Paper

RT^T_+ – Time Validity Constraints in RT^T Language

Anna Felkner and Adam Kozakiewicz

Research and Academic Computer Network (NASK), Warsaw, Poland

Abstract—Most of the traditional access control models, like mandatory, discretionary and role based access control make authorization decisions based on the identity, or the role of the requester, who must be known to the resource owner. Thus, they may be suitable for centralized systems but not for decentralized environments, where the requester and service provider or resource owner are often unknown to each other. To overcome the shortcomings of traditional access control models, trust management models have been presented. The topic of this paper is three different semantics (set-theoretic, operational, and logic- programming) of RT^T, language from the family of role-based trust management languages (RT). RT is used for representing security policies and credentials in decentralized, distributed access control systems. A credential provides information about the privileges of users and the security policies issued by one or more trusted authorities. The set-theoretic semantics maps roles to a set of sets of entity names. Members of such a set must cooperate in order to satisfy the role. In the case of logic-programming semantics, the credentials are translated into a logic program. In the operational semantics the credentials can be established using a simple set of inference rules. It turns out to be fundamental mainly in large- scale distributed systems, where users have only partial view of their execution context. The core part of this paper is the introduction of time validity constraints to show how that can make RT^T language more realistic. The new language, named RT^T_+ takes time validity constraints into account. The semantics for RT^T_+ language will also be shown. Inference system will be introduced not just for specific moment but also for time intervals. It will evaluate maximal time validity, when it is possible to derive the credential from the set of available credentials. The soundness and completeness of the inference systems with the time validity constraints with respect to the set-theoretic semantics of RT^T_+ will be proven.

Keywords—access control, inference system with time constraints, logic-programming semantics, role-based trust management, set-theoretic semantics.

1. Introduction

Guaranteeing that confidential data and services offered by a computer system are not made available to unauthorized users is an increasingly significant and challenging issue, which must be solved by reliable software technologies that are used for building high-integrity applications. The data, whether in electronic, paper or other form must be properly protected. The traditional solution to this problem are access control techniques, by which users are identified, and granted or denied access to a system, data and other resources, depending on their individual or group identity. This approach fits well into closed, centralized environments, in which the identity of users is known in advance.

Role-based access control (RBAC) model [1], [2] is the most flexible type of access control policy. It uses a user role to control of which users have access to particular resources. Access rights are grouped by the role name and access to resources is restricted to the users who are assigned to appropriate roles. This type of access control works well in a large-scale centralized system and is often used in enterprise environments. Quite the new challenges arise in decentralized and open systems, where the identity of users is not known in advance and the set of users can change. For example, consider a bookstore, in which students who are returning customers are eligible to get discount. However, when a person comes to the bookstore and she says that she is Mary Smith, then her identity itself will not help in deciding whether she is eligible for a discount or not. What can help in this particular situation are two credentials stating that she is a student (she has a student card) and that she owns a bookstore card. The identity of a user itself does not help in making decisions about their rights. What is needed to make such decisions is information about the privileges assigned to the user by other authorities, as well as trust information about the authority itself.

The term of *trust management* was introduced in 1996 by Blaze *et al.* in [3], who defined it as a unified approach to specify and interpret security policies, credentials and trust relationships. In trust management system an entity's privilege is based on its attributes instead of its identities. An entity's attributes are demonstrated through digitally signed credentials issued by multiple principals. A *credential* is an attestation of qualification, competence or authority issued to an individual by a third party. Examples of credentials in real life include identification documents, driver's licenses, membership cards, keys, etc. A credential in a computer system can be a digitally signed document. Such a concept of trust management has evolved since that time to a much broader context of assessing the reliability and developing trustworthiness for other systems and individuals [4]. In this paper, however, we will use the term trust management only in a meaning restricted to the field of access control.

The potential and flexibility of trust management approach stems from the possibility of *delegation*: a principal may transfer limited authority over a resource to other principals.

Such a delegation is implemented by means of an appropriate credential. This way, a set of credentials defines the access control strategy and allows deciding on who is authorized to access a resource, and who is not. RT languages combine trust management and RBAC features. To define a trust management system, a language is needed for describing entities (principals and requesters), credentials and roles, which the entities play in the system. Responding to this need, a family of role-based trust management languages has been introduced in [5]–[7]. The family consists of five languages: RT_0, RT_1, RT_2, RT^T, and RT^D, with increasing expressive power and complexity.

The core language of RT family is RT_0, described in detail in [7]. It allows describing localized authorities for roles, role hierarchies, delegation of authority over roles and role intersections. All the subsequent languages add new features to RT_0.

RT_1 introduces parameterized roles, which can represent relationships between entities.

RT_2 adds to RT_1 logical objects, which can be used to represent permissions given to entities with respect to a group of logically related objects (resources). Those extensions can help in keeping the notation concise, but do not increase the expressive power of the language, because each combination of parameters in RT_1 and each permission to a real instance of a logical object in RT_2 can be defined alternatively as a separate role in RT_0.

This paper focuses on RT^T languages, as it provides useful capabilities not found in any other languages: manifold roles to achieve both agreement of multiple principals from one set and from disjoint sets and role-product operators, which can express threshold and separation of duties policies. Similar to a role, which defines a set of principals, a manifold role defines a set of principal sets, each of which is a set of principals which cooperation satisfies the manifold role. A singleton role can be treated as a special case of a manifold role, which set of cooperating entities is a singleton set. This way, RT_0 credentials can also be expressed in RT^T language.

A threshold policy requires a specified minimum number of entities to agree on some fact, i.e., it requires agreement among k out of a set of entities that satisfy a specified condition, e.g., in a requirement that two different bank cashiers must authorize a transaction. Separation of duties policy requires a set of entities, each of which fulfills a specific role, to agree before access is granted. Both types of policies mean that some transactions cannot be completed by a single entity, because no single entity has all the access rights required to complete the transaction, that is why it is not possible to define it in RT_0.

RT^D provides mechanisms to describe delegation of role activations and selective use of role membership. This language is not covered in this paper.

A more detailed treatment of RT family can be found in [6]. The languages have a precise syntax and semantics definition. A set-theoretic semantics, which defines the meaning of a set of credentials as a function from the set of roles into the power set of entities, has been defined for RT_0 [8], [7] and we defined relational semantics, which apply also to other members of the family up to RT^T in [9].

