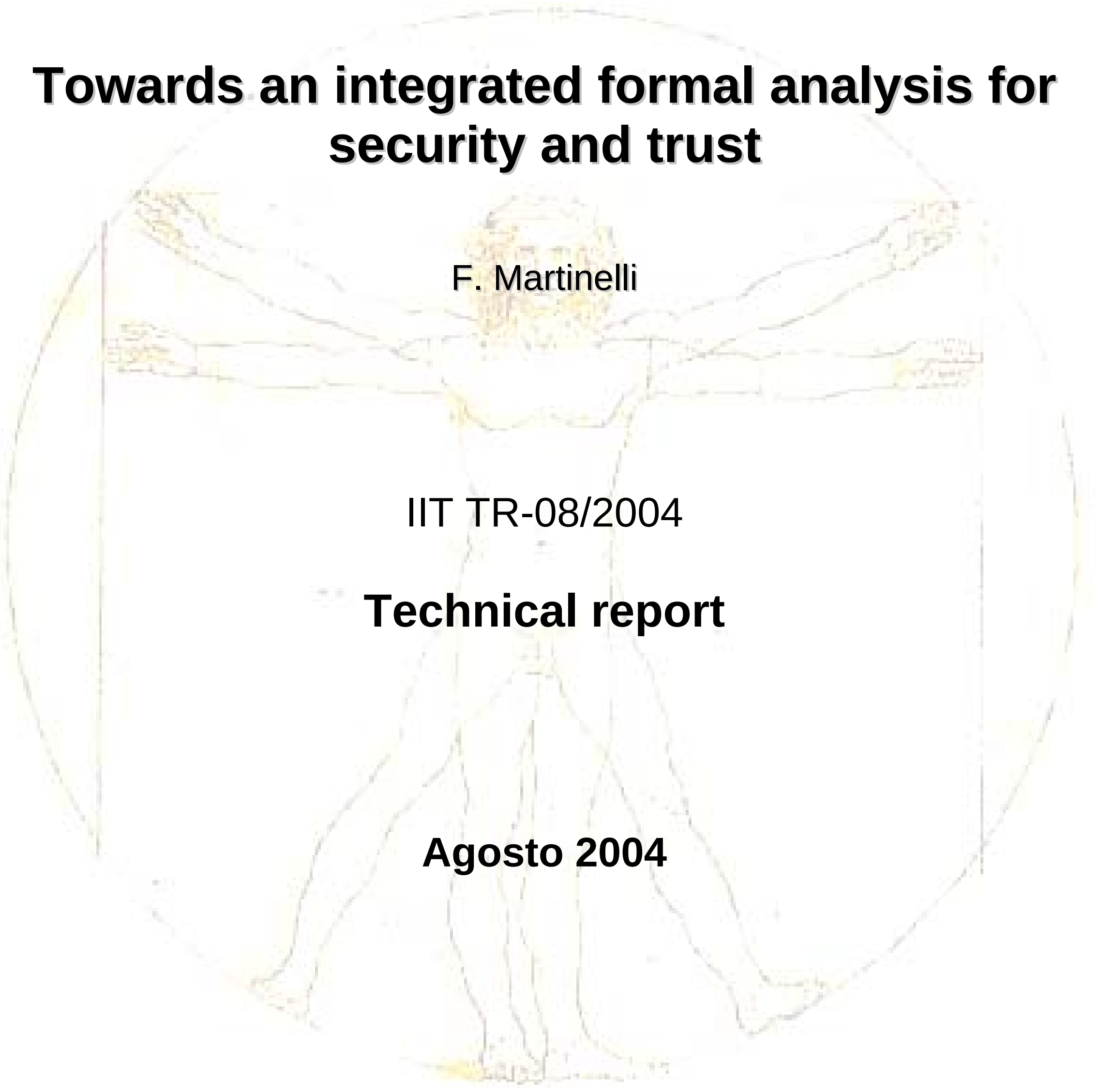
Consiglio Nazionale delle Ricerche



Towards an integrated formal analysis for security and trust

F. Martinelli

IIT TR-08/2004

Technical report

Agosto 2004

Istituto di Informatica e Telematica

Towards an integrated formal analysis for security and trust[⋆]

Fabio Martinelli

Istituto di Informatica e Telematica - C.N.R., Pisa, Italy
e-mail: Fabio.Martinelli@iit.cnr.it

Abstract. We aim at defining an integrated framework for the (automated) analysis for security and trust in complex and dynamic scenarios. In particular, we show how the same machinery used for the formal verification of security protocols may be used to analyze access control policies based on trust management.

