PAMMELA: Policy Administration Methodology using Machine Learning

Varun Gumma[1], Barsha Mitra[2], Soumyadeep Dey[3], Pratik Shashikantbhai Patel[2], Sourabh Suman[2], Saptarshi Das[4] and Jaideep Vaidya[5]

[1]*Department of Computer Science and Engineering, IIT Madras, Chennai, India*

[2]*Department of CSIS, BITS Pilani, Hyderabad Campus, Hyderabad, India*

[3]*Microsoft India Development Center, India*

[4]*JIS Institute of Advanced Studies and Research, JIS University, Kolkata, India*

[5]*MSIS Department, Rutgers University, New Brunswick, NJ, U.S.A.*

Keywords: ABAC, Policy Administration, Policy Augmentation, Policy Adaptation, Supervised Learning.

Abstract: In recent years, Attribute-Based Access Control (ABAC) has become quite popular and effective for enforcing access control in dynamic and collaborative environments. Implementation of ABAC requires the creation of a set of attribute-based rules which cumulatively form a policy. Designing an ABAC policy ab initio demands a substantial amount of effort from the system administrator. Moreover, organizational changes may necessitate the inclusion of new rules in an already deployed policy. In such a case, re-mining the entire ABAC policy requires a considerable amount of time and administrative effort. Instead, it is better to incrementally augment the policy. In this paper, we propose PAMMELA, a Policy Administration Methodology using Machine Learning to assist system administrators in creating new ABAC policies as well as augmenting existing policies. PAMMELA can generate a new policy for an organization by learning the rules of a policy currently enforced in a similar organization. For policy augmentation, new rules are inferred based on the knowledge gathered from the existing rules. A detailed experimental evaluation shows that the proposed approach is both efficient and effective.

1 INTRODUCTION

In any organization, it is of utmost importance to ensure that all accesses to resources take place in an authorized manner. This can be facilitated by deploying an access control model. Over the years, several access control models have been proposed among which the Role-Based Access Control (RBAC) (Sandhu et al., 1996) model became quite popular as an effective means of access control. In spite of its widespread popularity, RBAC suffers from a major drawback of being unsuitable for dynamic and collaborative environments.

The Attribute-Based Access Control (ABAC) model (Hu et al., 2014) was proposed to overcome the limitations of RBAC. This model allows users or subjects to access resources or objects based on the attributes of the subjects, objects and the environment. Each attribute is assigned a value or multiple values from a pre-defined set of values for every subject, object and environmental condition. In order to deploy ABAC, a set of rules is required. A rule defines the required attributes and the corresponding permissible values for each attribute for a specific type of access. The set of all rules collectively comprise a *policy*. While determining whether to allow or deny an access request, the attributes of the requesting subject, requested object as well as those of the environment in which the access request is made are taken into account. If each attribute has been assigned a specific value as per some rule, then the access request is allowed, otherwise, it is denied. The process of creating a policy for implementing ABAC is known as *policy engineering*. This can be carried out in two ways - *top-down* (Narouei et al., 2017) and *bottom-up* (Mocanu et al., 2015), (Xu and Stoller, 2015), (Gautam et al., 2017). Bottom-up policy engineering is also termed as *policy mining*. Combination of top-down and bottom-up approaches gives *hybrid policy engineering* (Das et al., 2018).

Gumma, V., Mitra, B., Dey, S., Patel, P., Suman, S., Das, S. and Vaidya, J.
PAMMELA: Policy Administration Methodology using Machine Learning.
DOI: 10.5220/0011272400003283
In *Proceedings of the 19th International Conference on Security and Cryptography (SECRYPT 2022)*, pages 147-157
ISBN: 978-989-758-590-6; ISSN: 2184-7711
Copyright © 2022 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

An organization intending to migrate to ABAC needs to design and implement an ABAC policy. Also, when an organization having an already deployed ABAC model, undergoes some changes, like opening of a new department or introduction of a new academic course, additional ABAC rules reflecting the changes need to be generated. Creating ABAC rules ab initio or completely re-mining an ABAC policy requires a considerable amount of administrative effort and time. These overheads can be substantially reduced if an existing policy is used as a reference for the creation of a new ABAC policy for migration. Also, instead of re-mining the complete policy in order to account for the organizational changes, it is more prudent to only create the additional rules. In both the scenarios, the existing ABAC policy can aid the process of creation of new ABAC rules, thereby effectively providing assistance in the task system administration.

In this paper, we propose a machine learning based methodology that will aid ABAC system administrators to efficiently augment an existing policy by including additional rules to accommodate various organizational alterations as well as to generate a new ABAC policy for an organization by referring to the existing policy of a similar organization. The key contributions of the paper are:

- We propose the *ABAC Policy Inference Problem* (ABAC-PIP) which takes as inputs an existing ABAC policy, a set of access requests and the attribute-value assignment information and creates a new set of ABAC rules that either augments the existing policy or constitutes a new policy. The rules thus created are considered as new since they are different from the existing ones in terms of certain attribute values.
- We propose a supervised learning based methodology for solving ABAC-PIP. We name our approach as ***P**olicy **A**dministration **M**ethodology using **M**achin**e** **L**e**a**rning* (PAMMELA). PAMMELA trains a machine learning classifier using an existing ABAC policy. The training includes both positive (rules that grant accesses) and negative (rules that deny accesses) rules. After training, PAMMELA creates a new set of access rules on being supplied with a set of access requests and attribute-value assignment information.
- We test our proposed policy inferring methodology on three manually crafted datasets that are created keeping in mind real-world scenarios. Our experiments show that PAMMELA shows promising results and provides a high degree of performance. We have experimented with a number of machine learning classifiers and show comparative results for all them.

