x_i : i \in N\}) \le \sqcup\{T(x_i) : i \in N\}$, since the opposite containment is immediate from monotonicity.

For a fixed expression p, world w and key k, the expression $[\![p]\!]_{w,l,k}$ is easily seen to be monotonic in l, i.e., if $l \le l'$ then $[\![p]\!]_{w,l,k} \subseteq [\![p]\!]_{w,l',k}$. Moreover, it is also continuous in l.

Lemma A.1: Suppose $l_0 \le l_1 \le \ldots$ is an increasing sequence of local name assignments and let $l_\omega = \sqcup_{m \in N} l_m$. For all principal expressions p, we have $[\![p]\!]_{w,l_\omega,k} = \bigcup_{m \in N} [\![p]\!]_{w,l_m,k}$.

Proof: By a straightforward induction on the structure of p. ∎

Given the world $w = (\ ; c)$, we define an operator T_w on the space of local name assignments LNA. For a local name assignment l, we define $T_w(l)$ to be the local name assignment such that for all $k \in K$ and $n \in N$, the set $T_w(l)(k, n)$ is the union of the sets $[\![p]\!]_{w,l,k}$ such that the formula $n \mapsto p$ is in $c(k)$. The following lemma is follows easily from Lemma A.1.

Lemma A.2: The mapping T_w is a continuous operator on $(LNA; \le)$.

The following lemma is almost immediate from the definitions.

Lemma A.3: A local name assignment l is consistent with a world w iff $T_w(l) \le l$.

Suppose $(X; \le)$ is a complete partial order with minimal element $\perp$. An element $x \in X$ is said to be a pre-fixpoint of an operator T on X if $T(x) \le x$; x is a fixpoint of T if $T(x) = x$. Given an operator T on X, define a sequence of elements $T \uparrow \alpha$, where α is an ordinal, as follows. For the base case, let $T \uparrow 0 = \perp$. For successor ordinals $\alpha + 1$, define $T \uparrow \alpha + 1 = T(T \uparrow \alpha)$. For limit ordinals λ, define $T \uparrow \lambda = \sqcup\{T \uparrow \alpha : \alpha < \lambda\}$. A well-known result (see [LNS82] for a discussion of its history) states that if T is continuous then then this sequences converges to the least pre-fixpoint of T, that convergence has taken place by $\alpha = \omega$, and that $T \uparrow \omega$ is in fact a fixed point of T. Thus, we obtain as a corollary of Lemma A.2 and Lemma A.3 that there exists a minimal local name assignment consistent with w, and that this local name assignment equals $T_w \uparrow \omega$. The second half of Theorem 3.1 is immediate from the earlier observation that $[\![p]\!]_{w,l,k}$ is monotonic in l. ∎

Theorem 3.2: AX_n (resp., AX_{inf}) is a sound and complete axiomatization of LLNC with respect to the open semantics if K is infinite (resp., K is finite).

Proof: We start with the completeness proof for AX_{inf}, so that we assume that K is infinite. We then show how to deal with AX_n. As usual, it suffices to show that if φ is AX_{inf}-consistent, then φ is satisfiable. In fact, we put a little extra work into our proof that φ is satisfiable so that we can prove Proposition 3.3 as well.

Let $Sub(\varphi)$ consist of all subformulas of φ. We say that a principal expression p' is a variant of p if $p \mapsto p'$ and $p' \mapsto p$ are both provable using only Reflexivity, Associativity, and Transitivity. The left-associative variant of a principal expression p is the one where we associate all terms to the left. Thus, $((n_1\text{'s } n_2)\text{'s } n_3)\text{'s } n_4$ is the left-associative variant of $n_1\text{'s } ((n_2\text{'s } n_3)\text{'s } n_4)$.

Define P to be the smallest set of principal expressions such that

1. if $p \mapsto q$ is in $Sub(\varphi)$ then p and q are in P,
2. if k cert $(n \mapsto p) \in Sub(\varphi)$ then $k\text{'s } n$ and $k\text{'s } p$ are in P,
3. if $p \in P$ and p' is the left-associative variant of p, then $p' \in P$,
4. P is closed under subexpressions, so that if $p\text{'s } q \in P$, then so are p and q,
5. if $k \in P$ is a key and $n \in P$ is a local name, then $k\text{'s } n \in P$.

For Proposition 3.3, it is necessary to get an upper bound on the size of P in terms of $|\varphi|$.

Lemma A.4: $|P| < 2|\varphi|^2$.

Proof: Let $|p|$ be the total number of expressions in $G \cup K \cup N$ that appear in p, counted with multiplicity. An easy proof by induction on structure shows that a principal expression p has at most $|p|$ subexpressions, at least one of which must be in $G \cup K \cup N$. For every other subexpression q, there is a unique left-associative variant q', which has at most $|q'| = |q| \le |p|$ subexpressions, each of which is associated to the left. Thus, starting with a principal expression p, the least set closed under clauses 3 and 4 above contains at most $|p|^2$ elements. Now a straightforward induction on the structure of φ shows that the least set P' closed under clauses 1–4 above has at most $|\varphi|^2$ expressions. Finally, it is easy to see that closing off under 5 gives us P, since the set that results after closing off under 5 is still closed under 1–4. Moreover, this final step adds at most $|\varphi|^2$ expressions $k\text{'s } n$, since both k and n must be subexpressions of φ. ∎

Let k_0 be some key not occurring in P. We use k_0 both to express emptiness of expressions in P and as the "current principal". Define P_1 to be the set of principal expressions $P \cup \{k_0\} \cup \{p\text{'s } k_0 : p \in P\}$. Let E be consist of the formulas $p\text{'s } k_0 \mapsto k_0$ for each $p \in P$. Note that all principal expressions occurring in the formulas in E are in P_1. Let S be an AX_{inf}-consistent set containing φ and, for every formula $\psi \in Sub(\varphi) \cup E$, either ψ or $\neg\psi$. Since φ is AX_{inf}-consistent, there must be some AX_{inf}-consistent set S of this form.

