Migrating from RBAC to temporal RBAC

ISSN 1751-8709
Received on 25th May 2016
Revised 13th December 2016
Accepted on 25th January 2017
E-First on 13th June 2017
doi: 10.1049/iet-ifs.2016.0258
www.ietdl.org

Barsha Mitra[1] ✉, Shamik Sural[2], Jaideep Vaidya[3], Vijayalakshmi Atluri[3]
[1]*Department of Computer Science & Information Systems, BITS Pilani-Hyderabad Campus, Hyderabad, India*
[2]*Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India*
[3]*MSIS Department, Rutgers University, USA*
✉ *E-mail: barsha.mitra@hyderabad.bits-pilani.ac.in*

Abstract: The last two decades have witnessed an emergence of role-based access control (RBAC) as the de facto standard for access control. However, for organisations already having a deployed RBAC system, in many cases it may become necessary to associate a temporal dimension with the existing access control policies due to changing organisational requirements. In such cases, migration from RBAC to a temporal extension of RBAC becomes essential. Temporal RBAC (TRBAC) is one such RBAC extension. The process of creating a set of roles for implementing a TRBAC system is known as *temporal role mining*. Existing temporal role mining approaches typically assume that TRBAC is being deployed from scratch and do not consider it as a migration from an existing RBAC policy. In this study, the authors propose two temporal role mining approaches that enable migration from RBAC to TRBAC. These approaches make use of conventional (non-temporal) role mining algorithms. Apart from aiding the migration process, deriving the roles in this manner allows the flexibility of minimising any desired role mining metric. They experimentally evaluate the performance of both of the proposed approaches and show that they are both efficient and effective.

1 Introduction

The primary goal of access control is to prevent unauthorised access to information and resources. Role-based access control (RBAC) [1] is one such access control model, and is considered the de facto access control model of choice in organisations of any size. The most essential component of RBAC is *roles*, by means of which RBAC ensures authorised user access. Roles are created by combining different permissions and users are selectively assigned to these roles. Thus, user access to system resources is granted via the roles assigned to them. *Role engineering* [2] is the process of deriving the roles required for deploying an RBAC system. It can either be top-down [3–6] or bottom-up. The top-down approach requires a considerable amount of human intervention, but the bottom-up approach can be automated. Role mining [7–10] is a bottom-up approach to role engineering.

Over the past few years, role mining has become quite popular as a means for deploying RBAC systems. It processes the input, which is a set of user–permission assignments (UPAs), in order to generate the output consisting of a set of roles (R), a user–role assignment (UA) relation and a role–permission assignment (PA) relation. UA represents the assignment of roles to users and PA depicts the component permissions of each role. Both UA and PA can be represented as Boolean matrices. The problem of deriving an optimal set of roles is known as the role mining problem (RMP). RMP aims to minimise the size of any one of the RBAC components or the cumulative size of several components such as the number of roles [9], summation of the sizes of the UA and the PA [11], weighted sum of sizes of role set and UA [12], sum of the sizes of role set, UA and PA [13] or the weighted structural complexity [8]. Corresponding to each such metric, a different variant of RMP has been proposed. In addition to these, RMP variants considering cardinality and separation of duty constraints are also present in the this literature [14–17]. Several algorithms have been proposed for solving the different RMP variants, such as permission clustering based approaches [18, 19], problem mapping based solution strategies [9, 20, 21] and matrix decomposition based role mining techniques [22, 23]. Mitra *et al.* [24] provide a comprehensive survey of the different approaches in role mining.

RBAC, though a popular model for access control, does not allow the imposition of any temporal, spatial or environmental constraint on the availability of roles to users. To overcome this shortcoming of RBAC, a number of extensions have been proposed. Temporal RBAC (TRBAC) [25] is one such extension. It restricts the time duration during which a role can be enabled. Though the assignment of roles to users takes place as is the case of conventional RBAC, the availability of the permissions included in those roles is temporally constrained. Users can invoke the permissions only when the associated roles are enabled. This enables the accurate specification of complex policies. Traditional role mining techniques are not capable of handling the presence of any time element in the input. Hence, these approaches cannot be directly used to derive roles having temporal constraints. In order to account for this shortcoming of conventional role mining, recently temporal role mining techniques have been proposed [26–28]. Temporal role mining processes a UPA relation containing a temporal component to produce a UA, a PA and a role enabling base (REB). The roles obtained from temporal role mining are called temporal roles. The REB contains the time interval sets during which each of the temporal roles is enabled. The existing temporal role mining algorithms either minimise the total number of roles such as Temporal Role Mining Problem - Many Valued Concepts (TRMP-MVC) [28] and the approaches proposed in [26, 27], or a weighted sum of the sizes of the REB and the PA (CO-TRAPMP-MVC of [28]). The approaches proposed in [26, 27] perform temporal role mining by including the temporally constrained UPAs of two or more users. In contrast to this, TRMP-MVC and CO-TRAPMP-MVC are based on the notion of many-valued concepts.

