The Semantics of Role-Based Trust Management Languages

Anna Felkner[1,2] and Krzysztof Sacha[1]

[1] Warsaw University of Technology,
Nowowiejska 15/19, 00–665 Warszawa, Poland
[2] Research and Academic Computer Network
Wąwozowa 18, 02-796 Warszawa, Poland
anna.felkner@gmail.com, k.sacha@ia.pw.edu.pl

Abstract. Role-based Trust management (RT) languages are used for representing policies and credentials in decentralized, distributed access control systems. RT languages combine trust management and role-based access control features. A credential provides information about the keys, rights and qualifications from one or more trusted authorities. The paper presents a set-theoretic semantics of Role-based Trust management languages, which maps a role to a set of sets of entity names. The semantics applies not only to the basic language of the family RT_0, but also to a much more sophisticated RT^T, which provides manifold roles and role-product operators to express threshold and separation-of-duty policies. A manifold role defines sets of entities whose cooperation satisfies the manifold role. It enables to express a such a condition, which need more than one member of a role to effectively fulfill the particular task.

1 Introduction

The problem of guaranteeing that confidential data stored in computer systems is not disclosed to unauthorized users is increasingly significant for the owning organizations and for the society. A usual solution to this problem is an implementation of some access control techniques, by which users are identified, and granted or denied access to a system data and other resources.

Traditional access control schemes, like Mandatory Access Control (MAC), Discretionary Access Control (DAC) and Role-Based Access Control (RBAC), make authorization decisions based on the identity, or the role of the requester, who must be known to the resource owner. The most flexible of those schemes is role-based access control system [15,7,8], which groups the access rights by the role name and grants access to a resource to those users only, who are assigned to a particular role. This type of access control works well in a centralized system and is often used in enterprise environments.

Quite new problems arise in decentralized, distributed and open systems, where the identity of the users is not known in advance and the set of users can change. For an example, consider a university, in which the students are enrolled and registered to particular faculties, and no central registry of all the students of that university exists. The policy of the university is such that a student is eligible to attend a lecture given at each faculty, regardless of the faculty to which he or she is actually registered. However,

T. Szmuc, M. Szpyrka, and J. Zendulka (Eds.): CEE-SET 2009, LNCS 7054, pp. 179–189, 2012.
© IFIP International Federation for Information Processing 2012

how could a faculty (the lecture owner) know that John Smith is eligible to attend the lecture, if his name is unknown to this faculty? The identity of the student itself does not aid in making a decision whether he or she is eligible to attend or not. What is needed to make such a decision is information about the rights assigned to John Smith by other authorities (is he or she registered to a faculty), as well as trust information about the authority itself (is the faculty a part of this university).

To overcome the drawbacks of traditional access control schemes, trust management models have been proposed [1,2,3,4,5,13], as an approach to make access decisions in decentralized and distributed systems. Trust management is a specific kind of access control, in which decisions are based on credentials (certificates) 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, social security cards, driver's licenses, membership cards, academic diplomas, certifications, security clearances, passwords and user names, keys, etc. A credential in a computer system can be a digitally signed document.

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 of deciding on who is authorized to access a resource, and who is not.

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 [12,11,14]. The family consists of five languages: RT_0, RT_1, RT_2, RT^T, RT^D, with increasing expressive power and complexity. All the languages have a precise syntax definition, but a satisfactory semantics definition is still missing. 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 only [14,9]. In this paper we define an elegant relational semantics, which applies not only to RT_0, but also to other members of the family up to RT^T.

The paper is structured as follows. The family of Role-based Trust management languages is described in Section 2 (including examples). Section 3, which is the core part of this paper, presents the set-theoretic semantics of RT^T language. Final remarks and plans for future work are given in Conclusions

2 Role-Based Trust Management Languages

Role-based Trust management languages are used for representing policies in distributed authorization systems. The languages combine features from trust management and role-based access control, and define a family of models of trust management systems with varying expressiveness and complexity.