It is to be noted here that the existing ABAC policy serving as a reference for the creation of the new rules is assumed to be correct and free from any form of inconsistency or noise. The reason is that the existing policy is assumed to have been deployed over a certain period of time and is considered to be a tried and tested means of access control enforcement.

The rest of the paper is organized as follows. Section 2 explores the different policy mining approaches present in the existing literature. In Section 3, we review the preliminary concepts related to ABAC. We formulate ABAC-PIP in Section 4 and present the corresponding solution strategy PAMMELA along with a discussion of the application scenarios in Section 5. Dataset description, evaluation metrics and experimental results are presented in Section 6. Finally, Section 7 concludes the paper with some insights into future research directions.

2 RELATED WORK

A considerable number of work has focused on developing techniques for creation of ABAC policies. Xu and Stoller proposed a strategy for mining ABAC policies from access logs (Xu and Stoller, 2014). Another work by Xu and Stoller has formulated the ABAC policy mining problem as an optimization problem and has presented *weighted structural complexity* as a policy quality metric (Xu and Stoller, 2015). Talukdar et al. (Talukdar et al., 2017) have proposed ABAC-SRM, a bottom-up policy mining method capable of creating generalized ABAC rules. Cotrini et al., in (Cotrini et al., 2018), have proposed Rhapsody, a policy mining technique that can handle sparse inputs and have defined a rule quality metric called *reliability*.

A scoring method for determining the quality of a policy and an attribute-based rule mining algorithm from the audit logs of an organization have been presented in (Sanders and Yue, 2019). This approach can minimize the under and over privileges for enforcing the principle of least privilege. The authors in (Alohaly et al., 2019) designed a methodology that can extract ABAC constraints in an automated manner from policies expressed in natural language. A constrained policy mining technique has been proposed in (Gautam et al., 2017). Lawal and Krishnan have proposed an approach for policy administration in ABAC via policy review (Lawal and Krishnan, 2021). Heutelbeck et al. (Heutelbeck et al., 2021) have designed a data structure for efficiently indexing policy documents and have proposed a method for finding the relevant policy documents for a particular access request. Some recent works have combined ABAC with blockchain (Kumar et al., 2021), (Rouhani et al., 2021) and have explored the application of ABAC in smart healthcare (Zhong et al., 2021).

Several incremental and adaptive policy generation techniques are present in the current literature. Das et al. (Das et al., 2017) have proposed a policy adaptation strategy between similar organizations. They have formulated the NP-complete Policy Adaptation Problem (PolAP) which aims at determining the value assignments of the attributes of each subject for a given ABAC policy and have proposed a heuristic algorithm for solving it. It can be noted that our proposed ABAC-PIP is different from PolAP both in terms of the inputs and the output. Das et al. have further extended their work in (Das et al., 2019) by considering environmental attributes and hierarchical relationships among subject attribute values. Batra et al. (Batra et al., 2021b) have proposed an incremental policy mining technique that is capable of creating new ABAC rules when a new permission is added or deleted or a new attribute value is added or deleted. In (Batra et al., 2021a), the authors have presented a strategy for determining policy similarity, have proposed two methods for performing policy reconciliation and also presented a policy migration technique.

The rapid growth of artificial intelligence has spurred the application of machine learning in the field of access control. In (Mocanu et al., 2015), the authors have presented a policy generation technique using Restricted Boltzmann Machines. Karimi et al. in (Karimi et al., 2021b) have proposed an automated policy learning method from access logs using unsupervised learning by considering both positive and negative rules. (Jabal et al., 2020a) presents *Polisma*, a framework for learning ABAC policies from access logs by using a combination of statistical, data mining and machine learning algorithms. In (Karimi et al., 2021a), the authors have designed an adaptive policy learning method for home Internet of Things (IoT) environment using reinforcement learning. Jabal et al. have proposed an approach known as FLAP (Jabal et al., 2020b) for collaborative environments. FLAP enables one organization to learn policies from another organization and perform policy adaptation via a policy learning framework by using local log or local policies or local learning or hybrid learning.

In this work, we formulate a policy creation problem variant and name it as the ABAC Policy Inference Problem (ABAC-PIP). We employ machine learning algorithms for solving ABAC-PIP in order to generate ABAC rules. To the best of our knowledge, this kind of problem formulation and the design of end-to-end machine learning based solution strategy for aiding system administrators have not been addressed in the existing literature.

3 ABAC PRELIMINARIES

The ABAC model consists of the following components:

- A set S of subjects. Each subject can be a human or a non-human entity.
- A set O of objects. Each object corresponds to a system resource that should be accessed in an authorized manner.
- A set E of environmental factors or conditions. Each condition can represent some temporal or spatial or some other kind of context in which a subject requests access to an object. For eg. location, time, etc.
- A set SA of subject attributes. Each subject attribute represents a property associated with a subject and can assume a single or multiple values from a set of values. These values are known as subject attribute values. If a subject attribute assumes a single value, it is known as *atomic valued*. If it is assigned multiple values for a subject simultaneously, it is known as *multi-valued*. An example of a subject attribute is $Department$. The attribute value set of $Department$ can include *Computer Science*, *Electronics*, and *Mechanical*.
- A set OA of object attributes. All concepts mentioned for subject attributes are applicable for object attributes as well. An example of an object attribute is *Type of Document* and its possible values can be *Project Plan*, *Budget*, *Expenditure Details*.
- A set EA of environmental attributes. All concepts discussed for subject attributes also apply for environmental attributes. Example of an environmental attribute is *Time of Doctor's Visit* having possible values as *Day Shift* and *Night Shift*.
- A function F_{sub} that assigns values to subject attributes for a subject. Formally, F_{sub}: S x SA $\rightarrow \{v_s \mid v_s$ is a subject attribute value$\}$. For eg., $F_{sub}(John, Department) = \{Computer\ Science\}$.
- A function F_{obj} that assigns values to object attributes for an object. Formally, F_{obj}: O x OA $\rightarrow \{v_o \mid v_o$ is an object attribute value$\}$. For eg., $F_{obj}(File1.doc,\ Type\ of\ Document) = \{Project\ Plan\}$.
- A function F_{env} that assigns values to environmental attributes. Formally, F_{env}: E x EA $\rightarrow \{v_e \mid v_e$ is an environmental attribute value$\}$.