An organisation wishing to migrate from RBAC to TRBAC has a deployed set of roles and consequently a UA and a PA. However, the existing set of roles could have become non-optimal due to incremental creation of roles over a period of time. Moreover, erroneous role or permission assignments are also often present in the RBAC system that could get migrated to the target TRBAC system. Therefore, in order to create an optimal and correct set of roles, it makes sense to perform temporal role mining rather than using the current UA, PA and merely associating an REB. In this study, an optimal set of roles implies that the set of roles obtained as output is minimal in terms of the optimisation metric considered during role creation. Moreover, a set of roles is said to be correct if none of the users acquires any extra privilege which s/he is not entitled to or loses any existing privilege which s/he is entitled to. Using the existing UA and PA and creating an REB may either result in a non-optimal set of roles and even worse, an incorrect set of roles where one or more users end up acquiring extra permissions or existing legal permissions for additional time duration. The input to the temporal role mining process, then, would be a set of temporal UPAs [26] obtained by adding a temporal component to the existing potentially non-optimal set of roles. The temporal roles could potentially be generated from these UPAs using the existing temporal role mining algorithms [27, 28]. However, the existing temporal role mining algorithms are only capable of minimising either the total number of roles or a weighted summation of the sizes of the PA and the REB. Hence, a methodology involving any of the currently available temporal role mining algorithms would fail to make use of the rich body of the literature related to traditional (non-temporal) role mining. Using non-temporal role mining algorithms allows the flexibility of minimising any desired metric while performing temporal role mining depending on the specific needs of the organisation. This would enable the organisation to deploy a desired flavour of TRBAC system based on its particular needs.

To facilitate the migration from RBAC to TRBAC, in this paper, we propose two temporal role mining algorithms making use of existing non-temporal role mining approaches. The temporal information associated with the input is considered in a phase distinct from the role creation phase as opposed to the existing temporal role mining approaches, where the time component is taken into account concurrently with the role creation phase. This separation of the two phases allows us to employ the available non-temporal role mining methods for temporal role generation. Non-temporal role mining is either carried out before considering the temporal dimension or after it. We name the former method as *Time-Agnostic Role Mining* (TARM) and the latter as *Snapshot-based Role Mining* (SNARM). Merging of temporal roles is done as a final step in both TARM and SNARM for further reducing the desired minimisation metric. Apart from the flexibility of minimising any temporal role mining metric, these approaches reduce the total number of temporal UPAs that need to be considered during temporal role creation. Consequently, the overhead of role generation is considerably reduced, which in turn lowers the time required to perform temporal role mining. This reduction in time is significant for datasets having large number of users and permissions. Thus, both TARM and SNARM essentially differ from the existing temporal role mining algorithms with respect to the manner of role creation. Furthermore, they both provide the flexibility to minimise a wide range of role mining metrics without the need to design separate algorithms for each of them. We evaluate the performance of both TARM and SNARM and compared it with previously proposed temporal role mining techniques using several benchmark datasets. It is worth noting that two different algorithms (TARM and SNARM) are proposed since they each outperform the other based on the interrelationships among the time component of the input. Thus depending on the nature of the input, the system administrator can take a decision regarding the choice of the appropriate algorithm.

The remaining of this paper is organised as follows. Section 2 discusses some preliminaries related to TRBAC and temporal role mining. The proposed TARM and SNARM approaches are presented in Sections 3 and 4, respectively. Performance of both algorithms is evaluated in Section 5. Finally, Section 6 concludes the paper and discusses directions for future research.

2 Preliminaries

In this section, we present a brief overview of TRBAC and temporal role mining.

TRBAC: Bertino *et al.* proposed TRBAC [25] as a temporal extension of RBAC. TRBAC allows roles to remain enabled for specific sets of time intervals and disables the roles for the remaining time. The enabling duration of each role is specified in the REB in terms of one or more periodic expressions. A periodic expression *PE* represents an infinite set of time intervals given in terms of *Calendars*. A calendar is a set of consecutive time intervals and can have any one of the following granularities: *Hours*, *Days*, *Weeks*, *Months* or *Years*. Two date expressions, namely, *begin* and *end* are associated with each periodic expression to bound the set of time intervals both from below and above. Thus, the resulting REB entry is $\langle [begin, end], PE \rangle$. Corresponding to each role *r*, one or more entries of the form $\langle [begin, end], PE, enable\ r \rangle$ are present denoting the time interval set during which *r* is enabled.

Temporal role mining: The input given to temporal role mining is a UPA relation containing temporal information. Such a UPA relation is known as temporal UPA or TUPA [26]. A TUPA can be represented in the form of a matrix whose rows and columns, respectively, denote users and permissions. In each entry (u, p) of the TUPA, a set of time intervals T_{up} is present. If permission *p* is assigned to user *u*, then T_{up} is non-empty and is represented using one or more periodic expressions. On the other hand, if *u* is not assigned *p*, then $T_{\text{up}} = \phi$. $\langle u, p, T_{\text{up}} \rangle$ is called a *triple* of the TUPA. From such a TUPA, temporal role mining derives a set of roles R, a UA, a PA and an REB. Depending on the specific optimality criterion to be minimised, several variants of the problem of temporal role mining have been proposed – temporal role mining problem (TRMP) ($|R|$) [26], generalised TRMP (the amount of mismatch between the input temporal UPAs and the ones computed by combining the UA, PA and REB) [27] or cumulative overhead of temporal roles and permissions minimisation problem [28].

3 Time-Agnostic Role Mining

A natural way of extending the non-temporal role mining algorithms to create temporal roles would be to generate an initial set of temporally unconstrained roles using any one of these algorithms and then associating a temporal component with each such role. In this section, we describe a temporal role mining approach which follows this method of role creation. We name this algorithm as *Time-Agnostic Role Mining* (TARM) since it initially ignores the time component of each UPA in the input TUPA matrix. In this approach, an initial set of roles is created which do not have any temporal component associated with them. We call such roles *time-agnostic roles*. In the second phase, a time constraint is added to each of the non-temporal roles to create the temporal role set. Since TARM is an iterative approach, these two phases are repeated until the time interval sets of all the triples of the TUPA have been completely included in a single temporal role or cumulatively included in several temporal roles. In the final phase, merging operations are carried out among the temporal roles. We now discuss each of these phases.

