Paper

Two Extensions of Trust Management Languages

Anna Felkner

Information Security Methods Team, Research and Academic Computer Network (NASK), Warsaw, Poland

https://doi.org/10.26636/jtit.2020.138719

Abstract—This article is focused on the family of role-based trust management languages (RT). Trust management languages are a useful method of representing security credentials and policies in large distributed access control mechanisms. They provide sets of credentials that are assigned to individual roles performed by the specific entities. These credentials provide relevant information about security policies issued by trusted authorities and define user permissions. RT languages describe the individual entities and the roles that these entities play in a given environment. A set of credentials representing a given security policy defines which entity has the necessary rights to access a specific resource and which entity does not have such rights. This study presents the results of research focusing on the potential of the family of RT languages. Its purpose is to show how security policies may be applied more widely by applying an inference system, and then using the extensions of the credentials, by taking into account time-related information or the conditions imposed with regard to the validity of such credentials. Each of these extensions can be used jointly or separately, offering even a wider range of opportunities.

Keywords*—*access control, conditional credentials, inference system with time constraints.

1. Introduction

Confidential services and data stored in a computer system should be available to authorized users only. Therefore, attempting to guarantee that kind of access control is an extremely important and difficult task. This problem should be resolved by using reliable and trusted software technologies that are relied upon to design high-integrity applications.

Traditional solutions used in the aforementioned scenarios include such access control as discretionary access control (DAC) [2], mandatory access control (MAC) [1], and role-based access control (RBAC) [3]) (which offers the highest degree of flexibility). These methods are characterized by the fact that after identification of the user (or the role they play), access to the resources or the system is granted or not. The owner of the specific resource must first know the identity of the requester. This approach may turn out to be sufficient in systems in which user identities are known. Unfortunately, this is not the case with open systems, where the user's identity is not known in advance, and where the resource owner and the requester often do not know each other. For example, if you are a student and you have a city card, you can use a scooter for free. When such a student visits a place renting electric scooters, it does not matter who they are. Their identity will not be required in order to get a free scooter ride. In this case, two types of credentials are relevant. One confirming that the person is a student, and the other in the form of a city card. Your identity is not useful for making the decision about the rights you are entitled to. The decision can be made based on a certificate or other credentials informing about your rights. These certificates must be issued by the relevant authority. Therefore, a different approach to managing access control has been defined.

Blaze *et al.* [4] defined *trust management* as a unified approach to define and interpret security credentials, policies and trust relationships. A *credential* certifies certain qualifications, competences or authority issued to somebody by a third party. Credentials provide information about the user's permissions and the security policies issued by at least one trusted authority. Identification documents, driver's licenses, academic diplomas, certificates, etc. may serve as good examples of credentials.

The task of the family of role-based trust management languages is to represent security credentials and policies in distributed access control systems, and to provide a set of credentials helping you assign roles to users. The core language of the family of RT languages is RT_0, as described in [5]. All subsequent languages add new features to RT_0. RT_1 adds parameterized roles to RT_0, representing relationships between the specific entities. RT_2 introduces logical objects that you can use to define permissions granted to entities that form a group of logically related objects. This article focuses on the RT^T language, because it provides useful features (not found in others languages): manifold roles and role-product operators that can express threshold values and separation of duties policies. A more detailed description of the RT family of languages may be found in [6].

Since the creation of the base RT family language, many different works have been created that are related to the potential uses of RT. These works include, inter alia, the role-based trust management model used in peer-to-peer networks [7], wireless sensor networks [8], [9] or cloud computing [10]. In this work, we present certain extensions of the most advanced RT family language. These extensions will enable the language to be used in many real-world situations. The study shows how the RT family of languages may become more easily applicable in real systems by enabling validity time limits and by introducing parameterization of credentials. So far, the proposed models did not allow for easy application of time restrictions in connection with security policies or credentials issued. It was also not easy to use conditional credentials. The proposed extensions of the basic language will be presented in the form of simple usage scenarios described in the subsequent sections of this document.

The remaining part of this paper is organized as follows: Section 2 describes the syntax of RT languages. The inference system over RT^T language is described in Section 3. Section 4 describes two extensions of the RT^T language (time validity and conditional credentials). Section 5 shows an inference system over the new RT^T_+ language. Use case scenarios that validate usability of the extensions are presented in Section 6. Final remarks are given in the conclusions.

2. The Syntax of *RT* Family Languages

Entities, roles, role names and credentials are the basic elements of RT languages. Entities may represent principals that define roles and issue credentials. They may also issue credentials and request access to resources. In a computer system, an entity may have the form of a program identified the user account. Roles are sets of entities with specific permissions, granted in accordance with the access control policy. Role names represent permissions that specific entities issue to other entities or entity sets. Credentials define a new role member (or a set of role members) and delegate authority to members of other roles.

