JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Mining Attribute Based Access Control Polices using Machine Learning

[1]Mrs Dipali Manish Patil, [2]Mrs Vrushali U Uttarwar , [3]Mrs Sapana Kolambe, [4]Mrs Nutan Chaudhari

[1]Assistant Prof, [2]Assistant Prof, [3]Assistant Prof, [4]SAP Consultant
[1,3]Dept of Information Technology, [2]Dept of Computer Engg
[1,2,3] DYPCOE,Akurdi, Pune, India, [4]Delivery Hero Pte. Ltd, Singapore

Abstract : *In machine learning, support vector machines is the technique with associated learning algorithms that analyze data & distinguish patterns, used for classification. As class labels are not given in existing system, proposed system uses Unsupervised learning method. Users allocate resources & User have attributes, SVM used to classify users depends upon attributes & policies. A new incremental, parallel and distributed SVM algorithm uses linear or non- linear hierarchy. The proposed system classifies very large datasets on standard personal computers. Support Vector Machine (SVM) algorithm used to classify users, resources based on attributes, it classifies users into access control & access grant permissions based on the access behaviour. The proposed system trains the rules of users .It first finds support set then it generate new rule according to support set and the behavioral access attributes.*

IndexTerms **- Machine learning, Unsupervised learning, Support Vector Machine, RBAC.**

I. Introduction

ABAC (attribute-based access control) is a flexible access control system that enhances security and information exchange [1]. ABAC also solves some of the issues that RBAC [2] has, such as role explosion [1], [3]. The merits of ABAC prompted the Federal Chief Information Officer Council to include it as a preferred access control model in the Federal Identity Credential and Access Management Roadmap and Implementation Guidance, ver. 2.0 [1], [4]. RBAC policy formulation by hand can be time-consuming and costly [5]. By partially automating the construction of RBAC policies, role mining algorithms have the potential to dramatically cut costs [5]. Role mining is an important research subject as well as a modest but fast increasing commercial industry sector (about $70 million) [5]. Manually developing ABAC policies can also be difficult [6] and costly [1]. The cost of developing ABAC policies might be reduced using ABAC policy mining methods.

Access control is a critical component of every information system, since it protects the underlying data against illegal access and changes [7]. One of the more recent approaches for describing access control rules, attribute based access control (ABAC), has demonstrated that it overcomes fundamental drawbacks in prior models [8]. In contrast to the discretionary access control (DAC) and mandatory access control (MAC) models, ABAC does not rely on user identities or rigorous rules to establish authorizations. Furthermore, by enabling the creation of flexible rules, it avoids issues such as role explosion in [8] role-based access management (RBAC). ABAC models, as the name implies, use user and resource attributes to decide whether an access request should be allowed or refused; in other words, attribute expressions are used to identify the sets of people and resources to which a policy applies[9,10,11]. For example, a policy such as "A manager can read any document in his/her department" may be directly translated to an ABAC policy: "if userType=manager, resourceType=document, userDepartment=resourceDepartment, action= read then PERMIT". As a result, ABAC is more versatile in setting access control policies than previous access control models, making it a potent access control model for boosting security

According to the proposed system project management sample policy, it detects unauthorized access using access control and access grant by asking access attributes. These access controlled access Grant labels are displayed in matrix format by SVM. The proposed algorithm constructs candidate rules using tuples made up of Users, Resources, and Operations. According to the proposed system, Generalize rule to cover additional tuple from user permission relation. SVM is used by the system to classify users into access control and access grant class labels. The system either trains the users' rules using the support set and behavioural access attributes, or it discovers new rules.

II. Literature Survey

Medvet et al. [12] recently developed an evolutionary, separate-and-conquer strategy for mining ABAC policies, based on the same policy language and case studies as [13]. During each cycle, a new rule is formed and the collection of access requests is reduced to a lower size. . Their work, like that of Xu and Stoller [13], is not capable of mining negative authorization rules, unlike our suggested technique. Furthermore, when compared to [13], there isn't much of a difference in terms of performance. As a result, we simply compare our results to [13].

