Reasoning about Naming and Time for Credential-based Systems*

Nathalie Chetcuti[1] and Fabio Massacci[2]

[1] Dep. de Informatique, Univ. Artois - France, chetcuti@cril.univ-artois.fr
[2] Dip. di Informatica e Telecomunicazioni, Univ. di Trento -Italy,
Fabio.Massacci@unitn.it

Abstract. Reasoning about trust management and credential-based systems such as SDSI/SPKI, is one of today's security challenges. The representation and reasoning problem for this (simple) public key infrastructure is challenging: we need to represent permissions, complex naming constructions ("Martinelli's office-mate is FAST's PC-Chair's Colleague"), intervals of time and metric time for expiration dates and validity intervals.

Such problem is only partly solved by current approaches. At first because they focus on Lamport and Rivest's SDSI and SPKI, the major goal being to show that the proposed logics and semantics captured exactly SPKI behavior or were better in this or that respect. Second, reasoning about time is missing. Complicated logics and algorithms are put in place for name resolution but it is always assumed that just the valid credentials are evaluated.

What we find missing is what Syverson termed an "independently motivated semantics". Here, we propose such a semantics with annexed logical calculi. The semantics has a natural intuitive interpretation and in particular can represent timing constraints, intersection of validity intervals and naming at the same time.

We also provide a logical calculus based on semantic tableaux with the appealing feature that the verification of credentials allows for the direct construction of a counter-model in the semantics when invalid requests are made. This combines semantic tableau method for modal and description logics with systems for reasoning about interval algebra with both qualitative and metric constraints.

1 Introduction

The security of credential-based systems is one of today's security challenges. This is mostly due to the disappearance of the traditional model of client/server interaction and its replacement by Service Oriented computing [**?**]: the important data is on some server which knows the clients and let them just have what they deserve. First of all, clients are no longer known by servers: the entire idea behind web services is that requests may come from everybody, provided they have the right credentials . Second, servers themselves are often distributed and their security policies may come from different sources and different administrative domains.

* F. Massacci has been partly supported by projects MIUR-FIRB "ASTRO" and EU IST-FET "WASP" and N. Chetcuti by a post-doc travel scholarship of UPS-Toulouse.

The traditional authentication and authorization questions have been transformed into another one about *trust management*: “Does the set of credentials about identifiers and about permissions proves that the request complies with the set of local policies?”

To perform these tasks without a centralized security infrastructure, a number of proposals have been put forward by security researchers (see the recent survey by Weeks [**?**]). One of the most cited work is Lampson and Rivest’s SDSI (Simple Distributed Security Infrastructure), later refined into an Internet RFC as SPKI (Simple Public Key Infrastructure) [**?**]. Many later proposals such as Binder [**?**] build upon the intuition of those works.

Loosely speaking, the appeal of SDSI/SPKI is to have distilled the concept of *local name* and to have reduced the traditional access and authorization decision into a problem of verifying the combination of *credentials linking local names and global names* and *credentials linking names and permissions*. For example, in FAST’s Chair, the Chair is a local name, which maps to an individual who is likely to be different from CSFW’s Chair. The individual standing for Chair may be linked by some certificate to an individual named Dimitrakos. Dimitrakos’ Colleague may also be mapped into more than one individual. Suppose now that any Dimitrakos’ Colleague was granted access to Martinelli’s Computer, should a claim from an FAST’s Chair’s Colleague be granted by the server? The reasoning is further complicated by time: one can be PC-chair or colleague for an interval of time and then something may change. Should the claim be granted in 2004?

The SDSI/SPKI proposal has been the subject of an intense debate and a number of researchers have formalized this proposal or its alternatives using logics to analyze and emphasize differences or subtle features. Abadi [**?**] has used a modal logic, which later Howell and Kotz [**?**] have modified, Halpern and Van der Meyden [**?**,**?**] have proposed another modal logic to reason about it, Jha, Reps and Stubblebine have used model checking for the verification for the time-free fragment of the logic [**?**]. Li et al. [**?**,**?**] have used Datalog.

So far, the approaches in Computer Security fora have focussed on using logics for giving “the” semantics to the operational description of SDSI/SPKI [**?**]. This has a number of drawbacks. For sake of example, to match the certificate treatment of SPKI, Halper and van der Meyden semantics [**?**, Sec. 3.2] is self-referential: a certificate is valid in a model because it is in the list of valid certificates. The English wording looks almost unacceptable though this is the exact description of the logical definition in the paper. This mixes syntax (the presence of a string in a set) and semantics (validity of the attribution of a key to a principal). The list of “valid” certificates describes what is in a set. In the world, the certificate may no longer describe the situation (for instance the key has been cracked and CRLs may not have been issued yet): syntactic would-be-valid certificates could well not be semantically valid.

A second limitation is the (missing) ability of reasoning about time. Obviously, for reasoning about validity interval one could simply apply the SPKI algorithm. But the same reasoning applies to any 40 pages paper describing a logic for the 4 pages name resolution algorithm. Time is the most intriguing aspect of certificates, in particular if we have name attribution certificates with validity periods that differs from validity periods of privileges attribution certificates. So, if Fabio’s co-author is Nathalie since

19/Jan/02, and Fabio's co-author can write chet-mass-03.tex up to 12/Jul/03 this is equivalent to say that the model could (semantically) satisfy also a certificate stating that Nathalie can write the paper from 19/Jan/02 to 12/Jul/03. The certificate may well not be in the list of valid certificates, as nobody might have actually issued it, but is nevertheless "semantically valid" as it would describe the status of the world. Even more sophisticated models such as Jha, Reps, and Stubblebine general authorization problems [**?**] cannot reason about such phenomena as their certificates are timeless.