The following are the types of credentials found in the RT family of languages. The first six appear in basic RT^T, and two additional ones are connected with the determination of the order in which credentials may appear. The individual credentials should be understood as follows:

$K.r \longleftarrow L$	– a member of role $K.r$ is entity L (simple membership).
$K.r \longleftarrow L.s$	– members of role $K.r$ are all members of role $L.s$ (simple inclusion).
$K.r \longleftarrow L.s.t$	– role $K.r$ contains role $M.t$ for each M, which is a member of role $L.s$ (linking inclusion).
$K.r \longleftarrow L.s \cap M.t$	– role $K.r$ contains all the entities that are members of role $L.s$ and role $M.t$ (intersection inclusion).
$K.r \longleftarrow L.s \odot M.t$	– to be a member of role $K.r$ you must be a union set of one member of role $L.s$ and one member of the role $M.t$.
$K.r \longleftarrow L.s \otimes M.t$	– role $K.r$ contains one member of role $L.s$ and one member of role $M.t$. The members of the roles must be different entities in this case.
$K.r \longleftarrow L.s \overset{\odot}{\rightarrow} M.t$	– to be a member of role $K.r$ you must be a union set of one member of role $L.s$ and one member of role $M.t$ – in this specific order.
$K.r \longleftarrow L.s \overset{\otimes}{\rightarrow} M.t$	– role $K.r$ contains one member of role $L.s$ and one member of role $M.t$ in this specific order. The members of the roles must be different entities in this case.

3. Inference System over RT^T Credentials

RT^T credentials define roles and roles represent permissions. The set of role members may be calculated by using an inference system. This system consists of an initial set of formulae recognized as true and a set of inference rules by means of which new formulae can be obtained.

If $\mathscr{S}$ is a set of given RT^T credentials, the application of inference rules in the inference system will allow to obtain new credentials from the given set $\mathscr{S}$. Derived credential s will be denoted by formula $\mathscr{S} \succ s$, which means that credential s may be obtained from set of credentials $\mathscr{S}$.

At the beginning, we have the following set of formulae: $s \in \mathscr{S}$ for each credential s in $\mathscr{S}$. The rules of the inference system are as follows:

$$\frac{s \in \mathscr{S}}{\mathscr{S} \succ s} \qquad (\mathbf{W_1})$$

$$\frac{\mathscr{S} \succ K.r \longleftarrow L.s \quad \mathscr{S} \succ L.s \longleftarrow X}{\mathscr{S} \succ K.r \longleftarrow X} \qquad (\mathbf{W_2})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s.t \qquad \mathscr{S} \succ L.s \longleftarrow M \\ \mathscr{S} \succ M.t \longleftarrow X\end{array}}{\mathscr{S} \succ K.r \longleftarrow X} \qquad (\mathbf{W_3})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s \cap M.t \qquad \mathscr{S} \succ L.s \longleftarrow X \\ \mathscr{S} \succ M.t \longleftarrow X\end{array}}{\mathscr{S} \succ K.r \longleftarrow X} \qquad (\mathbf{W_4})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s \odot M.t \qquad \mathscr{S} \succ L.s \longleftarrow X \\ \mathscr{S} \succ M.t \longleftarrow Y\end{array}}{\mathscr{S} \succ K.r \longleftarrow X \cup Y} \qquad (\mathbf{W_5})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s \otimes M.t \qquad \mathscr{S} \succ L.s \longleftarrow X \\ \mathscr{S} \succ M.t \longleftarrow Y \qquad X \cap Y = \phi\end{array}}{\mathscr{S} \succ K.r \longleftarrow X \cup Y} \qquad (\mathbf{W_6})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s \overset{\odot}{\rightarrow} M.t \qquad \mathscr{S} \succ L.s \longleftarrow X \\ \mathscr{S} \succ M.t \longleftarrow Y\end{array}}{\mathscr{S} \succ K.r \longleftarrow X \cup Y} \qquad (\mathbf{W_7})$$

$$\frac{\begin{array}{c}\mathscr{S} \succ K.r \longleftarrow L.s \overset{\otimes}{\rightarrow} M.t \qquad \mathscr{S} \succ L.s \longleftarrow X \\ \mathscr{S} \succ M.t \longleftarrow Y \qquad X \cap Y = \phi\end{array}}{\mathscr{S} \succ K.r \longleftarrow X \cup Y} \qquad (\mathbf{W_8})$$

The above inference system shows how new credentials can be inferred using all possible credentials found in RT^T. We add newly created credentials to the already existing set, thus expanding the initial set of credentials in a given security policy.

4. Credential Extensions

Here we will introduce two RT^T language extensions that may help apply this language more easily to systems with practical implications. These extensions include the validity of temporary credentials (time-based definition of policies and individual credentials is possible) and conditional credentials.

4.1. Time Validity in RT^T

The first extension shows how easy it is to limit the time of operation of a given credential. You can define different validity ranges for credentials. Time limits have already been used in RT_0 to some extent in [11]. However, this paper shows a different approach to time restrictions, so that they can be used in RT^T languages.
The introduction of such a restriction may result in obtaining a permission that is valid for a certain period of time only. It is quite natural to assume that permissions are not granted indefinitely. Temporary credentials can be written as; s **in** τ, which means that "the credential s is available at time τ". Time-dependent credentials sets are denoted by $\mathscr{TS}$. The newly created language is called RT^T_+. To lighten the notation, we can write s which means "s **in** $(-\infty,\ +\infty)$", i.e. credential without time limits.
The most commonly used trust management languages are monotonic, meaning that the addition of a new assertion to a query cannot cancel an action that has already been approved [12]. This precludes the use of permission negation. This is a potential source of problems, as any credential or policy statement added to the system can only increase the privileges and opportunities granted to others. This is a very impractical approach, although convenient for implementation and modeling. The time limits presented above can sometimes satisfy the need for negation without actually sacrificing the monotonicity of the system.
Time validity can be determined as follows:
$[t_1,t_2]; [t_1,t_2); (t_1,t_2]; (t_1,t_2); (-\infty,t]; (-\infty,t); [t,+\infty); (t,+\infty); (-\infty,\ +\infty); \tau_1 \cup \tau_2; \tau_1 \cap \tau_2; \tau_1 \setminus \tau_2$
and τ_1, τ_2 of any form in this list, with t ranging over time constants.

