Proceedings of the Fourth International Conference on Computing Methodologies and Communication (ICCMC 2020)
IEEE Xplore Part Number:CFP20K25-ART; ISBN:978-1-7281-4889-2



# *A Priority-based Approach for Detection of Anomalies in ABAC Policies using Clustering Technique*

K.Vijayalakshmi
Research Scholar, VISTAS, Chennai, India
Dept. of Computer Science
Arignar Anna Government Arts College, Cheyyar
Email: vijiyuvavelan@gmail.com

V.Jayalakshmi
School of Computing Sciences
Vels Institute of Science, Technology and Advanced Studies
VISTAS, Chennai, India
Email: jayasekar1996@yahoo.co.in

***Abstract*— Cloud computing offers several computing services like storage, networks, hardware, and software. The most beneficial cloud service is cloud storage. The organization or large industries can store their big data in cloud storage on pay for usage scheme. As the big data are outsourced in a distributed cloud environment, securing and protecting the big data is essential. The various access control models, which consist of a set of security policies, are used generally to protect the outsourced data. Anomalies in the security policies dilute the efficiency of the access security model. Developing an efficient access control model to protect the data is a challenging and ongoing process. The primary goal of this paper is to analyze and detect the important anomalies in Attribute-Based Access Control-ABAC Policies. This paper presents an approach that uses Priority-level to avoid the conflict in ABAC Policies. This approach groups the rules of ABAC policies based on Priority-level and similarity with the clustering technique, and detect the anomalies in each cluster rather than all rules, which made this approach efficient.**

***Keywords— ABAC Policy, access control model, anomalies, big data, cloud storage, clustering, priority-level, and security policy.***

## I. INTRODUCTION

With the growth of the internet and many new technologies, data sharing is well grown to manage the business and everything in an easy manner. Cloud, Fog computing, and other service platforms provide the data storage and sharing required by many business organizations and applications. Due to many security risks and threats, preventing data leakage is an ever-challenging job, and also it is essential to make the data owners trust the data-sharing service platforms. In social network environments, protecting the privacy of data is difficult than traditional data [1]. Data providers store only encrypted data in the clouds. Using a hybrid encryption technique may improve the security level [2]. Cloud and other service platforms use various access control models to protect data privacy and confidentiality in the distributed storage environment. The access control model is an efficient and traditional approach to protect shared data by allowing only authorized users. The various tasks of access control models are managing the risk condition when credential information of the user is lost, managing the revocation when multiple users are trying to get access to a single resource, and monitoring and managing the increased rate of new access rights [3]. In the last few decades, various access control models have been developed to secure the outsourced data in the distributed cloud storage environment. Some models are very good at securing the distributed outsourced big data, whereas some models have got great failure. Role-Based Access Control model (RBAC), Attribute-Based Access Control Model (ABAC), Ciphertext Policy Attribute-based Encryption (CP-ABE), Discretionary Access Control (DAC), Mandatory Access Control (MAC) are some existing access control models. MAC and DAC are good enough when the size of data, range of resources, and the number of users are very small. MAC gives its efficiency in securing multilevel database management systems. The main logic implemented in Multilevel Secure (MLS) applications is derived from MAC [4]. But nowadays, all are growing rapidly, and to manage the high range of resource requirements from a huge number of users, these approaches are not suitable, which need an emerging approach. RBAC is such a model that proved its efficiency in securing the outsourced data in the distributed storage environment[5]. Most of the access control models use either RBAC or ABAC or the integrated features of RBAC and ABAC [5].

RBAC maintains a permanent relationship between users, objects or subjects, roles, and permissions, whereas ABAC maintains the changeable mappings between users, roles, and permissions. RBAC model uses a new mid-layer called Role in between the user and permission [5]. The access control model with RBAC is well suited for the environment, which allows a large amount of data sharing and a large number of users. But today, emerging technology consists of cloud computing, mobile computing, Fog computing requires the access control model to be more efficient. Such access control models require the conceptual data and security policies of such an access control model need to be established with the attributes of objects, subjects, and environmental conditions [6]. RBAC is inefficient in supporting attribute-based policies in the huge distributed environment. In this situation, ABAC has been proposed to overcome the defect of RBAC and is well in establishing dynamic and attribute-based security policies. ABAC access control model allow or deny the operation on object requested by subject based on the valid attributes of subjects, objects, and environmental condition

and rules. The security policies are constructed as a set of conditions or rules that verify the list of attributes with the permitted set of values. The anomalies in rules of Policies decrease the efficiency of the ABAC model.