All the RT languages use the notion of a role to define sets of entities, which are members of this role. Entities in RT languages correspond to users in RBAC. Roles in RT can represent both - roles and permissions from RBAC. Moreover, RT_1 and RT_2 introduce attributes of a role, in an attempt to fulfill the Attribute-Based Access

Table 1. Supported features of RT languages

RT language	Supported features
RT_0	- localized authorities for roles, - role hierarchies, - delegation of authority over roles, - attribute based delegation of authority, - role intersections.
RT_1	features of RT_0 plus: - parameterized roles, - attribute-relationship based delegation, - attribute-field constraints.
RT_2	features of RT_1 plus: - logical objects.
RT^T	features of RT_0 plus: - manifold roles, - threshold policies, - separation-of-duty policies.
RT^D	features of RT_0 plus: - selective use of role membership, - dynamic credential delegation.

Control (ABAC) requirements. In ABAC systems, access control decisions are based on authenticated attributes of the entities.

RT_0 is the core language of RT family, described in detail in [14]. All the subsequent languages add new features to RT_0. A summary of the features supported by particular RT languages is shown in Table 1.

RT_1 adds to RT_0 parameterized roles, each of which can be described by a set of attributes. The attributes are typed, and can be integers, enumerations, floating point values, dates and times.

RT_2 further extends RT_1 to provide a notion of logical objects, which can group logically related entities, so that permissions to access specific resources can be assigned to them together.

RT^T provides manifold roles and role-product operators, which can express threshold and separation-of-duty policies. A manifold role is a role that can be satisfied by a set of cooperating entities, e.g. in a requirement that two different bank cashiers must authorize a transaction. A single-element role can be treated as a special case of a manifold role, whose set of cooperating entities is the singleton. Threshold policies require a specified minimum number of entities to agree on some fact. The concept of separation-of-duty is related to threshold policies. In the case of a separation-of-duty policy, entities from different sets must agree before access is granted. It means that some transactions can not be completed by a single entity. This implies that no single entity can have all the access rights required to complete such a transaction.

RT^D provides mechanism to describe delegation of rights and role activations, which can express selective use of capacities and delegation of these capacities. The semantics of this language is not covered in this paper.

The features of RT^T and RT^D can be combined together with the features of RT_0, RT_1 or RT_2. There are also few other languages based on RT_0, which have not been taken into account here. A more detailed overview of the Role-based Trust management family framework can be found in [12].

2.1 The Syntax of RT Languages

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 be 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 permissions issued by particular issuers. A role is described as a pair composed of an entity and a role name. **Credentials** define roles by pointing a new member of the role or by delegating authority to the members of other roles.

Table 2. Syntax of RT family

Language element	Notation
Entity name	$A, B, C \in \mathcal{E}$
Set of entieties	$U, V, W \subseteq \mathcal{E}$
Role name	$r, s, t \in \mathcal{R}$
Role	$A.r, B.s, C.t \in \mathcal{E} \times \mathcal{R}$
Role expression	$e ::= B \mid B.s \mid B.s.t \mid B.s \cap C.t \mid B.s \odot C.t \mid B.s \otimes C.t$
Credential	$c ::= A.r \leftarrow e$

In this paper, we use capital letters to denote entities and sets of entities (Table 2). Role names are denoted as identifiers beginning with a small letter. Roles take the form of an entity, or a set of entities, followed by a role name separated by a dot, e.g. A.r. Role expressions and credentials shown in Table 2 should be interpreted in the following way:

$A.r \leftarrow B$ – *simple member* – 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, as 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*.

$A.r \leftarrow B.s \cap C.t$ – *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*.

$A.r \leftarrow B.s \odot C.t$ role $A.r$ includes one member of role $B.s$ and one member of role $C.t$. This allows expressing the structure of a threshold.

$A.r \leftarrow B.s \otimes C.t$ 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. It enables to express separation-of-duty policies.

2.2 Examples

The models discussed in this paper can be, in general, very complex. Therefore, we present here only simplified examples, with the intention to illustrate the basic notions and the notation. The first example demonstrates the use of RT_0 credentials, the second and the third ones show the use of RT_1 credentials and the fourth example presents the use of RT^T credentials.