4.2. Conditional Credentials

The second, very intuitive and extremely important extension of the RT^T_+ language is connected with the fulfillment of the condition of one credential being based on availability or unavailability of another credential during the checking and implementation phase of this credential. This can be written as follows:

if $entity \in / \notin roles$ **then** $credential$

For example, we can use these credentials:

if $Kim{\in}L.controller\ and$
$Kim{\notin}L.specjalEmployees\ and$
$\{Claire,Rita\}{\in}L.specjalEmployees\ and$
$\{Claire,Rita\}{\notin}L.controller$
then $L.confirm \longleftarrow\ \{Claire,Rita,Kim\}$

to show that if Kim is a member of role $L.controller$ but not a member of role $L.specjalEmployees$, and a group of entities $\{Claire,Rita\}$ is a member of role $L.specjalEmployees$ but not a member of role $L.controller$, then the group of people consisting of $Claire$, $Rita$, and Kim can make the confirmation.

Another credential is:

if $Julia{\notin}L.active$ **then** $Julia.financial \longleftarrow L.assistspecialist$

stating that whenever $Julia$ is not an active worker (she is on holiday or is no longer working), her assistant can take care of all financial issues. $Julia's$ status is described during the execution context when the credential is used. The $Julia.financial \longleftarrow L.assistspecialist$ credential is available if $L.active \longleftarrow Julia$ cannot be inferred. If, in the execution context, we are able to obtain credential $L.active \longleftarrow Julia$ **in** τ, where τ is the validity time of the credential, then, credential $Julia.financial \longleftarrow L.assistspecialist$ is available at any time not included in τ.

This is also useful in a situation in which we want to check whether entity L is a member of role $K.r$ or not, and to add it if it is not.

if $L \notin\ K.r$ **then** $K.r \longleftarrow\ L$

if $Julia \notin\ L.active$ **then** $L.active \longleftarrow\ Julia$

which adds $Julia$ as an active worker.

This is a common conditional replacement scenario that is especially useful in the RT^T language, as the rule is used stating that a single person who plays an important role should be replaced by a pair (or more) of stand-ins acting together.

The two simple language extensions pertaining to the RT family, as presented above, although simple and intuitive, may considerably increase the applicability of RT languages in real systems. Both time constraints (which partly replace negation) and dependence on the enforceability of one credential based on the availability of another, find a lot of real system applications.

5. Inference System for RT^T Credentials with Time Validity

This section shows how we adjusted the RT^T inference system to the RT^T_+ credentials inference system.

If $\mathscr{TS}$ is a set of given RT_{+}^{T} credentials, the application of inference rules from the inference system will allow to obtain new credentials from the given set $\mathscr{TS}$. The obtained credential s valid during time t will be denoted by formula $\mathscr{TS} \succ_t s$, meaning that credential s may be derived from credential set $\mathscr{TS}$ at time t.

At the beginning, we have the following set of formulae: s **in** $\tau \in \mathscr{TS}$ for each credential s valid at time τ in $\mathscr{TS}$. The rules of the inference system are as follows:

$$\frac{s \textbf{ in } \tau \in \mathscr{TS} \quad t \in \tau}{\mathscr{TS} \succ_t s} \qquad (\mathbf{CW_1})$$

$$\frac{\mathscr{TS} \succ_t K.r \leftarrow L.s \quad \mathscr{TS} \succ_t L.s \leftarrow X}{\mathscr{TS} \succ_t K.r \leftarrow X} \qquad (\mathbf{CW_2})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s.t \quad \mathscr{TS} \succ_t L.s \leftarrow M \\ \mathscr{TS} \succ_t M.t \leftarrow X\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X} \qquad (\mathbf{CW_3})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s \cap M.t \quad \mathscr{TS} \succ_t L.s \leftarrow X \\ \mathscr{TS} \succ_t M.t \leftarrow X\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X} \qquad (\mathbf{CW_4})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s \odot M.t \quad \mathscr{TS} \succ_t L.s \leftarrow X \\ \mathscr{TS} \succ_t M.t \leftarrow Y\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X \cup Y} \qquad (\mathbf{CW_5})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s \otimes M.t \quad \mathscr{TS} \succ_t L.s \leftarrow X \\ \mathscr{TS} \succ_t M.t \leftarrow Y \qquad X \cap Y = \phi\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X \cup Y} \qquad (\mathbf{CW_6})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s \overset{\odot}{\to} M.t \quad \mathscr{TS} \succ_t L.s \leftarrow X \\ \mathscr{TS} \succ_t M.t \leftarrow Y\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X \cup Y} \qquad (\mathbf{CW_7})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ_t K.r \leftarrow L.s \overset{\otimes}{\to} M.t \quad \mathscr{TS} \succ_t L.s \leftarrow X \\ \mathscr{TS} \succ_t M.t \leftarrow Y \qquad X \cap Y = \phi\end{array}}{\mathscr{TS} \succ_t K.r \leftarrow X \cup Y} \qquad (\mathbf{CW_8})$$

The inference system presented above shows how new credentials can be inferred using all possible credentials found in RT_{+}^{T}. As it was the case in previous inference systems, we add the newly created credentials to those that already exist and, thus, expand our collection of credentials available within a specific security policy.