1.1 Our Contribution

Building upon previous formalizations by Abadi, Halpern and van der Meyden, Jha and Reps we provide a general model for reasoning about naming and identifiers, authorization, credentials, *and* time. We hope that this would provide the equivalent of what Syverson termed an "independently motivated semantics" [**?**]. Last but not least, we provide a method to reason about them. Nobody is interested in formalizing the SDSI/SPKI FAST's PC-Chair's ConfMan policies if we cannot then decide whether the remote user making a request signed with the key 0xF34567 is allowed to see the reviews of paper ES0345.pdf. For the naming and modal part we need to combine features for advanced work in modal and dynamic logics [**?**,**?**]. For the temporal part, as we shall see, this is a challenge where a CSP-based qualitative reasoning proper of Allen's Interval algebra [**?**] is not sufficient. TCPs (Temporal CSPs) and STPs (Simple Temporal Problem) are necessary to handle metric temporal relations [**?**].

In the rest of the paper we sketch the intuitions about SDSI/SPKI, we show the semantical model for credential based systems, and the intuitions behind it. We give a sound and complete calculus and conclude the paper with a brief discussion.

2 A Primer on SDSI/SPKI

The idea behind SDSI/SPKI [**?**] is that servers make access control decision by looking at public key credentials which either link identifiers to known roles or other identifiers or link identifiers with privileges.

Each principal has its set of *local names*[1], denoted by n, possibly with subscript. Name can be composed so that for Lamport, the local name Ron may map into Rivest and the name Ron′Buddy map into what Rivest consider a buddy. In SPKI, *Compound names* are denoted by the tuple construct `(name n1 n2 ...nk)` or by the equivalent expression $n'n_1'\ldots n_k$ where each $n-i$ is a local name, and n is either a local name or a public key. When n is a key we have a *fully qualified name*. The interpretation of a compound name depends on the agent except for fully-qualified names. So that the interpretation of Ron′poker-buddies by one agent depends on its interpretation of Ron, and may be different from another agent's interpretation of Ron and Ron′poker-buddies.

Strictly speaking, we have no agents interpeting names in SPKI but just keys. We may say that Ron's interpretation by the agent Lamport is mapped into the agent Rivest,

[1] In absence of a centralized naming authority there are no global names in SPKI whereas SDSI had global names such as DNS.

but what we can only say in practice is that Ron is mapped into the public key k_r. So that the public key k_l (which corresponds to what we call the agent Lamport) takes any credential verifiable with the public key k_r as a statement coming from his fellow Ron.

SPKI has other kinds of principals such as hashes of keys and threshold subject ("any m out of N of the following subjects") for joint signatures, or the reserved word "Self", representing the entity doing the verification. Here, along the same line of Jha, Reps and Stubblebine [**?**], we only consider compound names.

Credentials are represented by certificates. There are various types of certificates in SPKI: naming certificates, authorization certificates, and certificate revocation lists (CRLs). Here, we only treat the first two but the framework is designed to give a reasonable account of revocation list.

A *naming certificate* has the form of a cryptographically signed message with contents `(cert (issuer (name k n)) (subject p) valid)`, where `k` is a key (representing the issuer, whose signature is on the certificate), `n` is a local name, `p` is a fully-qualified name, and `valid` is an optional section describing temporal validity constraints on the certificate. The valid section describes an interval during which the certificate is valid, expressed by means of a `(not-before date)` and/or a `(not-after date)`. It may also include additional tests for the validation of certificates.

If `k` is a key, `p` is a fully-qualified name, `A` is an authorization (loosely speaking a set of actions), and `valid` is a temporal validity section then an *authorization certificate* is `(cert (issuer k) (subject p) (propagate) A valid)`. Intuitively, the issuer uses such a certificate to grant `p` the authority to perform the actions in `A`. Moreover, if the optional propagation field is present, then the subject is further authorized to delegate this authority to others[2].

The logical syntax is based on the proposals by Abadi [**?**], Halpern and van der Meyden [**?**,**?**], Jha, Reps et al. [**?**,**?**]. We have a set of actions A, a set of keys K, and a set of names N. A *principal* can be a key $k \in K$, a local name n where $n \in N$, or $p\,'\,q$ where p and q are principals.

The *atomic formulae* of our logic are $p \mapsto q$ and p perm a, where p and q are principals and a is an action. More complex formulae are built by the usual operators of negation, conjunction and disjunction. The intuition behind $p \mapsto q$, which is read "p speaks for q", is that for the current principal any authorization for p can be mapped into an authorization for q.

We have two forms of *certificates*, $\mathsf{cert}_k\,(p \mapsto q\,[t_b, t_e])$ to associate names to other names and $\mathsf{cert}_k\,(p\ \mathsf{perm}\ a\,[t_b, t_e])$ to associates permissions to names. Here, we allow one to use compound names as subjects and not just local names. The interpretation of $\mathsf{cert}_k\,(n \mapsto p\,[t_b, t_e])$ is that for the principal k the local name n is bound to the fully qualified name p for the validity period between the instant t_b and t_e. The interpretation of $\mathsf{cert}_k\,(p\ \mathsf{perm}\ a\,[t_b, t_e])$ is that k permits action a to p for the validity period of the certificate. It is possible to have also open ended certificates. Delegation can be treated by generalizing permission certificates replacing action a with recursive permissions.

[2] This unbounded delegation of powers has been mitigated by other authors. For instance, Li et al. [**?**,**?**] introduce the notion of delegation up to n steps.

With respect to the calculus in [**?**,**?**] we have eliminated the $says$ operator because it is subsumed by the certificate operator. See [**?**] for an automated reasoning method for some fragments of Abadi's *et al.* calculus.

3 Semantics

The motto of any credential-based security religion could be "Extra public-keys nulla salus" and we build upon this intuition by making a model where the basic domain is a set of keys, and where names connect keys with each other and with permissions.

Let's first give a model *without* time. So a *model* is a generalized Kripke structure that is a triple $\langle \mathbf{K}, \mathbf{LN}, \mathbf{A} \rangle$ where $\mathbf{K}$ is a set of real keys (such that for all $k \in K$ there is a $\mathbf{k} \in \mathbf{K}$ plus possibly additional keys). The naming relation $\mathbf{LN}$ is an indexed family of partial mappings from keys to subset of keys $N \longrightarrow \mathbf{K} \rightarrow 2^{\mathbf{K}}$. The grant relation $\mathbf{A}$ is a function mapping actions into mappings of keys into subset of keys $\mathbf{A} \longrightarrow \mathbf{K} \rightarrow 2^{\mathbf{K}}$. In the sequel we always use X as the syntactic value and $\mathbf{X}$ as the semantic counterpart.