1 Introduction

Computer security is a research area that is increasingly receiving the attention of researchers. Just to mention a very relevant topic, consider some security issues in ubiquitous computing systems: These consist of different entities that have to cooperate and share resources to achieve a certain goal. Cooperation is often enabled by trust relationships among entities. There is clearly a tight connection between the security mechanisms used to guarantee the confidentiality and integrity of information and mechanisms used to establish, manage and negotiate trust, reputation and recommendation among the different entities (e.g., see [8, 12, 23]).

In this paper we focus on the integrated formal modeling and analysis of security and trust. In particular, we uniformly model security protocols and some form of access control mechanisms based on trust management.

Formal languages for modeling distributes systems have been applied in the last decade to the analysis of cryptographic protocols. In this framework, cryptography is usually modeled by representing encryptions as terms of an algebra, e.g., $E(m, k)$ may represent the encryption of a message m with a key k (also denoted as $\{m\}_k$). Usually, the so-called perfect encryption abstraction is adopted: encryptions are considered as injective functions which can be inverted only by knowing the correct information,

[⋆] Work partially supported by MIUR project COVER; by CNR project "Tecniche e Strumenti Software per l'Analisi della Sicurezza delle Comunicazioni in Applicazioni Telematiche di Interesse Economico e Sociale" and by a CREATE-NET grant for the project "Quality of Protection (QoP)".

i.e. the decryption key. For instance, common inference rules for modeling the behavior of the encryption and decryption (in a shared-key schema) are the followings:

$$\frac{m \quad k}{E(m,k)} \quad \frac{E(m,k) \quad k}{m} \tag{1}$$

which should be read as: from a message m and a key k we can build the encryption $E(m,k)$; from an encryption $E(m,k)$ and a decryption key k we can obtain the encrypted message m.

The long standing tradition of modeling the specific features of cryptographic functions as term-rewriting rules met the powerful verification techniques developed for process algebras. As a matter of fact, several formal languages for describing communication protocols, for instance CSP [13], have been exploited for representing cryptographic protocols without changes in syntax or semantics: the inference rules have been given at the meta-level of the verification. Instead others, like the π–calculus [1] and the CCS [16, 14], have been effectively refined: the π–calculus have been equipped with two pattern matching constructs for modeling message splitting and shared-key decryption, respectively; the CCS has been equipped with an inference construct that permits to infer new messages from others, i.e.:

$$[m_1 \quad m_n \vdash_r x].P$$

which denotes a process that tries to deduce a message m from the messages in $m_1, \ldots, m_n$ and when it succeeds it substitutes this message for x in the process specification P. The language is called Crypto-CCS ([16]).

The inference relation could be defined in many ways. Often, we will consider the transitive closure of the entailment relations used in each process. This would give a a complex inference system. Such inference systems allow us to cope with the variety of different crypto-systems that can be found in the literature.

However, when one analyzes a security protocol, usually assumes public keys, digital certificates, and generally speaking credentials are already given, and does not check how these are formated/negotiated/managed. Such a limited view seems not completely appropriate for dynamic, fully interconnected systems, where access control policies may change and typically may also depend on credentials presented by users.

Similarly, when one wishes to formally analyze (e.g., see [3]) access control systems, the authentication mechanisms (usually a security protocol) is given for "secure", without further specification.

While separation of concerns is often desirable, this is not always possible. The interplay between security protocols and access control mechanisms/policies is crucial. Moreover, a good specification, validation, analysis and management framework should take an holistic point of view.

As a matter of fact, we show that the idea proposed by CryptoCCS of using inference constructs is also useful to model access control mechanisms based on credentials in distributed systems (e.g., see [19, 5]).