The first formula says that if we have credential s in the set of credentials $\mathscr{TS}$ at time τ and, at the same time, t is contained in τ, we can add credential $\mathscr{TS} \succ_t s$ to the set of our credentials.

The $(\mathbf{CW_2})$ formula says "if in the set of credentials $\mathscr{TS}$ credential $K.r \leftarrow L.s$ and credential $L.s \leftarrow X$ are available during the time t, then we can conclude that $K.r \leftarrow X$ is true during time t and we can add it to our set of credentials".

Way we proceed correspondingly by applying new credentials from the set of available $\mathscr{TS}$ credentials, which we add to our initial set.

5.1. Inferring Time Validity of Credentials

To determine the maximum validity time of a given credential s in the $\mathscr{TS}$ inference system, we need to strengthen the formula $\mathscr{TS} \succ_t s$ to $\mathscr{TS} \succ\!\succ_\tau s$, which means that in each time period $t \in \tau$ in which the credential set $\mathscr{TS}$ has semantics, you can deduce credential s from $\mathscr{TS}$. Below is the newly created inference system:

$$\frac{s \textbf{ in } \tau \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ_\tau s} \qquad (\mathbf{CWP_1})$$

$$\frac{\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \quad \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2} K.r \leftarrow X} \qquad (\mathbf{CWP_2})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow M \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow X\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X} \qquad (\mathbf{CWP_3})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \cap M.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow X\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X} \qquad (\mathbf{CWP_4})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \odot M.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow Y\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X \cup Y} \qquad (\mathbf{CWP_5})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \otimes M.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow Y \\ X \cap Y = \phi\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X \cup Y} \qquad (\mathbf{CWP_6})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \overset{\odot}{\to} M.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow Y\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X \cup Y} \qquad (\mathbf{CWP_7})$$

$$\frac{\begin{array}{c}\mathscr{TS} \succ\!\succ_{\tau_1} K.r \leftarrow L.s \overset{\otimes}{\to} M.t \\ \mathscr{TS} \succ\!\succ_{\tau_2} L.s \leftarrow X \quad \mathscr{TS} \succ\!\succ_{\tau_3} M.t \leftarrow Y \\ X \cap Y = \phi\end{array}}{\mathscr{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_3} K.r \leftarrow X \cup Y} \qquad (\mathbf{CWP_8})$$

$$\frac{\mathscr{TS} \succ\!\succ_{\tau_1} s \quad \mathscr{TS} \succ\!\succ_{\tau_2} s}{\mathscr{TS} \succ\!\succ_{\tau_1 \cup \tau_2} s} \qquad (\mathbf{CWP_9})$$

In order to infer the maximum validity of credentials, it is necessary to introduce the maximum validity of time at each stage of the inference process. The last rule $(\mathbf{CWP_9})$ must be applied as much as possible.

6. Use Case Scenarios

This section aims at presenting very simple cases in which the inference system is used in connection with RT_{+}^{T} language credentials. These scenarios are intended to verify usefulness of the presented RT^{T} language extensions.

6.1. First Scenario – Bank Security Policy

At first, the use of credentials without any restrictions will be shown. Then, we will add restrictions resulting

from the time when the person concerned performs a specific role.
Let us consider that we need two out of four guards to open the main treasury.

$$F.guards \longleftarrow F.guard \otimes F.guard \tag{1}$$

Furthermore, we want to have two guards and one main guard, who may also be a regular guard.

$$F.open \longleftarrow F.mGuard \odot F.guards \tag{2}$$

As may be observed, this can be written by using only one credential. There is no need to list all guards and determine the set of people needed to meet the requirements of his specific bank's security policy.
Let us now assign the people who perform the individual roles at the bank:

$$F.guard \longleftarrow \{Frank\} \tag{3}$$

$$F.guard \longleftarrow \{Susan\} \tag{4}$$

$$F.guard \longleftarrow \{Evan\} \tag{5}$$

$$F.guard \longleftarrow \{Victor\} \tag{6}$$

$$F.mGuard \longleftarrow \{Victor\} \tag{7}$$

$$F.mGuard \longleftarrow \{Eve\} \tag{8}$$

Taking into account the requirements of the bank's security policy, any pair from the set $\{Frank, Susan, Evan, Victor\}$ can act as two out of four guards. However, opening the main treasury requires the presence of $Victor$ (it may be a pair of people with $Victor$ or two other guards and $Victor$), or the presence of Eve as an additional guard.
If in our scenario, it is assumed that the bank's employees perform individual roles only for a certain period of time, which will render this example a closer reflection of reality. In fact, people at work also play a specific role at a certain point in time. To achieve that objective, we need to generalize credentials (3)–(8) by introducing time ranges for their validity:

$$F.guard \longleftarrow \{Frank\} \textbf{ in } \tau_1 \tag{9}$$

$$F.guard \longleftarrow \{Susan\} \textbf{ in } \tau_2 \tag{10}$$

$$F.guard \longleftarrow \{Evan\} \textbf{ in } \tau_3 \tag{11}$$

$$F.guard \longleftarrow \{Victor\} \textbf{ in } \tau_4 \tag{12}$$

$$F.mGuard \longleftarrow \{Victor\} \textbf{ in } \tau_5 \tag{13}$$

$$F.mGuard \longleftarrow \{Eve\} \textbf{ in } \tau_6 \tag{14}$$

stating that (3)–(8) are met only during τ_1, τ_2, τ_3, τ_4, τ_5, and during τ_6, respectively, while the credentials (1) and (2) are always valid, because they express certain facts that do not depend on time.
Now, using the credentials (1), (2) and (9)–(14), we can deduce those couples who can open the main treasury. The given set of credentials shows that the set $\{Frank, Susan, Victor\}$ can jointly open the treasury at time $\tau_1 \cap \tau_2 \cap \tau_5$ or $\{Susan, Victor\}$ during the time $\tau_2 \cap \tau_4 \cap \tau_5$.
Let us now move to the inference system with time-limited credentials. According to rule (**CWP₁**), we infer the validity period of the credentials in the following manner:

$$\frac{F.guards \longleftarrow F.guard \otimes F.guard \in \mathcal{TS}}{\mathcal{TS} \succ\succ F.guards \longleftarrow F.guard \otimes F.guard}$$

$$\frac{F.openTreasury \longleftarrow F.mGuard \odot F.guards \in \mathcal{TS}}{\mathcal{TS} \succ\succ F.openTreasury \longleftarrow F.mGuard \odot F.guards}$$

$$\frac{F.guard \longleftarrow \{Frank\} \textbf{ in } \tau_1 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_1} F.guard \longleftarrow \{Frank\}}$$

$$\frac{F.guard \longleftarrow \{Susan\} \textbf{ in } \tau_2 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_2} F.guard \longleftarrow \{Susan\}}$$

$$\frac{F.guard \longleftarrow \{Evan\} \textbf{ in } \tau_3 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_3} F.guard \longleftarrow \{Evan\}}$$

$$\frac{F.guard \longleftarrow \{Victor\} \textbf{ in } \tau_4 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_4} F.guard \longleftarrow \{Victor\}}$$

$$\frac{F.mGuard \longleftarrow \{Victor\} \textbf{ in } \tau_5 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_5} F.mGuard \longleftarrow \{Victor\}}$$

$$\frac{F.mGuard \longleftarrow \{Eve\} \textbf{ in } \tau_6 \in \mathcal{TS}}{\mathcal{TS} \succ\succ_{\tau_6} F.mGuard \longleftarrow \{Eve\}}$$

There is no need to use all credentials to infer whatever is of interest for us. For example, when we want to check the time interval in which two different guards may work together, we take into account credentials (1), (10), (12) and rule (**CWP₆**), and based thereon we conclude that:

$$\frac{\begin{array}{c}\mathcal{TS} \succ\succ F.guards \longleftarrow F.guard \otimes F.guard \\ \mathcal{TS} \succ\succ_{\tau_2} F.guard \longleftarrow \{Susan\} \\ \mathcal{TS} \succ\succ_{\tau_4} F.guard \longleftarrow \{Victor\} \\ \{Susan\} \cap \{Victor\} = \phi\end{array}}{\mathcal{TS} \succ\succ_{\tau_2 \cap \tau_4} \mathbf{F.guards} \longleftarrow \{\mathbf{Susan}, \mathbf{Victor}\}}$$

Now, we are interested in the initial question, i.e. who can open the main treasury and at what time. Therefore, we take the newly created credential and the credentials reflecting the conditions that need to be fulfilled in order to open the treasury (2), (13) and, using rule (**CWP₅**), we conclude that:

$$\frac{\begin{array}{c}\mathcal{TS} \succ\succ F.openTreasury \longleftarrow F.mGuard \odot F.guards \\ \mathcal{TS} \succ\succ_{\tau_5} F.mGuard \longleftarrow \{Victor\} \\ \mathcal{TS} \succ\succ_{\tau_2 \cap \tau_4} F.guards \longleftarrow \{Susan, Victor\}\end{array}}{\mathcal{TS} \succ\succ_{\tau_2 \cap \tau_4 \cap \tau_5} \mathbf{F.openTreasury} \longleftarrow \{\mathbf{Susan}, \mathbf{Victor}\}}$$

As a result of this inference, we see that the set of people who can jointly open the treasury is: $\{Susan, Victor\}$ in time: $\tau_2 \cap \tau_4 \cap \tau_5$.

6.2. Second Scenario – Company's Quality Policy

Company L has three roles: *employee*, *specjalist* and *controller*. The company's quality policy requires confirmation that the product is suitable for use when it has been checked by two employees, a specialist and a controller. It can be clearly stated that the two *employees* must be different persons. However, a *specialist* who is also an *employee* can be one of the two employees. The requirement for the *controller* is much more restrictive, namely they cannot simultaneously perform any of the other roles during the inspection. Therefore, we can save the company's control policy using the following set of credentials:

$$L.2Employees \longleftarrow L.employee \otimes L.employee \tag{15}$$

$$L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees \tag{16}$$

$$L.confirm \longleftarrow L.controller \otimes L.specjalEmployees \tag{17}$$

Let us now assign people who perform individual roles in the company:

$$L.employee \longleftarrow \{Claire\} \tag{18}$$

$$L.employee \longleftarrow \{Rita\} \tag{19}$$

$$L.specjal \longleftarrow \{Claire\} \tag{20}$$

$$L.controller \longleftarrow \{Kim\} \tag{21}$$

Now, using our credentials and using the inference system from Section 3, we can define a set of people who, acting together, can confirm the quality of products. Using credentials (15)–(21) and rule $(\mathbf{W_1})$, we can conclude that:

$$\frac{L.2Employees \longleftarrow L.employee \otimes L.employee \in \mathscr{S}}{\mathscr{S} \succ L.2Employees \longleftarrow L.employee \otimes L.employee}$$

$$\frac{L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees \in \mathscr{S}}{\mathscr{S} \succ L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees}$$

$$\frac{L.confirm \longleftarrow L.controller \otimes L.specjalEmployees \in \mathscr{S}}{\mathscr{S} \succ L.confirm \longleftarrow L.controller \otimes L.specjalEmployees}$$

$$\frac{L.employee \longleftarrow \{Claire\} \in \mathscr{S}}{\mathscr{S} \succ L.employee \longleftarrow \{Claire\}}$$

$$\frac{L.employee \longleftarrow \{Rita\} \in \mathscr{S}}{\mathscr{S} \succ L.employee \longleftarrow \{Rita\}}$$

$$\frac{L.specjal \longleftarrow \{Claire\} \in \mathscr{S}}{\mathscr{S} \succ L.specjal \longleftarrow \{Claire\}}$$

$$\frac{L.controller \longleftarrow \{Kim\} \in \mathscr{S}}{\mathscr{S} \succ L.controller \longleftarrow \{Kim\}}$$

Now, using credentials (15), (18), (19) and rule $(\mathbf{W_6})$ we can infer that:

$$\frac{\begin{array}{c}\mathscr{S} \succ L.2Employees \longleftarrow L.employee \otimes L.employee \\ \mathscr{S} \succ L.employee \longleftarrow \{Claire\} \\ \mathscr{S} \succ L.employee \longleftarrow \{Rita\} \\ \{Claire\} \cap \{Rita\} = \phi\end{array}}{\mathscr{S} \succ \mathbf{L.2Employees} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}\}}$$

Now, to appoint a set of people consisting of a specialist and two different employees, we need to use the newly created credential, add credentials (16), (20) and rule $(\mathbf{W_5})$:

$$\frac{\begin{array}{c}\mathscr{S} \succ L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees \\ \mathscr{S} \succ L.specjal \longleftarrow \{Claire\} \\ \mathscr{S} \succ L.2Employees \longleftarrow \{Claire, Rita\}\end{array}}{\mathscr{S} \succ \mathbf{L.specjalEmployees} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}\}}$$

Now, using the above credential and adding credentials (17), (21) and rule $(\mathbf{W_6})$ we can deduce the following:

$$\frac{\begin{array}{c}\mathscr{S} \succ L.confirm \longleftarrow L.controller \otimes L.specjalEmployees \\ \mathscr{S} \succ L.controller \longleftarrow \{Kim\} \\ \mathscr{S} \succ L.specjalEmployees \longleftarrow \{Claire, Rita\} \\ \{Kim\} \cap \{Claire, Rita\} = \phi\end{array}}{\mathscr{S} \succ \mathbf{L.confirm} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}, \mathbf{Kim}\}}$$

which confirms that the set of entities that can jointly confirm quality is: $\{Claire, Rita, Kim\}$.
As in the previous scenario, we also assume that *Claire* and *Rita* are employees only for a limited period of time. The same applies to *Claire* as a specialist and *Kim* as a controller. To do this, we need to generalize credentials (18)–(21) by introducing time ranges applying to their validity:

$$L.employee \longleftarrow \{Claire\} \ \mathbf{in} \ \tau_1 \tag{22}$$

$$L.employee \longleftarrow \{Rita\} \ \mathbf{in} \ \tau_2 \tag{23}$$

$$L.specjal \longleftarrow \{Claire\} \ \mathbf{in} \ \tau_3 \tag{24}$$

$$L.controller \longleftarrow \{Kim\} \ \mathbf{in} \ \tau_4 \tag{25}$$

stating that (18), (19), (20), and (21) are only available during τ_1, τ_2, τ_3, and during τ_4, respectively, while credentials (15), (16) and (17) are always valid, because they express certain facts that do not depend on time.
Let us use our time-dependent inference system. We can use credentials: (15)–(17) and (22)–(25). Using rule $(\mathbf{CWP_1})$ we can conclude that:

$$\frac{L.2Employees \longleftarrow L.employee \otimes L.employee \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ L.2Employees \longleftarrow L.employee \otimes L.employee}$$

$$\frac{L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees}$$

$$\frac{L.confirm \longleftarrow L.controller \otimes L.specjalEmployees \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ L.confirm \longleftarrow L.controller \otimes L.specjalEmployees}$$

$$\frac{L.employee \longleftarrow \{Claire\} \ \mathbf{in} \ \tau_1 \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ_{\tau_1} L.employee \longleftarrow \{Claire\}}$$

$$\frac{L.employee \longleftarrow \{Rita\} \ \mathbf{in} \ \tau_2 \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ_{\tau_2} L.employee \longleftarrow \{Rita\}}$$

$$\frac{L.specjal \longleftarrow \{Claire\} \ \mathbf{in} \ \tau_3 \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ_{\tau_3} L.specjal \longleftarrow \{Claire\}}$$

$$\frac{L.controller \longleftarrow \{Kim\} \ \mathbf{in} \ \tau_4 \in \mathscr{TS}}{\mathscr{TS} \succ\!\succ_{\tau_4} L.controller \longleftarrow \{Kim\}}$$