- A set P of permissions (operations). Common permission types can be *read*, *write*, *update*, *execute*, etc.
- A set $\mathcal{R}$ of rules. Each rule specifies whether a particular type of access is to be granted or denied. A rule that permits an access is a positive rule and a rule that disallows an access is a negative rule. All the rules cumulatively constitute an ABAC policy. An example ABAC rule can be of the following form - $<\{Department = Accounts, Designation = Accountant\}, \{Type = Payroll\ Data, Department = Any\}, view >$. In natural language, this rule translates to - *If a subject belongs to the Accounts department and has a Designation of Accountant, then she can view the payroll data of any employee belonging to any department.*

4 ABAC POLICY INFERENCE PROBLEM

Designing ABAC policies is not a trivial task and requires a considerable amount of administrative effort. The overhead associated with the process of policy generation can be reduced if some existing policy serves as a reference point or guideline based on which new policies can be inferred. This observation is applicable for augmenting an existing policy database by incrementally adding new rules as well as creating a new policy for an organization that has a similar structure as that of another organization where ABAC is already deployed. In order to achieve this goal of aiding the policy administration process, we propose the *ABAC Policy Inference Problem* (ABAC-PIP).

ABAC-PIP takes a deployed ABAC policy, a set of subjects, a set of objects, a set of subject attributes and their corresponding values, a set of object attributes and their corresponding values and a set of access requests as inputs and produces as output, for each access request, the set of permissions or operations required to carry out the designated task in case the access request is to be granted or the decision to deny the request if it is an unauthorized one. The formal problem definition is presented below.

ABAC-PIP: *Given an ABAC policy $\mathcal{P}$, a set S of subjects, a set O of objects, a set SA of subject attributes, a set OA of object attributes, a function F_{sub} defining the value assignment of the subject attributes of each subject, a function F_{obj} defining the value assignment of object attributes of each object, and a set $\mathcal{L}$ of access requests as inputs, determine, corresponding to each access request, the set of permissions* Pr*, associated with the access request if the access request is to be permitted or else the decision to deny the access request if it is not to be permitted.*

The following underlying assumptions have been made for ABAC-PIP.

- The existing ABAC policy $\mathcal{P}$ is correct and does not result either in any form of security breach or over-restrictive access decisions. The verification of the correctness of $\mathcal{P}$ is under not the scope of the current work.
- The access requests present in the set $\mathcal{L}$ are new access requests and the corresponding access decisions cannot be determined by the rules present in the already deployed ABAC policy $\mathcal{P}$.
- The access requests contained in $\mathcal{L}$ are derived from access logs of an organization and include both positive and negative access requests. Positive access requests are the ones which are to be granted and negative access requests are those which are to be denied.
- The new access requests are generated when an organization having a deployed ABAC model undergoes some changes which necessitate the creation of additional rules corresponding to the new access requests that will take place. An example of such a change can be the opening of a new department or the introduction of a new job designation.
- The new access requests can also be generated when an organization wishes to migrate to ABAC. This organization is similar in structure and workflow to another organization where ABAC has already been deployed.
- For the new access requests, the relevant information regarding the value assignment for the subject and object attributes are available.

In the following section, we discuss the applicability of ABAC-PIP to two policy administration scenarios and present the solution strategy that assists the corresponding policy administration tasks.

5 PROPOSED METHODOLOGY

We propose a machine learning based methodology for solving ABAC-PIP. Our approach is designed to help system administrators in creating ABAC policies and in the process, can reduce the overhead associated with policy creation. We name our proposed strategy as ***P**olicy **A**dministration **M**ethodology using **M**achin**e** **Le**a**rning* (PAMMELA). In this section, we first present a detailed overview of how PAMMELA solves ABAC-PIP and then elaborate on the scenarios where PAMMELA is applicable.

5.1 PAMMELA

PAMMELA is a supervised learning based methodology for solving ABAC-PIP. Our proposed strategy works in two phases. In the first phase, a machine learning classifier is trained using a set of labeled data. This labeled data is in the form of an ABAC policy consisting of several rules. Each rule defines the combination of the subject attribute-value pairs and the object attribute-value pairs for which a subject will be allowed to access and perform certain operations or acquire some permissions for an object. We refer to such rules as *positive rules*. The attributes are treated as features by the classifier. The positive rules help the classifier to learn under which conditions, an access request is to be granted and what are the corresponding permissions associated with the access.

In addition to this, it is also essential for the classifier to learn when an access request is to be denied. Such scenarios are covered by the *negative rules*. A negative rule specifies the attribute-value pairings for which an access request is not permissible. For negative rules, all those attribute-value pairs are considered which lead to unauthorized accesses, not just the ones which explicitly deny accesses. Such negative rules can be derived from the set of positive rules. If $\mathcal{U}$ denotes the set of all possible attribute-value pairings (both subject and object) and $\mathcal{PR}$ denotes the set of positive rules, then the set of negative rules $\mathcal{NR}$ = $\{\mathcal{U}\} \setminus \{\mathcal{PR}\}$. Though it is straightforward to derive the elements of $\mathcal{NR}$, the task, however, is not trivial. The reason behind this is that, generally in any organization, the set of rules disallowing accesses is much larger in size than the set of rules allowing accesses. In PAMMELA, the machine learning classifier is also trained using the negative rules. In this case, the classifier learns to output denial as a decision. If the classifier is trained using only the positive rules, then when a negative access request is given as input to the classifier, it may not be able to output the correct decision. The reason for this is during the training phase, the classifier, in such a case, never learns to output a deny decision for the specific attribute-value combinations.