3.1 Creation of time-agnostic roles

Given an input TUPA, a UPA matrix is created in this phase. The number of rows and columns of the UPA are equal to those of the TUPA. The entries of the UPA are filled up as follows: (i) (i, j) entry of UPA is set as 0 if (i, j) entry of TUPA is ϕ, and (ii) (i, j) entry of UPA is set as 1 if (i, j) entry of TUPA contains a non-empty set of time intervals. Thus, only the non-temporal information component of the TUPA is kept in the UPA.

Next from this UPA matrix, a set of roles is created by applying an existing non-temporal role mining algorithm. These roles are created independent of the temporal element of the input and hence are termed as time-agnostic roles. The user and permission sets of the temporal roles that are generated subsequently from the time-agnostic roles will be subsets of the user and permission sets of the time-agnostic roles. Thus in order to create the temporal roles, it is sufficient to consider those triples of the TUPA which correspond to the UPAs included in the time-agnostic roles. Hence, the number of triples of the TUPA which need to be taken into account is considerably reduced, thereby decreasing the overall searching overhead.

```
Input: TUPA
Output: R, UA, PA, REB
Require: $\Gamma$: set of triples of TUPA
Require: R′, UA′, PA′: non-temporal role mining output of a single iteration
Require: $R_I$: temporal role set created in one iteration
1: R $\leftarrow \phi$
2: while $\Gamma > \phi$ do
3:     Create UPA from TUPA
4:     Perform non-temporal role mining on UPA to obtain R′, UA′, PA′
5:     $R_I \leftarrow \phi$
6:     for each $r \in$ R′ do
7:         Create temporal roles for $r$ and add to $R_I$
8:     end for
9:     R $\leftarrow$ R $\cup R_I$
10:    Modify UA′ and PA′
11:    Add columns of UA′ to UA and rows of PA′ to PA
12:    Update TUPA
13: end while
14: Merge roles of R
15: Update UA, PA, REB
```

Fig. 1 *Algorithm 1: TARM*

3.2 Adding temporal constraints to time-agnostic roles

The time-agnostic roles generated in the previous phase are given as input to this phase. Corresponding to each time-agnostic role, one or more temporal roles are created and their enabling durations are added to the REB. For doing this, the time interval sets associated with the TUPA triples corresponding to the UPAs associated with a time-agnostic role are considered. We represent the set of such triples as Γ_r for a time-agnostic role *r*. The user and permission sets of the temporal roles which are created in this phase may vary from those of the time-agnostic roles. This happens because the time interval sets of the triples of Γ_r may not be identical.

From a time-agnostic role *r*, the temporal roles are created in the following manner:

- Search for a common time interval set *T* among the triples of Γ_r.
- If such a set of time intervals exists, create a temporal role having the same user and permission sets as that of *r*. This role is enabled for *T*.
- If no common duration is found, choose the time interval set T' present in the maximum number of triples of Γ_r.
- In the event of a tie, select the time interval set T'' which is of longer duration.
- Create a temporal role having a user set same as that of *r* and consisting of the common permissions assigned to these users. This role is enabled for either T' or T'' depending on whichever is selected.
- If no such common permission exists, create temporal roles for the individual users by including the respective permission sets assigned to them. Enabling duration of each of these roles is T' or T''.

Due to the generation of temporal roles, the time interval sets of triples of Γ_r can get split if T, T' or T'' are subsets of those time interval sets. Thus, the time interval sets of some of the triples can get completely included in some temporal roles and so are fully covered. For some triples, such an inclusion is partial and hence, they get partially covered. The TUPA is updated by removing the fully covered triples and retaining the uncovered portions of the partially covered triples. TARM is an iterative approach. So, the phases for creation of time-agnostic and temporal roles are repeated until all the triples of the TUPA are fully covered. These roles are added to the set R. Each temporal role r_i of R is represented as (U_i, P_i, T_i). U_i contains the users who are assigned r_i, the set of permissions included in r_i is denoted by P_i and T_i represents the time interval set during which r_i is enabled. R, UA, PA and REB are given as input to the third phase.

3.3 Merging of temporal roles

In the third and final phases of TARM, the temporal roles are merged. Merging of two roles $r_i = (U_i, P_i, T_i)$ and $r_j = (U_j, P_j, T_j)$ is carried out to create a role $r_k = (U_k, P_k, T_k)$ if any one of the four conditions is satisfied:

- If $U_i = U_j$ and $T_i = T_j$, then $U_k = U_i = U_j$, $P_k = P_i \cup P_j$ and $T_k = T_i = T_j$.
- If $P_i = P_j$ and $T_i = T_j$, then $U_k = U_i \cup U_j$, $P_k = P_i = P_j$ and $T_k = T_i = T_j$.
- If $U_i = U_j$, $P_i = P_j$, and T_i and T_j are consecutive or $T_i \cap T_j \neq \phi$, then $U_k = U_i = U_j$, $P_k = P_i = P_j$ and $T_k = T_i \cup T_j$.
- If $U_i = U_j$, $P_i = P_j$, and T_i and T_j are disjoint, then $U_k = U_i = U_j$, $P_k = P_i = P_j$ and $T_k = \{T_i, T_j\}$.