The intuition behind the set $\mathbf{K}$ should be obvious. The naming relation associates to each local name a mapping: who uses this name for whom. So $\mathbf{k}' \in \mathbf{LN}_n\mathbf{k}$ means that the principal associated to key $\mathbf{k}$ associates to the name n at least the key $\mathbf{k}'$. We have a set because the same name n may refer to many individuals as in $Ron\,'\,poker - buddies$. Furthermore note that the mapping may be partial because a principal may not associate anybody to a name. In the sequel, for simplicity we use the relation-oriented notation to describe the mapping: $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{LN}_n$.

The grant relation associates to each key the set of actions that the agent holding the key is willing to permit to other principals. So $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{A}(a)$ means that the principal associated to the key $\mathbf{k}$ is willing to permit action a to $\mathbf{k}'$.

Names can be lifted to compound names by giving a semantics for the $'$ operator:

- $\mathbf{N}(n) = \mathbf{LN}(n)$ where $n \in N$
- $\mathbf{N}(k) = \bigcup_{\mathbf{k}' \in \mathbf{K}} \{\langle \mathbf{k}', \mathbf{k} \rangle\}$ where $k \in K$
- $\mathbf{N}(p\,'\,q) = \bigcup_{\mathbf{k_p} \in \mathbf{K}} \{\langle \mathbf{k}, \mathbf{k_{pq}} \rangle \mid \langle \mathbf{k}, \mathbf{k_p} \rangle \in \mathbf{N}_p \text{ and } \langle \mathbf{k_p}, \mathbf{k_{pq}} \rangle \in \mathbf{N}_q\}$

The second rule says that syntactic keys are mapped into the corresponding semantics keys. The last rule is just composition: if $\mathbf{k}$ associate $\mathbf{k_p}$ to the name p and $\mathbf{k_p}$ associates $\mathbf{k_{pq}}$ to the name q then $\mathbf{k}$ should associate $\mathbf{k_{pq}}$ to the name $p\,'\,q$.

Proposition 1. *A principal $p_0\,'\,\ldots\,'\,p_l$ is equivalent to some principal $q_0\,'\,\ldots\,'\,q_m$ where q_0 is either a key or a name and $q_1, \ldots, q_m$ are names.*

So from now on we shall consider only principals in the latter form.

Figure **??** shows an example of a Kripke structure for credential systems without time. It describes the relation between two co-authors and a PC chair. Keys are represented as nodes of the graph and the permissions to read or write the paper labels each node. The key k_1 corresponds to Fabio Massacci, whereas the key k_3 is Fabio Martinelli which is $k_3\,'\,\mathsf{FAST\text{-}03}\,'\,\mathsf{co\text{-}chair}$. Note how the name Fabio is mapped into different keys.

To lift our structure to time we introduce the concept of a *trace* $\mathcal{M}$ that is a mapping from time to models. A trace associates to each instant of time a given model $\mathcal{M}@t =$

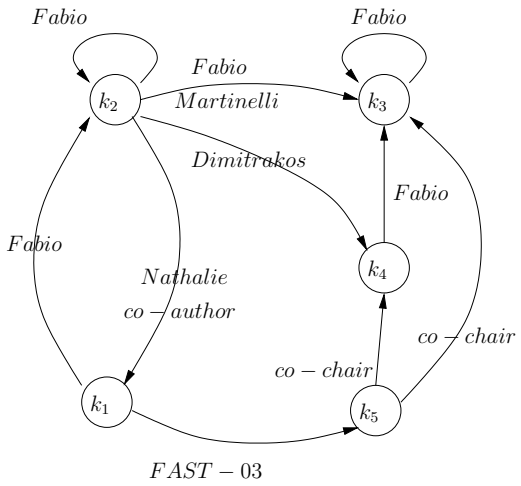

Fig. 1. Timeless Model

$\langle \mathbf{K}@t, \mathbf{LN}@t, \mathbf{A}@t \rangle$ where $t \in \mathbb{R}$. Then $\mathbf{K}@t : \mathbb{R} \longrightarrow \mathbf{K}$ is the set of real keys which are associated to the named keys at time t, $\mathbf{LN}@t$ is the local naming relation with the additional time parameter $\mathbb{R} \times N \longrightarrow \mathbf{K} \rightarrow 2^{\mathbf{K}}$ at time t and $\mathbf{A}@t$ is the grant relation: $\mathbb{R} \times A \longrightarrow \mathbf{K} \rightarrow 2^{\mathbf{K}}$.

The extension of the semantics to compound names *principals* is identical, except for the t subscript. In the sequel, we merge the principal and the time in the same subscript. So we write $\mathbf{N}_{p@t}$ instead of $\mathbf{N}@t(p)$, and similarly for other operators.

Intuitively, we can view the same formal structure from two different perspectives. At first glance, we have a timeline and at each time the entire world may change. This is hardly appealing: we are used that the world stays the same and something in it changes. The second perspective is more appealing: the model is composed by parts (keys, naming relations, and permission relations) that change over time.

In Figure **??** we show a timed version of the model shown in Figure **??** in which we have added the validity interval of each naming relation.

The timed model allows for fine grained distinctions that are not possible in the untimed model. If we consider key security properties like dynamic separation of duties, the timed models allows for a natural description of such constraints and situations.

For instance in the untimed model k_3 is always k_1 ′ FAST ′ Co-chair, and it is at the same time k_2 ′ Fabio and k_2 ′ Martinelli. In the timed modelthe concatenation of the validity periods is such that k_1 ′ FAST ′ Co-chair only maps into k_3 for the period $[7/03, 9/03]$ which is by far shorter then each validity period of the component naming relations. Equally, k_3 has never been at the same time k_2 ′ Martinelli and k_2 ′ Fabio (as in the untimed model), but either only one or the other. Thus k_3 has never got the permissions associated with both identifiers.