In order to check when two different employees can cooperate together, we use credentials (15), (22), (23) and rule (**CWP₆**) to infer:

$$\frac{\begin{array}{c}\mathcal{TS} \succ\!\succ L.2Employees \longleftarrow L.employee \otimes L.employee \\ \mathcal{TS} \succ\!\succ_{\tau_1} L.employee \longleftarrow \{Claire\} \\ \mathcal{TS} \succ\!\succ_{\tau_2} L.employee \longleftarrow \{Rita\}\{Claire\} \cap \{Rita\} = \phi\end{array}}{\mathcal{TS} \succ\!\succ_{\tau_1 \cap \tau_2} \mathbf{L.2Employees} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}\}}$$

In the next step, we use the above credential and add credentials (16), (24) and rule (**CWP₅**):

$$\frac{\begin{array}{c}\mathcal{TS} \succ\!\succ L.specjalEmployees \longleftarrow L.specjal \odot L.2Employees \\ \mathcal{TS} \succ\!\succ_{\tau_5} L.specjal \longleftarrow \{Claire\} \\ \mathcal{TS} \succ\!\succ_{\tau_1 \cap \tau_2} L.2Employees \longleftarrow \{Claire, Rita\}\end{array}}{\mathcal{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_5} \mathbf{L.specjalEmployees} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}\}}$$

Now, by using the above credential and adding credentials (17) and (25) and rule (**CWP₆**) we can deduce:

$$\frac{\begin{array}{c}\mathcal{TS} \succ\!\succ L.confirm \longleftarrow L.controller \otimes L.specjalEmployees \\ \mathcal{TS} \succ\!\succ_{\tau_6} L.controller \longleftarrow \{Kim\} \\ \mathcal{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_5} L.specjalEmployees \longleftarrow \{Claire, Rita\} \\ \{Kim\} \cap \{Claire, Rita\} = \phi\end{array}}{\mathcal{TS} \succ\!\succ_{\tau_1 \cap \tau_2 \cap \tau_5 \cap \tau_6} \mathbf{L.confirm} \longleftarrow \{\mathbf{Claire}, \mathbf{Rita}, \mathbf{Kim}\}}$$

which confirms that the set of entities that can jointly confirm quality is $\{Claire, Rita, Kim\}$ in time $\tau_1 \cap \tau_2 \cap \tau_3 \cap \tau_4$.
The above scenarios assume that the security policy credentials are permanently valid, but this is not required in any way. Some policies are always time-limited (e.g. seasonal sales) or are modified after a certain incident or to reflect changes affecting the company. Introduction of time-dependent credentials makes it very easy to define the company's security policy. As you can see in this simple scenario, you do not have to change the entire policy, but only take into account the time range. However, in the absence of time limits, the company policy should be changed, which would require, in a real-life scenario, a considerable amount of work to be performed by those involved. Such a transfer of rights, conducted for a certain period of time only, obviously is of great importance when replacing a specific person at their position. We can assign certain credentials to the person who replaces us for duration of our. Then, at the time of our return to work, the rights of the person who replaces us expire.

6.3. Third Scenario – Submitting an Proposal

The company policy states that in order to submit a proposal for co-financing a specific project, we must deliver it between 01/06/2019 and 31/07/2019 (τ_1).

$$P.validSend \longleftarrow P.send \textbf{ in } \tau_1$$

To write the proposal, we need a person who is part of the information security team (IST).

$$P.write \longleftarrow P.ist$$

We know that Mark works at IST.

$$P.ist \longleftarrow \{Mark\} \textbf{ in } \tau_2$$

We also know that Mark is going on vacation in July. During his absence, he will be replaced by Konrad.

$$\textbf{if } Mark \notin P.ist \textbf{ then } P.ist \longleftarrow \{Konrad\}$$

After being drawn up, the proposal must be checked by the IST manager. The manager cannot be the same person who has written the proposal.

$$P.check \longleftarrow P.ist \overset{\otimes}{\rightarrow} P.headIST$$

Luck is the name of the IST head.

$$P.headIST \longleftarrow \{Luck\} \textbf{ in } \tau_3$$

The proposal, after being checked by the manager, must be submitted to the project support team, so that a number be assigned thereto.

$$P.number \longleftarrow P.pst$$

After assigning the number, the proposal must be registered at the office.

$$P.register \longleftarrow P.office$$

Then, the proposal must be submit to the director, for sign-off.

$$P.signed \longleftarrow P.director$$

Once signed, the proposal is registered in the system by an IST employee.

$$P.send \longleftarrow P.ist$$

The specific sequence (creation, signing, numbering, registering and submission of the proposal) must be maintained. Considering the uniqueness of the people performing the individual roles at the specific steps, the order will look as follows:

$$\begin{array}{r}P.validSend \longleftarrow P.write \overset{\otimes}{\rightarrow} P.check \overset{\otimes}{\rightarrow} P.number \\ \overset{\otimes}{\rightarrow} P.register \overset{\otimes}{\rightarrow} P.send \textbf{ in } \tau_1\end{array}$$