The fourth condition is different from the remaining three conditions. If the enabling durations of the individual roles are disjoint, r_k is enabled for both T_i and T_j separately.

Algorithm 1 (see Fig. 1) shows the overall procedure for TARM. The set of all triples of the TUPA is denoted as Γ. R″, UA′ and PA′ constitute the output obtained from the non-temporal role mining algorithm in a particular iteration. R_I represents the set of temporal roles generated in a single iteration. Line 1 initialises R as an empty set. The *while loop* of Lines 2–13 iterates until Γ is not empty. A UPA matrix is created from the TUPA in Line 3. Time-agnostic role set R′, UA′ and PA′ are produced after performing non-temporal role mining on this UPA (Line 4). Line 5 initialises R_I as an empty set. The *for loop* of Lines 6–8 creates one or more temporal roles for each time-agnostic role *r* of R′. Line 9 adds these temporal roles to R and UA′ and PA′ are modified in Line 10. The columns and rows of UA′ and PA′ are, respectively, added to UA and PA in Line 11. Line 12 updates the TUPA. Finally, merging operations are carried out among the temporal roles and the UA, PA and REB are accordingly updated in Lines 14 and 15, respectively.

We illustrate the working of TARM using the TUPA matrix shown in Table 1. For the sake of brevity, we have only shown a single time interval corresponding to each non-empty entry of TUPA instead of a time interval set.

The UPA matrix created from this TUPA is shown in Table 2. When the Minimum Biclique Cover (MBC) based non-temporal role mining algorithm [20] is applied on this UPA in the first iteration, the following roles are obtained – $r_1 = (\{u_2, u_3\}, \{p_2, p_3\})$, $r_2 = (\{u_1, u_2, u_3\}, \{p_3\})$ and $r_3 = (\{u_1, u_2\}, \{p_1\})$. For r_1, no common duration exists among the triples $\langle u_2, p_2, \text{6 am–11 am}\rangle$, $\langle u_2, p_3, \text{5 am–7 am}\rangle$, $\langle u_3, p_2, \text{7 am–9 am}\rangle$ and $\langle u_3, p_3, \text{6 am–11 am}\rangle$. So, we search for the time interval that is associated with the maximum number of triples among these four triples. 7 am–9 am is such an interval since it is associated with three triples. However, among these three triples, only two are associated with both u_2 and u_3. So, a temporal role $(\{u_2, u_3\}, \{p_2\}, \{\text{7 am–9 am}\})$ is created. Next, for the time-agnostic role r_2, a common duration of 6 am–7 am exists among all the three associated triples. Thus, a temporal role $(\{u_1, u_2, u_3\}, \{p_3\}, \{\text{6 am–7 am}\})$ is created. For the last time-agnostic role r_3 obtained in this iteration, no common time interval exists between the two associated triples. Since each of the time interval is associated with the same number of triples (i.e. 1), the tie is broken by selecting the time interval 7 am–9 am since it is of longer duration. Thus, the temporal role obtained from r_3 is $(\{u_2\}, \{p_1\}, \{\text{7 am–9 am}\})$. After this, the TUPA is updated and in the subsequent iterations similar steps are performed. Proceeding in this manner, a set of nine roles is obtained after all the triples of the TUPA have been fully covered. Among these roles, $(\{u_3\}, \{p_3\}, \{\text{7 am–11 am}\})$ and $(\{u_1\}, \{p_3\}, \{\text{7 am–11 am}\})$ are merged to create the role $(\{u_1, u_3\}, \{p_3\}, \{\text{7 am–11 am}\})$ and roles $(\{u_2\}, \{p_2\}, \{\text{9 am–11 am}\})$ and $(\{u_2\}, \{p_2\}, \{\text{6 am–7 am}\})$ are merged to create the role $(\{u_2\}, \{p_2\}, \{\text{6 am–7 am, 9 am–11 am}\})$. Finally, a set of seven roles is obtained.

4 Snapshot-based role mining

In this section, we present an alternative temporal role mining approach which we name as *Snapshot-based Role Mining* (SNARM). SNARM consists of three phases – (i) creation of one or more UPA matrices from the input TUPA matrix, (ii) generation of roles from each of these UPAs by applying a non-temporal role mining algorithm, and (iii) merging of the roles to derive the final temporal role set. In the subsequent subsections, we give a description of each of these phases.

4.1 Creation of snapshot UPAs

Initially, the distinct sets of time intervals occurring in the TUPA are determined. Each distinct set of time intervals is considered as a snapshot of time. We denote the set of such time snapshots as U_S. For each snapshot T_l $(1 \le l \le |U_S|)$, a UPA matrix UPA$_l$ is created. We call such a UPA as snapshot UPA since it stores UPA information for a given snapshot of time. UPA$_l$ comprises of the same number of rows and columns as the TUPA. For each (i, j) entry of the TUPA which contains T_{ij}, (i) if $T_{ij} = T_l$, then entry (i, j) of UPA$_l$ is set as 1, and (ii) if $T_{ij} \ne T_l$, then entry (i, j) of UPA$_l$ is set as 0. The second step implies that, if $T_l \subset T_{ij}$ or $T_{ij} = \phi$, entry (i, j) of UPA$_l$ will contain 0.

