CERIAS Tech Report 2002-59

An Algorithm for Building User-Role Profiles in a Trust Environment.

by E Terzi, Y Zhong, B Bhargava, Pankaj, Pankaj, S Madria
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

An Algorithm for Building User-Role Profiles in a Trust Environment[1]

Evimaria Terzi[1], Yuhui Zhong[1], Bharat Bhargava[1], Pankaj[2], and Sanjay Madria[3]

[1] Center for Education and Research in Information Assurance and Security (CERIAS)
and Department of Computer Sciences,
Purdue University, West Lafayette, IN-47907, USA
{edt,zhong,bb}@cs.purdue.edu
[2]Department of Management Information Systems,
Krannert Graduate School of Management,
Purdue University, West Lafayette, IN – 47907, USA
pankaj@mgmt.purdue.edu
[3]Department of Computer Science, University of Missouri-Rolla,
Rolla, MO- 65409, USA
madrias@umr.edu

Abstract. A good direction towards building secure systems that operate efficiently in large-scale environments (like the World Wide Web) is the deployment of Role Based Access Control Methods (RBAC). RBAC architectures do not deal with each user separately, but with discrete roles that users can acquire in the system. The goal of this paper is to present a classification algorithm that during its training phase, classifies roles of the users in clusters. The behavior of each user that enters the system holding a specific role is traced via audit trails and any misbehavior is detected and reported (classification phase). This algorithm will be incorporated in the Role Server architecture, currently under development, enhancing its ability to dynamically adjust the amount of trust of each user and update the corresponding role assignments.

1 Introduction

The goal of this paper is to provide a solution to the problem of classifying role profiles. The notion of the role is adopted for wide environments like the World Wide Web (WWW) that allow for wide range information exchange and access. In such a context, where accessibility to large amount of data is provided, certain security concerns about the data itself are also raised. In such an environment, there is no guarantee that all the users entering a system behave appropriately. Establishing security and access control mechanisms in particular is a rather difficult task mainly because of two

[1] This research is supported by the CERIAS and NSF grants CCR-9901712 and CCR-0001788

Y. Kambayashi, W. Winiwarter, M. Arikawa (Eds.): DaWaK 2002, LNCS 2454, pp. 104–113, 2002.

reasons: 1) the large number of users that want to access the applications and 2) the fact that these users are mainly unknown. RBAC has emerged as a promising technology for efficient security management and enforcement [1,6] in such large-scale environments. Adapting RBAC to enforce security in the Web is increasingly seen as a solution for scalability issues, because permissions are associated to roles instead of users and the number of roles is much smaller compared to the number of users [28,29]. Therefore, the notion of role that can be assigned to a group of users is vital for such applications. However the problem that remains unsolved is how to trace the users' behavior dynamically and guarantee that their behavior is within the limits imposed by the role they hold. This paper deals mainly with this problem and proposes an algorithm that will be a part of the Role Server architecture as described below.

1.1 The Role Server Architecture

The formalization of trustworthiness of users that enter large scale environments is a major research direction. The main goals of this research involve: (a) designing a trustworthiness model, (b) determining the permission set granted to a user, and (c) evaluating the reliability of credentials that a user obtains from third parties. The overall goal of our research is to build a prototype of a Role Server that securely assigns roles to users. The main components of this Role Server are shown in Figure 1.

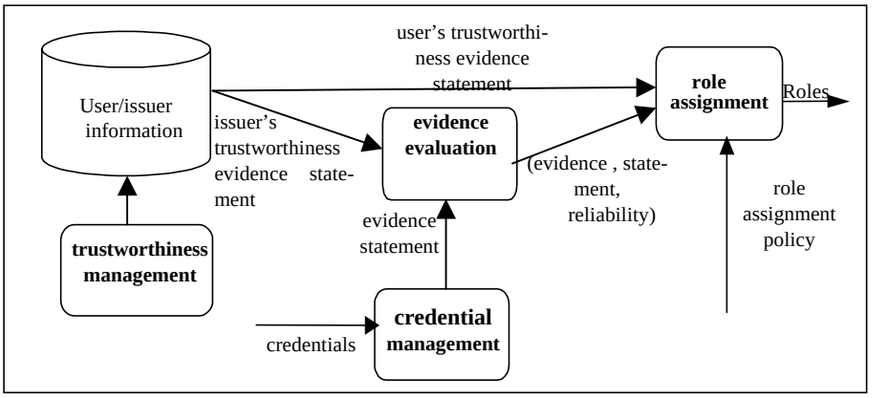

Fig. 1. Role Assignment

The main components of the Role Server architecture presented in Figure 1 are (a) the trustworthiness management, (b) the evidence evaluation, (c) the credential management, and (d) the role assignment component. The functionality of these components are described in [32].