This paper presents our ongoing research for developing a new access control model to protect data in a distributed storage platform. The initial component of our framework is to find possible anomalies in security policies. This paper categorizes the possible anomalies, collects the rules of ABAC policies and clusters rules based on their similarity and Priority-level. This approach detects anomalies in every cluster rather than every rule, which may improve the performance.

## II. RELATED WORK

According to the analysis of anomalies, Jonathan et al. (Moffett and Sloman, 1994) identify the following anomalies Conflict- in-Modality(action: allow or deny), Conflict-in-imperative and authority policy, Conflict-in-limited resources on-demand, Conflict-in-simultaneous tasks of single-subject [7]. In 2016 Khoumsi et al. framed two types of anomalies conflicting anomalies and non-conflicting anomalies. Maryem Ait El Hadj ( Meryeme Ayache, Yahya Benkaouz, Ahmed Khoumsi, and Mohammed Erradi, 2017) proposed an approach [8] to detect the anomalies in ABAC policies. Maryem Ait El Hadj uses equal weights for all categories(Subject, Object, Environmental conditions) to measure the similarity. Maryem Ait El Hadj and Mohammed Erradi[In 2018] proposed an approach [9] to detect and correct the anomalies in ABAC Security policies. This approach uses security policies and access logs as input to find a suspicious attack. Clustering is applied to group the security policies of similar concepts. In my view, this model only detects the anomalies in suspicious rules, which may lead the incorrect results. Thus anomalies of other rules(not found as suspicious rules) may not be detected an corrected. Even it's a time-consuming process; it is mandatory to detect and correct anomalies in all rules of security policies.

Contrary to the above-analyzed approaches, we use additional parameter Priority-level in each rule and cluster all rules based on similarity score and priority_level. Because of using additional parameter Priority_level, this approach avoids the conflict, makes the perfect clusters of similar rules, and reduces the computation time. Our proposed approach finds the anomalies of all rules instead of only suspicious rules to improve efficiency. Maryem Ait El Hadj and Mohammed Erradi have not implemented the Map-reduce method for the parallel distributed environment. Our future work is to use Map-Reduce to make the approach suitable for a fully distributed environment.

## III. PRELIMINARIES

### A. ABAC Model

The traditional authentication system uses the identity of the user or subject to make the decision (allow or deny) for a request of operation (read, write) on the object(resource: database, file). This access control is not enough to manage security in a large, distributed environment. This emerging technology needs additional information or attributes to trust or allow the user for data sharing. ABAC is such a model to protect the outsourced data in a distributed environment. ABAC access control model, shown in figure Fig. 1, contrasts with the traditional authentication system.

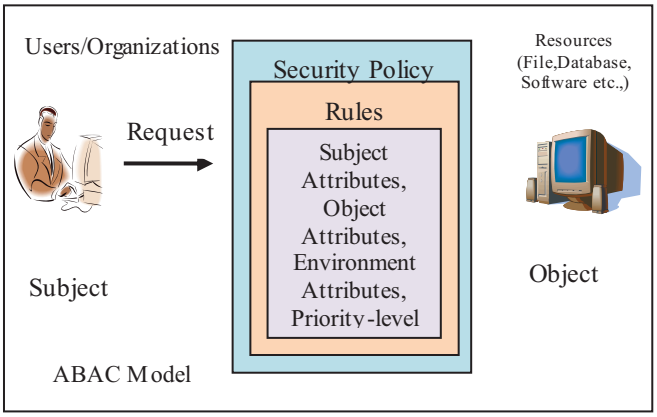


Fig. 1. Attribute-Based Access Control System

ABAC access control model allow or deny the operation on object requested by subject based on the valid attributes of subjects, objects, and environmental condition and rules. The security policies are constructed as a set of conditions that verify the list of attributes with the permitted set of values. The ABAC model is described in the following important jargon.