The paper is organized as follows. Section 2 consists of the role-based trust management language syntax and description of three semantics (relational, operational and logic-programming), including example. Section 3 describes time validity in RT^T language. Section 4 shows inference system over new RT_{+}^{T} language time constraints. An overview of the work related to RT systems and languages is given in Section 5. Final remarks are given in Conclusions.

2. The Syntax and Three Semantics of RT^T Language

Basic elements of RT languages are entities, role names, roles and credentials. *Entities* represent principals that can define roles and issue credentials, and requesters that can make requests to access resources. An entity can, e.g., be a person or program identified by a user account in a computer system or a public key. *Role names* represent permissions that can be issued by entities to other entities or groups of entities. *Roles* represent sets of entities that have particular permissions granted according to the access control policy. A role is described as a pair composed of an entity and a role name. *Credentials* define roles by appointing a new member of the role or by delegating authority to the members of other roles.

2.1. The Syntax of RT^T Language

In this paper, we use capital letters or nouns beginning with a capital letter (e.g., A,B) to denote entities and sets of entities. Role names are denoted as identifiers beginning with a small letter or just small letters (e.g., r,s). Roles take the form of an entity (the issuer of this role) followed by a role name separated by a dot (e.g., $A.r$). Credentials are statements in the language. A credential consists of a role, left arrow symbol and a valid role expression. There are six types of credentials in RT^T, which are interpreted in the following way:

$A.r \leftarrow B$	– *simple membership*: entity B is a member of role $A.r$.
$A.r \leftarrow B.s$	– *simple inclusion*: role $A.r$ includes (all members of) role $B.s$. This is a delegation of authority over r from A to B, because B may cause new entities to become members of the role $A.r$ by issuing credentials that define $B.s$.
$A.r \leftarrow B.s.t$	– *linking inclusion*: role $A.r$ includes role $C.t$ for each C, which is a member of role $B.s$. This is a delegation of authority from A to all the members of the role $B.s$. The expression $B.s.t$ is called a *linked role*.

$A.r \longleftarrow B.s \cap C.t$ – *intersection inclusion*: role $A.r$ includes all the entieties who are members of both roles $B.s$ and $C.t$. This is a partial delegation from A to B and C. The expression $B.s \cap C.t$ is called an *intersection role*.

$A.r \longleftarrow B.s \odot C.t$ – role $A.r$ can be satisfied by a union set of one member of role $B.s$ and one member of role $C.t$. A set consisting of a single entity satisfying the intersection role $B.s \cap C.t$ is also valid.

$A.r \longleftarrow B.s \otimes C.t$ – role $A.r$ includes one member of role $B.s$ and one member of role $C.t$, but those members of roles have to be different entities.

The models discussed in this paper can be, in general, very complex. Therefore, we present here only a simplified example, with the intention to illustrate the basic notions and the notation, with a focus on RT^T credentials.

Example 1 (Example of RT^T). Suppose that we need at least two out of four students to activate the subject. Using RT_0 credentials, we have to explicitly list all the students (four in this simple case) and choose two of them; this list needs to be changed each time members in the students role change. In RT^T only one credential is needed. Further, we want to have two students and one Ph.D. student, who can also (but does not have to) be a regular student. This requires just one more RT^T credential. The entire policy can be expressed as follows:

$$F.students \longleftarrow F.student \otimes F.student\,, \tag{1}$$

$$F.activeSubject \longleftarrow F.phdStudent \odot F.students\,. \tag{2}$$

Now, assume that the following credentials have been added:

$$F.student \longleftarrow \{Alex\}\,, \tag{3}$$

$$F.student \longleftarrow \{Betty\}\,, \tag{4}$$

$$F.student \longleftarrow \{David\}\,, \tag{5}$$

$$F.student \longleftarrow \{John\}\,, \tag{6}$$

$$F.phdStudent \longleftarrow \{John\}\,, \tag{7}$$

$$F.phdStudent \longleftarrow \{Emily\}\,. \tag{8}$$

Then one can conclude that, according to the policy, any pair of students from the set $\{Alex, Betty, David, John\}$ is sufficient to fulfill the role *F.students*, but to activate the subject it is required that either the pair includes *John*, or additionally *Emily* must also attend.

2.2. The Set-Theoretic Semantics of RT^T Language

The semantics of RT_0 has no potential to describe the meaning of RT^T, which supports manifold roles. Therefore, we define the meaning of a set of credentials as a relation over the set of roles and the power set of entities. Thus, we use a cartesian product of the set of roles and the power set of entities as the semantics domain of a RT language. The semantics mapping would associate a specific relation between roles and entities with each set of credentials. Such a relational approach allowed us to define a formal semantics of RT^T language presented in [9].

Example 2 (Set-theoretic semantics for Example 1). Computing consecutive relations S_i starts from an empty set, $S_0 = \phi$. According to Definition 2 from [9] only credentials 3 through 8 are mapped in S_0 into relation S_1:

$$\begin{aligned} S_1 = \{&(\{F\}, student, \{John\}), (\{F\}, student, \{Alex\}),\\ &(\{F\}, student, \{Betty\}), (\{F\}, student, \{David\}),\\ &(\{F\}, phdStudent, \{John\}),\\ &(\{F\}, phdStudent, \{Emily\})\}. \end{aligned}$$

Credential 1 adds the following instances to relation S_2:

$$\begin{aligned} S_2 = S_1 \cup \{&\\ &(\{F\}, students, \{John, Alex\}),\\ &(\{F\}, students, \{John, Betty\}),\\ &(\{F\}, students, \{John, David\}),\\ &(\{F\}, students, \{Alex, Betty\}),\\ &(\{F\}, students, \{Alex, David\}),\\ &(\{F\}, students, \{Betty, David\})\ \}. \end{aligned}$$

Credential 2 is resolved in S_3:

$$\begin{aligned} S_3 = S_2 \cup \{&\\ &(\{F\}, activeSubject, \{John, Alex\}),\\ &(\{F\}, activeSubject, \{John, Betty\}),\\ &(\{F\}, activeSubject, \{John, David\}),\\ &(\{F\}, activeSubject, \{John, Alex, Betty\}),\\ &(\{F\}, activeSubject, \{John, Alex, David\}),\\ &(\{F\}, activeSubject, \{John, Betty, David\}),\\ &(\{F\}, activeSubject, \{Emily, John, Alex\}),\\ &(\{F\}, activeSubject, \{Emily, John, Betty\}),\\ &(\{F\}, activeSubject, \{Emily, John, David\}),\\ &(\{F\}, activeSubject, \{Emily, Alex, Betty\}),\\ &(\{F\}, activeSubject, \{Emily, Alex, David\}),\\ &(\{F\}, activeSubject, \{Emily, Betty, David\})\ \}. \end{aligned}$$

The resulting relation S_3 cannot be changed using the given set of credentials, hence: $S_P = S_3$. Because the RT language considered in this example is RT^T, there is a set of sets of entities assigned to each role.