Define $S^+ = Cl(S; P_1)$ to be the smallest set of formulas containing S closed under Reflexivity, Transitivity, Left Motonocity, Converse of Globality, Globality, and Nonemptiness, in the sense that

(C1R) if $p \in P_1$, then $p \mapsto p \in S^+$,

(C1T) if $p \mapsto q$ and $q \mapsto r$ are both in S^+, then $p \mapsto r \in S^+$,

(C1LM) if $p \mapsto q \in S^+$, p's $r \in P_1$, and q's $r \in P_1$, then p's $r \mapsto q$'s $r \in S^+$,

(C1CG) if p's $g \in P_1$ for $g \in K \cup G$ then $g \mapsto p$'s $g \in S^+$,

(C1G) if p's $k \mapsto k \in S^+$ for some key k and p's $g \in P_1$, where $g \in K \cup G$, then p's $g \mapsto g \in S^+$,

(C1KL) if k cert $(n \mapsto p) \in S^+$ then $(k$'s $n \mapsto k$'s $p) \in S^+$,

(C1K) if $p \mapsto k' \in S^+$ and p's $k \in P_1$, then p's $k \mapsto k \in S^+$,

(C1N) if $\neg(p \mapsto q) \in S^+$ and q's $k \in P_1$, then q's $k \mapsto k \in S^+$,

(C1C) if p's $q \mapsto k_1 \in S^+$ and p's $k \in P_1$, then p's $k \mapsto k \in S^+$,

(C1NE) if p's $k \mapsto k$ and $k' \mapsto p$ are both in S^+, then $p \mapsto k' \in S^+$,

(C1KD) if k and k' are distinct keys in P, then $\neg(k \mapsto k') \in S^+$,

(C1LV) If p' is the left-associative variant of $p \in P$, then $p \mapsto p' \in S^+$ and $p' \mapsto p \in S^+$.

It is easy to see that S^+ is AX_{inf}-consistent, since S is and each of the closure rules emulates an axiom in AX_{inf}. Our goal now is to show that there exists a triple w, l, k such that $w, l, k \models \psi$ for all $\psi \in S$ (and thus, in particular, $w, l, k \models \varphi$).

Lemma A.5: If k_0 appears in the formula $p \mapsto q \in S^+$, then k_0 appears in both p and q.

Proof: An easy induction on the construction of S^+, using the fact that all principal expressions occurring in S^+ are in P_1 and k_0 appears only as the rightmost expression in a principal expression in P_1. ∎

By Lemma A.5, if $p \mapsto q \in S^+$ and one of the expressions p, q is in P (and thus does not mention k_0) then so is the other. Define a binary relation $\approx$ on P by defining $p \approx q$ if both $p \mapsto q$ and $q \mapsto p$ are in S^+. It is immediate from transitivity and reflexivity that $\approx$ is an equivalence relation on P. Given $p \in P$, we write $[p]$ for the equivalence class of p under $\approx$.

We classify the expressions in P as follows. Say that an expression p in P is empty (with respect to S^+) if $\neg(p$'s $k_0 \mapsto k_0)$ is in S^+. Say that p is key-equivalent if it is not empty and $k \mapsto p$ is in S^+ for some key k (by (C1NE) this implies $p \approx k$). Intuitively, the interpretation of an empty expression will be the empty set and the interpretation of a key-equivalent expression p such that $k \mapsto p \in S^+$ will be $\{k\}$. If p is neither empty nor key-equivalent, we say it is open. Clearly, every expression in P is either empty, key-equivalent, or open. Moreover, by (C1LM) and (C1T), if $p \approx q$ then p is empty, key-equivalent or open iff q is. In particular, we may sensibly refer to open $\approx$-equivalence classes of expressions in P.

Let O be the set of open equivalence classes of expressions in P. Note that if $K_\varphi \subseteq K$ consists of all the keys in K that appear in φ, then there are fewer than $2^{|\varphi|^2|K_\varphi|}$ equivalence classes of open expressions. For each class $c \in O$, let k_c be a fresh key. Intuitively, the key k_c will act as a canonical representative of the keys in the interpretation of an expression $p \in c$, in the sense that the interpretations of p's q and k_c's q will be the same for certain expressions q. Since K is infinite, we are guaranteed that we can always find keys k_c, but the argument works even if K is finite, as long as $|K| \geq 2|\mathcal{P}^2|$. (We also need to have a key in $K \setminus K_{\mathcal{P}}$ to be k_0.)

Define S^* to be consist of S^+ together with, for all $c \in \mathcal{O}$,

1. the formula $k_c \mapsto k_c$, and

2. the formulas $p \mapsto k_c$, where for some $q \in c$ we have $p \mapsto q \in S^+$.

It is easy to show that k_0 does not appear in any formula in $S^* - S^+$: Clearly k_0 does not appear in the formulas $k_c \mapsto k_c$ added by clause 1. If $p \mapsto q$ is a formula added by clause 2, then there is some equivalence class c and expression $q \in c$ such that $p \mapsto q \in S^+$. Since c is an equivalence class of expressions in $\mathcal{P}$, none of which contain k_0, the expression q does not contain k_0. It follows from Lemma A.5 that p does not contain k_0. Since $S^* - S^+$ contains no formulas involving k_0, S^* also satisfies the property stated for S^+ in Lemma A.5.

Define the local name assignment l as follows. Given a key k and local name n,

1. $l(k_0, n) = \{k' \in K \mid n \mapsto k' \in S^*\}$,

2. $l(k, n) = \{k' \in K \mid k\text{'s } n \mapsto k' \in S^*\}$ if $k \in \mathcal{P}$,

3. $l(k, n) = \{k' \in K \mid p\text{'s } n \mapsto k' \in S^* \text{ and } p \in c\}$ if $k = k_c$ for some $c \in \mathcal{O}$,

4. $l(k, n) = \emptyset$ for all other k.

Define the world $w = (\pi, c)$ by taking $\pi(g) = \{k \in K \mid g \mapsto k \in S^*\}$ and defining $c(k)$, for each key k, to be the set of formulas $n \mapsto p$ such that $(k \text{ cert } (n \mapsto p)) \in S^*$. Note for future reference that there exists a finite subset K_1 of K such that $l(n, k) \subseteq K_1$, $l(n, k) = \emptyset$ for $k \notin K_1$, $\pi(g) \subseteq K_1$, and $\pi(g) = \emptyset$ if g does not appear in Σ. Indeed, K_1 consists of the keys that appear in S^*, k_0, and the keys k_c for $c \in \mathcal{O}$.

Let $I(p) = \{k \in K \mid p \mapsto k \in S^*\}$.

Lemma A.6: If $p \in \mathcal{P}$, then p is empty iff $I(p) = \emptyset$.

Proof: If p is not empty, then it is either key-equivalent or open. If it is key-equivalent, we have already observed that there must exist some key k' such that $p \mapsto k' \in S^*$, so $I(p) \neq \emptyset$. If it is open, suppose it is in equivalence class c. Then $p \mapsto k_c \in S^*$, since $p \mapsto p \in S^+$ by (CIR). Again, it follows that $I(p) \neq \emptyset$.