It is possible to add more security constraints on the model. For instance, should we have persistence of global keys, i.e. should $\mathbf{K}@t = \mathbf{K}@t'$ hold? Basically this says that

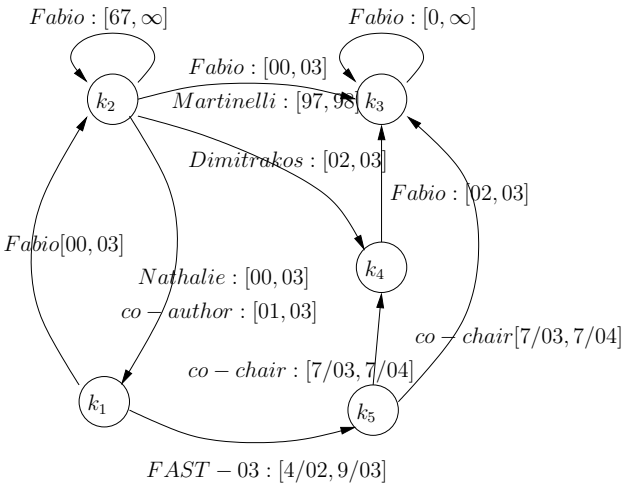

Fig. 2. Timed Model

all keys have been already invented, just not used. Another possibility is that the only possible keys are those listed in a given set of syntactic certificates, and thus $\mathbf{K}@t = K$ for all t.

Another question is whether the validity period of keys should always be a connected interval. Suppose that we have $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{N}_{\mathsf{FAST\,'\,Co\text{-}chair}@12/07/03}$, that is the real key $\mathbf{k}$ associates the key $\mathbf{k}'$ to the FAST ' Co-chair. After an year the certificate expires and we have $\langle \mathbf{k}, \mathbf{k}' \rangle \notin \mathbf{N}_{\mathsf{FAST\,'\,Co\text{-}chair}@12/07/04}$. Do we want to impose that for all $t \geq 11/07/2004$ we have $\langle \mathbf{k}, \mathbf{k}' \rangle \notin \mathbf{N}_{@tFAST\,'\,\mathsf{Co\text{-}chair}}$? If the answer is yes, this means that after a certificate is expired we would not accept another re-validation certificate for the same key. This is one possible scenario and there might be cases when this is desirable and cases when it is not.

All such possibilities can be captured with suitable axiom schemata. Different certificate theories (for instance persistence-of-syntactic-keys, no-revalidation) can be characterized by different axiom schemata.

We have now all the necessary material to give a semantics to *formulae*.

- $\mathcal{M}@t, \mathbf{k} \models p \mapsto q$ iff for all $\mathbf{k}' \in \mathbf{K}@t$, if $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{N}_{q@t}$ then $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{N}_{p@t}$
- $\mathcal{M}@t, \mathbf{k} \models p \ \mathsf{perm}\ a$ iff $\langle \mathbf{k}, \mathbf{k}' \rangle \in \mathbf{N}_{p@t}$ implies $a \in \mathbf{A}@t(\mathbf{k}')$

As for *certificates*, we evaluate them as follows:

- $\mathcal{M}@t \models \mathsf{cert}_k\,(p \mapsto q\ [t_b, t_e])$ if $t \in [t_b, t_e]$ implies $\mathcal{M}@t, \mathbf{k} \models p \mapsto q$
- $\mathcal{M}@t \models \mathsf{cert}_k\,(p\ \mathsf{perm}\ a\ [t_b, t_e])$ if $t \in [t_b, t_e]$ implies $\mathcal{M}@t, \mathbf{k} \models p\ \mathsf{perm}\ a$

Remark 1. Comparing with Halpern and van der Meyden proposals we have no syntactic list of valid certificates. A model is an independent entity from certificates. It has its properties and satisfies some formulae. If it satisfies the appropriate formulae, then

it also satisfies some certificates. In this way, as we said, a particular certificate theory characterizes a particular set of models.

Another characteristics of our model is that we worry about validity of certificates in model only if they refer to the current time. If a certificate is not at all applicable there is little interested in knowing whether it describes the current state of the world or not.

Then we can define the notion of *satisfaction by a trace* $\mathcal{M}$, that is the notion of satisfaction for all time instant.

– $\mathcal{M} \models \mathsf{cert}_k\,(c\,[t_b, t_e])$ iff $\forall t$, $\mathcal{M}@t \models \mathsf{cert}_k\,(c\,[t_b, t_e])$

As Jha and Reps [**?**,**?**] we have a notion of consequence for chains of certificates:

Definition 1. *A boolean combinations of certificates* χ *is a* logical consequence *of a set of certificates* $\mathcal{C}$ *if any trace which satisfies all certificates* $\mathcal{C}$ *also satisfies* χ.

If χ is a single certificates then we have exactly Jha and Reps notion of consequence. This generalized notions is more useful for deriving interesting consequences. For instance we can ask whether an invalid certificate implies that another certificate is also implicitly invalidated.

4 Semantic Tableaux

Once the model is in place, how do we know that a certificate is a logical consequence of a a set of physical certificates? Furthermore, if this is not the case, how can we get a counter-example?

We propose a calculus based on *semantic tableaux*. Intuitively, to prove that a certificate χ is a logical consequence of a set of other certificates $\mathcal{C}$ (see Def. **??**) we instead try to construct a model that falsifies χ and satisfies $\mathcal{C}$. If we succeed, then we have the counter example. If we fail and we used a fair and systematic procedure, we are sure that no such countermodel exists and the formula is valid.

If we drop the timed information, our calculus could be a tableaux sibling of the model checking procedure of Jha and Reps. The additional dificulty is that for each pair of constraints on the attempted counter-model we must check that

– either there is no temporal interaction and then we simply impose that the temporal constraints are not overlapping,
– or we update the untimed information with the required constraints during the overlapping validity intervals.

The construction starts from the formulae and then try to build the entire model and the trace by incrementally constructing the naming associations, by attributing permission and by determining temporal informations.