Example 1 (RT_0). A person has the right to attend a $lecture$, given at a university U, when he or she is a $student$ registered to a faculty of this university. To be able to fulfill the role of a $faculty$, an organization ought to be a $division$ of the university and should conduct $research$ activities. $John$ is a student registered to F, which is a $division$ of U, and which conducts $research$ activities. The following credentials prove that $John$ have the right to attend a $lecture$:

$$U.lecture \leftarrow U.faculty.student\,, \tag{1}$$
$$U.faculty \leftarrow U.division \cap U.research\,, \tag{2}$$
$$U.division \leftarrow F\,, \tag{3}$$
$$U.research \leftarrow F\,, \tag{4}$$
$$F.student \leftarrow John\,. \tag{5}$$

Example 2 (RT_1). The following example has been taken from [12]. A state university U, founded in 1955, gives special $privileges$ to graduates, who received a $diploma$ during the first four years of its operation, no matter which degree was conferred.

Such a policy can be expressed by a single credential with attributes assigned to a role:

$$U.privilages \leftarrow U.diploma(?, ?Year : [1955..1958])\,. \tag{6}$$

In this example $diploma$ is a role name that takes two parameters: The $degree$ and the $year$ of issue. The first question mark shows that the first attribute is insignificant. The second attribute ($year$), however, should take the values from 1955 through 1958.

Example 3 (RT_1). $John$ wants to share $pictures$ with his $friends$. However, he decided to restrict the access to his pictures to people over age 15.

$$John.pictures \leftarrow John.friends(?Age : [15..120])\,. \tag{7}$$

In this example, the acceptable values of the attribute Age are restricted to be in the range from 15 through 120.

Example 4 (RT^T). The following example has been adopted from [11]. A bank B has three roles: $manager$, $cashier$ and $auditor$. Security policy of the bank requires an $approval$ of certain transactions from a $manager$, two $cashiers$, and an $auditor$. The two $cashiers$ must be different. However, a $manager$ who is also a $cashier$ can serve as one of the two cashiers. The $auditor$ must be different from the other parties in the transaction.

Such a policy can be described using the following credentials:

$$B.twoCashiers \leftarrow B.cashier \otimes B.cashier\,, \tag{8}$$

$$B.managerCashiers \leftarrow B.manager \odot B.twoCashiers\,, \tag{9}$$

$$B.approval \leftarrow B.auditor \otimes B.managerCashiers\,. \tag{10}$$

Now, assume that the following credentials have been added:

$$B.cashier \leftarrow Mary\,, \tag{11}$$

$$B.cashier \leftarrow Doris\,, \tag{12}$$

$$B.cashier \leftarrow Alice\,, \tag{13}$$

$$B.cashier \leftarrow Kate\,, \tag{14}$$

$$B.manager \leftarrow Alice\,, \tag{15}$$

$$B.auditor \leftarrow Kate\,. \tag{16}$$

Then one can conclude that, according to the policy of B, the following sets of entities can cooperatively approve a transaction: $\{Mary, Doris, Alice, Kate\}$, $\{Mary, Alice, Kate\}$ and $\{Doris, Alice, Kate\}$.

3 Set-Theoretic Semantics of RT Languages

The syntax of a language defines language expressions, which are constructs that are used to communicate information [10]. The primary expressions of Role-based Trust management languages are credentials and sets of credentials, which are used as a means for defining roles.

The semantics of a language defines the meaning of expressions. Such a definition consists of two parts [10]: A semantic domain and a semantic mapping from the syntax to the semantic domain. The meaning of a language expression must be an element in the semantic domain.

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 semantic domain of a Role-based Trust management language. The semantic mapping would associate a specific relation between roles and entities with each set of credentials. Such a relational approach allows us to define a formal semantics of RT^T language.

3.1 The Semantics of RT_0

A set-theoretic semantics of RT_0, 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 originally defined in [14]. A definition quoted in this subsection is a modified version of the same semantics, which has been introduced in [9].