Conversely, suppose that $I(p) \neq \emptyset$. Thus, $p \mapsto k \in S^*$ for some key k. If $p \mapsto k \in S^+$, then by (CIK), p's $k_0 \mapsto k_0 \in S^+$, so p is not empty. If $p \mapsto k \notin S^+$, then $k = k_c$, and there is some $q \in c$ such that $p \mapsto q \in S^+$. Since q is open, q cannot be empty, so q's $k_0 \mapsto k_0 \in S^+$. Moreover, by (CILM), p's $k_0 \mapsto q$'s $k_0 \in S^+$. Thus, by (CIT), p's $k_0 \mapsto k_0 \in S^+$, so p is nonempty. ∎

Lemma A.7: For all expressions $p \in \mathcal{P}$, we have $[\![p]\!]_{w,l,k_0} = I(p)$.

Proof: We proceed by induction on $|p|$ (as defined in Lemma A.4). The claim is immediate from the definitions in case p is a global name or a local name. Suppose that p is a key k_1. Then $[\![p]\!]_{w,l,k_0} = \{k_1\}$. Since $k_1 \mapsto k_1 \in S^-$ by construction, it follows that $k_1 \in I(k_1)$. It remains to show that $I(k_1) \subseteq \{k_1\}$. Suppose $(k_1 \mapsto k) \in S^-$. By Lemma A.5, we cannot have $k = k_0$. Since S^+ is AX_{inf}-consistent and closed under (CIKD), if $k \in P$ we must have $k_1 = k$. The remaining possibility for k, that it equals k_c for some $c \in O$, cannot happen. For if so, only the second clause of the definition of S^- could explain $(k_1 \mapsto k) \in S^-$. But then we have $(k_1 \mapsto q) \in S^+$ for some $q \in c$. This contradicts the assumption that c is an equivalence class of open expressions.

Finally, suppose that $|p| > 1$. Let p' be the left-associative variant of p. It is clear from the semantics that $[\![p]\!]_{w,l,k_0} = [\![p']\!]_{w,l,k_0}$. Morover, (CILV) and (CIT) guarantee that $I(p) = I(p')$. Thus, it suffices to prove that $I(p') = [\![p']\!]_{w,l,k_0}$. Suppose that $p' = q$'s r. The definition of length guarantees that $|p'| = |p| > |q|$, so the induction hypothesis applies to q. Since p' is associated to the left, $r \in G \cup K \cup N$.

Suppose that $r = g \in G \cup K$. Note that $[\![q\text{'s } g]\!]_{w,l,k_0} = \emptyset$ if $[\![q]\!]_{w,l,k_0} = \emptyset$ and $[\![q\text{'s } g]\!]_{w,l,k_0} = [\![g]\!]_{w,l,k_0}$ if $[\![q]\!]_{w,l,k_0} \neq \emptyset$. We consider these two cases separately.

Suppose first that $[\![q]\!]_{w,l,k_0} = \emptyset$, so $[\![p']\!]_{w,l,k_0} = \emptyset$. By the induction hypothesis, $I(q) = \emptyset$. To show that $I(p') = \emptyset$, we show that p' is empty. Suppose not. Then (p')'s $k_0 \mapsto k_0 \in S^+$. Since S^+ contains either q's $k_0 \mapsto k_0$ or $\neg(q$'s $k_0 \mapsto k_0)$ and S^+ is AX_{inf}-consistent, by Nonemptiness(c), Associativity, and Transitivity, we must have q's $k_0 \mapsto k_0 \in S^+$. Thus, q is not empty. By Lemma A.6, $I(q) \neq \emptyset$, a contradiction. Hence, p' is empty. It now follows from Lemma A.6 that $I(p') = \emptyset$, as desired.

Consider next the case where $[\![q]\!]_{w,l,k_0} \neq \emptyset$, so $[\![p']\!]_{w,l,k_0} = [\![q\text{'s } g]\!]_{w,l,k_0} = [\![g]\!]_{w,l,k_0}$. To show that $[\![p']\!]_{w,l,k_0} = I(p')$, we show that $I(p') = I(g)$. The result then follows from the induction hypothesis.

By the induction hypothesis, $I(q) \neq \emptyset$, so by Lemma A.6, q is not empty. It follows from (CIG) that q's $g \mapsto g \in S^+$. Suppose that $k \in I(g)$. If $k \in P_1$, then $g \mapsto k \in S^+$, so by (CIT), q's $g \mapsto k \in S^+$ and $k \in I(p')$. If $k = k_c$ for some $c \in O$, then $g \mapsto q' \in S^+$ for some $q' \in c$. Thus, $p' \mapsto q' \in S^+$ by (CIT) and we obtain that $p' \mapsto k \in S^-$ by construction of S^-. Thus, $I(g) \subseteq I(p')$.

For the opposite containment, note that by (CICG) we have $g \mapsto q$'s $g \in S^+$. Arguing as above, we obtain using (CIT) that $I(g) \supseteq I(p')$. This completes the proof that $I(p') = I(g)$.

It remains to deal with the case that p' has of the form q's n, where n is a local name. There are three possibilities: q is empty, key-equivalent or open. If q is empty, then by Lemma A.6 and the induction hypothesis, $I(q) = \emptyset$ and $[\![q]\!]_{w,l,k_0} = \emptyset$. It follows that $[\![p']\!]_{w,l,k_0} = \emptyset$. Moreover, using Nonemptiness(c), Associativity, and Transitivity as above, it follows that p' is empty and hence by Lemma A.6, $I(p') = \emptyset$, as desired.

If q is key-equivalent, say $q \equiv k_1$, then $q \mapsto k_1 \in S^+$ and $k_1 \mapsto q \in S^+$. Using Key Distinctness and the consistency of S^+, it easily follows that $I(q) = \{k_1\}$. By the induction hypothesis, $[\![q]\!]_{w,l,k_0} = \{k_1\}$. Thus, $[\![p']\!]_{w,l,k_0} = l(k_1, n)$. By construction, $l(k_1, n) = I(k_1$'s $n) = I(p')$, as desired.

Finally, suppose that q is open. If $k \in I(p')$, then it is immediate from the construction that that $q \mapsto k_{[q]} \in S^-$ and $k \in l(k_{[q]}, n)$. By the induction hypothesis, $k_{[q]} \in [\![q]\!]_{w,l,k_0}$, so

$k \in [\![p']\!]_{w,l,k_0} = \cup_{k' \in [\![q]\!]_{w,l,k_0}} l(k',n)$. Thus, $I(p') \subseteq [\![p']\!]_{w,l,k_0}$ if p' is open.