Example 1. Indeed, consider a set of credentials, e.g. (signed) messages containing information about access rights. Assume that $\{A, ob_1, +\}_{pr(C)}$ means that the user C (via the signature with its private key $pr(C)$) asserts A has the right to access the object ob_1 and may grant this access to other users (this is denoted through the symbol $+$). A rule like:

$$\frac{\{A, ob_1, +\}_{pr(C)} \quad pr(C) \quad \{grant \quad B, ob_1\}_{pr(A)}}{\{B, ob_1, +\}_{pr(C)}}(acc_C)$$

may be used by the controller C to issue other access right credentials, after receiving an indication by A, i.e. the signed message $\{grant \quad B, ob_1\}_{pr(A)}$.

Thus, we may consider the inference rules as an abstract mechanism to express security policies usually defined using other mathematical models and logics (e.g., see [6, 19]).

In this paper we will deal in particular with the RT trust management system [12]. However, it should be noticed how the approach is very general. In particular, we will show also how to encode with inference systems the mechanisms for reasoning about trust proposed in [8].

Having a unique language will allow us to model the interplay between security protocols that use the trust relationships among different users, and the ways in which these relationships are created (that often rely on security/interaction protocols).

The fact that we can both model cryptography and some form of credential/trust management with the inference construct of CryptoCCS allows us to use the software tools and methodologies already developed for security protocols analysis to the more general case where credentials are explicitly managed. In particular, in [15] a software tool for automated security protocols analysis has been defined an in [17] has been extended to cope with a huge class of inference systems.

It is worthy noticing that the CryptoCCS has been previously defined to set up a uniform framework for the analysis of security properties and information flow (non-interference) with the same machinery (e.g., see [4]).

To sum up, the flexibility of the inference construct as a modeling tool may allow us to study and analyze uniformly several aspects of network/system security and trust.

The paper is organized as follows. Section 2 recalls the CryptoCCS language and some basic concepts about security protocol analysis. Section 3 shows how trust management specification and analysis may be performed using a similar machinery of the one for security protocols. Finally, Section 4 concludes the paper.

2 Crypto-CCS

Crypto-CCS [16, 14]is a slight modification of CCS process algebra [18], adopted for the description of cryptographic protocols.

The CryptoSPA model consists of a set of sequential agents able to communicate by exchanging messages.

The data handling part of the language consists of a set of inference rules used to deduce messages from other messages. We consider a set of relations among messages as: $\vdash_r \subseteq \mathcal{M}^{i_r+1}$, where r is the name of the rule and i_r the number of premises. For the sake of simplicity, we assume that $\vdash_r$ (for each $r \in \mathcal{R}$) is decidable.

2.1 The Language Syntax

CryptoSPA syntax is based on the following elements:

- A set Ch of channels, partitioned into a set I of input channels (ranged over by c) and a set O of output channels (ranged over by $\bar{c}$, the output corresponding to the input c);
- A set Var of variables, ranged over by x;
- A set $\mathcal{M}$ of messages, defined over a certain signature, ranged over by M, N, m, n ....

The set $\mathcal{L}$ of CryptoSPA terms (or processes) is defined as follows:

$$P, Q ::= \mathbf{0} \mid c(x).P \mid \bar{c}M.P \mid \tau.P \mid P \,|\, Q \mid P \backslash L \mid$$

$$A(M_1, \dots, M_n) \mid [\langle M_1, \dots, M_r \rangle \vdash_{rule} x]P; Q$$

where $M, M', M_1, \dots, M_r$ are messages or variables and L is a set of channels. Both the operators $c(x).P$ and $[\langle M_1 \dots M_r \rangle \vdash_{rule} x]P; Q$ bind variable x in P.

We assume the usual conditions about *closed* and *guarded* processes, as in [18]. We call $\mathcal{P}$ the set of all the *CryptoSPA* closed and guarded terms. The set of actions is $Act = \{c(M) \mid c \in I\} \cup \{\bar{c}M \mid \bar{c} \in O\} \cup \{\tau\}$ (τ is the internal, invisible action), ranged over by a. We define $sort(P)$

to be the set of all the channels syntactically occurring in the term P. Moreover, for the sake of readability, we always omit the termination $\mathbf{0}$ at the end of process specifications, e.g. we write a in place of $a.\mathbf{0}$. We give an informal overview of $CryptoSPA$ operators:

- $\mathbf{0}$ is a process that does nothing.
- $c(x).P$ represents the process that can get an input M on channel c behaving like $P[M/x]$).
- $\overline{c}m.P$ is the process that can send m on channel c, and then behaves like P.
- $\tau.P$ is the process that executes the invisible τ and then behaves like P.
- $P_1 \,|\, P_2$ (*parallel*) is the parallel composition of processes that can proceed in an asynchronous way but they must synchronize on complementary actions to make a communication, represented by a τ.
- $P \backslash L$ is the process that cannot send and receive messages on channels in L; for all the other channels, it behaves exactly like P;
- $A(M_1, \ldots, M_n)$ behaves like the respective defining term P where all the variables $x_1, \ldots, x_n$ are replaced by the messages $M_1, \ldots, M_n$;
- $[\langle M_1, \ldots, M_r \rangle \vdash_{rule} x]P; Q$ is the process used to model message manipulation as cryptographic operations. The process $[\langle M_1, \ldots, M_r \rangle \vdash_{rule} x]P; Q$ tries to deduce an information z from the tuple $\langle M_1, \ldots, M_r \rangle$ through the application of rule $\vdash_{rule}$; if it succeeds then it behaves like $P[z/x]$, otherwise it behaves as Q. The set of rules that can be applied is defined through an inference system (e.g., see Figure 1 for an instance).

2.2 The Operational Semantics of CryptoCCS

In order to model message handling (and so cryptography in an abstract way) we use a set of inference rules. Note that $CryptoCCS$ syntax, its semantics and the results obtained are completely parametric with respect to the inference system used. We present in Figure 1 an instance inference system, with rules: to combine two messages obtaining a pair (rule $\vdash_{pair}$); to extract one message from a pair (rules $\vdash_{fst}$ and $\vdash_{snd}$); to encrypt a message m with a key k obtaining $\{m\}_k$ and, finally, to decrypt a message of the form $\{m\}_k$ only if it has the same key k (rules $\vdash_{enc}$ and $\vdash_{dec}$, respectively).

In a similar way, inference systems can contain rules for handling the basic arithmetic operations and boolean relations among numbers, so that the value-passing CCS **if-then-else** construct can be obtained via the $\vdash_{rule}$ operator.

$$\frac{m \quad m'}{(m,m')}(\vdash_{pair}) \quad \frac{(m,m')}{m}(\vdash_{fst}) \quad \frac{(m,m')}{m'}(\vdash_{snd})$$

$$\frac{m \quad k}{\{m\}_k}(\vdash_{enc}) \quad \frac{\{m\}_k \quad k}{m}(\vdash_{dec})$$

Fig. 1. An example inference system for shared key cryptography.

$$(input)\frac{m \in \mathcal{M}}{c(x).P \xrightarrow{c(m)} P[m/x]} \qquad (output)\frac{}{\bar{c}m.P \xrightarrow{\bar{c}m} P} \qquad (internal)\frac{}{\tau.P \xrightarrow{\tau} P}$$

$$(\backslash L)\frac{P \xrightarrow{c(m)} P' \quad c \notin L}{P\backslash L \xrightarrow{c(m)} P'\backslash L} \qquad (|)_1\frac{P_1 \xrightarrow{a} P_1'}{P_1 \,|\, P_2 \xrightarrow{a} P_1' \,|\, P_2} \qquad (|)_2\frac{P_1 \xrightarrow{c(x)} P_1' \quad P_2 \xrightarrow{\bar{c}m} P_2'}{P_1 \,|\, P_2 \xrightarrow{\tau} P_1' \,|\, P_2'}$$

$$(Def)\frac{P[m_1/x_1,\ldots,m_n/x_n] \xrightarrow{a} P' \quad A(x_1,\ldots,x_n) \doteq P}{A(m_1,\ldots,m_n) \xrightarrow{a} P'}$$

$$(\mathcal{D})\frac{\langle m_1,\ldots,m_r\rangle \vdash_{rule} m \quad P[m/x] \xrightarrow{a} P'}{[\langle m_1,\ldots,m_r\rangle \vdash_{rule} x]P;Q \xrightarrow{a} P'}$$

$$(\mathcal{D}_1)\frac{\nexists m \text{ s.t. } \langle m_1,\ldots,m_r\rangle \vdash_{rule} m \quad Q \xrightarrow{a} Q'}{[\langle m_1,\ldots,m_r\rangle \vdash_{rule} x]P;Q \xrightarrow{a} Q'}$$