Table 1 Example TUPA matrix

	p_1	p_2	p_3
u_1	5 am–7 am	ϕ	6 am–11 am
u_2	7 am–9 am	6 am–11 am	5 am–7 am
u_3	ϕ	7 am–9 am	6 am–11 am

Table 2 UPA matrix created from TUPA of Table 1

	p_1	p_2	p_3
u_1	1	0	1
u_2	1	1	1
u_3	0	1	1

4.2 Mining of snapshot UPAs

The snapshot UPAs obtained from the previous phase are Boolean matrices. So, any conventional non-temporal role mining algorithm can be applied on each UPA$_l$ to create a set of snapshot roles R_l, a UA matrix UA$_l$ and a PA matrix PA$_l$. Each role of R_l is enabled for the time interval set T_l. After mining each of the UPA matrices, the combined set of temporal roles, denoted as R, is created by taking union of the individual snapshot role sets. The UA and the PA are generated by adding the columns and rows of the individual UA$_l$ and PA$_l$ matrices, respectively. In the REB, the time interval set T_l is associated with each role of R_l and this is done for the individual snapshot sets. As a final step, the temporal roles are merged using the procedure described in Section 3.3. In addition to reducing the number of roles, the merging phase can also be used for reducing the sizes of different RBAC components. Consequently, if the objective is to optimise any minimisation criterion other than the number of roles using SNARM and TARM, the size of the final output can be effectively reduced.

Algorithm 2 (see Fig. 2) shows the overall SNARM approach. Initially, sets R and U_S are initialised as empty sets (Lines 1 and 2, respectively). In the *for loop* of Lines 3–5, the TUPA is searched to find the snapshots which are added to U_S. For each T_l present in U_S, a UPA$_l$ is created (Line 7). After performing non-temporal role mining on UPA$_l$, a set of roles R_l, UA$_l$ and PA$_l$ are obtained (Line 8). The roles of R_l are added to R in Line 9 and the corresponding enabling durations are added to the REB (Line 10). After mining each UPA$_l$, the UA and the PA are created from the individual UA$_l$ and PA$_l$ in Line 12. The roles of R are further merged, if possible, in Line 13 and the UA, PA and REB are updated in Line 14.

For illustrating the working of SNARM, we use the TUPA matrix of Table 1. In this TUPA, three distinct time intervals are present – 5 am–7 am, 6 am–11 am and 7 am–9 am. Corresponding to each of these time intervals, a UPA matrix is created as shown in Tables 3a–c.

Mining the UPA of Table 3a using the MBC based approach, two roles are obtained – $(\{u_2\}, \{p_3\})$ and $(\{u_1\}, \{p_1\})$. The UPA of Table 3b gives the roles $(\{u_1, u_3\}, \{p_3\})$ and $(\{u_2\}, \{p_2\})$ and the UPA matrix of Table 3c produces the roles $(\{u_3\}, \{p_2\})$ and $(\{u_2\}, \{p_1\})$. The enabling duration of each of these roles is same as the time interval for which the UPA is created. These roles cannot be merged any further and the final role set consists of six temporal roles.

TARM and SNARM essentially differ from each other with respect to two aspects – the phase in which the temporal dimension of the input is to be taken into account and the region of the TUPA that needs to be inspected. The phase for considering the temporal component associated with the TUPA comes after the time-agnostic role creation procedure in case of TARM whereas this phase precedes the role creation phase for SNARM. TARM requires the inspection of only a subset of the entire set of triples of the TUPA for creating the temporal roles. In contrast to this, the entire TUPA needs to be scanned for creating the snapshot UPAs.

Depending on the requirements of the organisation implementing TRBAC, if an optimisation metric apart from the size of the final role set is to be minimised, then the non-temporal role mining approach that minimises the corresponding role mining metric can be used in Lines 4 and 8 of Algorithms 1 and 2 (Figs. 1 and 2), respectively. It may be noted that TARM can be modified for solving the generalised temporal role mining problem [27] by terminating the algorithm after a pre-defined number of triples of the TUPA have been fully covered. However, SNARM cannot be extended to solve this problem variant. The reason behind this is that TARM is an iterative approach as opposed to SNARM and

```
Input: TUPA
Output: R, UA, PA, REB
Require: $U_S$: set of snapshots of TUPA
 1: R ← $\phi$
 2: $U_S \leftarrow \phi$
 3: for each snapshot $T$ of TUPA do
 4:     $U_S \leftarrow U_S \cup \{T\}$
 5: end for
 6: for $l \leftarrow 1$ to $|U_S|$ do
 7:     Create UPA$_l$
 8:     Apply a non-temporal role mining algorithm on UPA$_l$ and create $R_l$, UA$_l$, PA$_l$
 9:     R ← R ∪ $R_l$
10:     Add $T_l$ in REB for each role of $R_l$
11: end for
12: Create UA and PA
13: Merge roles of R
14: Update UA, PA, REB
```

Fig. 2 *Algorithm 2: SNARM*

Table 3 UPA matrix for different time intervals of the TUPA in Table 1

	p_1	p_2	p_3
(a) Time interval 5 am–7 am			
u_1	1	0	0
u_2	0	0	1
u_3	0	0	0
(b) Time interval 6 am–11 am			
u_1	0	0	1
u_2	0	1	0
u_3	0	0	1
(c) Time interval 7 am–9 am			
u_1	0	0	0
u_2	1	0	0
u_3	0	1	0

Table 4 Real-world dataset description

Dataset	Users	Permissions	UPAs
apj	2044	1164	6841
emea	35	3046	7220
healthcare	46	46	1486
domino	79	231	730
firewall2	325	590	36,428
firewall1	365	709	31,951
americas small	3477	1587	105,205
americas large	3485	10,127	185,294
customer	10,021	277	45,427

only in case of iterative approaches, such modifications can be carried out.