According to rule (**CWP₁**), we can infer:

$$\frac{P.validSend \longleftarrow P.send \textbf{ in } \tau_1 \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ_{\tau_1} P.validSend \longleftarrow P.send}$$

$$\frac{P.write \longleftarrow P.ist \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.write \longleftarrow P.ist}$$

$$\frac{P.ist \longleftarrow \{Mark\} \textbf{ in } \tau_2 \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ_{\tau_2} P.ist \longleftarrow \{Mark\}}$$

$$\frac{\textbf{if } Mark \notin P.ist \textbf{ then } P.ist \longleftarrow \{Konrad\} \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ \textbf{if } Mark \notin P.ist \textbf{ then } P.ist \longleftarrow \{Konrad\}}$$

$$\frac{P.check \longleftarrow P.ist \overset{\otimes}{\rightarrow} P.headIST \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.check \longleftarrow P.ist \overset{\otimes}{\rightarrow} P.headIST}$$

$$\frac{P.number \leftarrow P.pst \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.number \leftarrow P.pst}$$

$$\frac{P.register \leftarrow P.office \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.register \leftarrow P.office}$$

$$\frac{P.signed \leftarrow P.director \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.signed \leftarrow P.director}$$

$$\frac{P.send \leftarrow P.ist \in \mathcal{TS}}{\mathcal{TS} \succ\!\succ P.send \leftarrow P.ist}$$

Now, by assigning specific individuals to each role, we can check who, when and under what conditions is capable of submitting the proposal (with receiving a grant being the final objective of the process).

As you can see in the scenario in which we have to meet different conditions, taking into account the importance of time and the order in which the credentials are made, you can model it using our inference system. These properties make it useful in real systems in which similar problems are faced.

7. Conclusions

In this article, we are developing the RT^T language by adding time limits affecting the validity of credentials, and by making the validity of a single credential dependent on the availability or unavailability of other credentials – all that in the context of execution during actual operation of a system. This approach shows the impact that small extensions may exert on the applicability of trust management languages.

Time restrictions are a reasonable extension of credentials, because credentials are granted to users for a predefined period of time, rather than indefinitely. As shown in the scenario, conditional credentials can greatly facilitate the automation of some processes. The inference systems presented here are simple, but well-founded theoretically.

The proposed model has been successfully applied to multi-valued logic, including uncertain information. In the near future, we plan to use it in other, more complex cases involving distributed systems whose policies are not trivial.

References

[1] X. Qian and T. F. Lunt, "A MAC policy framework for multilevel relational databases", *IEEE Trans. Knowl. and Data Engin.*, vol. 8, no. 1, pp. 3–15, 1996 (doi: 10.1109/69.485625).

[2] National Computer Security Center, "A guide to understanding discretionary access control in trusted systems", NCSC-TG-003, 1987 [Online]. Available: https://fas.org/irp/nsa/rainbow/tg003.htm

[3] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models", *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996 (doi: 10.1109/2.485845).

[4] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management", in *Proc. 17th IEEE Symp. on Secur. and Priv.*, Oakland, CA, USA, 1996, pp. 164–173 (doi: 10.1109/SECPRI.1996.502679).

[5] N. Li, W. Winsborough, and J. Mitchell, "Distributed credential chain discovery in trust management", *J. Comput. Secur.*, vol. 11, no. 1, pp. 35–86, 2003 (doi: 10.1145/502001.502005).

[6] N. Li, J. Mitchell, and W. Winsborough, "Design of a role-based trust-management framework", in *Proc. IEEE Symp. on Secur. and Priv.*, Berkeley, CA, USA, 2001, pp. 114–130 (doi: 10.1109/SECPRI.2002.1004366).

[7] S. Chithra, "A role based trust model for peer to peer systems using credential trees", *Int. J. of Comp. Theory and Engin.*, vol. 3, no. 2, pp. 234–239, 2011 (doi: 10.7763/IJCTE.2011.V3.310).

[8] A. Felkner, "How the role-based trust management can be applied to wireless sensor networks", *J. of Telecommun. and Inform. Technol.*, no. 4, 2012, pp. 70–77 [Online]. Available: http://dlibra.itl.waw.pl/dlibra-webapp/dlibra/docmetadata?id=1570

[9] K. Ezhil Vignesh and N. Radhika, "An improved role based trust management system using interactive artificial bee colony (I-ABC) algorithm for wireless sensor networks", *Res. J. of Appl. Sci. Engin. and Technol.*, vol. 10, no. 10, pp. 1175–1184, 2015 (doi: 10.19026/rjaset.10.1885).

[10] Pratap Kumar Behera, "A Novel Trust Based Access Control Model for Cloud Environment", M.Tech. Thesis, Department of Computer Science and Engineering National Institute of Technology Rourkela, Rourkela, India, 2015 [Online]. Available: https://core.ac.uk/download/pdf/80147859.pdf

[11] D. Gorla, M. Hennessy, and V. Sassone, "Inferring dynamic credentials for role-based trust management", in *Proc. of the 8th Conf. on Princip. and Pract. of Declar. Programm. ACM SIGPLAN 2006*, Venice, Italy, 2006, pp. 213–224 (doi: 10.1145/1140335.1140361).

[12] M. R. Czenko *et al.*, "Nonmonotonic trust management for P2P applications", in *Proc. 1st Int. Worksh. on Secur. Trust Manag. STM 2005*, Milan, Italy, 2005, pp. 113–130 (doi: 10.1016/j.entcs.2005.09.037).

Anna Felkner holds a Ph.D. degree in Information Technology from the Warsaw University of Technology and an M.Sc. degree from Białystok University of Technology. She is an Assistant Professor and head of the Information Security Methods Team. Her interests include access control, trust modeling, risk analysis and vulnerability management. She is the author of over 40 publications and has spoken at many conferences.

https://orcid.org/0000-0003-3813-4840
E-mail: anna.felkner@nask.pl
Information Security Methods Team
Research and Academic Computer Network (NASK)
Kolska 12
01-045 Warsaw, Poland