# A Semantics and a Calculi for Reasoning about Credential-based Systems

Nathalie Chetcuti-Sperandio

*Centre de Recherche en Informatique de Lens*
*Université d'Artois - France*
chetcuti@cril.univ-artois.fr

Fabio Massacci

*Dip. di Informatica e Telecomunicazioni*
*Università di Trento - Italy*
Fabio.Massacci@unitn.it

**Abstract**

Reasoning about 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, naming and identities of agents and complex naming constructions (Blackburn's office-mate is M4M's PC-Chair's Colleague), then we need to reason about intervals of time and metric time for expiration dates and validity intervals.

One of the limitation of many formalizations is their folding on Lampson 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. What we find missing is what Syverson termed an "independently motivated semantics". A semantics where models fitting SDSI would just be a particular subset of logical models and where other proposals could be equally well accomodated.

Here, we propose such an independently motivated 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 and advanced proposals that exploit both qualitative and metric constraints, whose integration is far from trivial.

# 1 Introduction

The security of credential-based systems is one of today's security challenges. Things are further complicated by the substantial disappearance of the traditional model of client/server interaction: 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.

The traditional authentication (who the sender is) and authorization (what it can do) questions have been transformed into another one about trust management: "Does the set of credentials about identity and about permission 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 [22]). One of the most frequently cited has been Lampson and Rivest's SDSI (Simple Distributed Security Infrastructure), which was later refined into an Internet RFC as SPKI (Simple Public Key Infrastructure) [8].

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 *M4M's Chair*, the *Chair* is a local name, which maps to an individual who is likely to be different from *TABLEAUX's Chair*. The individual standing for *Chair* may be linked by some certificate to an individual named *Areces*. *Areces' Colleague* may also be mapped into more than one individual. Suppose now that any *Areces' Colleague* was granted access to *Blackburn's Computer*, should a claim from an *M4M'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 [1] has used a modal logic, which later Howell and Kotz [12] have modified, Halpern and Van der Meyden [11,10] have proposed another modal logic to reason about it, Jha, Reps *et al.* have used model-checking for the verification of the time-free fragment of the tree [20]. Li et al. [16,17] have used logic programming and Datalog.

This is a knowledge representation and reasoning problem that is interesting because its solution requires the combination of many AI techniques, which so far have been rather separated. First of all, we need to represent permissions, second we need to represent naming and identities, third we need to reason about time.

Last but not least, we must provide methods to reason about them. Nobody is interested in formalizing the SDSI/SPKI M4M's PC-Chair's ConferencePaperMan-

ager 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 M4M345.pdf. For the naming and modal part we need to combine features for advanced work in modal and dynamic logics [18,6]. For the temporal part, as we shall see, this is a challenge where a CSP-based qualitative reasoning proper of Allen's Interval algebra [3,4] is not sufficient. TCSPs (Temporal CSPs) and STPs (Simple Temporal Problem) are necessary to handle metric temporal relations [7].

So far, the approaches proposed in Computer Security fora have focussed on using logics for giving *the* semantics to the operational description of SDSI/SPKI. This has created rather unwieldy proposals somehow bent and soldered on the application. In contrast, one needs a general framework to represent and reason about naming, identity and time, and then show that particular restrictions on the model allows us to capture the desired properties. If we look at modal logics, we don't have "the" semantics for knowledge, we have kripke structures and the particular notion of knowledge (positive or negative introspection etc.) that fits our application is obtained by imposing different restrictions on the model.

Building upon previous formalizations we provide a general model for reasoning about identities, authorization, credentials *and* time. We hope that this would provide the equivalent of what Syverson termed an "independently motivated semantics" [21]. The semantics has an intuitive look-and-feel and in particular is also able to reason naturally about time, permission and naming at the same 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 Allen's interval algebra and advanced proposals that exploit both qualitative and metric constraints [19,14].

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 the calculus and the proof that it is sound and complete before concluding.

## 2 A Primer on SDSI/SPKI

The idea behind SDSI/SPKI [8] 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 agent has its set of *local names*, denoted by $n$, possibly with subscript. Name can be composed so that for the agent Areces, the local name $Patrick$ may map into Blackburn and the agent $Patrick's\ Buddy$ may map into what Blackburn considers a buddy. In SPKI, *compound names* are denoted by the tuple construct `(name n1 n2 ...nk)` or by the equivalent expression $n_1's\ n_2's\ \ldots n_k$ where each `n-i` is a local name, and $n_1$ is either a local name or a public key. When $n_1$ 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 $Patrick's\ colleagues$ by one agent depends on its interpretation of $Patrick$, and may be different from another agent's

interpretation of $Patrick$ and $Patrick's\ colleagues$.

In absence of a centralized naming authority we don't have global names in SPKI and the only global entities are public keys. So, strictly speaking, we have no agents interpeting names in SPKI but just keys. We may say that $Patrick$'s interpretation of the agent Areces is mapped into the agent Blackburn, but what we can only say in practice is that $Patrick$ is mapped into the public key $k_p$. So that the public key $k_a$ (which corresponds to what we call the agent Areces) will take any credential verifiable with the public key $k_p$ as a statement coming from his fellow $Patrick$.