Definition 1. *The semantics of a set of credentials $\mathcal{P}$ is the least fixpoint of the following sequence of functions, which map roles to sets of entity names:*

1. *R_0 maps each role to an empty set ϕ,*
2. *$R_{i+1} \triangleq \bigoplus_{c \in \mathcal{P}} f(R_i, c)$,*

where $\bigoplus$ is the point-wise extension of a function and f is a function that, given a (partial) semantics R_i and a credential $A.r \leftarrow e$, returns all the entities that should be added to $R_i(A.r)$, as governed by e:

$$f(R_i, A.r \leftarrow B) \triangleq \{A.r \mapsto \{B\}\}\,, \tag{17}$$

$$f(R_i, A.r \leftarrow B.s) \triangleq \{A.r \mapsto R_i(B.s)\}\,, \tag{18}$$

$$f(R_i, A.r \leftarrow B.s.t) \triangleq \{A.r \mapsto \textstyle\bigcup_{C \in R_i(B.s)} R_i(C.t)\}\,, \tag{19}$$

$$f(R_i, A.r \leftarrow B.s \cap C.t) \triangleq \{A.r \mapsto R_i(B.s) \cap R_i(C.t)\}\,. \tag{20}$$

Although it has not been stated explicitly in [9], one can see that the argument of function R_i is a composition of an entity and a role name, and the value of function R_i is a subset of entities. Hence, the domain of function R_i is a cartesian product of the sets of entities $\mathcal{E}$ and role names $\mathcal{R}$, and the range of function R_i is the power set of entities:

$$R_i : \mathcal{E} \times \mathcal{R} \rightarrow 2^{\mathcal{E}}\,. \tag{21}$$

Such a functional semantics has no potential to describe the meaning of RT^T, which supports manifold roles and role-product operators.

3.2 The Semantics of RT^T

Let $\mathcal{E}$ be the set of entities and $\mathcal{R}$ be the set of role names. $\mathcal{P}$ is a set of RT-credentials, which describe the assignment of sets of entities to roles, issued by other entities (or rather sets of entities).

The semantics of $\mathcal{P}$, denoted by $\mathcal{S}_\mathcal{P}$, is defined as a relation:

$$\mathcal{S}_\mathcal{P} \subseteq 2^{\mathcal{E}} \times \mathcal{R} \times 2^{\mathcal{E}}\,, \tag{22}$$

An instance of this relation, e.g.: $(\{A\}, r, U)$, maps the role $\{A\}.r$ governed by entity $A \in \mathcal{E}$ to a set of entities $U \in 2^{\mathcal{E}}$. The entities of set U must cooperate together in order to satisfy the role. If the cardinality of set U is greater than one, the role $\{A\}.r$ is a manifold role. In case of RT_0, which does not support manifold roles, all sets of entities are singleton sets.

Another instance of the relation, e.g.: $(\{A, B\}, r, U)$, maps the role $\{A, B\}.r$ governed jointly by two entities $\{A, B\} \in 2^{\mathcal{E}}$ to a set of entities $U \in 2^{\mathcal{E}}$.

Denote the power set of entities by $\mathcal{F} = 2^{\mathcal{E}}$. Each element in $\mathcal{F}$ is a set of entities from $\mathcal{E}$ (a subset of $\mathcal{E}$). Each element in $2^{\mathcal{F}}$ is a set, compound of sets of entities from $\mathcal{E}$.

The semantics of $\mathcal{P}$ can now be described in an alternative way as a function:

$$\widetilde{\mathcal{S}}_{\mathcal{P}} : 2^{\mathcal{E}} \times \mathcal{R} \to 2^{\mathcal{F}}\,, \tag{23}$$

which maps each role from $2^{\mathcal{E}} \times \mathcal{R}$ into a set of subsets of entities. The members of each subset must cooperate in order to satisfy the role.

Knowing the relation $\mathcal{S}_{\mathcal{P}}$, one can define the function $\widetilde{\mathcal{S}}_{\mathcal{P}}$ as follows:

$$\widetilde{\mathcal{S}}_{\mathcal{P}}(U, r) = \{V \in 2^{\mathcal{E}} : (U, r, V) \in \mathcal{S}_{\mathcal{P}}\}\,. \tag{24}$$

The semantics of RT^T can now be defined formally in the following way.