For the opposite containment, suppose that $k \in [\![p']\!]_{w,l,k_0}$. This means that there is some key k' such that $k' \in [\![q]\!]_{w,l,k_0}$ and $k \in l(k',n)$. By the induction hypothesis, $k' \in I(q)$, so $q \mapsto k' \in S$. If $k' \in P_1$, then $q \mapsto k' \in S^+$ and (k')'s $n \mapsto k \in S^+$. (Since $q \in P$ and $q \mapsto k' \in S$, we cannot have $k' = k_0$, by Lemma A.5.) By (CILM), q's $n \mapsto (k')$'s $n \in S^+$, so by (CIT) we get q's $n \mapsto k \in S^+$. Hence, $k \in I(p')$. If $k' = k_c$, where c is an open equivalence class, then from $q \mapsto k' \in S$ it follows that $q \mapsto q' \in S^+$ for some $q' \in c$. From $k \in l(k_c,n)$ it follows that (r')'s $n \mapsto k \in S$ for some $r' \in c$. By construction of S^+ we must have (r')'s $n \in P_1$, and since $r' \equiv q'$, we have $q' \mapsto r' \in S^+$. By (CIT) we obtain $q \mapsto r' \in S^+$, and hence by (CILM) that q's $n \mapsto (r')$'s $n \in S^+$. Now notice that it follows from q's $n \mapsto (r')$'s $n \in S^+$ and (r')'s $n \mapsto k \in S$ that q's $n \mapsto k \in S$. If $k \in P_1$, this is immediate from (CIT). In case $k = k_d$ for some open class d, we have (r')'s $n \mapsto t \in S^+$ for some $t \in d$. But then q's $n \mapsto t \in S^+$ by (CIT); by definition of S we get that q's $n \mapsto k \in S$. This completes the proof. ∎

Lemma A.8: For all formulas $\psi \in Sub(\varphi) \cup E$, we have $\psi \in S$ iff $w,l,k_0 \models_o \psi$.

Proof: We first show that by induction on the structure of $\psi \in Sub(\varphi) \cup E$ that $\psi \in S$ iff $w,l,k_0 \models \psi$, and then show that the assignment l is consistent with w.

It is immediate from the construction of w that $w,l,k_0 \models \psi$ iff $\psi \in S$ for ψ of the form k cert $(n \mapsto p)$.

If ψ has the form $p \mapsto q$, note that $w,l,k_0 \models p \mapsto q$ iff $[\![p]\!]_{w,l,k_0} \subseteq [\![q]\!]_{w,l,k_0}$ iff (by Lemma A.7) iff $I(p) \subseteq I(q)$. Thus, it suffices to show that $I(p) \subseteq I(q)$ iff $p \mapsto q \in S^+$, for $p,q \in P$.

The "if" direction is immediate from (CIT): If $k \in I(q)$ then $q \mapsto k \in S$, so by (CIT) and the construction of S, $p \mapsto k \in S$ and thus $k \in I(p)$.

For the "only if" direction, suppose by way of contradiction that $I(p) \subseteq I(q)$ but $p \mapsto q \notin S^+$. Then, by construction, $\neg(p \mapsto q) \in S^+$. We consider three cases, depending on whether q is empty, key-equivalent, or open.

Note first that q cannot be empty: $\neg(p \mapsto q) \in S^+$, so by (CIN) we have q's $k_0 \mapsto k_0 \in S^+$.

Suppose that q is key-equivalent, with $k \mapsto q \in S^+$. If $p \mapsto k \in S^+$ then, by (CIT), $p \mapsto q \in S^+$, but this is not possible because S^+ is AX_{inf}-consistent. Thus $p \mapsto k \notin S^+$. Since $k \in P$, $p \mapsto k \notin S$, and thus $k \in I(p) - I(q)$, giving us the desired contradiction.

Finally, suppose q is open. By construction, $q \mapsto k_{[q]} \in S$. Moreover, we cannot have $p \mapsto k_{[q]} \in S$, for then there would exist $r \equiv q$ such that $p \mapsto r \in S^+$. Using (CIT), it would follow that $p \mapsto q \in S^+$, which is impossible since S^+ is AX_{inf}-consistent. Thus, $k_{[q]} \in I(p) - I(q)$, giving the required contradiction, and completing the proof in the case that ψ is of the form $p \mapsto q$.

If ψ is of the form $\neg\psi'$ or $\psi_1 \wedge \psi_2$, the result is immediate from the induction hypothesis (in the latter case, we need the fact that if $\psi_1 \wedge \psi_2 \in Sub(\varphi) \cup E$, then in fact $\psi_1 \wedge \psi_2 \in Sub(\varphi)$, so $\psi_1, \psi_2 \in Sub(\varphi)$ and the induction hypothesis applies). This completes the induction proof.

To show that the assignment l is consistent with w, suppose that $n \mapsto p \in c(k)$. Then, by construction, k cert $(n \mapsto p) \in S$. By (CIKL), we have k's $n \mapsto k$'s $p \in S^+$. By what we have just shown $w,l,k_0 \models k\text{'s } n \mapsto k\text{'s } p$. It follows that $w,l,k \models n \mapsto p$. Thus, l is consistent with w. ∎

Thus, we have shown that φ is satisfiable, completing the proof of Theorem 3.2 in the case that K is infinite. The same argument works without change if K is finite but $|K| \geq 2^{|\varphi|}$. (A consequence of this is that we do not need to use the axioms Witnesses and Current Principal to derive a valid formula φ in AX_n if $2^{|\varphi|} \leq |K|$.) Moreover, the proof shows that Proposition 3.3 holds if $|K| \geq 2^{|\varphi|}$.

Now suppose that $|K| < 2^{|\varphi|}$. We show that if φ is AX_n-consistent, then φ is satisfiable. The proof is in the spirit of that in the case of AX_{inf}, but simpler.

Now let P be the least set of principal expressions containing all principal expressions that appear in φ and closed under subexpressions. Let F consist of all formulas of the form $p \mapsto k'$ and $k\text{'s } p \mapsto k'$, where $p \in P$ and $k, k' \in K$. Let S be an AX_n-consistent set containing φ and, for every formula $\psi \in Sub(\varphi) \cup F$, either ψ or $\neg\psi$. Since φ is AX_n-consistent, there must be some AX_n-consistent set S of this form.