2 Related Work

Our research on building role profiles is primarily inspired by the work done in three main research areas, namely, data mining, classification and clustering; intrusion detection and; user profile building.

Clustering can be loosely defined as the process of organizing objects into groups whose members are similar in some way. An introduction to clustering techniques is given in [11,13,15,17] while a brief overview of the existing clustering techniques is given in [14,12]. Generally, there are two major styles of clustering: *partitioning* (often called *k-clustering*), in which every object is assigned exactly to one group, and *hierarchical clustering*, in which each group of size greater than one is in turn composed of smaller groups. Clustering algorithms for user profiles extraction have been employed in [16]. In order to extract users access profiles, the user sessions are clustered based on the pair-wise dissimilarities and using a fuzzy clustering algorithm. An extension of the previous work is given in [18] where new algorithms for fuzzy clustering of relational data are presented. The basics of fuzzy clustering techniques are presented in [19].

Closely related to our work is the research being conducted in the area of Intrusion Detection (ID). The major objective of these research efforts is to determine whether the behavior of a user within the system is normal or intrusive. There are mainly two techniques towards Intrusion Detection namely *anomaly detection* and *misuse detection.* A taxonomy and an overview of the existing ID Systems (IDS) is presented in [**7**]. Generally, anomaly detection identifies activities that vary from established patterns for users or group of users. Anomaly detection typically involves the creation of knowledge bases that contain the profiles of the monitored activities ([25]). Misuse detection on the other hand, involves the comparison of the users activities with the known behaviors of attackers attempting to penetrate a system ([20,21]). The IDSs specify whether the behavior of the user is (is not) within the acceptable limits of the behavior that the system can (cannot) tolerate. The classification algorithm presented in this paper is based on [31]. In this paper the authors build clusters that characterize normal and intrusive behaviors.

Building user profiles is another important related area of research. There is a considerable amount of work that has been done on building users group profiles for web sites access, and which mainly lays in the area of web mining. The researchers involved in this area have noticed that as the complexity of the Web sites increases the statistics provided by existing Web log analysis tools [1,2,3] may prove to be inadequate, and more sophisticated types of analyses is necessary. *Web Usage Mining*, which is application of data mining techniques to large Web data repositories, adds powerful techniques to the tools available to Web site administrators for analyzing Web Site usage. Web Usage Mining Techniques developed in [8,9,10,27,30] have been used to discover frequent itemsets, association rules [5], clusters of similar pages and users, sequential patterns [26], and perform path analysis. In web usage mining the group profiles are constructed based on the user's navigation patterns within the web site and not based on the actions that have been conducted by the users on the

system in terms of commands, memory and disk usage etc. In other words, the web profiles are based on the web log analysis, while the building role profiles in our case is based on the audit data analysis.

3 The Algorithm

The proposed algorithm can be described as follows:

The initial input to this algorithm is a data record of the audit log from which the system administrator can select *n* attributes which are the most appropriate ones. Therefore, each data-point is a value in the n-dimensional space. An additional field needs to be added in each record and this field will denote the role of the user the specific data record represents. Therefore a record – input to our algorithm will look as follows as $\mathbf{[X_1,X_2,\ldots X_n, R_1]}$. This implies that the values of the attributes of the log file for the specific user are X_1, X_2, … X_n and that the role of this user is R_1. The goal of the algorithm is the following: Having a certain number of such data records, a certain number of clusters should be formed. The minimum number of these clusters will be equal to the number of different roles. But this is just the lower bound. The upper bound can obviously be the number of users, a case that will rarely occur given that the behaviors of users holding the same role will not be completely diverse. Each cluster can contain users that belong only to one role, and each one of them will actually, be a profile for the behavior of the users that have acquired that specific role. After the training phase, the system should be able to deal with new users coming to the system on its own. The algorithm should also be able to perform often updates, without deteriorating the performance of the system. This requires that the algorithm should be incremental.