2.3. The Logic-Programming Semantics of RT^T

The second way that shows how the member sets of roles can also be calculated is to use a logic-programming semantics. The logic-programming semantics of RT_0 credentials was first introduced in [6]. A definition quoted in this subsection is a modified version of this semantics, which has been introduced in [8]. In this case the semantics is given indirectly. RT credentials are translated into a logic program and their semantics is obtained as the minimal Herbrand model of the translation. The main intention of this approach is to provide an implementation of credential resolution.

Definition 1: The logic-programming semantics of $\mathscr{P}$ is the minimal Herbrand model of $LP(\mathscr{P})$, the logic program defined as

$$LP(\mathscr{P}) = \bigcup_{c\in\mathscr{P}} lc(c)\,,$$

where function $lc(\cdot)$ translates every credential to a logic program clause as follows:

$$lc(A.r \leftarrow B) \triangleq r(A,B) : -$$

$$lc(A.r \leftarrow B.s) \triangleq r(A,\xi) : -s(B,\xi)$$

$$lc(A.r \leftarrow B.s.t) \triangleq r(A,\xi) : -s(B,\zeta), t(\zeta,\xi)$$

$$lc(A.r \leftarrow B.s \cap C.t) \triangleq r(A,\xi) : -s(B,\xi), t(C,\xi)$$

We decided to put some changes into logic-programming semantics for RT_0 and define the logic-programming semantics of RT^T.

$$\begin{aligned}
lc(A.r \leftarrow B) &= member(B, role(A,r))\\
lc(A.r \leftarrow B.s) &= member(X, role(A,r)) : -\\
&\quad member(X, role(B,s))\\
lc(A.r \leftarrow B.s.t) &= member(X, role(A,r)) : -\\
&\quad member(C, role(B,s)),\\
&\quad member(X, role(C,t))\\
lc(A.r \leftarrow B.s \cap C.t) &= member(X, role(A,r)) : -\\
&\quad member(X, role(B,s)),\\
&\quad member(X, role(C,t))\\
lc(A.r \leftarrow B.s \odot C.t) &= member(X \cup Y, role(A,r)) : -\\
&\quad member(X, role(B,s)),\\
&\quad member(Y, role(C,t))\\
lc(A.r \leftarrow B.s \otimes C.t) &= member(X \cup Y, role(A,r)) : -\\
&\quad member(X, role(B,s)),\\
&\quad member(Y, role(C,t)), X \backslash = Y
\end{aligned}$$

where $role(A,r)$ correspond to $A.r$, and $member(B, role(A,r))$ correspond to $A.r \leftarrow B$.
As in the case of the set-theoretic, we use Example 1 from Section 2 to illustrate the definition of RT semantics.

Example 3 (Logic-programming semantics for Example 1).

$$\begin{aligned}
lc(F.students \leftarrow F.student \otimes F.student) = {}&\\
&member(X \cup Y, role(F, students)) : -\\
&member(X, role(F, student)),\\
&member(Y, role(F, student)), X \backslash = Y
\end{aligned}$$

$$\begin{aligned}
lc(F.activeSubject \leftarrow F.phdStudent \odot F.students) = {}&\\
&member(X \cup Y, role(F, activeSubject)) : -\\
&member(X, role(F, phdStudent)),\\
&member(Y, role(F, students))
\end{aligned}$$

$$\begin{aligned}
lc(F.student \leftarrow Alex) &= member(Alex, role(F, student))\\
lc(F.student \leftarrow Betty) &= member(Betty, role(F, student))\\
lc(F.student \leftarrow David) &= member(David, role(F, student))\\
lc(F.student \leftarrow John) &= member(John, role(F, student))\\
lc(F.phdStudent \leftarrow John) &= member(John, role(F, phdStudent))\\
lc(F.phdStudent \leftarrow Emily) &= member(Emily, role(F, phdStudent))
\end{aligned}$$

The above rules can be easily implemented by using some prologue interpreter. Only minor syntactic changes (capital letters, etc.) are necessary.

2.4. Inference System over RT^T Credentials

RT^T credentials are used to define roles and roles are used to represent permissions. The semantics of a given set $\mathscr{P}$ of RT^T credentials defines for each role $A.r$ the set of entities, which are members of this role. The member sets of roles can also be calculated in a more convenient way by using an inference system, which defines an operational semantics of RT^T language. An inference system consists of an initial set of formulae that are considered to be true, and a set of inference rules that can be used to derive new formulae from the known ones.
Let $\mathscr{P}$ be a given set of RT^T credentials. The application of inference rules of the inference system will create new credentials, derived from credentials of the set $\mathscr{P}$. A derived credential c will be denoted using a formula $\mathscr{P} \succ c$, which should be read: credential c can be derived from a set of credentials $\mathscr{P}$.

Definition 2: The initial set of formulae of an inference system over a set $\mathscr{P}$ of RT^T credentials are all the formulae: $c \in \mathscr{P}$ for each credential c in $\mathscr{P}$. The inference rules of the system are the following:

$$\frac{c \in \mathscr{P}}{\mathscr{P} \succ c}, \qquad (W_1)$$

$$\frac{\mathscr{P} \succ A.r \leftarrow B.s \qquad \mathscr{P} \succ B.s \leftarrow X}{\mathscr{P} \succ A.r \leftarrow X}, \qquad (W_2)$$

$$\frac{\begin{array}{c}\mathscr{P} \succ A.r \leftarrow B.t \qquad \mathscr{P} \succ B.s \leftarrow C\\ \mathscr{P} \succ C.t \leftarrow X\end{array}}{\mathscr{P} \succ A.r \leftarrow X}, \qquad (W_3)$$