Fig. 2. Structured Operational Semantics for CryptoCCS (symmetric rules for $|_1, |_2$ and $\backslash L$ are omitted)

Example 2. Natural numbers may be encoded by assuming a single value 0 and a function $S(y)$, with the following rule: $\frac{x}{S(x)}$ *inc*. Similarly, we can define summations and other operations on natural numbers. ■

Example 3. We do not explicitly define equality check among messages in the syntax. However, this can be implemented through the usage of the inference construct. E.g., consider rule $\frac{x \quad x}{Equal(x,x)}$ *equal*. Then $[m = m']A$ (with the expected semantics) may be equivalently expressed as $[m \quad m' \vdash_{equal} y]A$ where y does not occur in A. Similarly, we can define inequalities, e.g., $\leq$, among natural numbers. ■

The operational semantics of a $CryptoCCS$ term is described by means of labeled transition relations, $P \xrightarrow{a} P'$, with the informal meaning that the process P may perform an action a evolving in the process P'. More formally, we consider a *labelled transition system* (*lts*, for short) $\langle \mathcal{P}, Act, \{\xrightarrow{a}\}_{a \in Act}\rangle$, where $\{\xrightarrow{a}\}_{a \in Act}$ is the least relation between $CryptoCCS$ processes induced by the axioms and inference rules of Figure 2.

2.3 Security protocol analysis

The security protocol analysis proposed in [16, 14] is based on the checking of following property:

$$\forall X \text{ s.t. } S \,|\, X \text{ satisfies } F$$

where F is a logical formula expressing the desired property. Often, when secrecy properties are considered, F models the fact that a given message, i.e. the secret to be verified, is not deducible from a given set of messages, i.e. the knowledge of the intruder X acquired during the computation with S. The verification of such property requires the ability of computing the closure of a inference systems, i.e. the possibility to iteratively apply the inference rules. Given a set $\mathcal{R}$ of inference rules, we consider the deduction relation $\mathcal{D}^{\mathcal{R}} \subseteq \mathcal{P}^{fin}(\mathcal{M}) \times \mathcal{M}$. Given a finite set of closed messages, say ϕ, then $(\phi, M) \in \mathcal{D}^{\mathcal{R}}$ if M can be derived by iteratively applying the rules in $\mathcal{R}$. Under certain assumption on the form of the rules, we may have that $\mathcal{D}^{\mathcal{R}}(\phi)$ is decidable.

2.4 Some assumptions on the inference system

Given a well-founded measure on messages, we say that a rule

$$r \doteq \frac{m_1 \quad \dots \quad m_n}{m_0}$$

is a S-rule (*shrinking* rule), whenever the conclusion is a proper subterm of one of the premises (call such premise *main*). The rule r is a G-rule (*growing* rule) whenever the conclusion is strictly larger than each of the premises, and all the variables in the conclusion must be in the premises.

Definition 1. *We say that an inference system enjoys a G/S property if it consists only of G-rules and S-rules, moreover whenever a message can be deduced through a S-rule, where one of the main premises is derived by means of a G-rule, then the same message may be deduced from the premises of the G-rule, by using only G-rules.*

Several of the inference systems used in the literature for describing cryptographic systems enjoy this restriction[1].

The main advantage of considering systems enjoying the G/S restriction is that proofs for messages may have a normal form, i.e., either:

[1] It is worthy noticing that in [9] a similar terminology has been used, and a restriction, called S/G, has been defined. However, this is rather different from ours and it is not well suited to model cryptographic systems.

– these consist of a sequence of applications of G-rules, or
– these consist of a sequence of applications of G-rules, followed by the application of an S-rule whose main premises are in ϕ and possibly followed by other applications of G-rules and S-rules.

Indeed, using G-rules for inferring the main premises of an S-rules, is un-useful. Thus, shrinking rules may be significantly applied only to messages in ϕ and to messages obtained by S-rules. However, since the measure for classifying the S-rules is well-founded then such a shrinking phase would eventually terminate when applied to a closed set of messages ϕ. After, only growing rules are possible. Thus, if the inference system enjoys the G/S restriction then $\mathcal{D}^R(\phi)$ is decidable when ϕ is finite. We may note that the inference system in page 1 enjoys the G/S restriction and so its deduction relation is indeed decidable.