For tableaux we shall use the usual terminology. For instance, see De Giacomo and Massacci [**?**] for the naming and permission part and Kautz and Ladkin [**?**] for the temporal part. So a *tableau* is a collection of branches, each intuitively corresponding to some potential counter-model. A *branch* has three components for *qualitative*

information (such as naming relations), for *qualitative temporal information* and for *quantitative temporal information*.

For the naming and permission information we have a triple $\langle(\mathsf{K}), \mathsf{N}, (\mathsf{F}, \mathsf{A})\rangle$ where

- K are the syntactic keys plus possibly some new keys
- a function $\mathsf{N} : N \longrightarrow 2^{\mathsf{K}\times\mathsf{K}}$ are the naming relation constructed so far,
- a function $\mathsf{A} : A \longrightarrow 2^{\mathsf{K}\times\mathsf{K}}$ are the permissions assigned so far
- $\mathsf{F} : \mathsf{K} \longrightarrow 2^{Fml}$ are the formulae (labelled with validity intervals) which we try to satisfy

If no contradiction is found at the end of the construction then the countermodel would be simply $\mathbf{K} := \mathsf{K}$, $\mathbf{LN} := \mathsf{N}$ and $\mathbf{A} := \mathsf{A}$.

For the qualitative temporal information about validity of certificates we have an *interval network* $\langle TI, \mathsf{E}\rangle$ where

- TI is a set of variables representing temporal intervals
- a function $\mathsf{E} : TI \times TI \longrightarrow 2^{Allen's\ relations}$ corresponds to the qualitative temporal relations that we have forced so far

If v is an interval variable by we represent its beginning and end point as v^- and v^+

Allen's interval relations are the following: before, after, meets, $\mathsf{met-by}$, $\mathsf{overlaps}$, $\mathsf{overlapped-by}$, starts, $\mathsf{started-by}$, during, $\mathsf{contains}$, $\mathsf{finishes}$, $\mathsf{finished-by}$, equals. Their interpretation is intuitive and we refer to Allen's work for additional explanations [**?**,**?**], and to Dechter and Meiri [**?**,**?**] for reasoning procedures. For instance v_1 meets v_2 means that when v_1 ends, v_2 starts.

For the metric temporal information we have a point network $\langle TP, \mathsf{E}_T\rangle$ where

- TP is a set of variables representing time points
- $\mathsf{E}_{TP} : TP \times TP \longrightarrow 2^{Intervals}$ represents the metric constraints between the time points that we contructed so far.

For example $E_{TP}(t, t') = \{[1, 5], [100, 201]\}$ means that t is distant by t' either for a value on the range $1 - 5$ or the range $100 - 201$. Using linear inequalities we would have $t + 1 \leq t' \leq t + 5$ or $t + 100 \leq t' \leq t + 201$

Initialization At the very start, K is the set of keys appearing in $\{\chi\} \cup \mathcal{C}$. The sets F, N, A, TI, and TP are empty.

For each certificate $\mathsf{cert}_k\ (c\ [t_b, t_e])$ in $\mathcal{C} \cup \{\chi\}$ a new interval variable v_c is added to TI and v_c^- and v_c^+ are added to TP. Then E, resp. E_{TP}, is enriched with the constraints existing between v_c, resp. v_c^- and v_c^+ , and the remaining interval, resp. point, variables. The intuition is that the interval variable v_c is used to define the validity period of the certificate.

Finally if χ, the certificate we are trying to disprove, has the form $\mathsf{cert}_{k_i}\ (c\ [t_{ib}, t_{ie}])$, we let $\mathsf{F}(k_i) \leftarrow \{\neg c : v_i\}$ where v_i is the interval corresponding to $[t_{ib}, t_{ie}]$. Intuitively, we want a model where the certificate is not valid in the given interval.

Remark 2. At this stage one may ask why do we need at all such validity intervals and such cumbersome notation of end points: we already know that a certificate spans an

interval of time by simply looking at it. This is actually correct except for one small but essential point: revocation certificates! The validity period is the maximal potential period in which a certificate *can* be valid. If certificates can be revoked, even if temporarily, we no longer knows what validity interval a certificate has just by looking at the certificate.

There is no space here to introduce revocation certificates but the machinery is necessary to scale up to a reasonable semantic account of revocation[3]. In particular the tricky bit is the validity period of revocation certificates. In a general model, there is no obligation[4] for revocation certificates to have validity period outlasting the validity period of the revoked certificate.

Suppose that cert_{k_3} (Dimitrakos′ Fabio $\mapsto$ Fabio $[1/1/02, 31/12/03]$) is revoked by a certificate whose validity interval is only $[1/1/03, 1/7/03]$. It is clear what happens in the interval $[1/1/02, 31/12/02]$: the speaks-for naming relation between Fabio and Dimitrakos′ Fabio is valid. It is also clear that in the period $[1/1/03, 1/7/03]$ this certificate is n longer valid[5]. What happens during the period $[2/7/03, 31/12/03]$ in which the revocation certificate is no longer valid? The natural semantical interpretation is that the original certificate is still valid.

4.1 Rules for Tableau Construction

After the initialization step the construction proceeds step-wise by the application of a rule and the checking of the consistency of the temporal information. It stops either when all the expressions were processed, or when an inconsistency is found.

We now consider the processing of atomic formulae only. The boolean operators are handled in the usual way: if a key must satisfy the conjunction of two formulae this means that the key must satisfy both formulae and thus both formulae are added to the branch at the appropriate key.

The only tricky bit for the read not familiar with tableau methods is the treatment of disjunction. Since disjunction means that either one or another formula must be satisfied we split the current branch in two. Formally this means that we duplicate every sets that we have constructed so far K, F, N, A, TI, and TP, and add one disjunct to one instance of F and the second disjunct to the second instance of F. Then the search continues by pushing one branch on the stack and by exploring the remaining branch. If the search for counter models is unsuccessful in the first branch, the search resumes the other alternative. Of course in any tableau implementation the structure is not duplicated and pointers are used.

To simplify the rules for formulae we need some abbreviations that describe possible relations between validity intervals of certificates.

Definition 2. *Let v, v' and v^* be interval variables.*

[3] Again in Halpern and Van der Meyden [**?**] we have a syntactic account of revocation, mostly because they have not considered time. In Jha et al, revocation is not treated.

