RMiner: A Tool Set for Role Mining

Ruixuan Li, Huaqing Li, Wei Wang, Xiaopu Ma, Xiwu Gu
School of Computer Science and Technology
Huazhong University of Science and Technology
Wuhan, Hubei 430074, China
rxli@hust.edu.cn, {lihuaqing, ybwei.wang, xpma}@smail.hust.edu.cn, guxiwu@hust.edu.cn

ABSTRACT
Recently, there are many approaches proposed for mining roles using automated technologies. However, it lacks a tool set that can be used to aid the application of role mining approaches and update role states. In this demonstration, we introduce a tool set, RMiner, which is based on the core of WEKA, an open source data mining tool. RMiner implements most of the classic and latest role mining algorithms and provides interactive tools for administrator to update role states. The running examples of RMiner are presented to demonstrate the effectiveness of the tool set.

Categories and Subject Descriptors
D.4.6 [**Operating Systems**]: Security and Protection—Access Controls; H.2.8 [**Database Management**]: Database Application—Data Mining

General Terms
Security, Management.

Keywords
Role-based access control, role engineering, role mining, tool set.

1. INTRODUCTION
The goal of role engineering is to configure a role-based access control (RBAC) system. Usually, role engineering process includes generating roles, assigning roles to users and updating role state. There are two basic approaches for role engineering: top-down approach and bottom-up approach. The top-down approach uses descriptions of business processes, security policies, and other business information to configure an RBAC system. The bottom-up approach uses data mining techniques to discover roles from the existing user-permission assignments and other business information. The bottom-up approach is commonly called role mining.

There are many algorithms have been proposed to mine the roles. The role mining algorithms discover roles from the real world data set that is usually with noise. It is something difficult to generate a role state that is accurately satisfied with the security needs for an RBAC system. Researches use the ratio of the number of generated roles exactly matching the original role sets to the number of original roles to evaluate a role mining algorithm. If the original role sets are not known, the weighted structural complexity (WSC) [4], which sums up the number of relationships in an RBAC state, is often used to evaluate the performance of the role mining algorithms. Nevertheless, role mining algorithms will not completely solve the role configuration problem. The administrators usually need to correct some improper roles generated by role mining algorithm. Therefore, we need a platform to implement the role mining algorithms and provide a tool set to aid the administrators to update the role state.

WEKA [1] is known as a collection of machine learning algorithms for data mining tasks. However, WEKA workbench platform is not suitable for role mining tasks. The reasons are as follows. Firstly, the input of WEKA is a set of instances that describes the relationship between attributes. An instance means the values on every attributes. In contrast, the input of role mining algorithms contains the user-permission assignments and some business information. If we use instances to indicate the assignments, we must list all the permissions, but their value is just 0 or 1. It is quit redundant and easy to mix assignments and attributes. Secondly, the output of role mining algorithms is a set of associated permissions. This is similar to association rule mining in WEKA, whereas the evaluation criterion in role mining is different from support and confidence that used in association rule mining. The rules with low frequency may be also a role in role mining field. Thirdly, there is lack of a common implementation of role mining algorithms, which makes it hard for researches to do a comparison experiments when designing a new algorithm. Therefore, we need to develop an independent tool set for role mining since the role mining algorithms cannot be migrated to WEKA platform directly.

In this demonstration, we show a role mining platform, namely RMiner. In RMiner, we implement most of the classical role mining algorithms, such as ORCA [2], HierarchicalMiner [3], CompleteMiner [4], FastMiner [4], Anti-apriori [5], GraphOptimization [6], and WeightedRoleMinning [7]. RMiner also provides a role state editor for updating the role state generated by the role mining algorithms. In addition, RMiner offers some abstract interfaces that the users can easily add new role mining algorithms, and discover the difference of the new algorithms compared with others using the visualization tools.

RMiner is implemented by Java. The program for RMiner can be downloaded from http://code.google.com/p/rminer/.

SACMAT'13, June 12-14, 2013, Amsterdam, The Netherlands.