SPKI has other kinds of names 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 *et al.* [20], 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. It may also describe a sequence of additional tests for the validation of certificates.

An *authorization certificate* has the form `(cert (issuer k) (subject p) (propagate) A valid)`, where `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, as above. The `(propagate)` section is optional. Intuitively, the issuer uses such a certificate to grant `p` the authority to perform the actions in `A`. Moreover, if the propagation field is present, then the subject is further authorized to delegate this authority to others.

## 3 Logical Syntax for SPKI

The syntax generalizes the proposals by Abadi [1], Halpern and van der Meyden [11,10], Jha, Reps *et al.* [13,20].

We have a set of actions $\mathfrak{A}$, a set of keys $\mathfrak{K}$, and a set of names $\mathfrak{N}$. Agents or *principals* are denoted by $k$ where $k \in \mathfrak{K}$, $n$ where $n \in \mathfrak{N}$, or $p's\ q$ where $p$ and $q$ are principals.

The *atomic formulae* of our logic are $p \mapsto q$ and $\mathsf{Perm}(p, a)$ where $p$ and $q$ are principals and $a$ is an action. Standard 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 agent currently evaluating the formula any authorization for $p$ can be mapped into an authorization for $q$.

We have two forms of *certificates*: $k \ \mathsf{cert}\ p \mapsto q : [t_1, t_2]$, the interpretation of which is that for the agent $k$ the local name $p$ is bound to the fully qualified name $q$ for the validity period between the instant $t_1$ and $t_2$, and $k \ \mathsf{cert}\ \mathsf{Perm}(p, a) : [t_1, t_2]$, the interpretation of which is that the agent $k$ permits action $a$ to $p$ for the validity period of the certificate. We allow one to use compound names as subjects and not just local names.

Note that a logical certificate does not (necessarily) correspond to a physical certificate. Indeed our major interest for formalizing a credential based system is the possibility of passing between the following two alternative formulations:

- given a set of physical certificates with certain validity intervals should a request for a given permission be granted in a given time interval;
- is a logical certificate about a given permission in a given time interval a logical consequence of a set of physical certificates with certain validity intervals.

Obviously the notion of consequence can only be defined once we have a semantics.

With respect to the calculus in [1,2] we have eliminated the $says$ operator because it is subsumed by the $\mathsf{cert}$ operator. See [18] for an automated reasoning method for some fragments of Abadi's *et al.* calculus.

## 4 Semantics

The motto of any credential-based system could be "Extra public-keys nulla salus" and our model builds upon this intuition by making a model where the basic domain is a set of real keys, and where names connects keys with each other and with permissions.

Looking from the perspective of modal logics of beliefs we can say that, loosely speaking, keys can be seen as individuals, authorizations as atomic concepts and names as roles connecting the individuals [9]. From the perspective of dynamic logic, keys are states, names are programs which allow us to pass from one state to another, $'s$ can be mapped in the operator for sequential composition, and certificates can be seen as some form of necessitation operator [15]. The major difference is that we can now name states. The speaks-for construct is slightly more subtle and has been widely used in grammar logics.

Let's first give a model *without* time. So a *model* is a generalized Kripke structure that is a triple $\langle \mathcal{K}, \mathcal{N}, \mathcal{A} \rangle$ where $\mathcal{K}$ is a set of real keys (such that for all $k \in \mathfrak{K}$ there is a $\underline{k} \in \mathcal{K}$). The naming relation $\mathcal{N}$ is a mapping from local names to subset of relations over keys $\mathcal{LN} \longrightarrow 2^{\mathcal{K}\times\mathcal{K}}$. In a equivalent terms one could say that it is an indexed family of relations. The grant relation $\mathcal{A}$ is a function mapping actions into into subset of keys $\mathcal{A} \longrightarrow 2^{\mathcal{K}}$.

The naming relation associates to each local name the corresponding key: $\langle k, k' \rangle \in \mathcal{N}(n)$, or equivalently $k \, \mathcal{N}_n \, k'$ if the principal associated to key $k$ associates to the name $n$ at least the key $k'$. We have a relation because the same name $n$ may refer to many individuals as in $Patrick's \ colleagues$. In the sequel we use the set-oriented notation: $k' \in \mathcal{N}(k, n)$.

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 $k' \in \mathcal{A}(k, a)$ means that the principal associated with the key $k'$ is permitted action $a$ by the principal associated with the key $k$.

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

- $\mathcal{N}(\underline{k}_1, k_2) = \{\underline{k}_2\}$ where $k_2 \in \mathfrak{K}$
- $\mathcal{N}(\underline{k}_1, n) = \mathcal{LN}(\underline{k}_1, n)$ where $n \in \mathfrak{N}$
- $\mathcal{N}(\underline{k}_1, p's\ q) = \bigcup_{\underline{k}_2 \in \mathcal{N}(\underline{k}_1, p)} \mathcal{N}(\underline{k}_2, q)$

The first rule says that syntactic keys are always mapped into the corresponding semantics keys. The last rule is a simple representation of composition: if $k_1$ associates $k_2$ to the name $p$ and $k_2$ associates $k_3$ to the name $q$ then $k_1$ should associate $k_3$ to the name $p's\ q$.