[4] A particular certificate theory may instead impose such obligation. There might be axiom schemata forcing properties of revocation certificates. Such schemata would be mapped into conditions for inconsistency in our tableaux setting.

[5] The relation may still be valid is there is some other valid certificate.

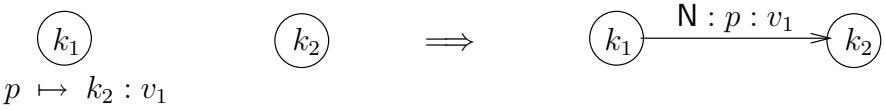

Fig. 3. Illustration of first rule for $\mapsto$ (the temporal graphs remain unchanged)

no overlapping validity: $v \cap v' = \emptyset$ *when the following constraint is satisfied:*

$$v\{\mathsf{before}, \mathsf{meets}, \mathsf{met}-\mathsf{by}, \mathsf{after}\}v'$$

overlapping validity: $v \cap v' \neq \emptyset$ *when the following constraint is satisfied:*

$$v \left\{ \begin{array}{c} \mathsf{starts}, \mathsf{started}-\mathsf{by}, \mathsf{finishes}, \mathsf{finished}-\mathsf{by}, \\ \mathsf{overlaps}, \mathsf{overlapped}-\mathsf{by}, \mathsf{during}, \mathsf{contains}, \mathsf{equals} \end{array} \right\} v'$$

containment: $v' \subseteq v$ *when the following constraint is satisfied:*

$$v'\{\mathsf{starts}, \mathsf{during}, \mathsf{finishes}, \mathsf{equals}\}v$$

The intuition of the first rule is that two intervals have no intersection if either one interval is before the other, or one interval just finishes when the other just starts.

Proposition 2. *To compute the overlap of validity periods we let* $v^* = v \cap v'$*. Then the following conditions hold (1) if* $v\{\mathsf{starts}, \mathsf{during}, \mathsf{finishes}, \mathsf{equals}\}v'$ *then* $v^* = v$*, (2) if* $v\{\mathsf{started}-\mathsf{by}, \mathsf{contains}, \mathsf{finished}-\mathsf{by}\}v'$ *then* $v^* = v'$*, (3) if* v $\mathsf{overlaps}$ v' *then* v^* $\mathsf{finishes}$ v *and* v^* starts v'*, (4) if* v $\mathsf{overlapped}-\mathsf{by}$ v' *then* v^* starts v *and* v^* $\mathsf{finishes}$ v'*.*

Certificate. The rule for certificates is the simplest: if $\mathsf{cert}_k\ (c\ [t_b, t_e]) \in \mathcal{C}$ then for the corresponding $v_c \in TI$, add $\{c : v_c\}$ to $\mathsf{F}(k)$. In words:if the certificate $\mathsf{cert}_k\ (c\ [t_b, t_e])$ is valid then the corresponding formula must be true for the corresponding key during the validity interval of the certificate.

Speaking for. The rules consider both positive and negative cases.

- if $p \mapsto k' : v \in \mathsf{F}(k)$ then add $\{p : v\}$ to $\mathsf{N}(k, k')$
- if $p \mapsto k'\,'\,q : v \in \mathsf{F}(k)$ then $\forall k^* \in \mathsf{K}$ such that $q : v' \in \mathsf{N}(k', k^*)$ either one has $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{p : v^*\}$ to $\mathsf{N}(k, k^*)$
- if $p \mapsto n\,'\,q : v \in \mathsf{F}(k)$ (where q can be null) then $\forall k' \in \mathsf{K}$ such that $n\,'\,q : v' \in \mathsf{N}(k, k')$ either one has $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{p : v^*\}$ to $\mathsf{N}(k, k')$
- if $\neg(p \mapsto q) : v \in \mathsf{F}(k)$ then for a new action a^* we add $\{p\ \mathsf{perm}\ a^*) : v, \neg(q\ \mathsf{perm}\ a^*) : v\}$ to $\mathsf{F}(k)$

The intuition behind the first rule is that if p is associated to k' for the key k then we add the labelling to the relation, tagged with the appropriate validity interval v. The graphical representation is shown in Figure **??**

The intuition behind the second rule is the following: suppose that you have a claim that for the key k the name p has been associated to the name $n\,'\,q$ for a certain validity

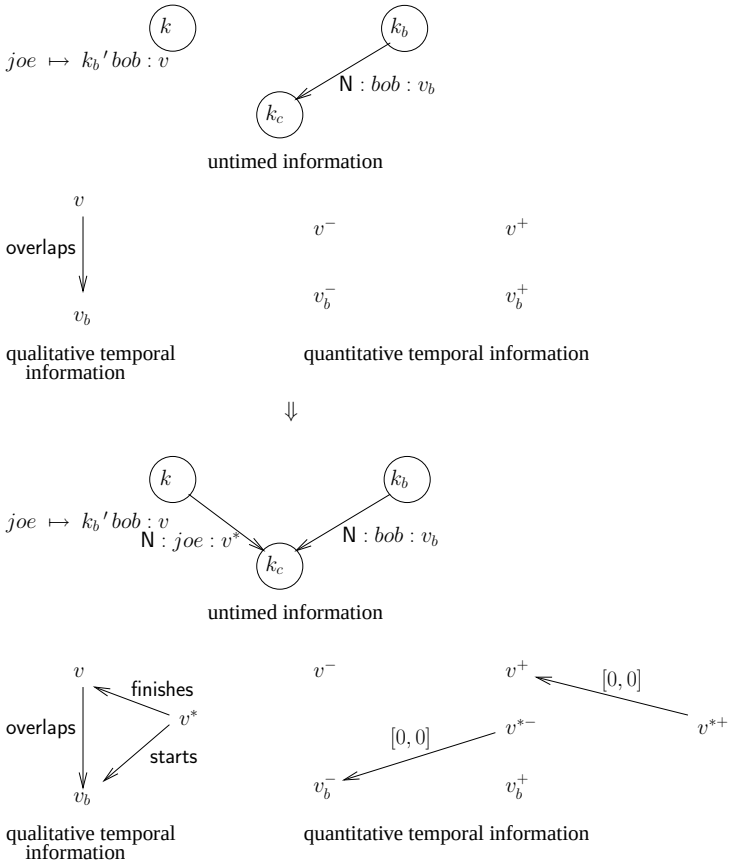

Fig. 4. Illustration of one of the rules for $\mapsto$ (where v overlaps v')

interval v. We can have a look at all naming relations between keys that have q as their name. These naming relations will also have their validity period, say v'. Now we have two possibilities. The first one is that the validity periods do not overlap (that is $v \cap v' = \emptyset$) and therefore there is nothing that we need to do. The second one is that the validity periods do overlap and then we must chain the two certificates for the overlapping periods, namely $v^* = v \cap v'$.