5 Experimental results

This section presents the experimental results for TARM and SNARM in terms of the number of roles produced and the overall execution time. The optimisation metric considered for the experiments is the total number of roles. For our experiments, we have considered nine benchmark datasets which have been widely used for evaluating performance of various non-temporal and temporal role mining algorithms such as [20, 27–29]. Moreover, we compare the performance of our proposed approaches with two temporal role mining methods such as the one proposed in [27] (which we refer to as GTRMP) and TRMP-MVCL [28]. Since GTRMP supersedes the approach proposed in [26] in terms of both capabilities and performance, we only compare against it instead of the approach in [26]. Similarly, since TRMP-MVCL gives better results than TRMP-MVCM of [28], we have chosen TRMP-MVCL between the two temporal role mining algorithms proposed in [28] for role minimisation. For each of the datasets, the number of users, permissions and UPAs are shown in Table 4.

Using the MBC based approach of [20], we have created a UA and a PA for each dataset. Since these datasets are non-temporal in nature, a temporal component has been synthetically added to each of them by generating the REBs. To do this, we have considered ten distinct sets of time intervals and three cases of time interval relationships – contained, overlapping and mixed. The number of distinct sets of time intervals could have been taken to be any number between 1 and 24 (considering 24 h in a day). However, we have fixed the number of distinct time interval sets at ten because we wanted to choose a number that more or less lies mid-way in the range of 1 and 24. The reason behind this is that we did not want the number of distinct time interval sets to be too high or too low. Too high number of distinct sets of time intervals seems to be infeasible for practical scenarios and too less number would not have reflected the unique aspects of the different algorithms considered for performance evaluation in a comprehensive manner. In case of contained and overlapping time interval relationships, ten contained and ten overlapping time interval sets are present, respectively. For the contained relationship, each role in the REB is enabled for any one of the contained time interval sets. Similarly, for the overlapping case, each role is enabled for an overlapping set of time intervals. However, for the creating the mixed case, five contained and five overlapping sets of time intervals have been considered so that the total number of distinct sets of time intervals remains ten. A total of 50% of the roles are enabled for any one of the contained time interval sets and the remaining 50% are enabled for an overlapping time interval set. For each of the cases, while generating the REBs, six out of the ten sets of time intervals have been assumed to be more frequently occurring than the remaining four.

Since the REBs are randomly generated, 30 different REBs are considered for each dataset. The UA and PA matrices generated as output of the MBC algorithm and the synthetically created REBs are combined to obtain the TUPA matrices. Thus for the contained case, any one or more of the contained time interval sets are associated with each temporal UPA of TUPA. Similarly, for the overlapping case, any one or more of the overlapping sets of time intervals is associated with each triple of TUPA. However, for the mixed case, any one contained set of time intervals (out of the five considered) is associated with approximately half of the total number of triples of the TUPA and any one overlapping time interval set (out of the five) is associated with the remaining triples of the TUPA. The temporal role mining algorithms take as input these TUPA matrices. While we have not derived a tight upper bound, since the original UA, PA, and REB used to obtain the TUPA is a valid solution, it can be considered as an upper bound. Thus, this UA, PA and REB combination serves as a baseline against which we can evaluate the performance of our algorithms. In our experiments, we analyse the goodness of the output produced by the proposed temporal role mining approaches by examining to what extent the output deviates from the optimal solution upper bound. The experiments were carried out on a machine having 2.7 GHz Intel Xeon processor with 8 GB RAM. As 30 REBs were created for each dataset, we have repeated each experiment 30 times and report the median number of roles and the average execution time over the 30 simulations.

For our proposed approaches, the MBC algorithm has been used for non-temporal role mining. In case of GTRMP, the value of the temporal mismatch limit has been set as 0. Our experiments have shown that TRMP-MVCL gives the least number of roles for $\theta = 0.1$ for these TUPA matrices. So in Tables 5 and 6, we present the number of roles and execution time of TRMP-MVCL for $\theta = 0.1$. The first column of the table shows the optimal solution upper bound which we refer to as OPT-UB.

Table 5 shows the number of roles produced by TARM and SNARM as well as those given by GTRMP and TRMP-MVCL. The number of roles produced by SNARM increases as the number of contained sets of time intervals increases. Thus, the number of roles obtained for the overlapping case is lower than that obtained for the mixed case which in turn is lower than the number of roles produced by the contained case. For TARM, the reverse trend is observed. As the number of overlapping time interval sets increases, the size of the role set produced by TARM also increases due to the greater amount of time interval splitting caused by overlapping sets of time intervals.

Between SNARM and TARM, the latter performs the best when only contained time interval sets are present. In fact, for the contained case, TARM produces a number of roles equal to OPT-UB. For the remaining two cases, SNARM gives fewer roles than TARM. This happens due to the complete lack of splitting in case of SNARM. TARM outperforms both GTRMP and TRMP-MVCL for the contained case and SNARM gives better results than GTRMP and TRMP-MVCL for the mixed and overlapping cases. However, for some datasets, TRMP-MVCL gives equal or lower number of roles than those produced by SNARM and TARM. Thus, it can be concluded that if only contained sets of time intervals are present in the TUPA, using TARM for role mining would give better results than the other approaches. If, however, overlapping or a mix of both types of time interval sets occurs in the input, SNARM would effectively minimise the number of roles. Since the interrelationships among the time interval sets have a significant impact on the performance of both SNARM and TARM, the availability of two different algorithms allows the flexibility of choosing the most suitable one depending on the nature of the input.