For supplying the input training data to the machine learning algorithm, we make use of categorical data. For each attribute value set, the categorical encoding starts increasing monotonically from 1 such that each subject or object attribute value is assigned a numerical value. For eg., If we have a subject attribute *Designation* having values *Assistant Professor*, *Associate Professor* and *Professor*, then *Assistant Professor* can be assigned the value 1, *Associate Professor* can be assigned the value 2 and *Professor* can be assigned the value 3. If a particular attribute is not applicable for a certain subject or object, then a special value *NA* (for Not Applicable) is assigned as the attribute value for that subject or object. The reason for using categorical encoding is a reduced input vector size in comparison to the vector size that is obtained using one-hot encoding.

After training, PAMMELA generates new rules in the second phase which is the testing phase for the machine learning classifier. Here, a set of access requests is given as input to the classifier. The combinations of the attribute values present in these access requests are different from those present in the existing policy rules. Such new combinations of values can occur when either the subject attribute value set(s) or the object attribute value set(s) or both are augmented with additional values.

Both positive and negative access requests are given as input to PAMMELA in the second phase. Positive access requests are those which are to be granted and negative access requests are those which are to be denied. The reason for including both positive and negative access requests is that when the already deployed policy cannot account for the organizational changes, then any type of access request needs to be appropriately classified. We assume that for each of the new access requests, the relevant attribute-value pair assignment information are available. Based on the learning achieved in the first phase, the new access requests and the attribute-value information, PAMMELA classifies each access request by determining whether it is to be allowed or disallowed. If the access request is to be disallowed, PAMMELA outputs the decision *NO*. On the other hand, if the access request is to be granted, PAMMELA outputs the set of permissions required to successfully execute the access request in addition to the decision of *YES*. Once the relevant decisions have been derived, the newly generated rules are presented to the system administrator for inclusion into the policy database. The workflow of PAMMELA is depicted in Figure 1.

5.2 Policy Administration via Augmentative Policy Inference

Undergoing structural and functional changes is not uncommon for any organization. Examples of such changes include but are not limited to the following - (i) an organization can start a new department, (ii) an organization can introduce a new job designation, (iii)

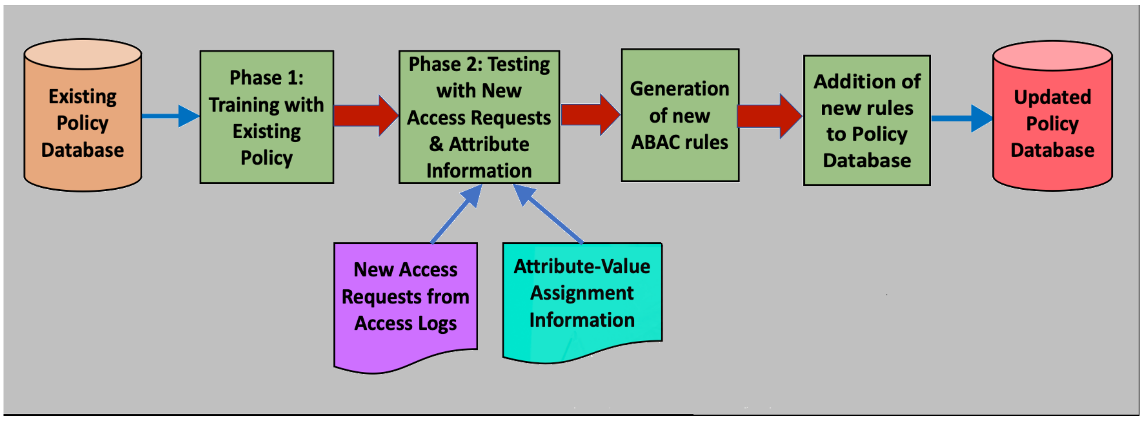
Existing Policy Database
Phase 1: Training with Existing Policy
Phase 2: Testing with New Access Requests & Attribute Information
Generation of new ABAC rules
Addition of new rules to Policy Database
Updated Policy Database
New Access Requests from Access Logs
Attribute-Value Assignment Information

Figure 1: Block Diagram depicting the workflow of PAMMELA.

an educational institute can introduce a new degree program, (iv) an educational institute can introduce a new undergraduate or graduate level course, (v) a hospital can start treating patients in a new specialization, etc. These types of changes increase the breadth of the workflow of the organization and may not be too frequent. However, whenever they do occur, there arises a need to modify or update the existing ABAC policy in order to prevent any form of unauthorized access or leakage of rights.

Updating the policy by re-executing the policy mining process involves a lot of overhead. Moreover, it increases the overall time required to manage the update process as well as the corresponding system downtime. Hence, it is prudent to design only the new rules and add them to the policy database. The types of changes considered here are not radical enough to completely alter the organizational structure. Therefore, the new rules that need to be designed will have some similarity with the existing rules and will differ from the current ones in terms of one or more attribute values. Also, the permission set associated with a new rule will be same as that of an equivalent existing rule.

Manually augmenting the policy database is a very challenging task even for a moderate sized organization. Our proposed methodology PAMMELA can perform automated augmentation of a deployed ABAC policy. We name this application of PAMMELA as *Augmentative Policy Inference*.