There must be some key $k_0 \in K$ such that for every local name in P and key $k \in K$, we have $n \mapsto k \in S$ iff $k_0\text{'s } n \mapsto k \in S$. For otherwise, for each key k, there is some local name n_k and key k_k such that either both $n_k \mapsto k_k$ and $\neg(k\text{'s } n_k \mapsto k_k)$ are in S or both $\neg(n_k \mapsto k_k)$ and $k\text{'s } n_k \mapsto k_k$ are in S. This means that S is inconsistent with the axiom Current Principal. Define the local assignment l so that $l(k,n) = \{k' : k\text{'s } n \mapsto k' \in S\}$. Similar to the case for AX_{inf}, define the world $w = (\sigma, c)$ by taking $\sigma(g) = \{k \in K \mid g \mapsto k \in S^+\}$ and defining $c(k)$, for each key k, to be the set of formulas $n \mapsto p$ such that $k \text{ cert } (n \mapsto p) \in S$.

Now we have the following analogue to Lemma A.8.

Lemma A.9: For all formulas $\psi \in Sub(\varphi) \cup F$, we have $\psi \in S$ iff $w,l,k_0 \models_o \psi$.

Proof: Again we first show that by induction on the structure of $\psi \in Sub(\varphi) \cup E$ that $\psi \in S$ iff $w,l,k_0 \models \psi$, and then show that the assignment l is consistent with w.

It is immediate from the construction of w that $w,l,k_0 \models \psi$ iff $\psi \in S$ for ψ of the form $k \text{ cert } (n \mapsto p)$.

We next show that the result holds if ψ is of the form $p \mapsto k'$, for $p \in P$, by induction on the structure of p. We strengthen the induction hypothesis to also show that $w,l,k_0 \models k\text{'s } p \mapsto k'$ iff $k\text{'s } p \mapsto k \in S$. If p is a key k_1, then $w,l,k_0 \models k_1 \mapsto k'$ iff $k' = k_1$ and by Reflexivity and Key Distinctness, $k_1 \mapsto k' \in S$ iff $k_1 = k'$. Similarly, $w,l,k_0 \models k\text{'s } k_1 \mapsto k'$ iff $w,l,k_0 \models k_1 \mapsto k'$ iff $k_1 \mapsto k' \in S$ iff $k\text{'s } k_1 \mapsto k' \in S$, by Transitivity, Key Globality, and Converse of Globality (using the fact that S is AX_n-consistent).

If p is a global identifier g, $w,l,k_0 \models g \mapsto k'$ iff $g \mapsto k' \in S$ by the definition of σ. The argument for $k\text{'s } g \mapsto k'$ is identical to the case that $p = k$.

If p is the local name n, then $w,l,k_0 \models n \mapsto k'$ iff $k' \in l(k_0,n)$ iff $k_0\text{'s } n \mapsto k' \in S$ iff $n \mapsto k' \in S$, by choice of k_0. Similarly, $w,l,k_0 \models k\text{'s } n \mapsto k'$ iff $k' \in l(n,k)$ iff $k\text{'s } n \mapsto k' \in S$.

Finally, if p is of the form $q\text{'s } r$, then $w,l,k_0 \models q\text{'s } r \mapsto k'$ iff there exists a key k'' such that $w,l,k_0 \models q \mapsto k''$ and $w,l,k_0 \models (k'')\text{'s } r \mapsto k'$ iff (by the induction hypothesis) there exists a key k'' such that $q \mapsto k'' \in S$ and $(k'')\text{'s } r \mapsto k' \in S$ iff $q\text{'s } r \mapsto k' \in S$. The "only if" direction of the last equivalence follows using Left Monotonocity and Transitivity; the "if" direction follows from Witnesses. The argument for k's $(q$'s $r) \mapsto k'$ is identical, using Associativity: $w, l, k_0 \models k$'s $(q$'s $r) \mapsto k'$ iff there exists a key k'' such that $w, l, k_0 \models k$'s $q \mapsto k''$ and $w, l, k_0 \models (k'')$'s $r \mapsto k'$ iff there exists a key k'' such that k's $q \mapsto k'' \in S$ and (k'')'s $r \mapsto k' \in S$ iff k's $(q$'s $r) \mapsto k' \in S$.

We now continue with our induction in the case that $p \mapsto q$. Note that $w, l, k_0 \models p \mapsto q$ iff $w, l, k_0 \models q \mapsto k'$ implies $w, l, k_0 \models p \mapsto k'$ for all $k' \in K$ iff (by the induction hypothesis) $q \mapsto k' \in S$ implies $p \mapsto k' \in S$ iff $p \mapsto q \in S$. The "only if" direction of the last equivalence follows immediately from Transitivity; the "if" direction follows from Witnesses.

We complete the induction proof by observing that if φ is of the form $\neg\varphi'$ or $\varphi_1 \wedge \varphi_2$, the result follows immediately from the induction hypothesis.

To show that l is consistent with w, suppose that $n \mapsto p \in c(k)$. By construction, this means that k cert $(n \mapsto p) \in S$. By Key Linking, we must also have k's $n \mapsto k$'s $p \in S$. By what we have just shown, $w, l, k_0 \models k$'s $n \mapsto k$'s p. It follows that $w, l, k \models n \mapsto p$. Thus, l is consistent with w. ∎

This completes the proof of Theorem 3.2 in the case that K is finite. Note that since we can assume without loss of generality that $|K| \geq 2|\varphi|^2$ here (otherwise the argument for the case that K is infinite applies) the proof also shows that Proposition 3.3 holds. ∎

Theorem 3.5: The same formulas are c-valid and o-valid; i.e., for all formulas φ, we have $\models_o \varphi$ iff $\models_c \varphi$.

Proof: We show that $\neg\varphi$ is o-satisfiable iff $\neg\varphi$ is c-satisfiable, which is equivalent to the claim. The direction from c-satisfiability to o-satisfiability is straightforward: Since for every world w the local name assignment l_w is w-consistent, it follows from $w, k \models_c \neg\varphi$ that $w, l_w, k \models_o \neg\varphi$. Thus, it remains to show that if $\neg\varphi$ is o-satisfiable, then it is c-satisfiable.

So suppose that $\neg\varphi$ is o-satisfiable. By Proposition 3.3, there is a world $w = (\pi, c)$, local name assignment l, and principal k such that $w, l, k \models_o \neg\varphi$ and a finite subset K' of K such that $l(k', n) \subseteq K'$ for all $k' \in K$ and $n \in N$, and $\pi(g) \subseteq K'$ for all global names g. By standard propositional reasoning, $\neg\varphi$ is equivalent to a disjunctive normal form expression in which the atoms are of the form $p \mapsto q$ and k_1 cert φ', where p and q are principal expressions, k_1 is a key, and φ' is a formula. If $w, l, k \models_o \neg\varphi$ then one of the disjuncts ψ is satisfied, i.e., $w, l, k \models_o \psi$. Suppose that ψ is the conjunction of the formulas in the set $A \cup B$, where

1. A is a set of formulas of the form $p \mapsto q$ or $\neg(p \mapsto q)$,
2. B is a set of formulas of the form k_1 cert φ' or $\neg(k_1$ cert $\varphi')$.