Notice that the algorithm operates under the assumption that every user can only hold one role, and therefore there cannot be entries in the audit data that correspond to the same user, and are of the form $\mathbf{[X_1,X_2,\ldots X_n, R_1]}$, $\mathbf{[X_1,X_2,\ldots X_n, R_2]}$ with $\mathbf{R_1 ? R_2}$.

3.1 The Phases

The algorithm consists of three phases. The first one is the training phase. During this phase the system is trained and the clusters that correspond to the role profiles are built based on the training data. The second phase is the classification phase. In this phase the algorithm actually detects whether the users behave according to the role that has been assigned to them. Since new users and new requirements arise, there is a need for periodical update. The frequency of functionality updates will depend on the system's requirements.

3.1.1 Training Phase – Building the Clusters

The algorithm starts by creating a number of dummy clusters, say *d*. This number *d* is equal to the number of discrete roles that can possibly exist in the system. The centroid of each one of these clusters is the mean vector, containing the average values of the selected audit data attributes) of all the users that belong to the specific role (note that in this stage the role of each user is known). It is also supposed that the training data consists of roles of the normal behaving entities. Although this may seem restrictive and enough to cause biased results, this is not the case since a legitimate user of a role $R_{1,}$ is classified as a malicious user when he/she tries to acquire one of the other roles. This allows abnormal behaviors for each role also to be considered.

After the first step *d* discrete clusters exist. The next step is to deal with each one of the data records of the training data separately. For each training data record, the distance between this record and each one of the clusters is calculated. This distance calculation takes into consideration only the previously selected *n* attributes of the data records that come from the log data. The cluster C_{cur} that is closer to the current data record R_{cur} is therefore specified. The data record is not assigned immediately into its closest cluster. One more check, regarding the role of the current record and the role represented by the cluster, needs to be done. In other words if the role of the record R_{cur} is the same as the role of the cluster C_{cur} then the current data record is assigned to this cluster, the mean vector of the cluster is reevaluated and the procedure continues with the next record of the training set that becomes the current record $R_{cur.}$ On the other hand, if the role of the current record R_{cur} is different from the role whose profile is represented by the cluster C_{cur} then a new cluster is created. This cluster will contain the current record and it will be another representative cluster of the role the record R_{cur} holds. This step is repeated until all the data that is available in the training set is processed.

Notice that form the above description of the above algorithm it can be concluded that each cluster contains behaviors of users that belong to a single role. However, there can be multiple different clusters that capture divergent behaviors of users holding a single role.

3.1.2 Classification Phase

The problem this phase deals with is that, given the behavior of a specific user *U*, represented by an *n-dimension* data point, the algorithm should verify whether the user *U* behaves according to the role he/she claims to have. If this is not the case, then the system should trigger an alarm, or take other measures as per the policy decided. Notice, that if the behavior of the user is not within the ranges of acceptable behavior of his role, the system is also able to determine which one of the other roles the user is attempting to acquire.

One of the ways of verifying (or not verifying) that a user behaves according to the role he/she has, is to calculate the distance between the *n* attributes that represent the user's *U* behavior in the audit data record with the centroids of the existing clusters.

Then take the closest cluster to the user *U* and check whether the role whose profile is represented by this cluster is the same as the role of the user *U*. If this is the case, then the user's behavior is within acceptable limits for the role he holds. Otherwise, a warning should be raised.

4 Experimental Results

The experimental data were collected from a standard NT server. We have experimented with 2000 records that have been selected for training the data and other 1800 records of user transactions have been used for testing. Three attributes of the audit log record have been selected as the most representative ones and they correspond to the command the user uses, the protocol he uses to access the server and the location he uses for his accesses.