$$\frac{\begin{array}{c}\mathscr{P} \succ A.r \leftarrow B.s \cap C.t \qquad \mathscr{P} \succ B.s \leftarrow X\\ \mathscr{P} \succ C.t \leftarrow X\end{array}}{\mathscr{P} \succ A.r \leftarrow X}, \qquad (W_4)$$

$$\frac{\begin{array}{c}\mathscr{P} \succ A.r \leftarrow B.s \odot C.t \qquad \mathscr{P} \succ B.s \leftarrow X\\ \mathscr{P} \succ C.t \leftarrow Y\end{array}}{\mathscr{P} \succ A.r \leftarrow X \cup Y}, \qquad (W_5)$$

$$\frac{\begin{array}{c}\mathscr{P} \succ A.r \leftarrow B.s \otimes C.t \qquad \mathscr{P} \succ B.s \leftarrow X\\ \mathscr{P} \succ C.t \leftarrow Y \qquad X \cap Y = \phi\end{array}}{\mathscr{P} \succ A.r \leftarrow X \cup Y}. \qquad (W_6)$$

There could be a number of inference systems defined over a given language. To be useful for practical purposes an inference system must exhibit two properties. First, it should be sound, which means that the inference rules could derive only formulae that are valid with respect to the semantics of the language. Second, it should be complete, which means that each formula, which is valid according to the semantics, should be derivable in the system.

All the credentials, which can be derived in the system, either belong to set $\mathscr{P}$, rule (W_1) or are of the type: $\mathscr{P} \succ A.r \longleftarrow X$, rules ($W_2$ through W_6). To prove the soundness of the inference system, one must prove that for each new formula $\mathscr{P} \succ A.r \longleftarrow X$, the triple (A,r,X) belongs to the semantics $S_{\mathscr{P}}$ of the set $\mathscr{P}$. To prove the completeness of the inference system over a set $\mathscr{P}$ of RT^T credentials, we must prove that a formula $P \succ A.r \longleftarrow X$ can be derived by using inference rules for each element $(A,r,X) \in S_{\mathscr{P}}$. Both properties have been shown in [10], proving that the inference system provides an alternative way of presenting the semantics of RT^T.

Example 4 (Inference system for Example 1). We use the inference system to formally derive a set of entities which can cooperatively activate a subject. To make long example shorter, let us use less credentials ((1), (2), (4), (6), and (7)). Using credentials (1), (2), (4), (6), and (7) according to rule (W_1) we can infer:

$$\frac{F.students \longleftarrow F.student \otimes F.student \in \mathscr{P}}{\mathscr{P} \succ F.students \longleftarrow F.student \otimes F.student}$$

$$\frac{F.activeSubject \longleftarrow F.phdStudent \odot F.students \in \mathscr{P}}{\mathscr{P} \succ F.activeSubject \longleftarrow F.phdStudent \odot F.students}$$

$$\frac{F.student \longleftarrow \{Betty\} \in \mathscr{P}}{\mathscr{P} \succ F.student \longleftarrow \{Betty\}}$$

$$\frac{F.student \longleftarrow \{John\} \in \mathscr{P}}{\mathscr{P} \succ F.student \longleftarrow \{John\}}$$

$$\frac{F.phdStudent \longleftarrow \{John\} \in \mathscr{P}}{\mathscr{P} \succ F.phdStudent \longleftarrow \{John\}}$$

Then, using credentials (1), (6) and (4) and rule (W_6) we infer:

$$\frac{\begin{array}{c}\mathscr{P} \succ F.students \longleftarrow F.student \otimes F.student \\ \mathscr{P} \succ F.student \longleftarrow \{John\} \\ \mathscr{P} \succ F.student \longleftarrow \{Betty\} \\ \{John\} \cap \{Betty\} = \phi\end{array}}{\mathscr{P} \succ \mathbf{F.students} \longleftarrow \{\mathbf{John}, \mathbf{Betty}\}}$$

In the next step we use the newly inferred credential and additionally credentials (2) and (7) with the rule (W_5):

$$\frac{\begin{array}{c}\mathscr{P} \succ F.activeSubject \longleftarrow F.phdStudent \odot F.students \\ \mathscr{P} \succ F.phdStudent \longleftarrow \{John\} \\ \mathscr{P} \succ F.students \longleftarrow \{John, Betty\}\end{array}}{\mathscr{P} \succ \mathbf{F.activeSubject} \longleftarrow \{\mathbf{John}, \mathbf{Betty}\}},$$

showing that the set of entities $\{John, Betty\}$ is sufficient to activate the subject.

3. Time Validity in RT^T

Inference rules with time validity for RT_0 were originally introduced in a slightly different way in [8]. In this paper we will try to extend the potential of RT^T language by putting time validity constraints into this language. In this case credentials are given to entities just for some fixed period of time. It is quite natural to assume that permissions are given just for fixed period of time, not for ever. Time dependent credentials take the form: c **in** v, meaning "the credential c is available during the time v". Finite sets of time dependent credentials are denoted by $\mathscr{CP}$ and the new language is denoted as RT^T_+. To make notation clear we write c to denote "c **in** $(-\infty, +\infty)$". Time validity can be denoted as follows:

$[\tau_1,\tau_2];[\tau_1,\tau_2);(\tau_1,\tau_2];(\tau_1,\tau_2);(-\infty,\tau];(-\infty,\tau);$
$[\tau,+\infty);(\tau,+\infty);(-\infty, +\infty);v_1 \cup v_2;v_1 \cap v_2;v_1 \backslash v_2$

and v_1, v_2 of any form in this list, with τ ranging over time constants.

Example 5 (Time validity for Example1). In our scenario, it is quite natural to assume that *Alex*, *Betty*, *David* and *John* are students only for a fixed period of time. The same with *John* and *Emily* as Ph.D. students. Thus, credentials (3)–(8) should be generalized to:

$$F.student \longleftarrow \{Alex\} \textbf{ in } v_1, \tag{9}$$

$$F.student \longleftarrow \{Betty\} \textbf{ in } v_2, \tag{10}$$

$$F.student \longleftarrow \{David\} \textbf{ in } v_3, \tag{11}$$

$$F.student \longleftarrow \{John\} \textbf{ in } v_4, \tag{12}$$

$$F.phdStudent \longleftarrow \{John\} \textbf{ in } v_5, \tag{13}$$

$$F.phdStudent \longleftarrow \{Emily\} \textbf{ in } v_6, \tag{14}$$

stating that credentials (3)–(8) are only available during v_1, v_2, v_3, v_4, v_5, and during v_6, respectively. On the other hand, credentials (1) and (2) are always valid, as they express some time-independent facts. Now, by using (1), (2) and (9)–(14), we want to be able to derive that for example the set $\{Alex, Betty, John\}$ can cooperatively activate the subject during all of the period: $v_1 \cap v_2 \cap v_5$ or $\{Betty, John\}$ during the time $v_2 \cap v_4 \cap v_5$.