We illustrate the second rule in the special case where v overlaps v' in Figure **??**. As we can see from the figure we have added the link between k and k^* but only for the overlapping interval v^*. The temporal information says that when v finishes then v^* also finish and when v' starts then v^* also starts.

Permission. These rules have the same flavour of the speaks for rules, excpet that they add permitted actions to each key rather then connecting keys with a naming relation.

– if k' perm $a : v \in \mathsf{F}(k)$ then add $\{a : v\}$ to $\mathsf{A}(k, k')$

– if $k'\,'\,q$ perm $a : v \in \mathsf{F}(k)$ then $\forall k^* \in \mathsf{K}$ such that $q : v' \in \mathsf{N}(k', k^*)$ then either $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{a : v^*\}$ to $\mathsf{A}(k, k^*)$
– if $n\,'\,q$ perm $a : v \in \mathsf{F}(k)$ (where q can be null) then $\forall k' \in \mathsf{K}$ such that $n\,'\,q : v' \in \mathsf{N}(k, k')$ then either $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{a : v^*\}$ to $\mathsf{A}(k, k')$

We also have a rule for negated permissions.

– if $\neg k'$ perm $a : v \in \mathsf{F}(k)$ then for a new interval $v^* \subseteq v$, add $\{\neg a : v^*\}$ to $\mathsf{A}(k, k')$
– if $\neg k'\,'\,q$ perm $a : v \in \mathsf{F}(k)$ then for a new $k^* \in \mathsf{K}$ and a new interval $v^* \subseteq v$, set $\mathsf{N}(k', k^*)$ to $\{q : v^*\}$ and $\mathsf{A}(k, k^*)$ to $\{\neg a : v^*\}$
– if $\neg n\,'\,q$ perm $a : v \in \mathsf{F}(k)$ (where q can be null) then for a new $k^* \in \mathsf{K}$ and a new interval $v^* \subseteq v$, set $\mathsf{N}(k, k^*)$ to $\{n\,'\,q : v^*\}$ and $\mathsf{A}(k, k^*)$ to $\{\neg a : v^*\}$

Rules for principals. These rules refine the naming relations eliminating compound names when validity intervals allow to do that.

– if $k^*\,'\,q : v \in \mathsf{N}(k, k')$ then add $\{q : v\}$ to $\mathsf{N}(k^*, k')$
– if $n\,'\,q : v \in \mathsf{N}(k, k')$ then $\forall k^* \in \mathsf{K}$ such that $n : v' \in \mathsf{N}(k, k^*)$ either $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{q : v^*\}$ to $\mathsf{N}(k^*, k')$
– if $q : v \in \mathsf{N}(k, k')$ and $q' : v' \in \mathsf{N}(k', k^*)$ then either $v \cap v' = \emptyset$ or let $v^* = v \cap v'$ and add $\{n\,'\,q : v^*\}$ to $\mathsf{N}(k, k^*)$.

Rule for consistency. A principal associated to a key cannot be permitted and forbidden the same action during overlapping periods.

– if $a : v \in \mathsf{A}(k, k')$ and $\neg a : v' \in \mathsf{A}(k, k')$ then

$$\mathsf{E}(v, v') = \mathsf{E}(v, v') \cap \{\mathsf{before}, \mathsf{after}, \mathsf{meets}, \mathsf{met} - \mathsf{by}\}$$

Finally, we must guarantee the consistency of the qualitative and the metric temporal information. To this extent we need to solve the corresponding constraint network and if the empty interval is present then the original information is inconsistent [**?**]. This algorithm is sound but not complete unless the interval network is at least *preconvex* [**?**]. To get completeness of the processing of the temporal information, we modify slightly some rules: whenever we need to add $v \cap v' = \emptyset$ we actually split it into $v\{\mathsf{before}, \mathsf{meets}\}v'$ or $v\{\mathsf{met} - \mathsf{by}, \mathsf{after}\}v'$.

Definition 3. *A branch is* saturated *when no new information can appear through the application of a rule. A branch is* closed *if an inconsistency is found ; it is* open *if it is saturated and not closed. A tableau is closed when all its branches are closed, it is open if one of its branches is such.*

Theorem 1. *If a boolean combinations of certificates χ has a closed tableau given a set of certificates $\mathcal{C}$ iff χ is a logical consequence of $\mathcal{C}$.*

5 Discussion and Conclusions

In the security literature there has been a number of proposals for the right logical account for SDSI/SPKI features. Abadi [**?**] has used the DEC-SRC calculus for access control [**?**]. However a number of problems have been found in the DEC-SRC calculus by other authors [**?**,**?**]. Howell and Kotz [**?**] have proposed an alternative semantics, but their solution is logically rather awkward (for instance it is not closed under the usual boolean operators) and does not give a reasoning procedure. Halpern and Van der Meyden [**?**,**?**] have refined the semantics of Abadi and their proposal is the basic reference for all subsequent works (including this one). They have proposed two modal logics to reason about it but their proposal is fairly tailored on the SDSI/SPKI framework. Jha, Reps and others [**?**,**?**] have given a pushdown automata procedure for SPKI certificates but have only focused on the normal trust relationship assuming time interval fixed. this is the most comprehensive treatment, and it is a decision procedure for time-free fragment of Halpern and van der Meyden logic. Dropping time from our framework it is a sibling of our model. However, in all papers the treatment of time is either absent (Abadi, Howell and Kotz, Jha and Reps) or refers essentially to the SPKI algorithms (Halpern and Van der Meyden).