In the case the inference system has not growing rules, we have decidability even in presence of a weaker form of *shrinking rules*. We say that a rule is *eq-shrink* whenever the conclusion has an equal or smaller size than one of the premises; moreover all the variables occurring in the conclusion must occur in at least of of the premises. In such a case the decision procedure simply consists of building the transitive closure of the inference rules.

3 Modeling several trust management languages

Through process algebras, one can formally specify communicating protocols and complex distributed systems. For instance, one could use CryptoCCS to describe the components and the communication interface of an access control mechanism as Policy Enforcement Point (PEP) or Policy Decision Point (PDP), e.g. see [21]. In particular, in trust management systems, where policies are given through credentials, this allow one to use the inference system of CryptoCCS to model also the trust engine used in these frameworks.

Let see how it works with two well known models.

3.1 RT_0: Role-based Trust Management

We show how inference rules can be conveniently used to model RT languages for trust management [12, 22, 11, 10]. In these languages, credentials carry information on policies to define attribute of principals starting from assertions of other principals. The notion of attribute is general enough to permit to use RT languages to model Role-based Access Control Mechanisms (RBAC), e.g. see [20]. As a matter of fact, an attribute could be considered as a role. Then one could use RT credential to express how principals are related to roles[2]. More precisely, we denote principals with A, B, C...; we denote role names with r, u, z.... A role takes the form of a principal followed by a role name, separated by a dot, e.g. $A.r$.

RT assumes four kind of credentials that express possible policy statements.

- $A.r \leftarrow D$ (**simple member**)
  This statement defines that D has role $A.r$.
- $A.r \leftarrow A_1.r_1$ (**simple containment**)
  This statement asserts that if D has role $A_1.r_1$ then it has role $A.r$. This kind of credential can be used to delegate the authentication of attributes from A to A_1.
- $A.r \leftarrow A_1.r_1.r_2$ (**linking containment**)
  This statement asserts that tf E has role $A_1.r_1$ and D has role $E.r_2$ then D has role $A.r$. This kind of credential may be used to delegate the assignment of $A.r$ role not to specific entities but to entities of a given role.
- $A.r \leftarrow A_1.r_1 \cap A_2.r_2$ (**Intersection containment**)
  This statement asserts that D has role $A_1.r_1$ and $A_2.r_2$ then D has role $A.r$.

Example 4. Consider the following set of credentials.

$$\begin{array}{l} Univ.stud \leftarrow FM \\ Shop.discount \leftarrow Univ.stud \end{array}$$

It follows that FM has role *Shop.discount*. So, the shop offers discounts to the students of the University.

The language for credentials has been equipped with several semantics. In particular, one semantics based on datalog is very similar to our inference rules (that in this case can be seen as datalog rules). So, we define one inference rule for each credential as follows.

[2] Similarly, credentials and attributes could be used to assign permissions to roles

$$\begin{array}{ll}
A.r \leftarrow D & \{D, r\}_A \\
A.r \leftarrow A_1.r_1 & \dfrac{\{y, r_1\}_A}{\{y, r\}_A} \\
A.r \leftarrow A_1.r_1.r_2 & \dfrac{\{z, r_1\}_{A_1} \quad \{y, r_2\}_z}{\{y, r\}_A} \\
A.r \leftarrow A_1.r_1 \cap A_2.r_2 & \dfrac{\{y, r_1\}_{A_1} \quad \{y, r_2\}_{A_2}}{\{y, r\}_A}
\end{array}$$

However, this requires a rule for each credential. We wish to fix from the very beginning the set of inference rules. Thus, we provide a slightly modified version of the inference system where we consider only 3 rules, one for each kind of credential defined in RT_0 (with the exception of the first kind of credentials that are simply messages).

$$\begin{array}{ll}
A.r \leftarrow D & \{D, r\}_A \\
A.r \leftarrow A_1.r_1 & \dfrac{\{y, r_1\}_A \quad \{r, A_1, r_1\}_A}{\{y, r\}_A} \\
A.r \leftarrow A_1.r_1.r_2 & \dfrac{\{z, r_1\}_{A_1} \quad \{y, r_2\}_z \quad \{r, A_1, r_1, r_2\}_A}{\{y, r\}_A} \\
A.r \leftarrow A_1.r_1 \cap A_2.r_2 & \dfrac{\{y, r_1\}_{A_1} \quad \{y, r_2\}_{A_2} \quad \{r, A_1, A_2, r_1, r_2\}_A}{\{y, r\}_A}
\end{array}$$

Note that, under the common measure of the size of terms, all the previous rules are *eq-shrink* rules and there are no *growing* rules. Thus, establishing whether a given principal, say D, has a certain role in a policy ϕ, i.e. $\{D, r\}_A \in \mathcal{D}(\phi)$ is decidable. This kind of analysis[3] is called *Simple Safety* in [10].