Note that, with this semantics any principal $p_1's\ p_2's\ \ldots's\ p_l$ is equivalent to some principal $q_1's\ q_2's\ \ldots's\ q_m$ where $q_1$ is either a key or a name and $q_2, \ldots, q_m$ are names. So from now on we shall consider only principals in the latter form.

To lift our structure to time we need to introduce the concept of a *trace* that is a mapping from time to models. A trace $\mathcal{M}$ associates to each instant of time a given model $\mathcal{M}_t = \langle \mathcal{K}_t, \mathcal{N}_t, \mathcal{A}_t \rangle$ where $t \in \mathbb{R}$. Then $\mathcal{K}_t : \mathbb{R} \longrightarrow \mathcal{K}$ is the set of real keys (such that for all $k \in \mathfrak{K}$ there is a semantic key $\underline{k} \in \mathcal{K}_t$) which are associated to the named keys at time $t$, $\mathcal{N}_t$ is the local naming relation with the additional time parametre $\mathbb{R} \times \mathfrak{N} \longrightarrow 2^{\mathcal{K} \times \mathcal{K}}$ at time $t$ and $\mathcal{A}_t$ is the grant relation: $\mathbb{R} \times \mathfrak{A} \longrightarrow 2^{\mathcal{K}}$. The extension of the semantics for naming *principals* is identical, except for the $t$ subscript.

At this stage the comparison with variations on modal logics is interesting.
For example should the validity period of keys always be a connected interval? For instance, suppose that we have that in our model we have
$k_1 \in \mathcal{N}_{11-May-2003}(k_2, FAST's\ Chair)$, that is the real key $k_1$, say FAST steering committee global key, associates the key $k_2$ to the $FAST's\ Chair$. After a year the certificate expires and we have $k_1 \notin \mathcal{N}_{11-May-2004}(k_2, FAST's\ Chair)$. Do we want to impose that for all $t \geq 11{-}May{-}2004$ we have $k_1 \notin \mathcal{N}_t(k_2, FAST's\ Chair)$? This means that after a certificate is expired we would not accept another revalidation certificate for the same key. This is one possible model and there might be cases when this is desirable and cases whether it is not.

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

- $\mathcal{M}_t, \underline{k}_1 \models p \mapsto q$ iff $\forall \underline{k} \in \mathcal{N}_t(\underline{k}_1, q), \underline{k} \in \mathcal{N}_t(\underline{k}_1, p)$
- $\mathcal{M}_t, \underline{k}_1 \models \mathsf{Perm}(p, a)$ iff $\forall \underline{k} \in \mathcal{N}_t(\underline{k}_1, p), a \in \mathcal{A}_t(\underline{k})$

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

- $\mathcal{M}_t \models k\ \mathsf{cert}\ p \mapsto q : [t_1, t_2]$ iff $t \in [t_1, t_2] \Rightarrow \mathcal{M}_t, \underline{k} \models p \mapsto q$
- $\mathcal{M}_t \models k\ \mathsf{cert}\ \mathsf{Perm}(p, a) : [t_1, t_2]$ iff $t \in [t_1, t_2] \Rightarrow \mathcal{M}_t, \underline{k} \models \mathsf{Perm}(p, a)$

Note that, in comparison with Halpern and van der Meyden proposals, we do not force certificates to be satified by a model. A model is an independent entity from certificates. It has its properties and satisfies some formulae. If it satisfies the appropriate formulae it will also satisfies some certificates. In this way, as we said, a particular certificate theory characterizes a particular set of models.

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

$$\mathcal{M} \models k \text{ cert } c : [t_1, t_2] \text{ iff } \forall t, \mathcal{M}_t \models k \text{ cert } c : [t_1, t_2]$$

As Jha, Reps *et al.* [13,20] we have a notion of consequence for chains of certificates:

**Definition 4.1** A certificate $\chi$ is a *logical consequence* of a set of certificates $\mathcal{C}$ if any trace which satisfies all the certificates of $\mathcal{C}$ also satisfies $\chi$.

## 5 Semantic Tableaux

Now comes the key question: how can we know that a certificate is a logical consequence of a set of physical certificates? Furthermore, if this is not the case, how can we get a *semantical counter-example*?

We propose a calculus based on *semantic tableaux*. Intuitively, to prove that a certificate $\chi$ is a logical consequence of a set of other certificates $\mathcal{C}$ (see Def. 4.1) we instead try to construct a model that falsifies $\chi$ and satisfies $\mathcal{C}$. If we succeed, then we have a 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.
For tableaux we shall use the usual terminology. For instance, see De Giacomo and Massacci [6] for the modal part and Kautz and Ladkin [14] for the temporal part.

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 information. So a *tableau* is a collection of branches, each intuitively corresponding to some possible counter-model. A *branch* has three components for *untimed information* (such as naming relations), for *qualitative temporal information* and for *quantitative temporal information*.