3.1. Set-Theoretic Semantics of RT^T_+

Now we can adapt our set-theoretic semantics of RT^T language to the new form of credentials. The semantics can be defined formally in the following way:

Definition 3: The semantics of a set of credentials $\mathscr{CP}$, denoted as $S_{\mathscr{CP}}$, is the smallest relation S_i, such as:

1. $S_0 = \phi$
2. $S_{i+1} = \bigcup_{(c \textbf{ in } v) \in \mathscr{CP}} f(S_i, c) \qquad$ for $i = 0, 1, \ldots$

that is closed with respect to function f, which describes the meaning of credentials in the same way as in [9].

3.2. Logic-Programming Semantics of RT_+^T

When considering the logic-programming semantics of RT_+^T, two possible scenarios must be analyzed: validation of authority at a given time instant and establishing authority for a period of time. In the first scenario, the logic-programming semantics is calculated at a precise time instant, by only considering those time-dependant credentials which are valid at that moment. In view of the fact that there will be no big changes, we will not provide a precise definition of the semantics. The second scenario is more complex, since it involves computing intersections of validity periods. Yet this case is as a future work. Feasibility of creating such semantics is underlined by the fact that development of an inference system for this case proved to be possible, as illustrated in the next section.

4. Inference System over RT_+^T Credentials

Now, we can adapt inference system over RT^T credentials to take time validity into account. Let $\mathscr{CP}$ be a given set of RT_+^T credentials. The application of inference rules of the inference system will create new credentials, derived from credentials of the set $\mathscr{CP}$. A derived credential c valid in time τ will be denoted using a formula $\mathscr{CP} \succ_\tau c$, which should be read: credential c can be derived from a set of credentials $\mathscr{CP}$ during the time τ.

Definition 4: The initial set of formulae of an inference system over a set $\mathscr{CP}$ of RT_+^T credentials are all in the form: c **in** $v \in \mathscr{CP}$ for each credential c valid in time v in $\mathscr{CP}$. The inference rules of the system are the following:

$$\frac{c\ \mathbf{in}\ v \in \mathscr{CP} \qquad \tau \in v}{\mathscr{CP} \succ_\tau c}, \qquad (CW_1)$$

$$\frac{\mathscr{CP} \succ_\tau A.r \leftarrow B.s \qquad \mathscr{CP} \succ_\tau B.s \leftarrow X}{\mathscr{CP} \succ_\tau A.r \leftarrow X}, \qquad (CW_2)$$

$$\frac{\mathscr{CP} \succ_\tau A.r \leftarrow B.s.t \qquad \mathscr{CP} \succ_\tau B.s \leftarrow C \qquad \mathscr{CP} \succ_\tau C.t \leftarrow X}{\mathscr{CP} \succ_\tau A.r \leftarrow X}, \qquad (CW_3)$$

$$\frac{\mathscr{CP} \succ_\tau A.r \leftarrow B.s \cap C.t \qquad \mathscr{CP} \succ_\tau B.s \leftarrow X \qquad \mathscr{CP} \succ_\tau C.t \leftarrow X}{\mathscr{CP} \succ_\tau A.r \leftarrow X}, \qquad (CW_4)$$

$$\frac{\mathscr{CP} \succ_\tau A.r \leftarrow B.s \odot C.t \qquad \mathscr{CP} \succ_\tau B.s \leftarrow X \qquad \mathscr{CP} \succ_\tau C.t \leftarrow Y}{\mathscr{CP} \succ_\tau A.r \leftarrow X \cup Y}, \qquad (CW_5)$$

$$\frac{\mathscr{CP} \succ_\tau A.r \leftarrow B.s \otimes C.t \qquad \mathscr{CP} \succ_\tau B.s \leftarrow X \qquad \mathscr{CP} \succ_\tau C.t \leftarrow Y \qquad X \cap Y = \phi}{\mathscr{CP} \succ_\tau A.r \leftarrow X \cup Y}. \qquad (CW_6)$$

All the credentials, which can be derived in the system, either belong to set $\mathscr{CP}$, rule (CW_1) or are of the type: $\mathscr{CP} \succ_\tau A.r \leftarrow X$, rules ($CW_2$ through CW_6). This new inference system mainly extends the inference rules from previous section, by replacing rules (W_i) with (CW_i) and considering only valid time-dependent credentials from $\mathscr{CP}$.

To prove the soundness of the inference system we must prove that for each new formula $\mathscr{CP} \succ_\tau A.r \leftarrow X$, the triple (A,r,X) belongs to the semantics $S_{\mathscr{CP}}$ of the set $\mathscr{CP}$. Let us first note that all the formulae $\mathscr{CP} \succ_\tau A.r \leftarrow X$, such as $A.r \leftarrow X \in \mathscr{CP}$ are sound. This is proven in Lemma 1.

Lemma 1: If $A.r \leftarrow X \in \mathscr{CP}$ then $(A,r,X) \in S_{\mathscr{CP}}$.

Proof: The relation $S_{\mathscr{CP}}$, which defines the semantics of $\mathscr{CP}$, is a limit of a monotonically increasing sequence of sets $S_0, S_1 \ldots$ such that $S_0 = \phi$. According to Definition 3: $f(S_0, A.r \leftarrow X) = (A,r,X)$. Hence, $(A,r,X) \in S_1$ and because $S_1 \subseteq S_{\mathscr{CP}}$ then $(A,r,X) \in S_{CP}$. ■

To prove the soundness of the inference system over $\mathscr{CP}$, we must prove the soundness of each formula $\mathscr{CP}_\tau \succ A.r \leftarrow X$, which can be derived from the set $\mathscr{CP}$. This is proven in Theorem 1.

Theorem 1 (soundness): If $\mathscr{CP} \succ A.r \leftarrow X$ then $(A,r,X) \in S_{\mathscr{CP}}$.