*Subject*: The term subject denotes an organization or a user who requests the resource.

*Object*: Resources such as file, software, database are called objects.

*Attributes*: Attributes describe the properties or characteristics of the subject or object. They are a collection of information defined as a pair of values(name of the attribute, the value of the attribute).

*Operation:* It is a function of request on the object is being executed (read, write, etc.)

*Policy*: It is a set of conditions or rules established to make a decision based on the values of the attributes.

*Environmental conditions*: They describes the characteristics of environmental conditions such as date of request, current time

### B. Problem statement

Previous approaches use only similarity values to group all similar rules. Our proposed approach uses additional parameter Priority-level to cluster the rules. With the use of additional parameter, our approach avoids that the conflict occurs when demand on limited object or resource, is avoided and also make the perfect clusters. Contrary to some existing approaches which detect the anomalies in all rules which require more computation time, Our approach detects anomalies in clusters rather than all rules to reduce the computation time.

### C. Security Policy

The security policy is a set of rules. Extensible Access Control Markup Language (XACML) is mostly used language to construct the Security Policy because of its simplicity and expressive power [10][11]. Each rule is constructed with categories Subject, Object, and Environmental conditions. In our approach, we use additional parameter Priority-level to construct the rule. The security policy SP is

$SP= \{R_1, R_2,\ldots, R_n\}$

Each Rule R is expressed as

$R= \{X_{op} \mid PR\}$

The decision X (allow, deny) is made for a request of operation (read, write) based on the predicate PR.

$PR= \{atr_1 \in Vatr_1, atr_2 \in Vatr_2, \ldots, atr_n \in Vatr_n \wedge$ Priority-level=non-empty-integer-value$\}$

So rule R can be written as

$R= \{X_{op} \mid atr_1 \in Vatr_1, atr_2 \in Vatr_2,\ldots, atr_n \in Vatr_n \wedge$ Priority-level=non-empty-integer-value$\}$

$atr_1, atr_2,\ldots, atr_n$ are the list of attributes belonging to any category(Subject, Object, Environmental Conditions)

$Vatr_1$, $Vatr_2$, $Vatr_n$ are the set of permitted values of the attributes $atr_1, atr_2,\ldots, atr_n$ respectively

*Example of a rule:*

$R= \{allow_{read} \mid$ Designation={Manager, Admin}, Department={Loan}, FileName={Customer_Savings}, Time={08.00-18.00}^Priority-level=1$\}$

In this above rule, Designation and Department are attributes of Subject, Filename is the attribute of Object, and Time is the attribute of Environmental conditions. This rule has the priority 1.

### D. Measuring Similarity Value

According to the previous research [12], the similarity value (SV) of two rules ($R_i$, $R_j$) is measured using the following formula-1

$$SV_R(R_i,R_j)=\sum_{t \in \{S,O,E\}} P_t SV_t(R_i,R_j) \text{ ------(1)}$$

The notations S, O, E are used to specify categories Subject, Object, and Environment respectively

The similarity value of the two rules is calculated by summing the product of probability P and similarity value SV of each category(S, O, and E). Hence the formula-1 can be expressed as

$$SV_R(R_i,R_j)=P_S SV_S(R_i,R_j) + P_O SV_O(R_i,R_j) + P_E SV_E(R_i,R_j)$$

The probability of each category can be assigned based on application-taken. Our approach assigns equal probability to all three categories. As the number of categories is three, the equal probability of each category is 1/3.

$$SV_R(R_i,R_j) = \frac{1}{3} SV_S(R_i,R_j) + \frac{1}{3} SV_O(R_i,R_j) + \frac{1}{3} SV_E(R_i,R_j) \text{ ------(2)}$$