Let K_ψ be the set of keys that appear in the formula ψ together with K' and k. Let N_ψ be the set of local names that appear in ψ. Define the world $w' = (\pi', c')$ as follows. Take the interpretation of global names π' to be equal to π, the interpretation of global names in w. Define c' by taking the set of certificates $c'(k')$ to be the empty if $k' \notin K_\psi$ and to consist of $c(k')$ together with all certificates of the form $n \mapsto p_{k'',}$ if $k' \in K_\psi$, $n \in N_\psi$, and $k'' \in l(n, k')$, where $p_{k'',}$ is a principal expression of the form (k'')'s (k'')'s $\ldots (k'')$ that does not appear in

. (Clearly we can make the expression su ciently long so as to ensure it does not appear in .) Clearly $[_{k^0 2 K} c(k^0)$ is nite.

We show that $w^0; k \models_c$. It follows from this that $w^0; k \models_c :$. Note rst that from the fact that $c(k^0)$ $c^0(k^0)$ for all k^0, it follows that $w^0; k \models_c k^0$ cert for all formulas k^0 cert in B . Moreover, if : (k^0 cert) is in B then, since the expressions $p_{k^{00};}$ on the right-hand side of the certi cates in $c^0(k^0)$ $c(k)$ do not appear in it follows that $w^0; k \models_c$: (k^0 cert). Thus $w^0; k \models_c B$.

It remains to show that the formulas in A are satis ed. To show this, we show that

$$l_{w^0}(n; k^0) = l(n; k^0) \text{ for all } n \; 2 \; N \quad \text{and } k^0 \; 2 \; K \; . \tag{2}$$

It easily follows from (2), the fact that all keys in are in K^0, and the fact that global names have the same interpretation in w and w^0 that $[\![p]\!]_{w^0; l_{w^0}; k^0} = [\![p]\!]_{w; l; k^0}$ for all principal expressions p occurring in A and all keys k^0 2 K . This in turn is easily seen to imply that $w^0; k \models_c A$.

It remains to prove (2). It is almost immediate from the de nition of l^0 that $l_{w^0}(n; k^0)$ $l(n; k^0)$ for all n 2 N and k^0 2 K . For the opposite containment, we prove by induction on j that $(T_{w^0} " j)(n; k^0)$ $l(n; k^0)$ for all j 2 N , n 2 N , and k^0 2 K . The base case j = 0 is trivial. For the induction step, suppose that $j = j^0 + 1$ and k^{00} 2 $(T_{w^0} " j)(n; k^0)$. Thus, k^{00} 2 $(T_{w^0}(T_{w^0} " j^0))(n; k^0)$, which means that k^{00} 2 $[\![p]\!]_{w^0; T_{w^0} " j^0; k^0}$ for some principal expression p such that n 7! p 2 $c^0(k^0)$. There are two possibilities: (1) n 7! p 2 $c(k^0)$ or (2) n 7! p 2 $c^0(k^0)$ $c(k^0)$. In case (2), p must be of the form $p_{k_1;}$ so $[\![p]\!]_{w^0; T_{w^0} " j^0; k^0} = fk_1 g$ and $k_1 = k^{00}$. But in this case, by construction, k^{00} 2 $l(n; k^0)$. In case (1), using the induction hypothesis and the fact that global names and keys in p have the same interpretation in w and w^0 (this interpretation being a subset of K^0), we get that $[\![p]\!]_{w^0; T_{w^0} " j^0; k^0}$ $[\![p]\!]_{w; l; k^0}$. Thus, k^{00} 2 $[\![p]\!]_{w; l; k^0}$. Because l is w-consistent and n 7! p 2 $c(k^0)$, we again obtain that k^{00} 2 $l(n; k^0)$, as required.

Since $l_{w^0}(n; k^0)$ is the union of the $(T_{w^0} " j)(n; k^0)$, it follows that $l_{w^0}(n; k^0) = l(n; k^0)$. This completes the proof of (2). ∎

Proposition 3.8: Let be any c-satis able boolean combination of formulas of the form k cert , and let be any boolean combination of formulas of the form p 7! q where neither p nor q contains a local name. Then $\models_c$) i $\models_c$.

Proof: Clearly $\models_c$ implies $\models_c$) . For the converse, suppose by way of contradiction that $\models_c$) and there is a world w = (; c) and a principal k such that $w; k \models_c :$. Since is assumed to be c-satis able, there exists a world $w^0 = ({}^0; c^0)$ and a principal k^0 such that $w^0; k^0 \models_c$. Let w^{00} be the world (; c^0). Then a straightforward induction shows that for all principal expressions p not containing a local name, we have $[\![p]\!]_{w^{00}; l_{w^{00}}; k} = [\![p]\!]_{w; l_w; k}$. Moreover, for all keys k_1 and formulas , we have $w^{00}; k \models_c k_1$ cert i $w^0; k^0 \models_c k_1$ cert . It follows that $w^{00}; k \models_c$ ^ : , giving us our desired contradiction. ∎

Theorem 4.1: Suppose $k_1; k_2$ are principals, w = (; c) is a world, and p is a principal expression. Let E_w be the set of all the formulas g 7! k for all global names g and keys k 2 (g) and the formulas k cert for all keys k and formulas 2 c(k). The following are equivalent:

1. $k_1 \in \mathrm{REF2}(k_2, \ , c, p)$,

2. $w, k_2 \vdash_c p \mapsto k_1$,

3. $w', k_2 \vdash_c p \mapsto k_1$ for all worlds $w' \quad w$,

4. $E_w \vdash_c k_2$'s $p \mapsto k_1$,

5. $E_w \vdash_o k_2$'s $p \mapsto k_1$.