PAMMELA can determine the relevant decisions for both positive and negative accesses, thereby augmenting the policy database by inferring new ABAC rules. Other than providing the categorical encoding for the new attribute values, our proposed approach significantly reduces the degree human intervention in the policy augmentation process. We assume that the subject attribute and the object attribute value sets are augmented with new values corresponding to the various organizational changes.

We elaborate the observations made above using the following examples. Suppose an *XYZ* university has two departments - Computer Science and Electronics. The university has deployed the ABAC model for access control. We consider one subject attribute *User-Type* having values *Faculty*, *Student* and *Teaching-Assistant* and one object attribute *Resource-Type* having values *Question-Paper*, *Answer-Script*, *Assignment* and *Mark-Sheet*. The university management decides to open a new department, Information Technology. The rules governing the access of the different resources by the users belonging Computer Science and Electronics, will no doubt be applicable for Information Technology. The new rules will have the *Department* attribute value as *Information Technology*. Another example can be that, if the Computer Science department wishes to introduce a new evaluation component, *Quiz*, then the rules for *Quiz* will be similar to those associated with the access of the existing component *Assignment*.

5.3 Policy Administration via Adaptive Policy Inference

When an organization intends to migrate to ABAC, a policy needs to be designed ab initio using a policy generation approach. The input to the policy creation process can be an access log consisting of a set of access requests. A considerable amount of administrative overhead associated with this task can be reduced if some already deployed ABAC policy is available which can serve as a guideline. Henceforth, we shall refer to this existing policy as *reference policy* and the new policy that is to be designed as *target policy*. Note that the structure of the organization where the reference policy is deployed is assumed to be similar to the structure of the organization that wishes to implement ABAC. In such a scenario, the rules present in the reference policy can be adapted to generate the target policy. The adaptation process will handle the dissimilarities present between the rules of the reference policy and the target policy in terms of the subject and object attribute values.

We explain the adaptive policy inference task using an example. We take the example of XYZ university mentioned in Sub-section 5.2 having two departments, Computer Science and Electronics. The subject as well as object attributes and their corresponding values are the same as those mentioned in the previous sub-section. Suppose, another educational institute, *PQR* wishes to adopt the ABAC model. *PQR* has two departments, Mechanical Engineering and Civil Engineering. *PQR* has the same subject and object attributes as those of XYZ. The subject attribute values of *PQR* are also same as those of XYZ. For the object attribute values, *PQR* has an additional value *Presentation* apart from those present for XYZ. The evaluation component *Presentation* requires the same type of accesses as that of *Assignment*. Thus the rules that need to be implemented for *PQR* have the same structure as those of XYZ and hence, can be inferred by taking the rules of XYZ as reference. In this way, the reference policy of XYZ can be appropriately adapted to create the target policy for *PQR*.

PAMMELA can effectively handle policy adaptation. The reference policy will serve as an input to the machine learning classifier in the training phase. In order to generate the target policy, the positive and negative access requests present in the access logs of the organization that wishes to implement ABAC as well as the attribute-value assignment information will be supplied to the classifier in the testing phase. Based on the learning accomplished during training, the classifier will output either a grant decision along with the set of associated permissions or a deny decision. This output combined with the attribute-value assignment information will create the rules for the target policy. One assumption in this context is that the set of permissions associated with each rule of the target policy is the same as that of a corresponding rule of the reference policy.

Several recent works like (Das et al., 2017), (Das et al., 2019), (Jabal et al., 2020a) and (Jabal et al., 2020b) have addressed the problem of policy adaptation. The approaches proposed in (Das et al., 2017) and (Das et al., 2019) do not make use of machine learning. The techniques in (Jabal et al., 2020a) and (Jabal et al., 2020b) use machine learning along with a number of heuristic methods. To the best of our knowledge, our proposed methodology PAMMELA is the first end-to-end machine learning based strategy for policy adaptation.

5.4 Discussion

PAMMELA offers the advantage of significant reduction of administrative effort for augmentative and adaptive policy inferences by taking the ABAC rules currently present in the existing policy database as input. Needless to say that if these tasks are carried out manually, it requires a huge amount of effort to painstakingly scan through the policy database to determine which existing rules are most similar to the rules that are to be created and generate the new rules. This makes the entire process very time consuming and quite prone to errors. The tasks can be accomplished using heuristic methods but at the cost of designing several functions covering different scenarios and running several experiments to determine the best possible strategy. If each strategy executes for a considerable amount of time, then selection of the most suitable approach will involve a lot of time overhead even before the actual augmentation or adaptation process. PAMMELA eliminates all the above mentioned issues by providing an end-to-end intelligent solution for aiding system administrators. It is to be noted that the rules present in the existing policy are considered to be correct and consistent. Thus, the scenario of any error present in the existing policy migrating to the newly derived rules does not occur and hence has not been considered in the current work.

6 EXPERIMENTAL EVALUATION

In this section, we discuss the datasets as well as the metrics used for evaluating the performance of PAMMELA and then report the experimental findings.

6.1 Dataset Description

For the purpose of experimentation, we have manually created three datasets. Each dataset consists of two parts - (i) an ABAC policy consisting of a number of rules, and (ii) a set of access requests along with attribute-value assignment information. The ABAC policy is used in the training phase of PAMMELA and corresponds to the existing or the reference policy. The set of access requests and the attribute-value assignment information are used in the testing phase.

We have not used any synthetic dataset generator or simulator for dataset creation. The datasets have been created keeping real-world scenarios in mind so that they are similar to actual physical organizations. We have generated the set of subjects, objects, atomic valued subject and object attributes and their corresponding values in such a way that they mimic real-

Table 1: Attribute-value Count for Training Data.