For the "untimed" information we have a triplet $\langle \mathtt{K}, \mathtt{N}, (\mathtt{F}, \mathtt{A}) \rangle$ where

- $\mathtt{K}$ is the set of the syntactic keys plus possibly some new keys,
- the function $\mathtt{N} : \mathtt{K} \times \mathtt{K} \longrightarrow 2^{\mathfrak{N}}$ is the local naming relation constructed so far,
- the function $\mathtt{A} : \mathtt{K} \times \mathtt{K} \longrightarrow 2^{\mathfrak{A}}$ corresponds to the permissions assigned so far,
- $\mathtt{F} : \mathtt{K} \longrightarrow 2^{Formulae}$ corresponds to the formulae (labelled with validity intervals) which we try to satisfy.

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

- $\mathtt{V}$ is a set of variables representing temporal intervals,
- a function $\mathtt{E} : \mathtt{V} \times \mathtt{V} \longrightarrow 2^{Allen's\ s\ relations}$ corresponds to the qualitative temporal

relations that we have forced so far.

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

Allen's interval relations are the following: before, after, meets, met − by, overlaps, overlapped − by, starts, started − by, during, contains, finishes, finished − by, equals. Their interpretation is intuitive and we refer to Allen's work for additional explanations [3,4]. 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 $\langle \mathtt{V}_T, \mathtt{E}_T \rangle$ where

- $\mathtt{V}_T$ is a set of variables representing temporal points
- $\mathtt{E}_T : \mathtt{V}_T \times \mathtt{V}_T \longrightarrow 2^{Intervals}$ represent the metric constraints between the time points that we contructed so far.

After an 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.

### *5.1 Initialization*

At the very start, K is the set of keys appearing in $\{\chi\} \cup \mathcal{C}$. F, N, A, V and $\mathtt{V}_T$ are empty.

For each certificate $k$ cert $c : [t_1, t_2]$ in $\{\chi\} \cup \mathcal{C}$, a new interval variable $v_\omega$ is added to V and $v_\omega^-$ and $v_\omega^+$ are added to $\mathtt{V}_T$. Then E, resp. $\mathtt{E}_T$, is enriched with constraints existing between $v_\omega$, resp. $v_\omega^-$ and $v_\omega^+$, and the remaining interval, resp. point, variables. The basic intuition is that the interval variable $v_\omega$ is used to define the validity period of the certificate.

**Example 5.1** Let $\mathcal{C}_N = \{k_1 \text{ cert } (p \mapsto k_3) : [1,4]\}$ and $\mathcal{C}_A = \{k_2 \text{ cert } \mathsf{Perm}(k_1\,'s\ p, a) : [2,6]\}$. Then $\mathtt{V} = \{v_1, v_2\}$ where $v_1$ corresponds to $[1,4]$ and $v_2$ to $[2,6]$, $\mathtt{V}_T = \{v_1^-, v_1^+, v_2^-, v_2^+\}$, $\mathtt{E}(v_1, v_2) = \{\mathsf{overlaps}\}$, $\mathtt{E}_T(v_1^-, v_1^+) = \{[3,3]\}$, $\mathtt{E}_T(v_2^-, v_2^+) = \{[4,4]\}$, $\mathtt{E}_T(v_1^-, v_2^-) = \{[1,1]\}$ (the remaining metric constraints are deducible from $\mathtt{E}_T$), see Fig. 1.

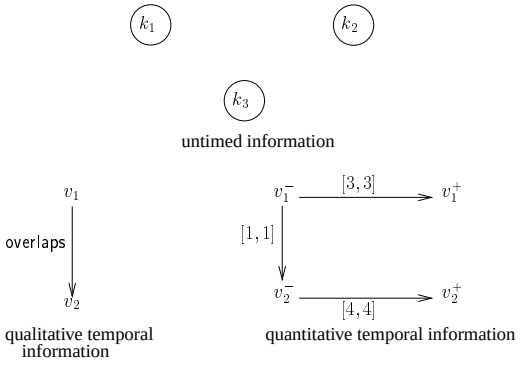


Fig. 1. Illustration of example 5.1

So, looking at the figure, at the beginning we have the three keys of the two certificates, and for the moment, no naming relation between them.

We introduce an interval variable for the validity of the first, resp. second, certificate: $v_1$, resp. $v_2$. Then we say that the two intervals overlap (as [1,4] overlaps [2,6]). For the metric constraints we say that the difference in time between the beginning of $v_1$ and the beginning of $v_2$ ranges in the interval $[1, 1]$, because $v_1 = [1, 4]$ and $v_2 = [2, 4]$ and $2 - 1 = 1$. Equally, the duration of $v_1$, that is the difference between $v_1^- = 1$ and $v_1^+ = 4$, lasts between $[3, 3]$.

Finally if $\chi$, the certificate we are trying to disprove, has the form $k_i$ cert $c : [t_{i1}, t_{i2}]$, we add $\{\neg c : v_i\}$ to $\mathtt{F}(k_i)$ where $v_i$ is the interval corresponding to $[t_{i1}, t_{i2}]$. Intuitively, we want a model where the certificate is not valid in the given interval.