Proof: Like the proof of Theorem 1 in [10], but relying on the above Lemma 1 instead of Lemma 1 from [10]. ■

To prove the completeness of the inference system over a set $\mathscr{CP}$ of RT_+^T credentials, we must prove that a formula $\mathscr{CP} \succ A.r \leftarrow X$ can be derived by using inference rules for each element $(A,r,X) \in S_{\mathscr{CP}}$. This is proven in Theorem 2.

Theorem 2 (completeness): If $(A,r,X) \in S_{\mathscr{CP}}$ then $\mathscr{CP} \succ A.r \leftarrow X$.

Proof: Like the proof of Theorem 2 in [10], but relying on the above Lemma 1 instead of Lemma 1 from [10]. ■

4.1. Inferring Time Validity of Credentials

This inference system evaluates maximal time validity, when it is possible to derive the credential c from $\mathscr{CP}$. It enhances formula $\mathscr{CP} \succ_\tau c$ to $\mathscr{CP} \succ\!\succ_v c$, specifying that at any time $\tau \in v$ in which $\mathscr{CP}$ has a semantics, it is possible to infer the credential c from $\mathscr{CP}$. To make notation clear we write $\succ\!\succ$ to denote $\succ\!\succ_{(-\infty,+\infty)}$. The inference rules of the system are the following:

$$\frac{c\ \mathbf{in}\ v \in \mathscr{CP}}{\mathscr{CP} \succ\!\succ_v c}, \qquad (CWP_1)$$

$$\frac{\mathscr{CP} \succ\!\succ_{v_1} A.r \leftarrow B.s \qquad \mathscr{CP} \succ\!\succ_{v_2} B.s \leftarrow X}{\mathscr{CP} \succ\!\succ_{v_1 \cap v_2} A.r \leftarrow X}, \qquad (CWP_2)$$

$$\frac{\mathscr{CP} \succ\!\succ_{v_1} A.r \leftarrow B.s.t \qquad \mathscr{CP} \succ\!\succ_{v_2} B.s \leftarrow C \qquad \mathscr{CP} \succ\!\succ_{v_3} C.t \leftarrow X}{\mathscr{CP} \succ\!\succ_{v_1 \cap v_2 \cap v_3} A.r \leftarrow X}, \qquad (CWP_3)$$

$$\frac{\mathscr{CP} \succ\!\succ_{v_1} A.r \leftarrow B.s \cap C.t \qquad \mathscr{CP} \succ\!\succ_{v_2} B.s \leftarrow X \qquad \mathscr{CP} \succ\!\succ_{v_3} C.t \leftarrow X}{\mathscr{CP} \succ\!\succ_{v_1 \cap v_2 \cap v_3} A.r \leftarrow X}, \qquad (CWP_4)$$

$$\frac{\mathscr{CP} \succ\!\succ_{v_1} A.r \leftarrow B.s \odot C.t \qquad \mathscr{CP} \succ\!\succ_{v_2} B.s \leftarrow X \qquad \mathscr{CP} \succ\!\succ_{v_3} C.t \leftarrow Y}{\mathscr{CP} \succ\!\succ_{v_1 \cap v_2 \cap v_3} A.r \leftarrow X \cup Y}, \qquad (CWP_5)$$

$$\frac{\mathscr{CP} \succ\!\!\succ_{v_1} A.r \leftarrow B.s \otimes C.t \quad \mathscr{CP} \succ\!\!\succ_{v_2} B.s \leftarrow X \quad \mathscr{CP} \succ\!\!\succ_{v_3} C.t \leftarrow Y \quad X \cap Y = \phi}{\mathscr{CP} \succ\!\!\succ_{v_1 \cap v_2 \cap v_3} A.r \leftarrow X \cup Y}, \qquad (CWP_6)$$

$$\frac{\mathscr{CP} \succ\!\!\succ_{v_1} c \quad \mathscr{CP} \succ\!\!\succ_{v_2} c}{\mathscr{CP} \succ\!\!\succ_{v_1 \cup v_2} c}. \qquad (CWP_7)$$

The key rule is (CWP_1). It claims that $\mathscr{CP}$ can be used whenever it is valid. Rules (CWP_2) - (CWP_6) simply claim that an inference rule can be used only when all its premises are true and that the validity of the resulting credentials is the intersection of validity periods of all the premises. Finally, the rule (CWP_7) claims that if a credential c can be inferred both with validity v_1 and with validity v_2, then c can be inferred with validity $v_1 \cup v_2$. $\mathscr{CP} \succ\!\!\succ_v$ generalises $\mathscr{CP} \succ_\tau$. They are both equivalent whenever $v = [\tau, \tau]$. Because several possible ways may exist to infer a certain c from $\mathscr{CP}$, all providing a different period of validity, the rule (CWP_7) can be used several times to broaden c's validity.

Definition 5 (maximal inference): An inference terminating in $\mathscr{CP} \succ\!\!\succ_v c$ is called maximal if and only if:

1) there exists no $v' \supset v$ such that $\mathscr{CP} \succ\!\!\succ_{v'} c$, and
2) every sub-inference terminating in $\mathscr{CP} \succ\!\!\succ_{v''} c'$, for $c' \neq c$, which does not use c in its premises, is maximal.

The first condition ensures that the rule (CWP_7) has been used as much as possible to infer the validity of c. The second condition ensures that this property is propagated through the whole inference tree. Maximal inferences guarantee that v in (CWP_1) is the maximal time validity for $A.r \leftarrow X$.
For these inferences we can prove soundness and completeness of $\mathscr{CP} \succ\!\!\succ_v$ by means of Theorem 3, which proof relies on the following Lemma.

Lemma 2: $\mathscr{CP} \succ_\tau c$ implies that there exists a v containing τ such that $\mathscr{CP} \succ\!\!\succ_v \; c$.

Proof: It suffices to replicate inference for $\mathscr{CP} \succ_\tau c$, replacing every appearance of rule (CW_i) with (CWP_i), and v will be the intersection of the validity of all the credentials $\mathscr{CP}$ used in the inference and will be at least $[\tau, \tau]$. ■

Theorem 3 ([soundness and completeness for maximal inferences): Let $\mathscr{CP} \succ\!\!\succ_v c$ be a maximal inference and set $\mathscr{CP}$ of RT_+^T credentials be defined. Then $\mathscr{CP} \succ_\tau c$ if and only if $\tau \in v$.