Proof: The presentation of REF2 in Figure 1 is still slightly informal, combining recursion and nondeterminism. To make it fully precise, define a computation tree of REF2 to be a finite tree labelled by expressions of the form "$k_1 \in \mathrm{REF2}(k_2, \ , c, p)$", such that if N is a node so labelled, then one of the following four conditions holds:

1. p is a key k, we have $k = k_1 = k_2$, and N is a leaf of the tree,

2. p is a global name g and $k_1 \in \ (g)$,

3. p is a local name n and $c(k_2)$ contains a formula $n \mapsto q$ and N has exactly one child, labelled "$k_1 \in \mathrm{REF2}(k_2, \ , c, q)$",

4. p is of the form q's r and N has exactly two children, labelled "$k \in \mathrm{REF2}(k_2, \ , c, q)$" and "$k_1 \in \mathrm{REF2}(k, \ , c, r)$", for some key k.

We take $k_1 \in \mathrm{REF2}(k_2, \ , c, p)$ to mean that there exists a computation tree of REF2 with root labelled "$k_1 \in \mathrm{REF2}(k_2, \ , c, p)$".

Given a world $w = (\ , c)$ and $m \in \mathbb{N}$, let $l_m = T_w \uparrow m$. The following result establishes a correspondence between the stages of the computation of l_w and the computation trees of REF2. The proof is by a straightforward induction on m, with a subinduction on the structure of p.

Lemma A.10: For all $m \in \mathbb{N}$, keys k_1, k_2, worlds $w = (\ , c)$, and principal expressions p, we have $k_1 \in [\![p]\!]_{w, l_m, k_2}$ iff there exists a computation tree of REF2 of height at most m whose root is labelled "$k_1 \in \mathrm{REF2}(k_2, \ , c, p)$".

Using the fact that $l_w = \sqcup\{l_m : m \in \mathbb{N}\}$, Lemma A.1, and Lemma A.10, we obtain the equivalence between (1) and (2).

The proof of the implication from (2) to (3) is by a straightforward induction on the structure of p; that is, for fixed $w' \quad w$, we show by induction on the structure of p that if $w, k_2 \vdash_c p \mapsto k_1$ then $w', k_2 \vdash_c p \mapsto k_1$. The opposite implication from (3) to (2) is trivial, since $w \quad w$. For the implication from (3) to (4), suppose that (3) holds and (4) does not. Then for some world w' and key k we have $w', k \vdash_c E_w$ and $w', k \vdash_c \neg(k_2\text{'s } p \mapsto k_1)$. The latter implies $w', k_2 \vdash_c \neg(p \mapsto k_1)$. Since $w', k \vdash_c E_w$, it follows that $w' \quad w$. Thus, by (3), $w', k_2 \vdash_c p \mapsto k_1$, contradicting our assumption. The implication from (4) to (3) is immediate, since $w', k_2 \vdash_c E_w$ for all $w' \quad w$. Finally, the equivalence between (4) and (5) is just a special case of Theorem 3.5. ∎

Proposition 5.1: If M represents w and l then for all principal expressions p and $x, y \in K \cup G \cup N$ we have $M \models \tau_{x,y}(p)$ iff $x, y \in K$ and $w, l, x \models p \mapsto y$.

Proof: By a straightforward induction on the structure of p. The base cases, where $p \in K \cup G \cup N$, are immediate from the definition of "represents" and the semantics of the logic. The inductive case, where $p = q$'s r, is immediate from the semantics and the definition of the translation. ∎

Theorem 5.2: The minimal Herbrand model M_w of Φ_w represents w and l_w.

Proof: (Sketch) The proof proceeds by showing a direct correspondence between the construction of the minimal Herbrand model of Φ_w and the fixpoint construction of l_w.

The theory of logic programming [Llo87] associates with the Horn theory Φ_w an operator Θ_w on the space of Herbrand models on the vocabulary V, defined by $name(x, y, z) \in \Theta_w(M)$ if there exists a substitution instance of a formula in Φ_w of the form $B \Rightarrow name(x, y, z)$ such that $M \models B$. The least Herbrand model M_w of Φ_w is then equal to $\Theta_w \uparrow \omega = \bigcup_{m \in N} \Theta_w \uparrow m$, where $\Theta_w \uparrow 0 = \emptyset$ and $\Theta_w \uparrow m + 1 = \Theta_w(\Theta_w \uparrow m)$ for $m \geq 0$.

Let T_w be the operator on local name assignments defined in the proof of Theorem 3.1. Using Proposition 5.1 to handle the rules in Φ_w corresponding to certificates, we may then show by a straightforward induction on m that for all $m \geq 1$, the Herbrand model $\Theta \uparrow m$ represents the world w and the local name assignment $T_w \uparrow m$. It follows that $M_w = \Theta \uparrow \omega$ represents $l_w = T_w \uparrow \omega$. ∎

Theorem 6.1: AX^{self}_{inf} (resp., AX^{self}_{n}) is a sound and complete axiomatization of $LLNC^s$ with respect to the open semantics if K is infinite (resp., K is finite).

Proof: The argument is very similar to that in the proof of Theorem 3.2. First suppose that K is infinite.

We add the following clauses to the definition of P:

6. $Self \in P$,

7. if $n \in P$ is a local name then $Self$'s $n \in P$.

We also add the following clauses to the definition of S^+, corresponding to the new axioms for $Self$.

(C1SP) if $Self$'s $p \in P$ then $Self$'s $p \mapsto p \in S^+$ and $p \mapsto Self$'s $p \in S^+$,

(C1PS) if p's $Self \in P$ then p's $Self \mapsto p \in S^+$ and $p \mapsto p$'s $Self \in S^+$,

(C1SE) if $Self \mapsto p \in S^+$ and p's $k \mapsto k \in S^+$ then $p \mapsto Self \in S^+$.

Lemma A.5 still applies. The definitions following this lemma, up to and including that of S are unchanged. However, the construction of the model changes slightly. We no longer use k_0 to represent the "current principal", instead, we use the key k that the construction associates with $Self$. This could be either a key in P_1 or one of the keys k_c for $c \in O$, depending on whether Self is key-equivalent or open. Note that we cannot have Self empty (thanks to the Identity axiom). If Self is key-equivalent, then by (C1KD) it is equivalent to at most one key $k \in P$. In this case, we define $k = k$. If Self is open we define k to be k_c, where $c = [\![\mathtt{Self}]\!]$.

We now define w and l exactly as before, except that we now set $l(k_0; n) = \emptyset$, since we no longer use k_0 as the "current principal." The following lemma is the analogue of Lemma A.7.

Lemma A.11: For all expressions $p \in P$, we have $[\![p]\!]_{w;l;k} = I(p)$.

Proof: The proof is very similar to that of Lemma A.7; we just describe the modifications required. The base cases for p a global name or a key are identical.