### *5.2 Building rules*

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

**Definition 5.2** Let $v_1$, $v_2$ and $v_3$ be interval variables.

- $v_1 \cap v_2 = \emptyset$ when the following constraint is satisfied: $v_1\{\mathsf{before}, \mathsf{meets}, \mathsf{met-by}, \mathsf{after}\}v_2$
- $v_1 \cap v_2 \neq \emptyset$ when the following constraint is satisfied: $v_1\{\mathsf{overlaps}, \mathsf{overlapped-by}, \mathsf{starts}, \mathsf{started-by}, \mathsf{during}, \mathsf{contains}, \mathsf{finishes}, \mathsf{finished-by}, \mathsf{equals}\}v_2$
- $v_2 \subseteq v_1$ when the following constraint is satisfied: $v_2\{\mathsf{starts}, \mathsf{during}, \mathsf{finishes}, \mathsf{equals}\}v_1$

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

**Proposition 5.3** *To compute the overlap of validity periods we let* $v_3 = v_1 \cap v_2$.

- *if* $v_1\{\mathsf{starts}, \mathsf{during}, \mathsf{finishes}, \mathsf{equals}\}v_2$ *then* $v_3 = v_1$
- *if* $v_1\{\mathsf{started-by}, \mathsf{contains}, \mathsf{finished-by}\}v_2$ *then* $v_3 = v_2$
- *if* $v_1$ $\mathsf{overlaps}$ $v_2$ *then* $v_3$ $\mathsf{finishes}$ $v_1$ *and* $v_3$ $\mathsf{starts}$ $v_2$
- *if* $v_1$ $\mathsf{overlapped-by}$ $v_2$ *then* $v_3$ $\mathsf{starts}$ $v_1$ *and* $v_3$ $\mathsf{finishes}$ $v_2$

We have now all the necessary machinery to introduce the rules.

#### *5.2.1 Rule for certificates*

If $k$ cert $c : [t_1, t_2] \in \mathcal{C}$ then for the corresponding $v \in \mathtt{V}$, add $\{c : v\}$ to $\mathtt{F}(k)$

The basic intuition is that if the certificate $k$ cert $c : [t_1, t_2]$ is valid then the corresponding formula must be true for the corresponding key during the validity interval of the certificate.

#### *5.2.2 Rules for formulae*

We now consider the processing of atomic formulae only as the boolean operators are handled in the usual way: for instance 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. As for disjuntion it simply splits the branch in two.

**Speaking for**

All the rules follow the same pattern. For each constraints on the model we either check that there is no temporal interaction or, if there is some, we update the untimed information during the overlapping intervals. Step by step this fills the naming relations between keys and specify the timing relations between the various validity intervals. The rules consider both positive and negative parts:

- if $p \mapsto k' : v \in \mathtt{F}(k)$ then add $\{p : v\}$ to $\mathtt{N}(k, k')$
- if $p \mapsto k's\ q : v \in \mathtt{F}(k)$ then $\forall k_3 \in \mathtt{K} \cup \mathtt{K}, q : v' \in \mathtt{N}(k', k_3)$ either one has $v \cap v' = \emptyset$ or let $v_3 = v \cap v'$ and add $\{p : v_3\}$ to $\mathtt{N}(k, k_3)$
- if $p \mapsto n's\ q : v \in \mathtt{F}(k)$ (where $q$ can be null) then $\forall k' \in \mathtt{K} \cup \mathtt{K}, n's\ q : v' \in \mathtt{N}(k, k')$ either one has $v \cap v' = \emptyset$ or let $v_3 = v \cap v'$ and add $\{p : v_3\}$ to $\mathtt{N}(k, k')$
- if $\neg(p \mapsto q) : v_1 \in \mathtt{F}(k_1)$ then for a new action $a_\omega$, add $\{\ \mathsf{Perm}(p, a_\omega) : v_1, \neg\ \mathsf{Perm}(q, a_\omega) : v_1\}$ to $\mathtt{F}(k_1)$

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 2.

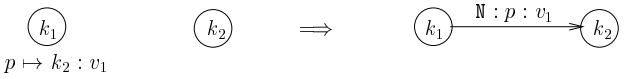


Fig. 2. Illustration of first rule for $\mapsto$ (the temporal graphs remain unchanged)

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 $n's\ q$ for a certain validity 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 period do overlap and then we must chain the two certificates for the overlapping periods, namely $v_3 = v \cap v'$.

We illustrate the second $[\mapsto]$ rule in the special case where $v_1$ $\mathsf{overlaps}$ $v_2$ in Figure 3. As we can see from the figure we have added the link between $k_1$ and $k_3$ but only for the overlapping interval $v_3$. The temporal information says that when $v_1$ finishes then $v_3$ also finishes and when $v_2$ starts then $v_3$ also starts.