The notion of ABAC policy mining was initially introduced by Xu and Stoller [13]. The goal of ABAC mining is to make the transition from a current access control paradigm to the ABAC framework as painless as possible by partially automating the process. Their policy mining method works like this at a high level. They begin by creating an Access Control List (ACL) from an ABAC policy and attribute data, which they refer to as the User Permission Relation. Then, while iterating through the tuples in the provided User Permission Relation, their policy mining method chooses a user permission tuples to serve as the seed for developing a candidate rule. By substituting conjuncts in attribute expressions with restrictions, this candidate rule is then generalized. The purpose of their generalization procedure is to broaden the rule's scope in terms of the extra tuples that the rule may cover in the User Permission Relation. The collection of candidate rules is then optimized by deleting redundant rules and combining pairs of rules to cover the whole ACL. If a rule covers cases in the User Permission Relationship that are also covered by another rule, it is redundant. By taking the union of conjuncts in those rules for each attribute, two separate rules with the same constraints are combined. Negative authorizations, on the other hand, are not handled by their algorithm. Furthermore, the ABAC policy mining technique described in [13] is very heuristic and difficult to understand.

Vaidya et al. described the Basic-RMP challenge as the task of determining the smallest collection of roles from a given UPA[14]. Edge-RMP [15, 16] is a version of Basic-RMP that tries to reduce |UA|+|PA| as well as the number of roles. The cardinality of a set S is denoted by the symbol |S|. Colantonio et al. have developed cost-based criteria for mining best set of roles [17]. Another metric introduced by Molloy et al. is Weighted Structural Complexity (WSC) [18], which is the weighted sum of the number of elements in R, UA, PA, and other components of an RBAC system. The restriction of RBAC policy mining is that, in order to get RBAC configuration from the supplied User Permission Assignments, role mining difficulties only evaluate positive authorizations, or what permissions are awarded to users based on their roles. When it comes to obtaining ABAC policy, our ABAC mining technique takes into account both positive and negative authorizations[19], which employed the Apriori algorithm [20] to find statistical patterns from access records of lab doors at a research lab, was an early effort on using association rule mining to ABAC rules. The dataset included 25 actual doors and 29 users who opened the doors with the use of a smartphone app and Bluetooth. By comparing mined rules with existing rules, the authors were able to discover policy misconfigurations.

The authors of [21] presented Rhapsody, a tool that employs the Apriori method as well. Rhapsody uses log mining to construct ABAC rules with little over-privilege. It does not, however, give a weighted system for balancing between under- and over-privilege, nor does it take into account huge and complicated privilege spaces.

III. Proposed System

Block Diagram of the proposed system is shown below in fig 3.1

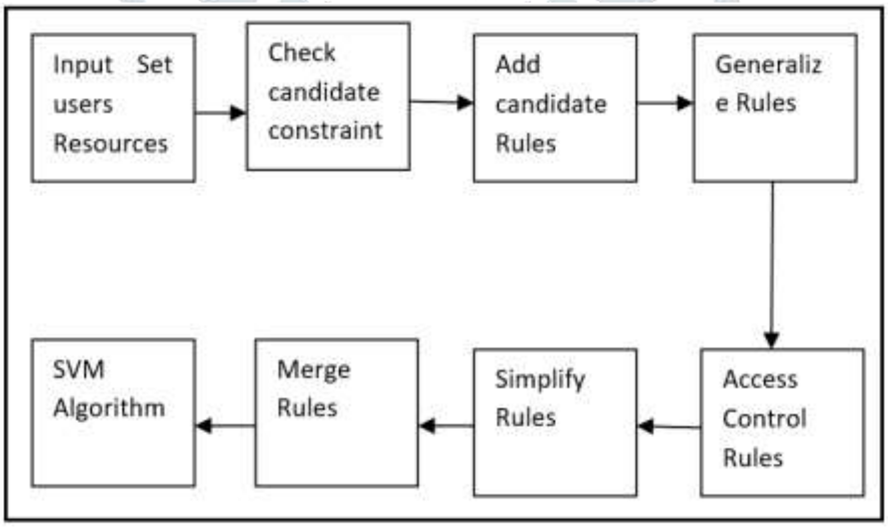

Figure 3.1 Block Diagram

Step by step working of the system is as follow.

Step1: Give Input set Users & Resources.

An access control list (ACL) policy is a tuple (U,R, Op,UP_0),
where
U = a set of users
R = a set of resources
Op=a set of operations.
UP_0=user-permission relation

$UP_0 \leq U \times R \times O_p$..(1)

Step 2: Check Candidate Constraint.

Ensures that Values in column satisfy certain conditions. The function candidate Constraint (r, u) returns a set containing all the atomic constraints that hold between resource r and user u.

Step 3: Add Candidate Rules

e_u=computeUAE(s_u,U)
e_r=computeRAE(s_r,R)
e_u=User attribute expression(UAE)
e_r=Resource attribute expression(RAE)
s_u=Set of Users
s_r=Set of Resources

Step 4: Generalize Rule:-Generalize rule ρ by adding some formulas from cc to its constraint and eliminating conjuncts for attributes used in those formulas. Generate policies containing rules whose meanings overlap.