4.1 Experiment I

Problem Statement: The goal of the first experiment is to test the classification accuracy of the proposed algorithm. Given a training data set what is the accuracy of classification that can be achieved when the testing set contains new records, other than those included in the training set?
Hypothesis: Intuitively it is expected that when the testing and the training data are identical the classification accuracy of the algorithm would be 100%, since the system has a clear idea of how to classify each one of the testing records. The higher the percentage of new (previously unknown) records in the testing set the less the classification accuracy of the algorithm.

Another parameter that is expected to have an impact to the classification accuracy of the algorithm is the number of distinct roles of the system. The more the number of roles, the less accuracy expected. And this is because the borders between the clusters become less distinct as the number of roles increase. Large number of roles inevitably means more clusters, whose centroids are closer to each other, making classification process less accurate.

Both hypotheses are experimentally tested below:
Input Parameters: For the first experiment we have used the training data set consisting of 2000 records describe above. For the testing phase we have substituted a certain percentage (ranging from 0 to 90%) of the training data set with new records from the other set of 1800 records.
Regarding the roles the input parameters were the following:

- For the experiment with 6 roles we discriminate six roles from the audit data ("system administrators", "faculty members", "staff", "graduate students", "undergraduate students", "visitors"). These roles are discriminated from the audit data and verified by the system administrator himself.
- For the 4 role experiments we have put some of the above roles under the same new (hyper)-role. In this case the roles that have been discriminated are: ("system ad-

ministrators", "academic staff", "students", "visitors"). Here the "academic staff" (hyper)-role refers to both the faculty and the staff members of the university. The same way the (hyper)-role "students" includes both the graduate and the undergraduate students referred previously.

- Finally the two-roles that have been discriminated for the two role experiments are the "recognized users" and the "visitors". The correspondence to the above categories is rather straightforward.

Output – Data Observations: The output data of the first experiment have been gathered and presented in Figure 2. The y-axis of the graph shows the classification accuracy of the algorithm for the various percentage of new records inserted in the testing phase (x-axis) and for different number of roles (2, 4, 6). For example, it is found that the classification accuracy is 80% when there are 4 distinct roles in the system and 50% of new records inserted in the testing data set with respect to the training data set. This implies that the testing set consists of 1000 records that are the same as in the records used in the training phase; and additional 1000 records that are new to the system and not used for training. Classification accuracy of 80% means that 1600 out of the total 2000 records of the testing data set were classified correctly.

Conclusions: The experimental results that we gather verify that our classification algorithm is capable of achieving satisfactory classification rates. The main parameters that have an impact on the performance of the algorithm are the percentage of new records in the test data with respect to the records of the training data set, and the number of distinct roles. The less the number of roles and the less new data records the better the classification rate obtained.

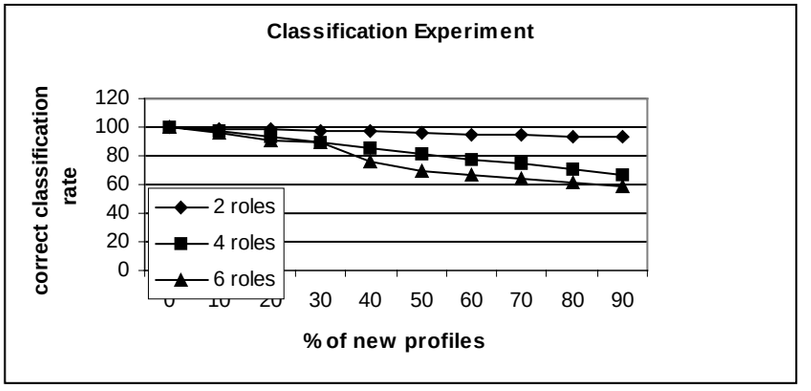

Fig. 2. Classification Experiment

4.2 Experiment II

Problem Statement: The second experiment's goal is to test the ability of the algorithm to point out misbehaviors of the users and to additionally specify the type of their misbehavior.

Hypothesis: Based on our intuition and on the results obtained from the previous experiment, we hypothesize that the two main parameters that have an impact on the performance of the algorithm in this experiment are the percentage of misbehaviors that appear in the testing data set, and the number of discrete roles that are discriminated in the system.