**Permission**

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

- if $\mathsf{Perm}(k_2, a) : v_1 \in \mathtt{F}(k_1)$ then add $\{a : v_1\}$ to $\mathtt{A}(k_1, k_2)$

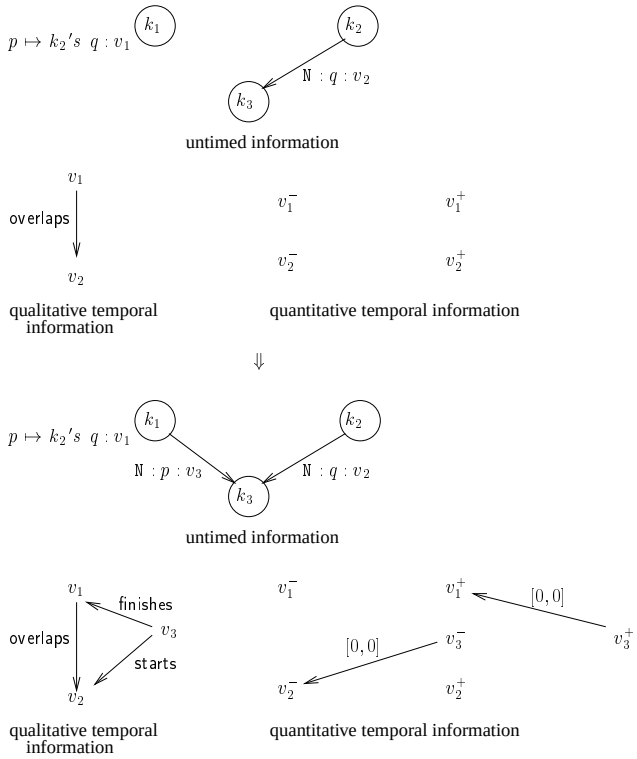


Fig. 3. Illustration of one of the rules $[\mapsto]$ (where $v_1$ overlaps $v_2$)

- if $\mathsf{Perm}(k_2's\ q, a) : v_1 \in \mathtt{F}(k_1)$ then $\forall k_3 \in \mathtt{K}$ such that $q : v_2 \in \mathtt{N}(k_2, k_3)$ either one has $v_1 \cap v_2 = \emptyset$ or let $v_3 = v_1 \cap v_2$ and add $\{a : v_3\}$ to $\mathtt{A}(k_1, k_3)$
- if $\mathsf{Perm}(n's\ q, a) : v_1 \in \mathtt{F}(k_1)$ (where $q$ can be null) then $\forall k_2 \in \mathtt{K}$ such that $n's\ q : v_2 \in \mathtt{N}(k_1, k_2)$ either one has $v_1 \cap v_2 = \emptyset$ or let $v_3 = v_1 \cap v_2$ and add $\{a : v_3\}$ to $\mathtt{A}(k_1, k_2)$

We also have rules for negated permissions:

- if $\neg\ \mathsf{Perm}(k_2, a) : v_1 \in \mathtt{F}(k_1)$ then for a new interval $v_\omega \subseteq v_1$, add $\{\neg a : v_\omega\}$ to $\mathtt{A}(k_1, k_2)$
- if $\neg\ \mathsf{Perm}(k_2's\ q, a) : v_1 \in \mathtt{F}(k_1)$ then for a new $k_\omega \in \mathtt{K}$ and a new interval $v_\omega \subseteq v_1$, set $\mathtt{N}(k_2, k_\omega)$ to $\{q : v_\omega\}$ and $\mathtt{A}(k_1, k_\omega)$ to $\{\neg a : v_\omega\}$
- if $\neg\ \mathsf{Perm}(n's\ q, a) : v_1 \in \mathtt{F}(k_1)$ (where $q$ can be null) then for a new $k_\omega \in \mathtt{K}$ and a new interval $v_\omega \subseteq v_1$, set $\mathtt{N}(k_1, k_\omega)$ to $\{n's\ q : v_\omega\}$ and $\mathtt{A}(k_1, k_\omega)$ to $\{\neg a : v_\omega\}$

**Principals**

These rules basically refine the naming relations eliminating or creating compound names when validity intervals allow to do that.

- if $k_3's\ q : v_1 \in \mathtt{N}(k_1, k_2)$ then add $\{q : v_1\}$ to $\mathtt{N}(k_3, k_2)$
- if $n's\ q : v_1 \in \mathtt{N}(k_1, k_2)$ then $\forall k_3 \in \mathtt{K} \cup \mathtt{K}$ such that $n : v_2 \in \mathtt{N}(k_1, k_3)$ either one has $v_1 \cap v_2 = \emptyset$ or let $v_3 = v_1 \cap v_2$ and add $\{q : v_3\}$ to $\mathtt{N}(k_3, k_2)$
- if $q_1 : v_1 \in \mathtt{N}(k_1, k_2)$, $q_2 : v_2 \in \mathtt{N}(k_2, k_3)$ then either one has $v_1 \cap v_2 = \emptyset$ or let

$v_3 = v_1 \cap v_2$ and add $\{n's\ q : v_3\}$ to $\mathtt{N}(k_1, k_3)$

### *5.3 Consistency*

#### *5.3.1 Rule for consistency*

A principal associated to a key cannot be permitted and forbidden the same action at the same time:

if $a : v_1 \in \mathtt{A}(k_1, k_2)$ and $\neg a : v_2 \in \mathtt{A}(k_1, k_2)$ then

$$\mathtt{E}(v_1, v_2) = \mathtt{E}(v_1, v_2) \cap \{\mathsf{before}, \mathsf{after}, \mathsf{meets}, \mathsf{met-by}\}$$