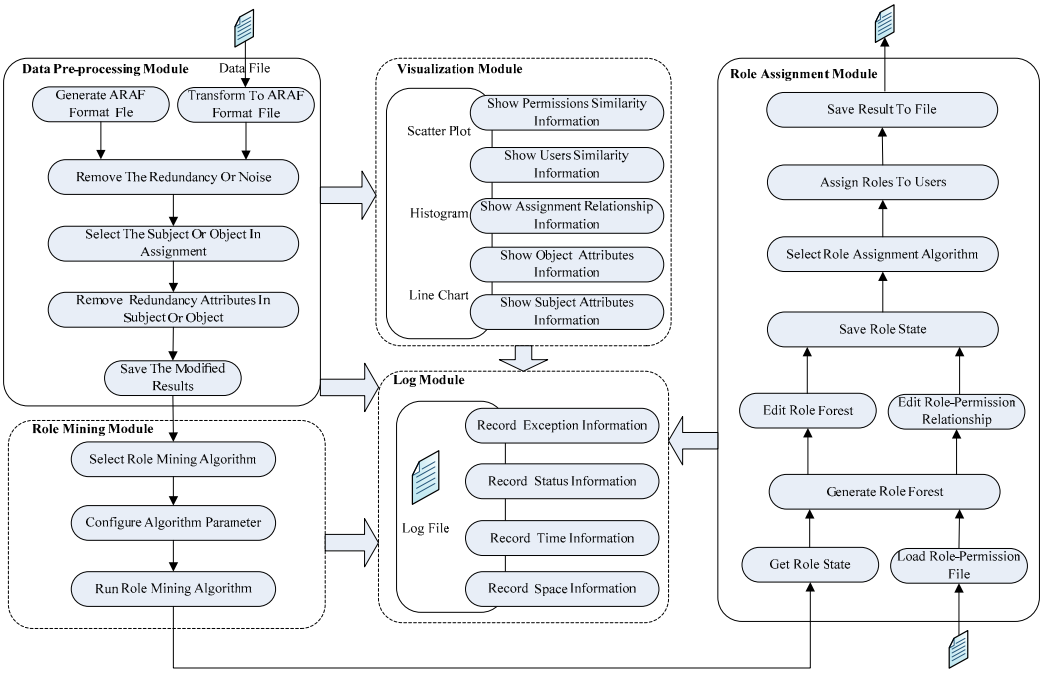

Figure 1. The framework of RMiner.

2. THE FRAMEWORK OF RMINER

In RMiner, we want to provide a tool set to help researchers or administrators do role engineering work. The platform of RMiner can also do some comparison experiments to make them understand the characteristics of the role mining algorithms clearly. The framework of RMiner, as shown in Figure 1, includes five parts: data pre-processing, role mining, role assignment, visualization, and log.

Before using the RMiner, the users must transform the input user-permission relationship to ARAF file format that will be introduced in Section 3.1. The users can also use the artificial user-permission assignments generated by RMiner automatically if they don't offer real user-permission relationships. In general, if the users want to calculate the accuracy of the algorithm, they need to provide the predefined set of roles. After that, the users can remove the redundancy or noise in the input manually. Then the users need to select the subject and object in the assignments and remove the redundant attributes. When the input data are ready, RMiner will use the selected algorithm to generate roles. In the process of role mining, the users can adjust the parameters of the selected algorithm. The output of the selected algorithm contains: ROLES (set of roles), UA (user to role assignments) and PA (role to permission assignments). If these outputs are not satisfied with the users' security needs, the users can update them through a visualization tool, namely role editor. In the stage of role edit, the users can use the role mining result or load the role to permission file for role assignment. If the users want to edit the result, they can edit the role to permission relationship shown in table and edit the role state shown as role forest in RMiner. After editing the role to permission relationship and role state, the users need to select an algorithm for role assignment. The role assignment algorithm builds the Integral UA and PA relationship according to the initial user-permission relationship. The users can save the results into a specified file.

After loading the role to permission assignment relationship and the attributes of the subject and object, the users can view the attribute and assignment relationship information through the visualization module of RMiner. In this module, the users can also know the similarity information of the users and the permissions in the user-permission relationship. RMiner shows the information via scatter plot, histogram or line chart. Through the permission similarity information shown in RMiner, the users can estimate the number of the original roles, which will be useful if they just want to know the number of the roles roughly and don't want to run the role mining algorithms. Besides the above four modules, the log module records the running information, which contains exception, status, time and space information.