Attribute-Type	**Attribute-Name**	**Number of Values in**		
		University Dataset 1	**University Dataset 2**	**Company Dataset**
Subject	Designation	3	5	12
	Post	X	6	X
	Department	4	5	5
	Course	X	120	X
	Degree	2	2	X
	Year	4	4	X
	Project-Name	X	X	6
Object	Resource-Type	7	8	8
	Department	4	5	4
	Course	X	120	X
	Degree	2	2	X
	Year	4	4	X
	Project-Name	X	X	6

life organizations. The rules as well as the access requests of each dataset are similar to actual organizational accesses and hence can be considered to be semantically meaningful. For each dataset, we have considered only one permission corresponding to the ability of a subject to access a given object or the denial of access. However, PAMMELA is capable of handling multi-permission scenarios as well. The datasets are:

University Dataset 1: We have designed this dataset to mimic the workflow of an educational institute.

University Dataset 2: Another dataset has been designed keeping in mind the structure of an educational institute and has been named as University Dataset 2. This dataset is much more detailed than University Dataset 1 both in terms of the attributes, their corresponding values as well as the number of rules present in the training data.

Company Dataset: This dataset has been created keeping in mind the structure and workflow of a software company and incorporates features and access rules similar to a real-life organization.

Table 1 shows the number of values associated with each attribute in the training data for a specific dataset. A 'X' in a table cell indicates that the corresponding attribute is not present for the dataset indicated by the column. Table 2 shows the number of rules present in the training data and the total number of as well as the number of positive and negative access requests present in the test data for each dataset. If a certain attribute is present in a dataset but is not applicable for a particular subject or object, then the value *Not Applicable* is assigned for that attribute.

Table 3 presents the number of new values introduced for the different attributes in the test data for the three datasets in the following format - attribute name: number of new attribute values. In the test data of each dataset, one new attribute value has been introduced in each access request.

6.2 Evaluation Metrics

The performance of PAMMELA is evaluated using the metrics of Accuracy, Precision, Recall and F1-score, as discussed below.

YES and NO respectively denote the grant and deny decisions given as outputs by PAMMELA. The terminologies and metrics are:

- **True Positive Accesses (TPA):** These are the positive access requests corresponding to which PAMMELA gives the output YES and are instances of correct classifications.
- **True Negative Accesses (TNA):** These are the negative access requests corresponding to which PAMMELA gives the output NO and are instances of correct classifications.
- **False Positive Accesses (FPA):** These are the negative access requests corresponding to which PAMMELA gives the output YES, i.e., misclassifications resulting in security breach.
- **False Negative Accesses (FNA):** These are the positive access requests corresponding to which PAMMELA gives the output NO, i.e., misclassifications resulting in an over-restrictive system.
- **Accuracy:** It is the ratio of the correctly classified access requests to the total number of access requests. It denotes the capability of the classifier to make correct decisions. Mathematically, $Accuracy = \frac{TPA+TNA}{TPA+FPA+TNA+FNA}$
- **Precision:** It is the ratio of the correctly classified positive access requests to the total number of access requests for which the output is YES. Precision is inversely proportional to the degree of security breach occurring in the system. Mathematically, $Precision = \frac{TPA}{TPA+FPA}$
- **Recall:** It is the ratio of the correctly classified positive access requests to the total number of positive

Table 2: Training Data Rule Count and Test Data Access Request Count.

Dataset	Rule Count in Training Data	Access Request Count in Test Data		
		Total	Positive	Negative
University Dataset 1	53	1010	598	412
University Dataset 2	156808	483	308	175
Company Dataset	1616	287	95	192

Table 3: New Attribute Value Count for Test Data.

Attribute-Type	Attribute-Name : New Value Count		
	University Dataset 1	*University Dataset 2*	*Company Dataset*
Subject	Designation : 2, Department : 5, Degree : 2	Designation : 1, Department : 1, Post : 1, Course : 21	Designation: 4, Project-Name: 1
Object	Resource-Type: 3, Department : 4, Degree : 2	Resource-Type : 4, Department : 1, Course : 21	Resource-Type : 7, Project-Name : 1

access requests. Recall is inversely proportional to the degree of over-restrictiveness of the system. Mathematically, $Recall = \frac{TPA}{TPA+FNA}$

- **F1-score:** It is calculated as the weighted average of precision and recall. F1-score balances the relative trade-off between precision and recall. It is calculated as $F1-score = \frac{2*Precision*Recall}{Precision+Recall}$

6.3 Results

In this section, we present the experimental results. We have used the following machine learning classifiers to evaluate the performance of PAMMELA - Artificial Neural Network (ANN), Decision Tree (DT), Random Forest (RF), Extra Trees (ET), Gradient-Boosting (GB) and XGBoost (XGB). The implementation was done using scikit-learn library and the experiments were conducted on a MacBook Pro laptop having 2.3 GHz, 8 cores, intel core i9 processor, 16 GB RAM and macOS 11.4 as the operating system. We report the performance of PAMMELA in terms of Accuracy (Acc), Precision (Pre), Recall (Rec) and F1-score (F1-s) in Table 4 for the three datasets.