3.2 Josang *et al.* topologies

We also show how the trust model of Josang *et al.* [8] can be managed in our framework. The authors suggest trust is always linked to a purpose. The most natural situation is when one trusts another for performing a certain function/task. This may be expressed as $A \xrightarrow{f} D$, i.e. A trusts D for performing f. Moreover, it is often common that one, say A, asks another, say D, for suggesting/reccomendating a third one for doing a given task, i.e. f. This could be expressed by the following credential $A \xrightarrow{r,f} D$.

[3] Actually, that work considers a dynamic set of policies. However, the analysis technique adopted is actually based on a subset of the set of prolog rules that represent the initial problem. Thus, we are also able to manage it.

The main idea is that when one calculates whether a given chain trust exists, it must always consider that the last step in the chain is a functional trust one, while all the others are recommendation steps. Thus, we have another kind of credential like $A \xrightarrow{r} B \xrightarrow{f} D$, , expressing the fact that A trusts D for performing f via the recommendation of B.

$$
\begin{array}{ll}
A \xrightarrow{f} D & \{f, D\}_A \\
A \xrightarrow{r,f} D & \{r, D, f\}_A \\
\dfrac{A \xrightarrow{r,f} B \quad B \xrightarrow{r,f} D}{A \xrightarrow{r,f} D} & \dfrac{\{r, B, f\}_A \quad \{r, D, f\}_B}{\{r, D, f\}_A} \\
\dfrac{A \xrightarrow{r,f} B \quad B \xrightarrow{f} D}{A \xrightarrow{r} B \xrightarrow{f} D} & \dfrac{\{r, B, f\}_A \quad \{f, D\}_B}{\{r, B, f, D\}_A}
\end{array}
$$

As in the previous case, the deduction relation of this set of rules is decidable. This gives us an alternative strategy w.r.t. the one presented in [8].

As in [8], one could insert further information into the credentials, as measure of trust. For instance, credentials could be enhanced with such information and rules could derived the trust measure of resulting credentials in the appropriate way. For instance, consider the following credential enhanced with a trust measure, i.e.: $A \xrightarrow{r,f,m} B$. Then the transitive composition rule could be the following:

$$
\frac{A \xrightarrow{r,f,m_1} B \quad B \xrightarrow{r,f,m_2} D}{A \xrightarrow{r,f,m_3} D}
$$

where m_3 is a function of m_1, m_2, for instance $m_3 = min\{m_1, m_2\}$.

If the set of possible trust values is finite, then the deduction relation is still decidable. More complex trust measures can be found in [7]. Clearly, one may try to define specific strategies for each set of inference rules. However, we argue that the mechanisms we used are general enough to deal with common trust management systems.

4 Conclusions and future work

We have presented an approach for the analysis of security protocols where credentials are explicitly managed. In the simplest case, such credentials are PKI certificates or XML security assertions. In more complex scenarios, these could be statements expressing access control policies, as in RT languages. Our framework is supported by an analysis tool called PaMoChSA [15].

Our results may be considered as a step towards the creation of a uniform and automated specification and verification framework for the evaluation of network and system security. This would be of help for formally measuring the quality of protection of complex architectures. We plan to develop a full language for specifying software architectures equipped with our analysis techniques. We plan also to integrate and facilitate the specific analysis techniques for trust management protocols, as given in [22], into our framework.

As a specific application, we note XML documents may be seen as terms in a given signature and thus analyzed with our tool. Recently, Gordon *et al.* (see [2]) developed an analysis for authentication properties of web services using a similar process algebra equipped with an additional prolog-like construct. We can see our work as an extension of theirs in order to model also trust relationships for web services.

References

[1] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 1999.