#### *5.3.2 Consistency of the temporal information*

Finally we must guarantee the consistency of the temporal information. Roughly speaking the qualitative and the metric temporal information are processed separately, then combined and, if new information appeared, updated, otherwise the computation stops. If the empty relation is present then the original information is inconsistent [14]. This algorithm is sound but not complete unless the interval network is at least *preconvex* [5]. To get completeness of the processing of the temporal information, we modify slightly some rules: whenever we need to add $v_1 \cap v_2 = \emptyset$ we actually split it into $v_1\{\mathsf{before}, \mathsf{meets}\}v_2$ or $v_1\{\mathsf{met-by}, \mathsf{after}\}v_2$.

**Definition 5.4** 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.

In Fig. 4 we have an example of an open tableau.

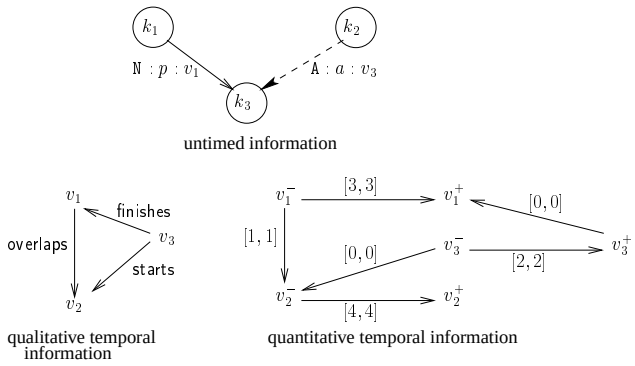


Fig. 4. A (open) tableau for example 5.1

## 6 Soundness and Completeness

**Definition 6.1** A branch $B$ is SAT if there is a trace $\mathcal{M}$ and a mapping $\mu$ from $B$ to $\mathcal{M}$ such that:

- $\forall t \in \mathbb{R}$, $\mu(\mathtt{K}) \subseteq \mathcal{K}_t$
- if $n : v \in \mathtt{N}(k_1, k_2)$ then $\forall t \in \mu(v)$, $\mu(k_2) \in \mathcal{LN}_t(\mu(k_1), \mu(n))$

- if $a : v \in \mathtt{A}(k_1, k_2)$ then $\forall t \in \mu(v)$, $\mu(k_2) \in \mathcal{A}_t(\mu(k_1), \mu(a))$
- if $f : v \in \mathtt{F}(k)$ then $\forall t \in \mu(v)$, $\mathcal{M}_t, \mu(k) \models \mu(f)$
- if $\neg f : v \in \mathtt{F}(k)$ then $\forall t \in v^*$, $\mathcal{M}_t, \mu(k) \not\models \mu(f)$ where $v^* \subseteq \mu(v)$

and $\mu$ satisfies the temporal information:

- if $\mathsf{r} \in \mathtt{E}(v_1, v_2)$ then $\mu(v_1)\, \mathsf{r}\, \mu(v_2)$
- if $v \in \mathtt{E}_T(e_1, e_2)$ then $\mu(e_2) - \mu(e_1) \in \mu(v)$

**Lemma 6.2 (Satisfiability preservation)** *If a branch $B$ is SAT then applying a rule to $B$ yields a SAT branch.*

**Proof.** (sketch) Assume $B$ is a SAT branch. There is some trace $\mathcal{M}$ and some mapping $\mu$ from $B$ to $\mathcal{M}$ for which Def. 6.1 holds. Using the semantics (Section 4) we can show that it still holds after applying a rule to $B$, possibly by extending $\mathcal{M}$ or $\mu$. □

**Theorem 6.3 (Soundness)** *If a certificate $\chi$ has a closed tableau given a set of certificates $\mathcal{C}$ then $\chi$ is a logical consequence of $\mathcal{C}$.*

**Proof.** (sketch) The usual proof consists in showing that an unvalid certificate yields an open tableau. If $\chi$ is not a logical consequence of $\mathcal{C}$ then $\mathcal{C}$ and $\neg\chi$ are satisfiable by some trace $\mathcal{M}$. So the initial branch of the tableau for $\chi$ given $\mathcal{C}$ is SAT. Then we get the conclusion thanks to lemma 6.2 and to the saturation of the tableau. □

As for completeness, we first saturate the tableau (*i.e.* all the branches of the tableau). If one branch $B$ is open then we can build a trace from it such that $B$ is SAT w.r.t. some mapping.

**Definition 6.4** An *instantiation* of the temporal networks of a branch associates a real interval to each interval variable and a real number to each point variable such as to satisfy the qualitative and metric constraints between the temporal variables.