Table 6 shows the performance comparison in terms of the average execution time of the four temporal role mining approaches. Between SNARM and TARM, the former method executes in a lower amount of time. This occurs because non-temporal role mining is carried out for a much higher number of times in TARM than SNARM. Also, the creation of the snapshot UPAs in SNARM takes lesser time than generation of temporal roles corresponding to the time-agnostic roles in TARM. SNARM and TARM, in general, takes lesser amount of time to execute than both GTRMP and TRMP-MVCL. In very few cases, GTRMP and TRMP-MVCL has a lower execution time than the two proposed algorithms.

It can be observed from the experimental results, the number of roles, in general, produced by any one or both of TARM and SNARM is either the lowest or among the lowest over all the four algorithms considered for performance evaluation. For execution time also, a similar trend is observed. For the *apj* dataset, though the execution time of GTRMP is the lowest, the number of roles given by GTRMP is much higher than SNARM for all cases of time interval relationships and is higher than TARM for the contained case. Moreover, in only one case (mixed type of time interval relationship for the *healthcare* dataset), TRMP-MVCL performs better than both TARM and SNARM in terms of the number of roles as well as the average execution time. For the cases where TRMP-MVCL generates an equal number of roles as those produced by TARM and SNARM, TRMP-MVCL, on an average, takes longer to execute than the two proposed algorithms.

Table 5 No. of roles given by TARM, SNARM, GTRMP and TRMP-MVCL

Dataset	Contained					Overlapping				Mixed			
	OPT-UB	TARM	SNARM	GTRMP	TRMP-MVCL	TARM	SNARM	GTRMP	TRMP-MVCL	TARM	SNARM	GTRMP	TRMP-MVCL
apj	456	456	458	468	459	536	456	469	458	512	457	468	458
emea	34	34	34	36	34	34	34	36	34	34	34	36	34
healthcare	15	15	16	17	15	28	15	17	15	25	16	16	15
domino	20	20	20	21	20	52	20	21	20	47	20	20	20
firewall2	10	10	10	10	10	17	10	10	10	15	10	10	10
firewall1	69	69	72	77	75	219	69	77	74	171	70	78	75
americas small	211	211	234	256	227	788	216	246	222	586	224	250	226
americas large	421	421	426	525	437	587	423	523	431	534	426	526	436
customer	276	276	276	278	277	276	276	278	277	276	276	277	277

Table 6 Average execution time (in seconds) of TARM, SNARM, GTRMP and TRMP-MVCL

Dataset	Contained				Overlapping				Mixed			
	TARM	SNARM	GTRMP	TRMP-MVCL	TARM	SNARM	GTRMP	TRMP-MVCL	TARM	SNARM	GTRMP	TRMP-MVCL
apj	4.43	4.40	0.31	25.75	5.66	4.47	0.55	22.44	5.25	4.47	0.44	23.65
emea	0.04	0.03	0.71	0.21	0.04	0.03	0.74	0.16	0.04	0.03	0.73	0.18
healthcare	<0.01	<0.01	0.01	0.18	<0.01	<0.01	0.01	0.10	<0.01	<0.01	0.01	0.13
domino	<0.01	<0.01	0.02	0.01	0.02	<0.01	0.03	0.01	0.02	0.01	0.03	0.01
firewall2	0.04	0.04	0.14	82.33	0.13	0.04	0.18	77.94	0.10	0.04	0.17	73.97
firewall1	0.18	0.19	0.48	67.32	1.18	0.19	1.09	53.19	0.84	0.17	0.80	55.47
americas small	7.80	7.43	12.02	2679.13	139.15	8.11	23.27	1394.03	46.66	7.71	17.34	1987.33
americas large	175.48	180.27	259.07	3344.03	230.42	177.45	357.83	1767.18	218.15	177.07	298.33	2947.63
customer	4.63	4.72	39.96	28.18	4.25	4.43	40.18	23.52	4.25	4.67	41.25	23.12

Thus, in general, our proposed algorithms outperform the existing temporal role mining algorithms.

6 Conclusion and future work

In this paper, we have proposed two temporal role mining algorithms, TARM and SNARM, which can be used for migrating from RBAC to TRBAC. Starting with an existing UA and PA and a newly created REB, both of these approaches use non-temporal role mining methods in conjunction with temporal element consideration procedures to obtain a set of temporal roles which can be used for deploying the new TRBAC system. Results obtained from benchmark datasets indicate that our proposed approaches perform better than existing temporal role mining algorithms.

The algorithms presented in this paper as well as the existing ones can be augmented in future for enforcing Temporal separation of duty constraints and also for performing incremental temporal role mining in order to specify several organisational polices and for workflow management. Moreover, top-down temporal role engineering approaches can be designed which can be used along with the bottom-up temporal role mining methods for creating semantically meaningful temporal roles. In the future, we plan to address these issues.