Proof: By induction on the depth of $\mathscr{CP} \succ\!\!\succ_v c$. For the base case, $\mathscr{CP}$ must contain a credential c **in** v. If $\tau \in v$ we can trivialy conclude thanks to (CW_1). By induction, $\mathscr{CP} \succ_\tau A.r \leftarrow X$ if and only if $\tau \in v$. And vice versa, assuming by contradiction that there is a $\tau' \notin v$ such that $\mathscr{CP} \succ_{\tau'} c$; but then the inference leading to $\mathscr{CP} \succ\!\!\succ_v c$ would not be maximal, because Lemma 2 would contradict the assumption.

For the inductive step, we prove by case analysis on the last rule used. Analysis of (CWP_i) for $i = 2...6$ is trivial, as it adapts the reasoning from proof in [10] in the same way as done above for the base case. The most difficult cases are when using rule (CWP_7). If $\mathscr{CP} \succ\!\!\succ_v c$ terminates with an appearance of (CWP_7), then $v = v_1 \cup v_2$. This case is particular, because formulae $\mathscr{CP} \succ\!\!\succ_{v_1} c$ and $\mathscr{CP} \succ\!\!\succ_{v_2} c$ are not maximal. Let $\mathscr{CP} \succ_\tau c$. By Lemma 2, there exists a v' containing τ such that $\mathscr{CP} \succ\!\!\succ_{v'} c$. Now, it is that $v' \subseteq v$, otherwise $\mathscr{CP} \succ\!\!\succ_v c$ would not be maximal. And vice versa, let $\tau \in v$ and let $\mathscr{CP} \succ\!\!\succ_{v'} c$ be the deepest sub-inference of $\mathscr{CP} \succ\!\!\succ_v c$, which premises do not require c (hence, $\mathscr{CP} \succ\!\!\succ_{v'} c$ has been obtained by using (CWP_i), for $i \neq 7$) and such that $\tau \in v'$. By definition of the rules of inference system (inferring time validity), each of these premises has a time validity containing τ_i; since these premises have been obtained by maximal inferences, by induction we can replace $\succ\!\!\succ \ldots$ with $\succ_\tau$. Now, we have to use (CW_i) and conclude. ■

Example 6 (Time validity in inference system for Example 1). Let us get back to our example and to make long example shorter, let us use less credentials: (1), (2), (10), (12), and (13). According to rule (CWP_1) we can infer:

$$\frac{F.students \leftarrow F.student \otimes F.student \in \mathscr{CP}}{\mathscr{CP} \succ\!\!\succ F.students \leftarrow F.student \otimes F.student}$$

$$\frac{F.activeSubject \leftarrow F.phdStudent \odot F.students \in \mathscr{CP}}{\mathscr{CP} \succ\!\!\succ F.activeSubject \leftarrow F.phdStudent \odot F.students}$$

$$\frac{F.student \leftarrow \{Betty\} \textbf{ in } v_2 \in \mathscr{CP}}{\mathscr{CP} \succ\!\!\succ_{v_2} F.student \leftarrow \{Betty\}}$$

$$\frac{F.student \leftarrow \{John\} \textbf{ in } v_4 \in \mathscr{CP}}{\mathscr{CP} \succ\!\!\succ_{v_4} F.student \leftarrow \{John\}}$$

$$\frac{F.phdStudent \leftarrow \{John\} \textbf{ in } v_5 \in \mathscr{CP}}{\mathscr{CP} \succ\!\!\succ_{v_5} F.phdStudent \leftarrow \{John\}}$$

When we want to check if two different students can cooperate, from credentials (1), (10), (12) and rule (CWP_6) we infer:

$$\frac{\begin{array}{c}\mathscr{CP} \succ\!\!\succ F.students \leftarrow F.student \otimes F.student \\ \mathscr{CP} \succ\!\!\succ_{v_2} F.student \leftarrow \{Betty\} \\ \mathscr{CP} \succ\!\!\succ_{v_4} F.student \leftarrow \{John\} \\ \{Betty\} \cap \{John\} = \phi\end{array}}{\mathscr{CP} \succ\!\!\succ_{v_2 \cap v_4} \mathbf{F.students} \leftarrow \{\mathbf{Betty}, \mathbf{John}\}}$$

In the next step we use it and additionally credentials (2), (13) and rule (CWP_5):

$$\frac{\begin{array}{c}\mathscr{CP} \succ\!\!\succ F.activeSubject \leftarrow F.phdStudent \odot F.students \\ \mathscr{CP} \succ\!\!\succ_{v_5} F.phdStudent \leftarrow \{John\} \\ \mathscr{CP} \succ\!\!\succ_{v_2 \cap v_4} F.students \leftarrow \{Betty, John\}\end{array}}{\mathscr{CP} \succ\!\!\succ_{v_2 \cap v_4 \cap v_5} \mathbf{F.activeSubject} \leftarrow \{\mathbf{Betty}, \mathbf{John}\}}$$

showing that the set of entities that can cooperatively activate a subject is: $\{Betty, John\}$ during the time: $v_2 \cap v_4 \cap v_5$.

5. Related Work

Traditional access control systems usually rely on RBAC model [1], [2], which groups the access rights by the role name and limits the access to a resource to those users, who are assigned to a particular role.

The term trust management was first applied in the context of distributed access control in [3]. The first trust management system described in the literature was PolicyMaker [11], which defined a special assertion language capable of expressing policy statements, which were locally trusted, and credentials, which had to be signed using a private key. The next generation of trust management languages were KeyNote [12], which was an enhanced version of PolicyMaker, SPKI/SDSI [13] and a few other languages [14]. All those languages allowed assigning privileges to entities and used credentials to delegate permissions from its issuer to its subject. What was missing in those languages was the possibility of delegation based on attributes of the entities and not on their identity.

Trust management, introduced in [3], has evolved since that time to a much broader context of assessing the reliability and developing trustworthiness for other systems and individuals [4]. In this paper, however, we used the term trust management only in a meaning restricted to the field of access control.

The meaning of roles in RT captures the notion of groups of users in many systems and has been borrowed from RBAC approach. The core language of RT family is RT_0, described in detail in [7]. It allows describing localized authorities for roles, role hierarchies, delegation of authority over roles and role intersections. All the subsequent languages add new features to RT_0. A more detailed overview of the RT family framework can be found in [5], [6], [15].