The results show that PAMMELA provides the highest accuracy of 88.4% for University Dataset 1, 89% for University Dataset 2 and 88.2% for Company Dataset across all classifiers. The highest precision recorded for the University Datasets is more than 97% and is 86.7% for the Company Dataset. This implies that the number of misclassifications in terms of the false positive accesses is quite low. The recall achieved by our proposed approach is less than 90% for the University Datasets and less than 80% for the Company Dataset. This shows that PAMMELA can have a tendency to make over-restrictive access decisions which though not desirable but will not lead to security breaches. It should be noted here that our proposed methodology is designed to help system administrators in policy augmentation and policy adaptation. Instead of performing these administrative tasks either manually or heuristically, the system administrator can obtain the output from PAMMELA and manually inspect only a fraction of all the access requests that are misclassified. The accuracy, precision and recall values of Table 4 indicate that the misclassified access requests constitute only a small percentage of the entire access request set. Thus, it is evident that PAMMELA will accurately create majority of the rules and the administrator will need to manually create only a few rules, reducing the overall administrative effort to a great extent.

We next give examples of a few sample augmented rules generated by PAMMELA corresponding to each dataset. An example positive rule and an example negative rule of each dataset are listed below. Here, the common subject and object attributes are represented as $Subject.Attribute-Name$ and $Object.Attribute-Name$ respectively.

University Dataset 1:

- $\{Designation = Student, Subject.Department = Information\ Technology, Subject.Degree = BTech, Subject.Year = First\}, \{Resource\text{-}Type = Assignment, Object.Department = Information\ Technology, Object.Degree = BTech, Object.Year = First\}, \{Access = Allow\}$
- $\{Designation = Officer, Subject.Department = Accounts, Subject.Degree = NA, Subject.Year = NA\}, \{Resource\text{-}Type = Question\ Paper, Object.Department = Information\ Technology, Object.Degree = BTech, Object.Year = First\}, \{Access = Deny\}$

University Dataset 2:

- $\{Designation = Faculty, Post = Associate\ Professor, Subject.Course = Parallel\ Computing, Subject.Department = CSE, Subject.Degree = NA, Subject.Year = NA\},$

Table 4: Experimental Results.

Clas-sifier	University Dataset 1				University Dataset 2				Company Dataset			
	Acc	*Pre*	*Rec*	*F1-s*	*Acc*	*Pre*	*Rec*	*F1-s*	*Acc*	*Pre*	*Rec*	*F1-s*
ANN	0.730	0.971	0.560	0.710	0.890	0.951	0.873	0.910	0.763	0.667	0.568	0.614
DT	0.864	0.932	0.831	0.879	0.716	0.758	0.815	0.786	0.882	0.867	0.758	0.809
RF	0.880	0.969	0.824	0.891	0.706	0.928	0.584	0.717	0.871	0.815	0.789	0.802
ET	0.884	0.969	0.831	0.895	0.737	0.979	0.601	0.744	0.857	0.793	0.768	0.781
GB	0.881	0.935	0.860	0.895	0.489	0.604	0.575	0.589	0.777	0.696	0.579	0.632
XGB	0.864	0.932	0.831	0.879	0.652	0.893	0.516	0.654	0.836	0.761	0.737	0.749

{*Resource-Type* = *AnswerSheet*, *Object.Course* = *Parallel Computing*, *Object.Department* = *CSE*, *Object.Degree* = *BTech*, *Object.Year* = *Fourth*}, {*Access* = *Allow*}

- {*Designation* = *Teaching Assistant*, *Post* = *PhD*, *Subject.Course* = *Software Engineering*, *Subject.Department* = *Information Technology*, *Subject.Degree* = *BTech*, *Subject.Year* = *Third*}, {*Resource-Type* = *Assignment*, *Object.Department* = *CSE*, *Object.Course* = *Operating Systems*, *Object.Degree* = *BTech*, *Object.Year* = *Third*}, {*Access* = *Deny*} (The initial part of the rule refers to a Ph.D. student acting as a teaching assistant for the third year BTech course Software Engineering offered by the Information Technology department.)

Company Dataset:

- {*Designation* = *Programmer*, *Subject.Project-Name* = *P*3, *Subject.Department* = *Development*}, {*Resource-Type* = *Project-Detail*, *Object.Project-Name* = *P*3, *Object.Department* = *Development*}, {*Access* = *Allow*}
- {*Designation* = *Database Engineer*, *Subject.Project-Name* = *NA*, *Subject.Department* = *Technical*}, {*Resource-Type* = *Salary-Detail*, *Object.Project-Name* = *P*3, *Object.Department* = *Human Resource*}, {*Access* = *Deny*} (Here, we have assumed that a database engineer is not specific to a particular project, but rather is associated with all projects of the company.)

The training time for PAMMELA depends on the number of rules and the number of attributes present in the training data. The number of subjects and objects have no effect on the training time. We observed that for ANN, PAMMELA took approximately 130 seconds to complete the training on University Dataset 2. This is the highest recorded training time for our experiments. For the other classifiers, PAMMELA gave much lower training execution times on all the datasets. The reason for this is that the University Dataset 2 is the largest dataset in terms of the number of attributes and rules. The average testing time for PAMMELA was less than 1 second. Of course, the testing time is dependent upon the number of access requests considered. The train and test time indicate that experimenting with multiple classifiers will not be too time consuming. A system administrator can experiment with a number of classifiers for PAMMELA before selecting the best possible option. Once training has been done, PAMMELA can keep on generating new rules without any further training. However, if the policy used for training needs to be changed altogether (may happen if the organizational structure or workflow undergoes some radical modifications), then the classifier will need to be re-trained.

7 CONCLUSION

In this paper, we have proposed the ABAC Policy Inference Problem (ABAC-PIP) that aims to derive a new set of attribute based rules from an existing policy. We have proposed an end-to-end supervised learning based methodology, PAMMELA for solving ABAC-PIP. System administrators can use PAMMELA for augmentative as well as adaptive policy inference when an organization undergoes some changes or an organization migrates to ABAC by adapting the policy of a similar organization. Experimental results show that PAMMELA can be effectively used for aiding system administrators.