3. THE DESIGN OF RMINER

3.1 The Data Structure of RMiner

In order to maintain independence and generality of data set for role mining process, we develop the assignment relationship with attribute file format (ARAF) based on attribute relation file format (ARFF) of WEKA. The only difference between ARAF and ARFF is the concept of latitude. We use latitude to indicate the two assignment sides, and the attributes of a side are described by ARFF format. Figure 2 shows an example of user-permission assignments with user attributes using ARAF format. An ARAF file can be divided into two parts. The first part is the assignment, including assignment name, assignment subject, assignment object and the subject to object assignment relationships which are described by 0/1 matrix. The second part is the attribute information of latitude. This information includes attribute name, attribute data type, such as numeric, string, time and enumerated, and the value of attributes for every instance in latitude.

The latitude of ARAF can be user, role and permission. Thus, we can use ARAF file to indicate the user to permission relationships, role to permission relationships and user to role relationships. In

```
@assignment upa
@latitude user {Ann,Bob,Carl}
@latitude permission {rAcc,wAcc,cdAcc,cTrans,rTrans}
@matrix
1,1,0,1,0
1,1,0,1,0
1,0,1,0,1
@attributes of user
@attribute age numeric [10, 80]
@attribute department {HR, BM}
@data
45,HR
20,HR
29,BM
```

Figure 2. An example of ARAF.

general, the assignment in an RBAC system will be relatively sparse. The matrix can be represented using the pressurized reduced storage method. We only record the object labels in every subject assignment.

In the implementation of RMiner, the ARAF file format is corresponding to the data structure for the assignments. It includes sub-structure latitude, matrix, attributes and instances.

3.2 Data Pre-processing Tool

The RMiner interface is shown in Figure 3. We can see that there are five option cards in RMiner: Preprocess, RoleMining, Assignment, Visualize and About. The Preprocess part is to choose and modify the data set used in role mining algorithm. The RoleMining part is to generate roles using the selected algorithm. The Assignment part is to assign the generated roles to users and update the role state. The Visualize part is to provide the interactive 2D display for the role mining process.

Figure 3 also shows the module of data pre-processing interface. The data pre-processing tool in RMiner is to help users clear the noise in the data set. It provides several predefined real data sets and an artificial data generation tool. These will facilitate the use of the RMiner working environment. The users can load the user-permission relationship file, or generate user-permission relationship with RMiner automatically. The users can also edit the user-permission information with the edit button.

3.3 Role Mining Platform

After the preprocessing, the data set is used as the input for role mining algorithm. The users select the role mining algorithm, configure the algorithm parameters, and activate the algorithm. RMiner will automatically execute the selected algorithm, and record the running time, memory space usage and other evaluation information. The users can use the visualization tool to view the generated role set. The same data set can be activated by different role mining algorithms. Thus, we can do the comparison experiments on different algorithms. The generated role set will be formatted as assignment structure ARAF, and their latitudes are role and permission respectively.

In RMiner, we have implemented 13 role mining algorithms, as shown in Figure 4. CompleteMiner, FastMiner and HPRoleMinimization can output role set. ORCA, GraphOptimization, HierarchicalMiner, HPEdgeMinimization, FeatureMiner and YeHRMiner can output role set and role hierarchies. StateMiner and MinimalPerturbation consider perturbation, and WeightedRoleMinning considers the weights of permissions. CCRMimplement and ThesisAlgorithmImplement considers constraints. Anti-apriori mines the constraints in Role-Based Access Control. The users are free to add new role mining algorithms to RMiner platform.

3.4 Role Assignment Tool

The role set generated by role mining algorithm may not fully meet the security needs of the real application. In order to increase the interpretability and generalization ability, the users can manually update the role state in RMiner. As shown in Figure 5, RMiner provides a role editor for users. Role editor uses the partial order of permissions in role sets to build the role forest, which will help user understand the role hierarchy and the weights of roles. The users can use role editor to add, delete or modify the role set. In Figure 5, the left part is the role forest, in which the users can edit every role, and the right part is the role to permission relationship, in which the green part is inherited from parent roles and cannot be edited. After completing modifying the role set, the users can assign the roles to users according to the original user-permission assignments. As we know, this process is a set covering problem. RMiner provides greedy and genetic algorithms to solve the role assignment problem.