**Definition 6.5** Let $B$ be an open branch. A *trace $\mathcal{M}$ associated with $B$* is built in the following way: let $\iota$ be an instantiation of the temporal networks [5]

- $\forall t \in \mathbb{R}$, $\mathcal{K}_t = \mathtt{K}$
- $\forall t \in \mathbb{R}$, $\mathcal{LN}_t(k_1, n) = \{k_2 |$ there is $v$ such that $t \in \iota(v)$ and $n : v \in \mathtt{N}(k_1, k_2)\}$
- $\forall t \in \mathbb{R}$, $\mathcal{A}_t(k_1, a) = \{k_2 |$ there is $v$ such that $t \in \iota(v)$ and $a : v \in \mathtt{A}(k_1, k_2)\}$

**Lemma 6.6** *If $B$ is an open branch then $B$ is SAT.*

**Proof.** (sketch) Let $\mathcal{M}$ be a trace associated with $B$ w.r.t. a temporal instantiation $\iota$. Consider the mapping $\mu$ from $B$ to $\mathcal{M}$ where $\mu$ is identity except for temporal intervals and points where $\mu$ coincide with $\iota$. Then using the saturation of the branch and the absence of inconsistency we can show that $B$ is SAT by induction on principals and statements. □

**Theorem 6.7 (Completeness)** *If a certificate $\chi$ is a logical consequence of a set of certificates $\mathcal{C}$ then $\chi$ has a closed tableau given $\mathcal{C}$.*

**Proof.** (sketch) Like for the soundness proof, the completeness one goes the other way round: let $B$ be an open branch of a tableau for $\chi$ given $\mathcal{C}$. From the initialization of the tableau, lemma 6.6 and Def. 6.1, we get that $\neg\chi$ and $\mathcal{C}$ are satisfiable by some trace. Hence $\chi$ is not a logical consequence of the set of certificates $\mathcal{C}$. □

## 7 Conclusions

In the security literature there has been a number of proposals for the right logical account for SDSI/SPKI features. Abadi [1] has used the DEC-SRC calculus for access control [2]. However a number of problems have been found in the DEC-SRC calculus by later authors [18,11]. Howell and Kotz [12] 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 [11,10] have proposed two modal logics to reason about it but their proposal is fairly tailored on the SDSI/SPKI framework and again no automated reasoning procedure have been given. Jha, Reps *et al.* [13,20] have given a pushdown automata procedure for, loosely speaking, time-free fragment of Halpern and van der Meyden logic. In all papers the treatment of time is either absent (Abadi, Howell and Kotz, Jha and Reps) or rather sketchy (Halpern and Van der Meyden).

Conceptually, it is possible to have here 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 [17].

An intriguing subject of future research is the usage of *symbolic validity intervals*. For instance, if we had $k$ cert $M4M's\,co{-}chair \mapsto Areces : [12/04/04, NextM4M]$ and then have symbolic constraints on intervals such as $NextM4M$ is non-overlapping with 2004 and overlaps with part of 2002 and starts after [appointment-day,end-of-conference]. The calculus we presented can indeed cope with such constraints.

Building upon previous attempt of formalizations we provide a general model for reasoning about naming and identities, 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 [6] with systems for reasoning about the interval algebra by Allen [3,4] and advanced proposals that exploit both qualitative and metric constraints [19,14].

## 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*.

[4] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–844, 1983.

[5] J.-F. Condotta. The Augmented Interval and Rectangle Networks. In *KR'2000*, 2000.

[6] 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.

[7] R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Networks. *Artificial Intelligence*, 49(1–3):61–95, 1991.

[8] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen. *SPKI Certificate Theory*, September 1999. IEFT RFC 2693.

[9] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. Reidel, 1983.

[10] J. Halpern and R. van der Meyden. A Logical Reconstruction of SPKI. In *CSFW'01*.

[11] J. Halpern and R. van der Meyden. A Logic for SDSI's Linked Local Name Spaces. *JCS*, 9(1/2):105–142, 2001.

[12] J. Howell and D. Kotz. A Formal Semantics for SPKI. In *ESORICS 2000*, 2000.

[13] S. Jha and T. Reps. Analysis of SPKI/SDSI Certificates Using Model Checking. In *IEEE CSFW-2002*, 2002.

[14] H. A. Kautz and P. B. Ladkin. Integrating Metric and Qualitative Temporal Reasoning. In *AAAI-91*, 1991.

[15] D. Kozen and J. Tiuryn. Logic of programs. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume II, chapter 14, pages 789–840. 1990.

[16] N. Li. Local Names in SPKI/SDSI. In *IEEE CSFW 2000*, pages 2–15, 2000.

[17] N. Li, B. N. Grosof, and J. Feigenbaum. A Practically Implementable and Tractable Delegation Logic. In *IEEE SSP 2000*, pages 27–42, 2000.

[18] F. Massacci. Tableaux Methods for Formal Verification of Multi-Agent Distributed Systems. *JLC*, 8(3):373–400, 1998.

[19] I. Meiri. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. In *AAAI'91*, 1991.

[20] S. Schwoon, S. Jha, T. Reps, and S. Stubblebine. On Generalized Authorization Problems. In *CSFW 2003*, 2003.

[21] P. Syverson. The Use of Logic in the Analysis of Cryptographic Protocols. In *IEEE SS& P-91*, pages 156–170, 1991.

[22] S. Weeks. Understanding Trust Management Systems. In *IEEE SS&P-2001*, 2001.