With respect to the trust management systems of Li et al. [**?**,**?**] we only have propositional rules. In contrast, Li et al. constructions based on logic programming and Datalog allows for quantification. Since term creating functions are absent, quantification is a just a more efficient and compact representation of the propositional version but does not introduce new possibilities. However, the semantics is just the term algebra of the Datalog programs representing a policy. It could be intriguing to have a independent semantics for the bounded delegation framework.

Conceptually, it is possible to lift our framework to first order reasoning over objects and it would be interesting to derive syntactic restrictions on quantification in certificates that would allow for the the same decidability results based on Datalog in Li el al [**?**]. The lifting of permissions can be done without difficulties: one could simply import techniques from first order modal logics. Basically, we could have certificates like $\mathsf{cert}_k\ (p\ \mathsf{perm}\ a(o)\ [t_b, t_e])$ when permission to perform action a is only granted on object o and $\mathsf{cert}_k\ (p\ \mathsf{perm}\ \lambda x.a(x)\ [t_b, t_e])$ when permission is granted on all objects.

The lifting of naming association is the tricky bit, as there is no simple security intuition on the meaning of "key k is associated to name n for the object o" (though it is possible to write formulae linking them). A convincing semantics could then be given only for a convincing security intuition of the above concept. Even looking at neighbor field, such as grammar logics or description logics we don't have anything remotely resembling such ternary relation.

Once this framework were lifted to first logic would then be interesting to analyze the relations between the two frameworks. In particular one could then try to derive syntactic restriction on the general model that would make possible to use the Datalog representation and inference engine. Building on this result one could then show the relation with other works on credential based systems that are based on logic and that cope with the proper aspect of trust negotiation (which is neglected in the original SPKI proposal) such as the work of Yu et al. [**?**] and Bonatti and Samarati [**?**]

An intriguing subject of future research is the usage of *symbolic validity intervals*. For instance, if we had $\mathsf{cert}_k\,(\mathsf{FAST'co\text{-}chair} \mapsto \mathsf{Martinelli}\,[12/07/04, \mathsf{NextFAST}])$ and then have symbolic constraints on intervals such as NextFAST is non-overlapping with 2004 and overlaps with part of 2002 and starts after [appointment-day,end-of-conference]. The calculus we have present can indeed cope with such constraints. The lifting of the SPKI framework to symbolic validity intervals looks promising for some distributed applications.

Building upon previous attempt of formalizations we provided a general model for reasoning about naming and identifiers, authorization, credentials, and time. We show how to construct a general reasoning method for the logic that combines advanced tableaux methods for modal and description logics [**?**] with systems for reasoning about the interval algebra by Allen [**?**] and advanced proposals that exploit both qualitative and metric constraints [**?**,**?**].

References

1. M. Abadi. On SDSI's Linked Local Name Spaces. *JCS*, 6(1-2):3–21, 1998.
2. M. Abadi, M. Burrows, B. Lampson, and G. D. Plotkin. A Calculus for Access Control in Distributed Systems. *TOPLAS*, 15(4):706–734, 1993.
3. J.F. Allen. An Interval-Based Representation of Temporal Knowledge. In *IJCAI'81*, 1981.
4. J.F. Allen. Maintaining Knowledge about Temporal Intervals. *CACM*, 26(11):832–844, 1983.
5. Piero Bonatti and Pierangela Samarati. Regulating service access and information release on the Web. In *ACM CCS-00*, pages 134–143. ACM Press, 2000.
6. J.-F. Condotta. The Augmented Interval and Rectangle Networks. In *KR'2000*, 2000.
7. G. De Giacomo and F. Massacci. Combining Deduction and Model Checking into Tableaux and Algorithms for Converse-PDL. *Inf. and Comp.*, 162(1–2):117–137, 2000.
8. R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Networks. *AIJ*, 49(1–3):61–95, 1991.
9. J. DeTreville. Binder, a logic based security language. In *IEEE S&P-2002*, 2002.
10. C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen. *SPKI Certificate Theory*, September 1999. IEFT RFC 2693.
11. D. Georgakopoulos and M. Papazouglou. Service-oriented computing. *CACM*, 2003. To appear.
12. J. Halpern and R. van der Meyden. A Logic for SDSI's Linked Local Name Spaces. *JCS*, 9(1/2):105–142, 2001.
13. J. Halpern and R. van der Meyden. A Logical Reconstruction of SPKI. In *CSFW'01*, 2001.
14. J. Howell and D. Kotz. A Formal Semantics for SPKI. In *ESORICS 2000*, 2000.
15. H. A. Kautz and P. B. Ladkin. Integrating Metric and Qualitative Temporal Reasoning. In *AAAI-91*, 1991.
16. N. Li. Local names in spki/sdsi. In *IEEE CSFW 2000*, pages 2–15, 2000.
17. F. Massacci. Tableaux Methods for Formal Verification of Multi-Agent Distributed Systems. *JLC*, 8(3):373–400, 1998.
18. I. Meiri. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. In *AAAI'91*, 1991.
19. J. Feigenbaum Ninghui Li, B. N. Grosof. A practically implementable and tractable delegation logic. In *IEEE S&P 2000*, pages 27–42, 2000. Full version appears in ACM TISSEC.
20. T. Reps S. Jha. Analysis of spki/sdsi certificates using model checking. In *IEEE CSFW-2002*, 2002.

21. S. Schwoon, S. Jha, T. Reps, and S. Stubblebine. On generalized authentication problems. Technical report, Univ. of Winsconsin, 2003. To appear in CSFW 2003.
22. Paul Syverson. The use of logic in the analysis of cryptographic protocols. In *IEEE S& P-91*, pages 156–170, 1991.
23. S. Weeks. Understanding trust management systems. In *IEEE S&P-2001*, 2001.
24. Ting Yu, Marianne Winslett, and Kent E. Seamons. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. *ACM TISSEC*, 6(1):1–42, 2003.