3.5 Visualization

Visualization tool in RMiner can assist the users to analysis the mining results more clearly. It translates the data of role mining process into a variety of graphics that indicates the status of the process and reveals the inner nature and regularity. Figure 6 shows the permission similarity information on the university_runningexample dataset. Both X-axis and y-axis represent permissions. The deeper the color of the block is, the bigger the probability of forming a role is. The process of building an RBAC system involves assignment relationships and business information. Visualization tool provides a scatter plot, histogram line and statistical analysis, which facilitates the understanding of the role mining process.

4. CONCLUSIONS AND FUTURE WORK

RMiner provides user a visual role mining and role state updating platform. The design of RMiner is according to the building process of an RBAC system. Users can use the platform to build an RABC system or carry out comparison experiments to analysis the difference between role mining algorithms. RMiner is a unified verification platform that simplifies the process of role mining research and experimental work. However, RMiner only provides a common tool for building an RBAC system. It is not fully adapt to the changes in the RBAC system. At the same time, there is still some inadequate in the data pre-processing and role state updating processes. The data pre-processing almost relies on the manual manipulation. It is hard to apply to large data set. The display of role editor on large data sets is something complex, which will weaken the interaction and the user experience. We will try to improve these in the future work.

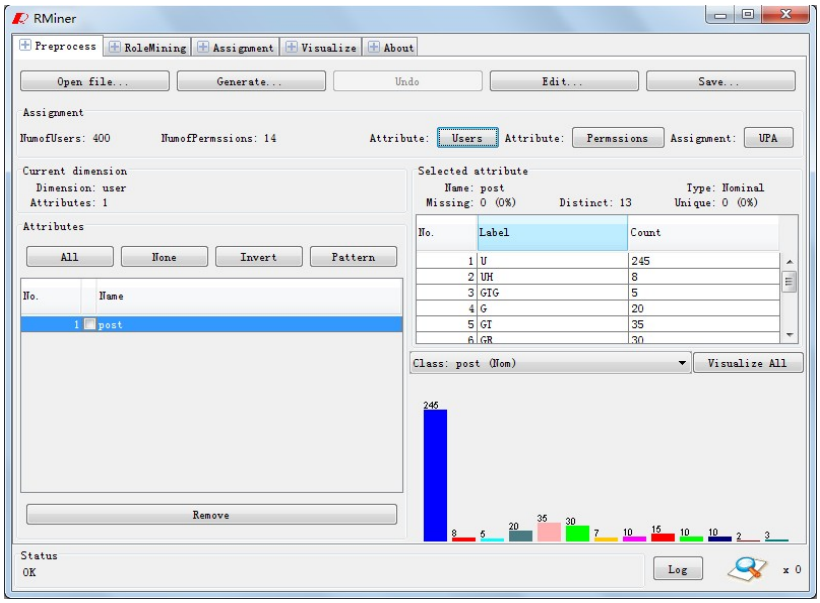

Figure 3. The module of data pre-processing.

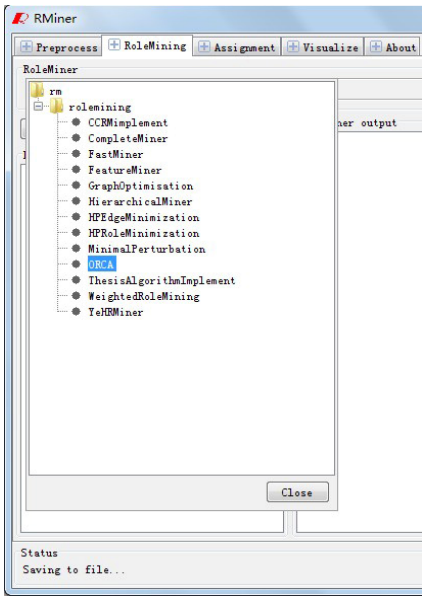

Figure 4. The module of role mining.