In the future, we intend to apply deep learning, reinforcement learning and incremental learning for inferring ABAC policies. Another direction of future research can be attempting to adapt the ABAC policies of multiple organizations for a single target organization. This will be a challenging task as it will require resolving the conflicts among the rules of different organizations.

ACKNOWLEDGMENTS

Research reported in this publication was supported by the National Institutes of Health under award R35GM134927, and a research gift received from Cisco University Research. The content is solely the responsibility of the authors and does not necessarily represent the official views of the agencies funding the research.

REFERENCES

Alohaly, M., Takabi, H., and Blanco, E. (2019). Towards an automated extraction of ABAC constraints from natural language policies. In *ICT Systems Security and Privacy Protection*, pages 105–119.

Batra, G., Atluri, V., Vaidya, J., and Sural, S. (2021a). In-memory policy indexing for policy retrieval points in attribute-based access control. In *Int. Conf. on Information Systems Security*, pages 99–120.

Batra, G., Atluri, V., Vaidya, J., and Sural, S. (2021b). Incremental maintenance of ABAC policies. In *11th ACM Conf. on Data and Application Security and Privacy*, pages 185–196.

Cotrini, C., Weghorn, T., and Basin, D. (2018). Mining ABAC rules from sparse logs. In *2018 IEEE European Symposium on Security and Privacy*, pages 31–46.

Das, S., Sural, S., Vaidya, J., and Atluri, V. (2017). Policy adaptation in attribute-based access control for inter-organizational collaboration. In *IEEE 3rd Int. Conf. on Collaboration and Internet Computing*, pages 136–145.

Das, S., Sural, S., Vaidya, J., and Atluri, V. (2018). Hype: A hybrid approach toward policy engineering in attribute-based access control. *IEEE Letters of the Computer Society*, pages 25–29.

Das, S., Sural, S., Vaidya, J., and Atluri, V. (2019). Policy adaptation in hierarchical attribute-based access control systems. *ACM Trans. on Internet Technology*, 19(3).

Gautam, M., Jha, S., Sural, S., Vaidya, J., and Atluri, V. (2017). Poster: Constrained policy mining in attribute based access control. In *ACM Symposium on Access Control Models and Technologies*, pages 121–123.

Heutelbeck, D., Baur, M. L., and Kluba, M. (2021). In-memory policy indexing for policy retrieval points in attribute-based access control. In *26th ACM Symposium on Access Control Models and Technologies*, pages 59–70.

Hu, V. C., Ferraiolo, D., Kuhn, D. R., Schnitzer, A., Sandlin, K., Miller, R., and Scarfone, K. (2014). Guide to Attribute-Based Access Control (ABAC) definition and considerations. Technical report, NIST Special Publication.

Jabal, A. A., Bertino, E., Lobo, J., Law, M., Russo, A., Calo, S., and Verma, D. (2020a). Polisma - a framework for learning attribute-based access control policies. In *European Symposium on Research in Computer Security*, volume 12308, pages 523–544.

Jabal, A. A., Bertino, E., Lobo, J., Verma, D., Calo, S., and Russo, A. (2020b). FLAP – a federated learning framework for attribute-based access control policies.

Karimi, L., Abdelhakim, M., and Joshi, J. (2021a). Adaptive ABAC policy learning: A reinforcement learning approach.

Karimi, L., Aldairi, M., Joshi, J., and Abdelhakim, M. (2021b). An automatic attribute based access control policy extraction from access logs. *IEEE Trans. on Dependable and Secure Computing*.

Kumar, R., Palanisamy, B., and Sural, S. (2021). BEAAS: Blockchain enabled attribute-based access control as a service. In *IEEE International Conference on Blockchain and Cryptocurrency*, pages 1–3.

Lawal, S. and Krishnan, R. (2021). Enabling flexible administration in ABAC through policy review: A policy machine case study. In *7th IEEE Int. Conf. on Big Data Security on Cloud, IEEE Int. Conf. on High Performance and Smart Computing, and IEEE Int. Conf. on Intelligent Data and Security*, pages 69–74.

Mocanu, D. C., Turkmen, F., and A.Liotta (2015). Towards ABAC policy mining from logs with deep learning. *Intelligent Systems*, pages 124–128.

Narouei, M., Khanpour, H., Takabi, H., Parde, N., and Nielsen, R. (2017). Towards a top-down policy engineering framework for attribute-based access control. *ACM Symposium on Access Control Models and Technologies*, pages 103–114.

Rouhani, S., Belchior, R., Cruz, R. S., and Deters, R. (2021). Distributed attribute-based access control system using permissioned blockchain. *Special Issue on Emerging Blockchain Applications and Technology, World Wide Web*, 24:1617–1644.

Sanders, M. W. and Yue, C. (2019). Mining least privilege attribute based access control policies. In *35th Annual Computer Security Applications Conf.*, pages 404–416.

Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. (1996). Role-Based Access Control Models. *IEEE Computer*, 29(2):38–47.

Talukdar, T., Batra, G., Vaidya, J., Atluri, V., and Sural, S. (2017). Efficient bottom-up mining of attribute based access control policies. *IEEE Int. Conf. on Collaboration and Internet Computing*, pages 339–348.

Xu, Z. and Stoller, S. (2015). Mining attribute-based access control policies. *IEEE Trans. on Dependable and Secure Computing*, 12(5):533–545.

Xu, Z. and Stoller, S. D. (2014). Mining attribute-based access control policies from logs. In *28th Annual IFIP WG 11.3 Working Conf. on Data and Applications Security and Privacy*, pages 276–291.

Zhong, H., Zhou, Y., Zhang, Q., Xu, Y., and Cui, J. (2021). An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare. *Future Generation Computer Systems*, 115:486–496.