Step 5: Access control the rules:-the proposed system can grant or deny access to objects using access control rules. Access control rules are written in terms of Allow or Deny decisions.

Step 6: Simplify Rules:-The function simplify Rules attempts to simplify all of the rules in Rules. It updates its argument Rules in place, replacing rules in Rules with simplified versions when simplification succeeds .It returns a Boolean indicating whether any rules were simplified.

Step 7: Merge rules. :-The function merges Rules (Rules) attempts to reduce the WSC weighted structural complexity of Rules by removing redundant rules and merging pairs of rules. Merge Rules (Rules) updates its argument Rules in place, and it returns a Boolean indicating whether any rules were merged.

Step 8: Support Vector Machine Algorithm:-In machine learning, support vector machines is method with associated learning algorithms that analyze data and recognize patterns, used for classification. Users allocate resources & User has attributes, SVM used to classify resources based on attributes & policies. A new incremental, parallel and distributed SVM. Algorithm using linear or nonlinear kernels proposed aims at classifying very large datasets on standard personal computers.

Algorithm :

```
Step1: Input: Training dataset represented by A and D matrices

f = access frequency of users
u =set of users
r =set of resources

Step2: Starting with $u_{0\ \varepsilon\ R}{}^{n+1}$ and i=0
Step 3: Repeat
        1. $u_{i+1} = u_i - \delta^2 f(u)^{-1} \Delta f(u_i)$\\ classifies users into groups
        2. i+i+1\\ increment to next step
Until $\Delta f(u_i) = 0$
Step4: Return ui;
```

IV. Results and Discussion

As per proposed system ,project management sample policy takes input from the process of registration, resource visiting ,resource uploading it takes parameters from these process as input to rule mining which records rules depending on visiting directory it maintains server log of resource server log site. The data in this process maintained at data repository .As per proposed system project management sample policy takes input from this process, it detects unauthorized access using access control & access grant by asking security question. SVM displays these access controlled access Grant labels in matrix format.

The time required to detect unauthorized access is minimum in project management sample policy using support vector machine.In existing system the time required to control the access is more as compared to proposed system.As number of directories increases the time requied to control the the access using attribute based access control using unsupervised learning .

Table 4.1 Results comparision

Num of Dir	Existing System time (ms)	Proposed System time (ms)
3	395 mili seconds	370 mili seconds
4	420 mili seconds	390 mili seconds
5	410 mili seconds	380 mili seconds
6	425 mili seconds	405 mili seconds
7	500 mili seconds	470 mili seconds