Input Parameters: For the second experiment we have used the same training data set constituting 2000 data records. For the testing phase we do not use any of the additional 1800 new data records. Instead we have modified the role attribute of a certain percentage of the records of the initial data set varying from 0 to 90%. Therefore, if an initial record of the training data set is like $[X_1,X_2,X_3,R_1]$ and this records is going to be changed for the testing phase, then its changed format is $[X_1,X_2,X_3,R_2]$. The algorithm is expected to point out that this is a misbehavior in the system of the form $R_1 \rightarrow R_2$. The rest of the input parameters remain the same as in experiment I.

Output – Data Observations: We compare the algorithm's output regarding the misbehavior of the users with the initial unmodified records and check the output of the algorithm in terms of how successfully the misbehaved users were classified. Figure 3 presents the output of the experiment II. The y-axis of the graph shows the misclassification accuracy, while the x-axis shows the percentage of the original data records that have been changed to be misbehaviors for their roles.

Conclusions: The experimental results have pointed out the ability of the algorithm to find misbehaviors in the system and to point out the type of this misbehavior in terms of the role that a user is illegally trying to acquire. The number of distinct roles and the percentage of misbehaviors play an important role in this context.

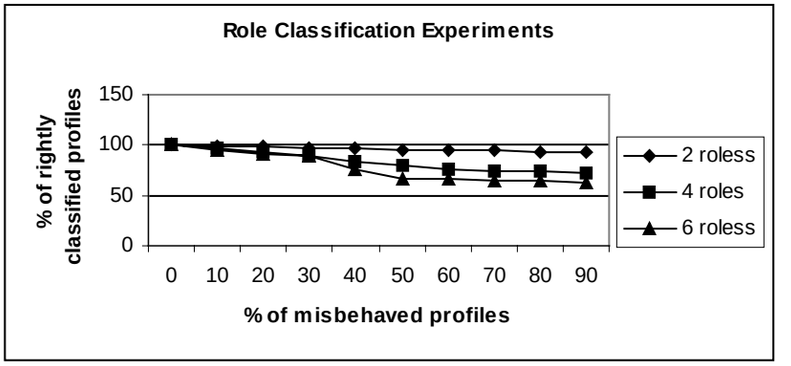

Fig. 3. Misclassification Experiment

5 Conclusions and Future Work

This paper presents an algorithm for building user role profiles. This algorithm will be used in the Role Server architecture [32] and can also be adopted within any RBAC system to examine whether a user behaves according to the role he/she holds. The algorithm has been presented in detail and evaluated by a small-scale experimental process.

Among our future research plans are extensive experimental evaluation of the algorithm and its incorporation as a component of the trustworthiness manager of the Role Server presented in Figure 1. We also plan to compare the performance of the algorithm with similar classification algorithms and cross validate experiments. Attempts will be made to check the impact of larger data sets, number of roles and attributes, on the accuracy of the algorithm. Experiments with larger data sets would give a better insight into other aspects of the algorithm, e.g., efficiency, accuracy and scalability etc.

Acknowledgements. We would like to acknowledge Dr. Mukesh Mohania, IBM India Research Laboratory, IIT Delhi Campus, Hauz Khas, New Delhi, India for his useful contribution to this research through numerous discussions and written exchanges.

References

1. Funnel web professional. http://www.activeconcepts.com
2. Hit list commerce. http://www.marketwave.com
3. Webtrends log analyzer. http://www.webtrends.com
4. Role Based Access Control website, http://csrc.nist.gov/rbac/, 2001.
5. R. Agrawal and R. Srikant: "Fast Algorithms for mining association rules". In *Proc. Of the 20th VLDB Conference*, pages 487-499, Santiago, Chile, 1994
6. G-J. Ahn, R. Sandhu: "Role-based Authorization Constraints Specification". *ACM Transactions on Information and System Security*, Vol. 3, No. 4, ACM, November 2000.
7. S. Axelsson: " Intrusion Detection Systems: A Survey and Taxonomy", Technical Report No 99-15, Dept. of Computer Engineering, Chalmers University of Technology, Sweden, March 2000
8. A. Buchner and M. Mulvenna: "Discovering internet marketing intelligence throughout online analytical web usage mining". SIGMOD Record, 27(4):54-61, 1998.
9. M.S. Chen, J.S. Park and P.S. Yu: "Data Mining for path traversal patterns in web environment". In 16th International Conference on Distributed Computing Systems, pages 385-392, 1996.
10. R. Cooley, B. Mobasher and J. Srivastave: "Data preparation for mining world wide web browsing patterns". Knowledge and Information Systems, 1(1), 1999.
11. B.S. Everitt: "Cluster Analysis", Halsted Press, Third Edition, 1993
12. D. Fasulo: "An Analysis of Recent Work on Clustering Algorithms", TR. Computer Sciences Dpt., Washington University, 1999.