Time-dependant credentials were introduced in [8] but just for RT_0 language. Because RT^T language is more complex, powerful and it allows to express security policies more suited to real needs, we decided to develop extensions to this specific language, which has not been done before.

6. Conclusions

This paper deals with modeling of trust management systems in decentralized and distributed environments. The modelling framework is the RT^T language from a family of role-based trust management. Three types of semantics for a set of RT^T credentials have been introduced in the paper. A set-theoretic semantics of RT^T has been defined as a relation over a set of roles and a power set (set of sets) of entities. All the members of a set of entities related to a role must cooperate in order to satisfy the role. In the case of logic-programming semantics, RT credentials are translated into a logic program. This way, our definitions cover the full potential of RT^T, which supports the notion of manifold roles and it is able to express structure of threshold and separation-of-duties policies. Using RT^T one can define credentials stating that an action is allowed if it gets approved by members of more than one role. This enables defining complex trust management models in a real environment. An operational semantics of RT^T is defined as a inference system, in which credentials can be established from an initial set of credentials using a simple set of inference rules. The core part of the paper is a formal definition of a sound and complete inference system, in which credentials can be derived from an initial set of credentials using a set of inference rules. The semantics is given by the set of resulting credentials of the type $A.r \leftarrow X$, which explicitly show a mapping between roles and sets of entities. Using RT^T_+ one can define credentials, which state that an action is allowed if it gets approval from members of more than one role. This improves the possibility of defining complex trust management models in a real environment. The goal of this paper is the introduction of time validity constraints to show how that can make RT^T language more realistic. The properties of soundness and completeness of the inference system with respect to the semantics of RT^T_+ are proven. Inference systems presented in this paper are simple, but well-founded theoretically. It turns out to be fundamental mainly in large-scale distributed systems, where users have only partial view of their execution context.

Acknowledgements

This work is partially funded by the Polish Ministry of Science and Higher Education as part of project “Secure workstation for special applications”, grant number OR00014011.

References

[1] D. F. Ferraiolo, R. S. Sandhu, S. I. Gavrila, D. R. Kuhn, and R. Chandramouli, “Proposed NIST standard for role-based access control”, *ACM Trans. Inf. Syst. Secur.*, no. 3, pp. 224–274, 2001.

[2] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, “Role-based access control models”, *IEEE Computer*, no. 2, pp. 38–47, 1996.

[3] M. Blaze, J. Feigenbaum, and J. Lacy, “Decentralized trust management”, in *Proc. 17th IEEE Symp. Secur. Privacy*, Oakland, CA, USA, 1996, pp. 164–173.

[4] W. M. Grudzewski, I. K. Hejduk, A. Sankowska, and M. Wańtuchowicz, *Trust Management in Virtual Work Environments: A Human Factors Perspective*. CRC Press Taylor & Francis Group, 2008.

[5] N. Li and J. Mitchell, “RT: a role-based trust-management framework”, in *Proc. 3rd DARPA Inform. Surviv. Conf. Exp.*, IEEE Computer Society Press, Oakland, CA, USA, 2003, pp. 201–212.

[6] N. Li, J. Mitchell, and W. Winsborough, “Design of a role-based trust-management framework”, in *Proc. IEEE Symp. Secur. Privacy*, IEEE Computer Society Press, Oakland, CA, USA, 2002, pp. 114–130.

[7] N. Li, W. Winsborough, and J. Mitchell, “Distributed credential chain discovery in trust management”, *J. Comput. Secur.*, no. 1, pp. 35–86, 2003.

[8] D. Gorla, M. Hennessy, and V. Sassone, “Inferring dynamic credentials for role-based trust management”, in *Proc. 8th ACM SIGPLAN Conf. Princip. Pract. Declar. Program. PPDP 2006*, Venice, Italy, 2006, pp. 213–224.

[9] A. Felkner and K. Sacha, “The semantics of role-based trust management languages”, in *Proc. CEE-SET 2009*, Kraków, Poland, 2009, pp. 195–206, (preprints).

[10] A. Felkner and K. Sacha, “Deriving RT^T credentials for role-based trust management”, *e-Inf. Softw. Engin. J.*, vol. 4, pp. 9–19, 2010.

[11] M. Blaze, J. Feigenbaum, and M. Strauss, "Compliance checking in the policymaker trust management system", in *Proc. 2nd Int. Conf. Financial Cryptography*, London, United Kingdom, 1998, pp. 254–274.

[12] M. Blaze, J. Feigenbaum, and A. D. Keromytis, "The role of trust management in distributed systems security", in *Secure Internet Programming*, J. Vitek, C. D. Jensen, Eds. Springer, 1999, pp. 185–210.

[13] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, and R. L. Rivest, "Certificate chain discovery in SPKI/SDSI", *J. Comp. Secur.*, no. 9, pp. 285–322, 2001.

[14] P. Chapin, C. Skalka, and X. S. Wang, "Authorization in trust management: features and foundations", *ACM Comp. Surv.*, no. 3, pp. 1–48, 2008.

[15] M. R. Czenko, S. Etalle, D. Li, and W. H. Winsborough, "An Introduction to the Role Based Trust Management Framework RT", Tech. Rep. TR-CTIT-07-34, Centre for Telematics and Information Technology University of Twente, Enschede, 2007.

Anna Felkner graduated from the Faculty of Computer Science of Białystok University of Technology (M.Sc., 2004) and the Faculty of Electronics and Information Technology of Warsaw University of Technology (Ph.D., 2010). At present she is an Assistant Professor at Network and Information Security Methods Team in NASK Research Division. Main scientific interests concern the security of information systems, especially access control and trust management.
E-mail: anna.felkner@nask.pl
Research and Academic Computer Network (NASK)
Wąwozowa st 18
02-796 Warsaw, Poland

Adam Kozakiewicz got his M.Sc. in Information Technology and Ph.D. in Telecommunications at the Faculty of Electronics and Information Technology of Warsaw University of Technology, Poland. Currently he works at NASK as Assistant Professor and Manager of the Network and Information Security Methods Team, also as part-time Assistant Professor at the Institute of Control and Computation Engineering at the Warsaw University of Technology. His main scientific interests include security of information systems, parallel computation, optimization methods and network traffic modeling and control.
E-mail: adam.kozakiewicz@nask.pl
Research and Academic Computer Network (NASK)
Wąwozowa st 18
02-796 Warsaw, Poland