The similarity value of each category (C) is calculated using the following formula-3

$$SV_c(R_i,R_j) = \sum_{atr \in \{ATc(Ri) \cap ATc(Rj)\}} P_{atr} SV_{atr}(R_i,R_j) \text{ ----- (3)}$$

The similarity value of two rules for each category is calculated by summing the probability ( $P_{atr}$) and similarity value ($SV_{atr}$) of every attribute (atr), where atr is common to both two rules $R_i$, $R_j$) of category C. $ATc(R_j)$ is the set of attributes of the category C in rule $R_j$

The similarity value of an attribute of a category C is calculated using the following formula-4. Let NSV be the number of values the same for every common attribute in both $R_i$ and $R_j$, and NDV be the number of distinct values for every common attribute in both $R_i$ and $R_j$.

$$SV_{atr}(R_i,R_j)= \frac{NSV}{NDV} \text{ ------ (4)}$$

*Example for calculating similarity value:*

$R_1=\{allow_{read} \mid$ Designation = {Manager,Admin}, Departmnet = {Loan}, FileName={Customer_Savings}, Time={08.00-18.00}^Priority-level=1$\}$

$R_2=\{allow_{read} \mid$ Designation ={Admin}, Departmnet={Loan}, FileName={Customer_Savings}, Time={08.00-18.00}^Priority-level=2$\}$

The similarity value of the categories Subject, Object, and Environmental conditions are calculated using formula-3

The similarity value of category Subject (Designation and Department are the attributes common to both $R_1$ and $R_2$) is

$$SV_S(R_1,R_2)= \sum_{atr \in \{Designation,Department\}} P_{atr} SV_{atr}(R_1,R_2)$$

There are two common attributes in the category Subject ($SV_S$), and if equal probabilities are used, then the probability of every attribute is 1/2.

$$\begin{aligned} SV_S(R_1,R_2) &= P_{Designation} SV_{Designation}(R_1,R_2) \\ &\quad + P_{Department} SV_{Department}(R_1,R_2) \\ &= \frac{1}{2} SV_{Designation}(R_1,R_2) \\ &\quad + \frac{1}{2} SV_{Department}(R_1,R_2) \end{aligned}$$

The similarity value of rules R1 and R2 for the category Object $SV_O$ (FileName is the only attribute of Object) is

$$SV_O(R_1,R_2)=P_{FileName} SV_{FileName}(R_1,R_2)$$

As the probability $P_{FileName}=1$,

$$SV_O(R_1,R_2)=SV_{FileName}(R_1,R_2)$$

The similarity value of rules R1 and R2 for the category Environmental Condition (Time is the only attribute and the probability $P_{Time}=1$ ) is

$$SV_E(R_1,R_2)=P_{Time} SV_{Time}(R_1,R_2) = SV_{Time}(R_1,R_2)$$

The similarity value of rules R1 and R2 for the attributes Designation, Department, FileName, and Time are calculated using the formula-4

$$SV_{Designation}(R_1,R_2) = \frac{NSV(Designation)}{NDV(Designation)} = \frac{|\{Admin\}|}{|\{Manager,Admin\}|} = \frac{1}{2}$$

$$SV_{Department}(R_1,R_2) = \frac{NSV(Department)}{NDV(Department)} = \frac{|\{Loan\}|}{|\{Loan\}|} = 1$$

$$SV_{FileName}(R_1,R_2) = \frac{NSV(FileName)}{NDV(FileName)} = \frac{|\{Customer\_Savings\}|}{|\{Customer\_Savings\}|} = 1$$

$$SV_{Time}(R_1,R_2) = \frac{NSV(Time)}{NDV(Time)} = \frac{|\{8.00:18.00\}|}{|\{8.00:18.00\}|} = 1$$