13. A.D. Gordon: “Classification: Methods for Exploratory Analysis of Multivariate Data”. Chapman and Hall, 1981.
14. J. Han and M. Kambler: “Data Mining, Concepts and Techniques “, Morgan Kaufmann Publishers
15. A.K. Jain and R.C. Dubes: “Algorithms for Clustering Data”. Prentice Hall, 1988.
16. A. Joshi and R. Krishnapuram: “On Mining Web Access Logs”. ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2000
17. L. Kaufman and P.J. Rousseeuw: “Finding Groups in Data: An Introduction to Cluster Analysis”, John Wiley & Sons, Inc, 1990.
18. R. Krishnapuram, A. Joshi, L. Yi : “A Fuzzy Relative of the k-Medoids Algorithm with Application to Web Document and Snippet Clustering”.
19. R. Krishnapuram and J. M. Keller: “A possibilistic approach to clustering”. IEEE Transactions on Fuzzy Systems, 1(2):98-110, May 1993
20. S. Kumar and E. Spafford: “A pattern Matching Model for Misuse Intrusion Detection”. In Proc. Of the 17th National Computer Security Conference, pp. 11-21, 1994.
21. S. Kumar and E. Spafford: “A software Architecture to Support Misuse Intrusion Detection”. Dpt. Of Computer Sciences, Purdue University, CDS_TR_95-009, 1995.
22. Terran Lane and Carla E. Brodley: “Temporal sequence learning and data reduction for anomaly detection”, ACM Transactions on Information Systems Security 2(3) (Aug. 1999), Pages 295 - 331.
23. W. Lee and S.J. Stolfo: “A Framework for Constructing Features and Models for Intrusion Detection Systems”, ACM Transactions on Information and Security, Vol.3, No. 4, pp. 227- 261, November 2000,
24. T. Lunt: “Detecting Intruders in Computer Systems”. Conference on Auditing and Computer Technology, 1993
25. T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes and T. Garvey: “A real-time intrusion detection expert system (IDES) – final technical report. Technical Report, Computer Science Laboratory, SRI International, Menlo Park, California, 1992.
26. H. Mannila, H. Toivonen and A.I. Verkamo: “Discovering Frequent Episodes in Sequences”. In Proc. Of the 1st International Conference on Knowledge Discovery and Data Mining, 1997.
27. O. Nasraoui, R. Krishnapuram and A. Joshi: “ Mining web access logs using fuzzy relational clustering algorithm based on robust estimator”. In 18th International World Wide Web Conference, Toronto , Canada, 1999.
28. J. S. Park, R. S. Sandhu, S. Ghanta: “RBAC on the Web by Secure Cookies”. Proc. of the IFIP Workshop on Database Security, pp. 49-62, 1999.
29. J. S. Park, R. S. Sandhu: “Smart certificates: Extending x.509 for secure attribute services on the web”. Proc. of the 22nd NIST-NCSC National Information Systems Security Conference, Arlington, VA, October 1999.
30. C. Shahabi, A. Zarkesh, J. Adibi and V. Shah: “Knowledge Discovery from users web page navigation”. In Workshop on Research Issues in Data Engineering, Birmingham, England , 1997.
31. N. Ye and X. Li: “ A Scalable Clustering Technique for Intrusion Signature Recognition”, Proc. Of Workshop on Information Assurance and Security, NY – USA, June 2001
32. Y. Zhong, B. Bhargava, M. Mahoui: “Trustworthiness Based Authorization on WWW”. Proc. of the Workshop on Security in Distributed Data Warehousing. New Orleans, 2001