[2] Karthikeyan Bhargavan, Cedric Fournet, and Andrew D. Gordon. A semantics for web services authentication. In *Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 198–209. ACM Press, 2004.

[3] P. A. Bonatti and P. Samarati. Logics for authorizations and security. In *Logics for Emerging Applications of Databases*, LNCS. Springer-Verlag, 2003.

[4] R. Focardi, R. Gorrieri, and F. Martinelli. Non interference for the analysis of cryptographic protocols. In *Proceedings of 27th International Colloquium in Automata, Languages and Programming*, volume 1853 of *Lectures Notes in Computer Science*, pages 354–372, 2000.

[5] R. Gorrieri and F. Martinelli. Process algebraic frameworks for the specification and analysis of cryptographic protocols. In *MFCS*, LNCS 2747. Springer-Verlag, 2003.

[6] Halpern and van der Meyden. A logic for SDSI's linked local name spaces. In *PCSFW: Proceedings of The 12th Computer Security Foundations Workshop*. IEEE Computer Society Press, 1999.

[7] Audun Josang. The consensus operator for combining beliefs. *Artif. Intell.*, 141(1):157–170, 2002.

[8] Audun Jsang, Elizabeth Gray, and Michael Kinateder. Analysing topologies of transitive trust. In *Proc. of the* 1^{st} *workshop on Formal Aspects in Security and Trust (FAST2003)*, 2003.

[9] D. Kindred and J. M. Wing. Fast, automatic checking of security protocols. In *Second USENIX Workshop on Electronic Commerce*, pages 41–52, Oakland, California, 1996.

[10] Ninghui Li, John Mitchell, and William H. Winsborough. Beyond proof-of-compliance: Safety and availability analysis in trust management. In *IEEE Symposium on Research in Security and Privacy.* 2003.

[11] Ninghui Li and Mahesh V. Tripunitara. Security analysis in role-based access control. In *ACM Symposium on Access Control Models and Techniques (SACMAT 2004).* 2004.

[12] Ninghui Li, William H. Winsborough, and John C. Mitchell. Distributed credential chain discovery in trust management. *Journal of Computer Security*, 1:35–86, 2003.

[13] Gavin Lowe. Breaking and fixing the Needham Schroeder public-key protocol using FDR. In *Proceedings of Tools and Algorithms for the Construction and the Analisys of Systems*, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer Verlag, 1996.

[14] F. Martinelli. Analysis of security protocols as *open* systems. *Theoretical Computer Science*, 290(1):1057–1106, 2003.

[15] F. Martinelli, M. Petrocchi, and A. Vaccarelli. PaMoChSA: A tool for verification of security protocols based on partial model checking. 2001. Tool Demo at the 1st International School on Formal Methods for the Design of Computer, Communication and Software Systems: Process Algebras.

[16] Fabio Martinelli. Languages for description and analysis of authentication protocols. In P. Degano and U. Vaccaro, editors, *Proceedings of 6th Italian Conference on Theoretical Computer Science*, pages 304–315. World Scientific, 1998.

[17] Fabio Martinelli. Symbolic semantics and analysis for crypto-ccs with (almost) generic inference systems. In *Proceedings of the 27th international Symposium in Mathematical Foundations of Computer Sciences(MFCS'02)*, volume 2420 of *LNCS*, pages 519–531, 2002.

[18] R. Milner. *Communication and Concurrency.* International Series in Computer Science. Prentice Hall, 1989.

[19] P. Samarati and S. De Capitani di Vimercati. Access control: Policies, models, and mechanisms. In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design*, LNCS 2171. Springer-Verlag, 2001.

[20] Ravi Sandhu, Venkata Bhamidipati, Edward Coyne, Srinivas Ganta, and Charles Youman. The arbac97 model for role-based administration of roles: preliminary description and outline. In *Proceedings of the second ACM workshop on Role-based access control*, pages 41–50. ACM Press, 1997.

[21] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, B.V. C. de Laat, M. Holdrege, and D. Spence. RFC 2904 AAA authorization framework. 2000.

[22] William H. Winsborough and Ninghui Li. Safety in automated trust negotiation. In *IEEE Symposium on Security and Privacy.* 2004.

[23] M. Winslett. An introduction to automated trust negotiation. In *Workshop on Credential-Based Access Control Dortmund, October 2002.*