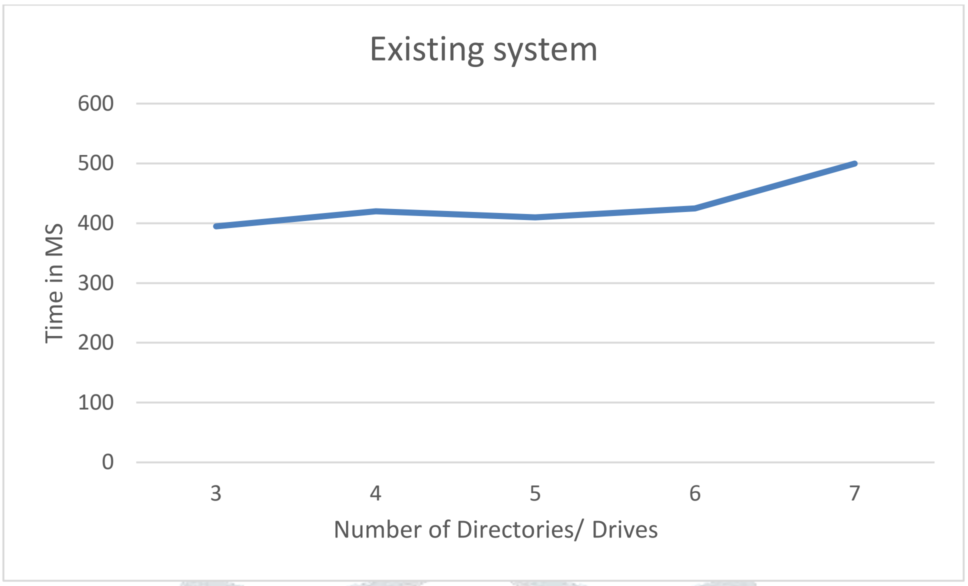

Figure 4.1 Results analysis of Existing system

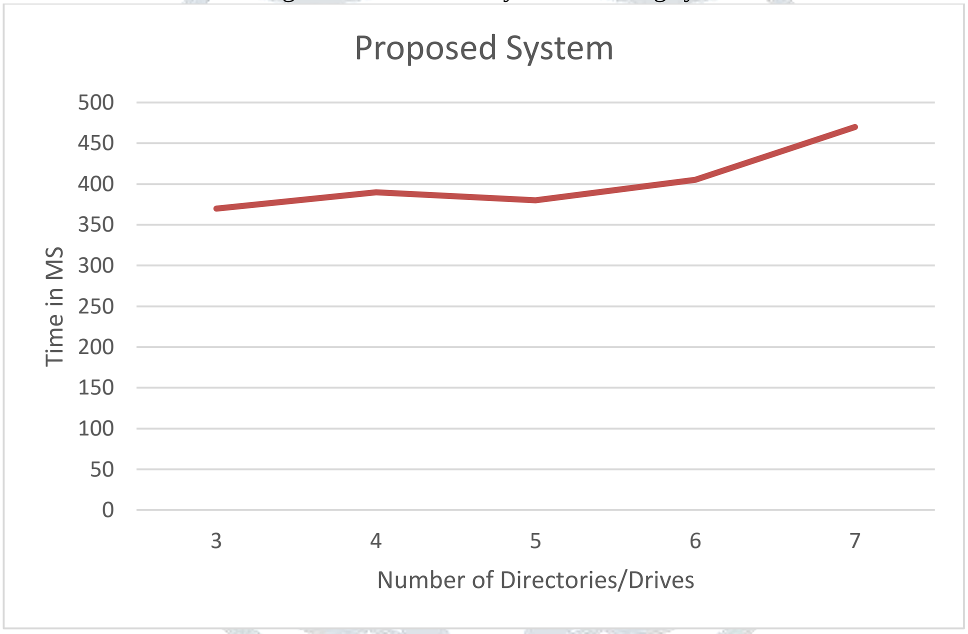

Figure 4.2 Results analysis of Proposed system

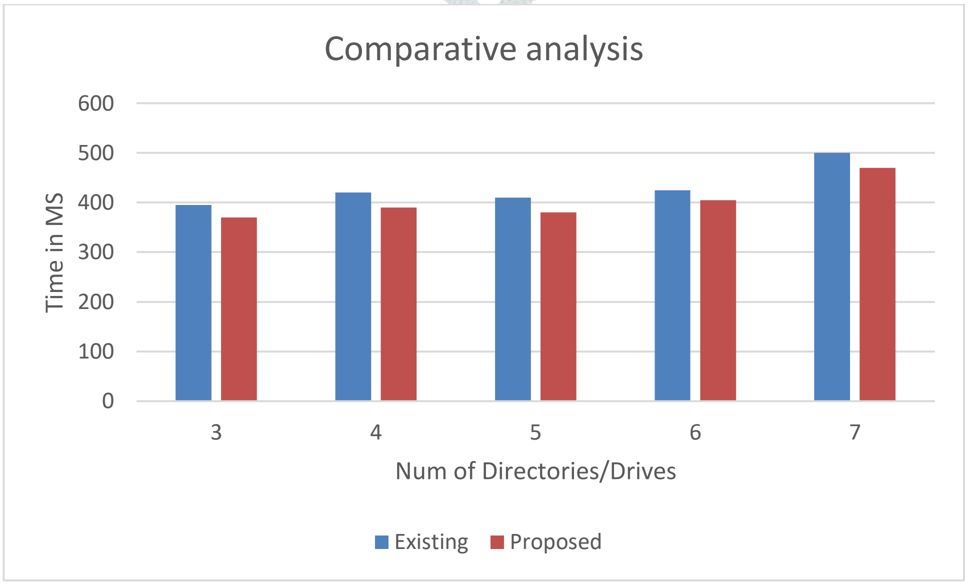

Figure 4.3 Comparative analysis

V. Conclusion and Future Scope

This System presents an ABAC policy mining algorithm. Experiments with sample policies and synthetic policies demonstrate the algorithms effectiveness. Machine learning based Support Vector Machine (SVM) algorithm used to classify users , resources based on attributes ,It uses Highest quality rules & policies for the classification of Users & Resources. Unsupervised learning method is used to classify users into access control & access grant according to access behaviour. In future Decision Tree Algorithm with SVM for better classification of the results can be done, also attribute based encryption/decryption can be added for more secure access control policies.