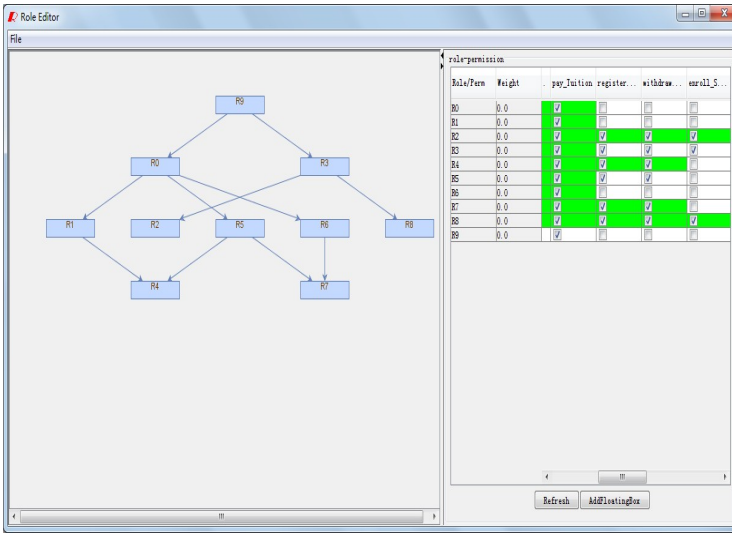

Figure 5. Role editor and assignment in RMiner.

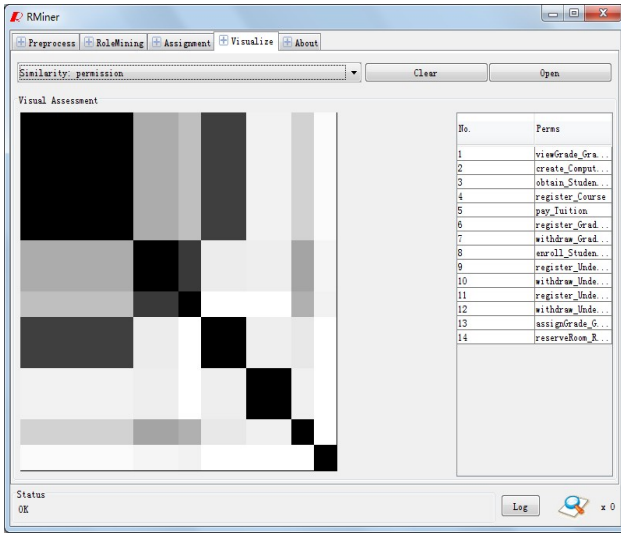

Figure 6. Permission similarity information.

Acknowledgements

This work is supported by National Natural Science Foundation of China under grants 61173170 and 60873225, National High Technology Research and Development Program of China under grant 2007AA01Z403, and Innovation Fund of Huazhong University of Science and Technology under grants 2012TS052 and 2012TS053.

5. REFERENCES

[1] Bouckaert, R., Frank, E., Hall, M., Kirkby, R., Reutemann, P., Seewald, A., and Scuse, D. 2008. *WEKA Manual for Version 3-6-0*. University of Waikato, Hamilton, New Zealand, 2008.

[2] Schlegelmilch, J., and Steffens, U. 2005. Role mining with ORCA. In *Proceedings of the 10th ACM Symposium on Access Control Models and Technologies* (Stockholm, Sweden, June 2005). SACMAT'05, ACM, 168-176.

[3] Molloy, I., Chen, H., Li, T., Wang, Q., Li, N., Bertino, E., Calo, S., and Lobo, J. 2008. Mining roles with semantic meanings. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies* (Estes Park, CO, USA, June 2008). SACMAT'08, ACM, 21-30.

[4] Vaidya, J., Atluri, V., and Warner, J. 2006. Roleminer: mining roles using subset enumeration. In *Proceedings of the 13th ACM Conference on Computer and Communications Security* (Alexandria, VA, USA, October 2006). CCS'06. ACM, 144-153.

[5] Ma, X., Li, R., Lu, Z., and Wang, W. 2012. Mining Constraints in Role-Based Access Control. *Mathematical and Computer Modelling*, Elsevier, 55, 1-2 (2012), 87-96.

[6] Zhang, D., Ramamohanarao, K., and Ebringer, T. 2007. Role engineering using graph optimisation. In *Proceedings of the 12th ACM Symposium on Access Control Models and Technologies* (Sophia Antipolis, France, June 2007). SACMAT'07, ACM, 139-144.

[7] Ma, X., Li, R., Lu, Z. 2010. Role Mining Based on Weights. In *Proceedings of the 15th ACM Symposium on Access Control Models and Technologies* (Pittsburgh, PA, USA, June 2010). SACMAT'10, ACM, 65-74.