When $p = n$ is a local name, we proceed as follows. There are two possibilities, depending on whether $k \in P$ or not. Suppose first that $k \in P$. Then we have k Self and, by (C1LM) and (C1SP), k's n Self's n n. It then follows by (C1T) and construction of l that $n \mapsto k \in S$ iff k's $n \mapsto k \in S$ iff $k \in l(k; n)$, as required.

If $k = k_c$ for c an open class, we proceed as follows. If $k \in I(n)$, then we consider two cases, depending on whether $k \in P_1$. If $k \in P_1$, then $n \mapsto k \in S^+$ and it follows that Self's $n \mapsto k$ by (C1SP) and (C1T). Since Self Self it is immediate that $k \in [\![p]\!]_{w;l;k}$. Alternatively, if $k = k_d$, for $d \in O$, then we have $n \mapsto q \in S^+$ for some $q \in d$. By (C1SP) and (C1T) it follows that Self's $n \mapsto q \in S^+$, hence Self's $n \mapsto k \in S$. As before, this implies that $k \in [\![n]\!]_{w;l;k}$.

For the opposite inclusion, suppose that $k \in [\![n]\!]_{w;l;k}$. Since we are assuming that Self is open, there must be some q Self such that q's $n \mapsto k \in S$. By (C1LM), we have Self's $n \mapsto q$'s $n \in S^+$. It follows using (C1T) that Self's $n \mapsto k \in S$, hence $n \mapsto k \in S$. This completes the argument for the base case of n a local name.

There is now an additional base case for $p =$ Self. Here, note that $[\![\mathtt{Self}]\!]_{w;l;k} = \{k\}$. We therefore need to show that Self $\mapsto k \in S$ iff $k = k$. When $k \in P_1$, we have Self k, so Self $\mapsto k \in S$ iff $k \mapsto k$, and the claim follows by (C1KD) and (C1T) as in the base case for keys. The alternative is that $k = k_c$ for $c = [\![\mathtt{Self}]\!] \in O$. Since have Self $\mapsto k_c \in S$ by construction of S, it remains to prove that if Self $\mapsto k \in S$ then $k = k_c$. Now we cannot have Self $\mapsto k \in S$ for $k \in P_1$, for then by the argument above that Self is nonempty and (C1SE), we have $k \mapsto$ Self $\in S^+$, contradicting the assumption that c is open. Thus, we must have $k = k_d$ for some $d \in O$. In this case, there exists $q \in d$ such that Self $\mapsto q \in S^+$. Since d is open, we have q's $k_0 \mapsto k_0 \in S^+$, hence $q \mapsto$ Self $\in S^+$ by (C1SE). Thus, Self q, and it follows that $d = c$, hence $k = k$ as required. This completes the argument for the base case where $p =$ Self.

The inductive case is exactly as before, except that we need to consider the new case p's Self. Here, we note that $[\![p\text{'s Self}]\!]_{w;l;k} = [\![p]\!]_{w;l;k}$. Thus, by the induction hypothesis, we are required to prove that $p \mapsto k \in S$ iff p's Self $\mapsto k \in S$. This follows using (C1PS) and (C1T). ∎

The remainder of the proof in the case that K is infinite proceeds as before, using k in place of k_0.

If K is finite, the proof is even closer to that for the logic without Self. As sketched in the main text, because S is consistent, it follows from Identity, Witnesses, and Self-is-key that there must be some key $k \in K$ such that $\mathit{Self} \mapsto k \in S$. For this key k, we must have k's $n \mapsto k \in S$ iff $n \mapsto k \in S$. Thus, k plays the role of k_0 in the earlier argument. (Note that we now no longer need CurrentPrincipal to ensure the existence of k_0.) The rest of the argument is unchanged.) ∎

Acknowledgments

Work on this paper was done while the second author was with the School of Computing Sciences, University of Technology, Sydney. This work was supported in part by NSF under grant IRI-96-25901 and by a UTS internal research grant. A preliminary version of this paper appeared in the Proceedings of the 12th IEEE Computer Security Foundations Workshop, 1999, pp. 111–122.

References

[Aba98] M. Abadi. On SDSI's linked local name spaces. Journal of Computer Security, 6(1–2):3–21, 1998.

[ABLP93] M. Abadi, M. Burrows, B. Lampson, and G. D. Plotkin. A calculus for access control in distributed systems. ACM Transactions on Programming Languages and Systems, 15(4):706–734, 1993.

[BFL96] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In Proceedings 1996 IEEE Symposium on Security and Privacy, pages 164–173, 1996.

[Bir67] G. Birkhoff. Lattice Theory. American Mathematical Society, Providence, R.I., 3rd edition edition, 1967.

[EK76] M. H. van Emden and R. A. Kowalski. The semantics of predicate logic as a programming language. Journal of the ACM, 23(4):733–742, 1976.

[FHMV95] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. Reasoning about Knowledge. MIT Press, Cambridge, Mass., 1995.

[GH93] A. J. Grove and J. Y. Halpern. Naming and identity in propositional logics, Part I: the propositional case. Journal of Logic and Computation, 3(4):345–378, 1993.

[Gro98] SPKI Working Group. Simple public key infrastructure, internet draft. at http://www.ietf.org/html.charters/spki-charter.html, 1998.

[HM90] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. Journal of the ACM, 37(3):549–587, 1990. A preliminary version appeared in Proc. 3rd ACM Symposium on Principles of Distributed Computing, 1984.

[HvdM99] J. Y. Halpern and R. van der Meyden. Adding revocation and timestamps to a logic for sdsi's linked local name spaces. unpublished manuscript, 1999.

[HvdMS99] J.Y. Halpern, R. van der Meyden, and F. Schneider. Logical foundations for trust management. manuscript, 1999.

[LABW92] B. Lampson, M. Abadi, M. Burrows, and E. Wobber. Authentication in distributed systems: Theory and practice. ACM Transactions on Computer Systems, 10(4):265–310, 1992.

[Llo87] J. W. Lloyd. Foundations of Logic Programming. Springer-Verlag, Berlin, 2nd edition edition, 1987.

[LNS82] J.-L. Lassez, V. L. Nguyen, and E. A. Sonenberg. Fixed point theorems and semantics: a folk tale. Information Processing Letters, 14(3):112–116, 1982.

[RL96] R.L. Rivest and B. Lampson. SDSI — a simple distributed security infrastructure. at http://theory.lcs.mit.edu/ cis/sdsi.html, 1996.

[Ull88] J. D. Ullman. Principles of Database and Knowledge Base Systems, Volume I. Computer Science Press, 1988.

[Ull89] J. D. Ullman. Principles of Database and Knowledge Base Systems, Volume II: The New Technologies. Computer Science Press, 1989.