7 References

[1] Sandhu, R.S., Coyne, E.J., Feinstein, H.L., *et al.*: 'Role-based access control models', *IEEE Comput.*, 1996, **29**, (2), pp. 38–47

[2] Coyne, E.J.: 'Role engineering'. Proc. of 1st ACM Workshop on Role-Based Access Control, 1995, pp. 15–16

[3] Narouei, M., Takabi, H.: 'Towards an automatic top-down role engineering approach using natural language processing techniques'. Proc. of 20th ACM Symp. on Access Control Models and Technologies, 2015, pp. 157–160

[4] Neumann, G., Strembeck, M.: 'A scenario-driven role engineering process for functional RBAC roles'. Proc. of 7th ACM Symp. on Access Control Models and Technologies, 2002, pp. 33–42

[5] Roeckle, H., Schimpf, G., Weidinger, R.: 'Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization'. Proc. of 5th ACM Workshop on Role-Based Access Control, 2000, pp. 103–110

[6] Strembeck, M.: 'Scenario-driven role engineering', *IEEE Secur. Privacy*, 2010, **8**, (1), pp. 28–35

[7] Frank, M., Buhmann, J.M., Basin, D.: 'Role mining with probabilistic models', *ACM Trans. Inf. Syst. Secur.*, 2013, **15**, (4), pp. 1–28

[8] Molloy, I., Chen, H., Li, T., *et al.*: 'Mining roles with multiple objectives', *ACM Trans. Inf. Syst. Secur.*, 2010, **13**, (4), pp. 36:1–36:35

[9] Vaidya, J., Atluri, V., Guo, Q.: 'The role mining problem: a formal perspective', *ACM Trans. Inf. Syst. Secur.*, 2010, **13**, (3), pp. 27:1–27:31

[10] Blundo, C., Cimato, S.: 'A simple role mining algorithm'. Proc. of 25th ACM Symp. on Applied Computing, 2010, pp. 1958–1962

[11] Vaidya, J., Atluri, V., Guo, Q., *et al.*: 'Edge-RMP: Minimizing administrative assignments for role-based access control', *J. Comput. Secur.*, 2009, **17**, (2), pp. 211–235

[12] Lu, H., Hong, Y., Yang, Y., *et al.*: 'Towards user-oriented RBAC model', *J. Comput. Secur.*, 2015, **23**, (1), pp. 107–129

[13] Zhang, D., Ramamohanarao, K., Ebringer, T.: 'Role engineering using graph optimisation'. Proc. of 14th ACM Symp. on Access Control Models and Technologies, 2007, pp. 139–144

[14] Harika, P., Nagajyothi, M., John, J.C., *et al.*: 'Meeting cardinality constraints in role mining', *IEEE Trans. Dependable Secur. Comput.*, 2015, **12**, (1), pp. 71–84

[15] Hu, J., Khan, K.M., Bai, Y., *et al.*: 'Constraint-enhanced role engineering via answer set programming'. Proc. of 7th ACM Symp. on Information, Computer and Communications Security, 2012, pp. 73–74

[16] Sarana, P., Roy, A., Sural, S., *et al.*: 'Role mining in the presence of separation of duty constraints'. Proc. of 11th Int. Conf. on Information Systems Security, 2015, pp. 98–117

[17] Blundo, C., Cimato, S.: 'Constrained role mining'. Proc. of 8th Int. Workshop on Security and Trust Management, 2012, pp. 289–304

[18] Vaidya, J., Atluri, V., Warner, J., *et al.*: 'Role engineering via prioritized subset enumeration', *IEEE Trans. Dependable Secur. Comput.*, 2010, **7**, (3), pp. 300–314

[19] Zhang, W., Chen, Y., Gunter, C., *et al.*: 'Evolving role definitions through permission invocation patterns'. Proc. of 18th ACM Symp. on Access Control Models and Technologies, 2013, pp. 37–48

[20] Ene, A., Horne, W., Milosavljevic, N., *et al.*: 'Fast exact and heuristic methods for role minimization problems'. Proc. of 13th ACM Symp. on Access Control Models and Technologies, 2008, pp. 1–10

[21] Huang, H., Shang, F., Liu, J., *et al.*: 'Handling least privilege problem and role mining in RBAC', *J. Comb. Optim.*, 2013, **30**, (1), pp. 63–86

[22] Lu, H., Vaidya, J., Atluri, V.: 'Optimal Boolean matrix decomposition: application to role engineering'. Proc. of 24th IEEE Int. Conf. on Data Engineering, 2008, pp. 297–306

[23] Lu, H., Vaidya, J., Atluri, V.: 'An optimization framework for role mining', *J. Comput. Secur.*, 2014, **22**, (1), pp. 1–31

[24] Mitra, B., Sural, S., Vaidya, J., *et al.*: 'A survey of role mining', *ACM Comput. Surv. (CSUR)*, 2016, **48**, (4), p. 50

[25] Bertino, E., Bonatti, P.A., Ferrari, E.: 'TRBAC: a temporal role-based access control model', *ACM Trans. Inf. Syst. Secur.*, 2001, **4**, (3), pp. 191–233

[26] Mitra, B., Sural, S., Atluri, V., *et al.*: 'Toward mining of temporal roles'. Proc. of 27th Annual IFIP WG 11.3 Working Conf. on Data and Applications Security and Privacy, 2013, pp. 65–80

[27] Mitra, B., Sural, S., Atluri, V., *et al.*: 'The generalized temporal role mining problem', *J. Comput. Secur.*, 2015, **23**, (1), pp. 31–58

[28] Mitra, B., Sural, S., Vaidya, J., *et al.*: 'Mining temporal roles using many-valued concepts', *Comput. Secur.*, 2016, **60**, pp. 79–94

[29] Molloy, I., Li, N., Li, T., *et al.*: 'Evaluating role mining algorithms'. Proc. of 14th ACM Symp. on Access Control Models and Technologies, 2009, pp. 95–104