Definition 2. *The semantics of a set of credentials $\mathcal{P}$, denoted by $\mathcal{S}_{\mathcal{P}}$, is the smallest relation S_i, such that:*

1. $\mathcal{S}_0 = \phi$,
2. $\mathcal{S}_{i+1} = \bigcup_{c \in \mathcal{P}} f(\mathcal{S}_i, c)$, *for* $i = 0, 1, \ldots$,

which is closed with respect to function f, which describes the meaning of credentials in the following way ($U, V, W, \ldots$ are sets of entities, may be singletons):

$$f(\mathcal{S}_i, U.r \leftarrow V) = \{(U, r, V)\}\,, \tag{25}$$

$$f(\mathcal{S}_i, U.r \leftarrow V.s) = \{(U, r, W) : (V, s, W) \in \mathcal{S}_i\}\,, \tag{26}$$

$$f(\mathcal{S}_i, U.r \leftarrow V.s.t) = \bigcup_{W:(V,s,W)\in\mathcal{S}_i} \{(U, r, X) : (W, t, X) \in \mathcal{S}_i\}\,, \tag{27}$$

$$f(\mathcal{S}_i, U.r \leftarrow V.s \cap W.t) = \{(U, r, X) : (V, s, X) \in \mathcal{S}_i \wedge (W, t, X) \in \mathcal{S}_i\}\,, \tag{28}$$

$$f(\mathcal{S}_i, U.r \leftarrow V.s \odot W.t) = \{(U, r, X \cup Y) : (V, s, X) \in \mathcal{S}_i \wedge (W, t, Y) \in \mathcal{S}_i\} \tag{29}$$

$$f(\mathcal{S}_i, U.r \leftarrow V.s \otimes W.t) = \{(U, r, X \cup Y) : (V, s, X) \in \mathcal{S}_i \wedge (W, t, Y) \in \mathcal{S}_i \wedge (X \cap Y) = \phi\}\,. \tag{30}$$

3.3 Examples

We use example 1 and example 4 from section 2.2 to illustrate the definition of RT semantics.

Example 1 (RT_0). The sequence of steps to compute consecutive relations $\mathcal{S}_i$ is shown in Table 3. Consecutive sections of the table describe relations $\mathcal{S}_0$ through $\mathcal{S}_3$. The rows of the table correspond to entities (principals) and the columns correspond to role names. This way, a cell of the table shows the set of entities, which are members of the respective role issued by the corresponding principal.

Table 3. The relations $\mathcal{S}_0$ through $\mathcal{S}_3$

Relation	Entity	lecture	faculty	student	division	research
$\mathcal{S}_0$	**U**	ϕ	ϕ	ϕ	ϕ	ϕ
	F	ϕ	ϕ	ϕ	ϕ	ϕ
	John	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{S}_1$	**U**	ϕ	ϕ	ϕ	$\{F\}$	$\{F\}$
	F	ϕ	ϕ	$\{John\}$	ϕ	ϕ
	John	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{S}_2$	**U**	ϕ	$\{F\}$	ϕ	$\{F\}$	$\{F\}$
	F	ϕ	ϕ	$\{John\}$	ϕ	ϕ
	John	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{S}_3$	**U**	$\{John\}$	$\{F\}$	ϕ	$\{F\}$	$\{F\}$
	F	ϕ	ϕ	$\{John\}$	ϕ	ϕ
	John	ϕ	ϕ	ϕ	ϕ	ϕ

The starting relation $\mathcal{S}_0$ is, by definition, empty. According to Definition 2, only credentials 3, 4 and 5, are mapped in $\mathcal{S}_0$ into nonempty sets by function f. These sets are shown in relation $\mathcal{S}_1$ in Table 3. In $\mathcal{S}_1$, credential 2 is mapped into instance $(\{U\}, faculty, \{F\})$ of relation $\mathcal{S}_2$, and in $\mathcal{S}_2$, credential 1 is mapped into instance $(\{U\}, lecture, \{John\})$. The resulting relation $\mathcal{S}_3$ cannot be changed using the given set of credentials, hence:

$$\mathcal{S}_P = \mathcal{S}_3 \,. \tag{31}$$

Because the RT language considered in this example is RT_0, all the sets of entities assigned to roles are singleton sets.