REFERENCES

1. V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (abac) definition and considerations (final draft)," National Institute of Standards and Technology, NIST Special Publication 800-162, Sep. 2013.

2. R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-based access control models," IEEE Computer, vol. 29, no. 2, pp. 38–47, Feb. 1996.

3. NextLabs, Inc., "Managing role explosion with attribute based access control," Jul. 2013, http://www.slideshare.net/nextlabs/ managing-role-explosion-with-attribute-based-access-control.

4. Federal Chief Information Officer Council, "Federal identity credential and access management (ficam) roadmap and implementation guidance, version 2.0," Dec. 2011, http://www.idmanagement.gov/documents/ ficam-roadmap-and-implementation-guidance.

5. S. Hachana, N. Cuppens-Boulahia, and F. Cuppens, "Role mining to assist authorization governance: How far have we gone?" International Journal of Secure Software Engineering, vol. 3, no. 4, pp. 45–64, October-December 2012.

6. M. Beckerle and L. A. Martucci, "Formal definitions for usable access control rule sets—From goals to metrics," in Proceedings of the Ninth Symposium on Usable Privacy and Security (SOUPS). ACM, 2013, pp. 2:1–2:11

7. E. Ferrari. Access Control in Data Management Systems. Synthesis Lectures on Data Management, 2(1):1–117, Jan. 1, 2010. issn: 2153-5418.

8. V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. Guide to attribute based access control (abac) definition and considerations (draft). NIST special publication, 800(162), 2013.

9. X. Jin, R. Krishnan, and R. Sandhu. A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC. In IFIP Annual Conference on Data and Applications Security and Privacy, LNCS, pages 41–55. Springer, Berlin, Heidelberg, July 11, 2012.

10. D. Servos and S. L. Osborn. HGABAC: Towards a Formal Model of Hierarchical Attribute-Based Access Control. In International Symposium on Foundations and Practice of Security, LNCS, pages 187–204. Springer, Cham, Nov. 3, 2014.

11. X. Zhang, Y. Li, and D. Nalla. An Attribute-based Access Matrix Model. In Proceedings of the 2005 ACM Symposium on Applied Computing, SAC '05, pages 359–363, New York, NY, USA. ACM, 2005

12. E. Medvet, A. Bartoli, B. Carminati, and E. Ferrari. Evolutionary inference of attribute-based access control policies. In International Conference on Evolutionary Multi-Criterion Optimization, pages 351–365. Springer, 2015.
13. Z. Xu and S. D. Stoller. Mining Attribute-Based Access Control Policies. IEEE Transactions on Dependable and Secure Computing, 12(5):533–545, Sept. 2015. issn: 1545-5971.
14. J. Vaidya, V. Atluri, and Q. Guo. The Role Mining Problem: A Formal Perspective. ACM Trans. Inf. Syst. Secur., 13(3):27:1– 27:31, July 2010. issn: 1094-9224.
15. H. Lu, J. Vaidya, and V. Atluri. Optimal Boolean Matrix Decomposition: Application to Role Engineering. In 2008 IEEE 24th International Conference on Data Engineering, pages 297– 306, Apr. 2008.
16. J. Vaidya, V. Atluri, Q. Guo, and H. Lu. Edge-RMP: Minimizing administrative assignments for role-based access control. Journal of Computer Security, 17(2):211–235, Jan. 1, 2009. issn: 0926-227X.
17. A. Colantonio, R. Di Pietro, and A. Ocello. A Cost-driven Approach to Role Engineering. In Proceedings of the 2008 ACMSymposium on Applied Computing, SAC '08, pages 2129– 2136, New York, NY, USA. ACM, 2008.

18. I. Molloy, N. Li, Y. A. Qi, J. Lobo, and L. Dickens. Mining Roles with Noisy Data. In Proceedings of the 15th ACM Symposium on Access Control Models and Technologies, SACMAT '10, pages 45–54, New York, NY, USA. ACM, 2010.
19. Lujo Bauer, Scott Garriss, and Michael K Reiter. 2011. Detecting and resolving policy misconfigurations in access-control systems. *ACM Transactions on Information and System Security (TISSEC)* 14, 1 (2011), 2.
20. Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proceedings of the International Conference on Very Large Data Bases*, *VLDB*, Vol. 1215. 487–499.
21. Carlos Cotrini Jimenez, Thilo Weghorn, and David A. Basin. 2018. Mining ABAC Rules from Sparse Logs. *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)* (2018), 31–46.