Example 4 (RT^T). The sequence of steps to compute consecutive relations $\mathcal{S}_i$ in this example can be represented in a table similar to Table 3. However, all the roles in this example are issued by a single entity B, hence; there is no use of showing other entities. Therefore, each section of Table 4 has exactly one row, which corresponds to B. The columns of the table correspond to role names. A cell of the table shows the set of sets of entities that cooperatively can satisfy the respective role issued by B.

Credentials 11 through 16 are mapped in $\mathcal{S}_0$ into relation:

$$\begin{aligned}\mathcal{S}_1 = \{\ &(\{B\}, cashier, \{Mary\}),\ (\{B\}, cashier, \{Doris\}),\\ &(\{B\}, cashier, \{Alice\}),\ (\{B\}, cashier, \{Kate\}),\\ &(\{B\}, manager, \{Alice\}),\ (\{B\}, auditor, \{Kate\})\ \}\,.\end{aligned} \tag{32}$$

The mapping of credential 8 in $\mathcal{S}_1$ adds the following instances:

$$\begin{aligned}\mathcal{S}_2 = \mathcal{S}_1 \cup \{\ &\\ &(\{B\}, twoCashiers, \{Mary, Doris\}),\ (\{B\}, twoCashiers, \{Mary, Alice\}),\\ &(\{B\}, twoCashiers, \{Mary, Kate\}),\ (\{B\}, twoCashiers, \{Doris, Alice\}),\\ &(\{B\}, twoCashiers, \{Doris, Kate\}),\ (\{B\}, twoCashiers, \{Alice, Kate\})\ \}\,.\end{aligned} \tag{33}$$

Table 4. The relations $\mathcal{S}_0$ through $\mathcal{S}_4$

	cashier	**manager**	**auditor**	**twoCashiers**	**managerCashiers**	**approval**
$\mathcal{S}_0$	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
$\mathcal{S}_1$	$\{Mary\}$ $\{Doris\}$ $\{Alice\}$ $\{Kate\}$	$\{Alice\}$	$\{Kate\}$	ϕ	ϕ	ϕ
$\mathcal{S}_2$	$\{Mary\}$ $\{Doris\}$ $\{Alice\}$ $\{Kate\}$	$\{Alice\}$	$\{Kate\}$	$\{Mary, Doris\}$ $\{Mary, Alice\}$ $\{Mary, Kate\}$ $\{Doris, Alice\}$ $\{Doris, Kate\}$ $\{Alice, Kate\}$	ϕ	ϕ
$\mathcal{S}_3$	$\{Mary\}$ $\{Doris\}$ $\{Alice\}$ $\{Kate\}$	$\{Alice\}$	$\{Kate\}$	$\{Mary, Doris\}$ $\{Mary, Alice\}$ $\{Mary, Kate\}$ $\{Doris, Alice\}$ $\{Doris, Kate\}$ $\{Alice, Kate\}$	$\{Mary, Doris, Alice\}$ $\{Mary, Alice\}$ $\{Mary, Kate, Alice\}$ $\{Doris, Alice\}$ $\{Doris, Kate, Alice\}$ $\{Alice, Kate\}$	ϕ
$\mathcal{S}_4$	$\{Mary\}$ $\{Doris\}$ $\{Alice\}$ $\{Kate\}$	$\{Alice\}$	$\{Kate\}$	$\{Mary, Doris\}$ $\{Mary, Alice\}$ $\{Mary, Kate\}$ $\{Doris, Alice\}$ $\{Doris, Kate\}$ $\{Alice, Kate\}$	$\{Mary, Doris, Alice\}$ $\{Mary, Alice\}$ $\{Mary, Kate, Alice\}$ $\{Doris, Alice\}$ $\{Doris, Kate, Alice\}$ $\{Alice, Kate\}$	$\{Mary, Doris$ $Alice, Kate\}$ $\{Mary, Alice, Kate\}$ $\{Doris, Alice, Kate\}$

The mappings of credentials 9 in $\mathcal{S}_2$ and 10 in $\mathcal{S}_3$ can be calculated analogously. The six sets in column **managerCashiers** are the union sets of set $\{Alice\}$ and the six sets from column **twoCashiers**. The three sets in column **approval** are the union sets of set $\{Kate\}$ and these sets from column **managerCashiers**, which are disjoint with set $\{Kate\}$. The resulting relation $\mathcal{S}_4$ cannot be changed using the given set of credentials, hence: $\mathcal{S}_P = \mathcal{S}_4$.

Because the RT language considered in this example is RT^T, there are sets of sets of entities assigned to roles. The interpretation of results shown in Table 4 is such that there are three sets of entieties, enumerated in the right bottom cell of the table, which can cooperatively approve a transaction.

4 Conclusions

This paper deals with modeling of trust management systems in decentralized and distributed environments. The modeling framework is a family of Role-based Trust management languages.

The core part of the paper is a definition of formal semantics for a set of Role-based Trust management credentials, which is based on a set-theoretic interpretation. The semantics has been defined as a relation between roles and sets of entities. Members of such a set must cooperate in order to satisfy the role. This way, our definition covers not only the basic RT_0 language, but also the more powerful RT^T, which provides the notion of manifold roles and is able to express structure of threshold and separation-of-duty policies. 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.

References

1. Blaze, M., Feigenbaum, J., Lacy, J.: Decentralized Trust Management. In: Proc. of the 17th IEEE Symposium on Security and Privacy, pp. 164–173. IEEE Computer Society Press, Oakland CA (1996)
2. Blaze, M., Feigenbaum, J., Keromytis, A.D.: The Role of Trust Management in Distributed Systems Security. In: Ryan, M. (ed.) Secure Internet Programming. LNCS, vol. 1603, pp. 185–210. Springer, Heidelberg (1999)
3. Chadwick, D., Otenko, A., Ball, E.: Role-Based Access Control with X.509 Attribute Certificates. IEEE Internet Comput. 2, 62–69 (2003)
4. Chapin, P., Skalka, C., Wang, X.S.: Authorization in Trust Management: Features and Foundations. ACM Comput. Surv. 3, 1–48 (2008)
5. Felkner, A.: Modeling Trust Mangement in Computer Systems. In: Proc. of the 9th Int. PhD Workshop OWD 2007, Conference Archives PTETiS, vol. 23, pp. 65–70 (2007)
6. Felkner, A.: Set-Theoretic Semantics of Role-Based Trust Management. In: Proc. of the 10th Int. PhD Workshop OWD 2008, Conference Archives PTETiS, vol. 25, pp. 567–572 (2008)
7. Ferraiolo, D.F., Kuhn, D.R.: Role-based Access Control. In: Proc. of the 15th National Computer Security Conference, pp. 554–563 (1992)
8. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S.I., Kuhn, D.R., Chandramouli, R.: Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and System Security 3, 224–274 (2001)
9. Gorla, D., Hennessy, M., Sassone, V.: Inferring Dynamic Credentials for Role-Based Trust Management. In: Proc. of the 8th ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, pp. 213–224. ACM (2006)
10. Harel, D., Rumpe, B.: Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff. Weizmann Science Press of Israel, Jerusalem (2000)
11. Li, N., Mitchell, J.: RT: A Role-Based Trust-Management Framework. In: Proc. of the 3rd DARPA Information Survivability Conference and Exposition, pp. 201–212. IEEE Computer Society Press, Oakland CA (2003)
12. Li, N., Mitchell, J., Winsborough, W.: Design of a Role-Based Trust-Management Framework. In: Proc. of the IEEE Symposium on Security and Privacy, pp. 114–130. IEEE Computer Society Press, Oakland CA (2002)
13. Li, N., Winsborough, W., Mitchell, J.: Beyond Proof-of-Compliance: Safety and Availability Analysis in Trust Management. In: Proc. of the IEEE Symposium on Security and Privacy, pp. 123–139. IEEE Computer Society Press, Oakland CA (2003)
14. Li, N., Winsborough, W., Mitchell, J.: Distributed Credential Chain Discovery in Trust Management. Journal of Computer Security 1, 35–86 (2003)
15. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-Based Access Control Models. IEEE Computer 2, 38–47 (1996)