By substituting the above similarity values, we calculate the followings

$$SV_S(R_1,R_2) = \frac{1}{2} SV_{Designation}(R_1,R_2) + \frac{1}{2} SV_{Department}(R_1,R_2) = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times 1 = \frac{1}{4} + \frac{1}{2} = 0.25 + 0.5 = 0.75$$

$$SV_O(R_1,R_2) = SV_{FileName}(R_1,R_2) = 1$$

$$SV_E(R_1,R_2) = SV_{Time}(R_1,R_2) = 1$$

The similarity value of rules ($R_1$, $R_2$) is

$$SV_R(R_1,R_2) = \frac{1}{3} SV_S(R_1,R_2) + \frac{1}{3} SV_O(R_1,R_2) + \frac{1}{3} SV_E(R_1,R_2) = \frac{1}{3} \times 0.75 + \frac{1}{3} \times 1 + \frac{1}{3} \times 1 = 0.25 + 0.33 + 0.33 = 0.91$$

## IV. PROPOSED APPROACH

Many Clustering techniques such as the KNN(K-Nearest Neighbouring) algorithm, *K*-means algorithm, Hierarchical clustering algorithm, Density-based clustering algorithm are good in forming clusters [13]. Our approach cluster rules based on similarity value and Priority-level. We develop a new algorithm that clusters the rules with two parameters similarity value and priority-level. All previous research of clustering with similarity value [14] uses the threshold value is 0.8. They cluster the two rules if the similarity value is above 0.8. But our approach groups the two rules, if the similarity value of the two rules is greater than the threshold value 0.8 and the priority-level of two rules, is equal. Otherwise, they are not considered as similar rules. The Priority-level is used to avoid the conflict that occurs when two or more requests of the same operation (read) upon the limited object (File: Customer).

### *A. Algorithm*

```
Our Proposed Algorithm:
Input:
  SP={R1,R2,….,Rn} /* SP is security policy
                   contains set of Rules*/
Output:
  C={C1,C2,…,Ck}  // C is a Set of Clusters
Priority-Similarity-based Algorithm:
 1. k=0;
      /* Initialize that all rules are not
          clustered */
 2. For i=1 to n do
 3. clustered[i]=false;
 4. first_similarity[i]=false;
 5. Next i
 6. i=0;
 7. For Rule1=Ri to Rn-1 do
 8. i=i+1;
 9. j=i+1;
10.For Rule2=Ri+1 to Rn do
11.sv=SVR(Rule1,Rule2);
12.pl1= Priority-level (Rule1);
13.Pl2= Priority-level (Rule2);
14.pdiff=|pl1-pl2|;
15.If sv>0.8 and pdiff=0 then
    //find the first similar rule for Rule1
16.If first_similarity[i]=false then
   // Initialize the Cluster Ck
17.k=k+1;
18.Ck= {Rule1, Rule2};
19.first_similarity[i]=true;
20.else
 /* The similar rule Rule2 is joined with the
              cluster Ck */
21.Ck=Ck U {Rule2};
22.End If
     // Rule1 and Rule2 are clustered
23.clustered[i]=true;
24.clustered[j]=true;
25.End If
26.j=j+1;
27.Next Rule2
 // One Rule may be in more than one cluster
28.If clustered[i]=false then
29.Ck= {Rule1};
30.clustered[i]=true;
      /* Every rule must be contained in at
           least one Cluster */
31.End If
32.Next Rule1
```

```
// Check whether the last rule is clustered
33.If clustered[n]==false then
34.K=k+1;
35.Ck= {Rulen};
36.clustered[n]=true;
37.End If
38.End.
```

Our proposed algorithm takes the set of rules $SP=\{R_1, R_2, \ldots, R_n\}$ as input and produces a set of clusters of rules $C=\{C_1, C_2, \ldots, C_k\}$ as the output. `clustered[i]` is a boolean value and used to verify that the rule $R_i$ is contained in at least one cluster, and `first_similarity[i]` is also a boolean value and used to initialize every new cluster. `Priority-level` $(R_i)$ is the Priority-level of rule $R_i$, and $SV_R(Rule1, Rule2)$ is the similarity value calculated by using the formula-1 mentioned above.

Our proposed algorithm uses similarity value and the priority-level of two rules to cluster similar rules. In contrary to the other existing approaches, we use priority level to avoid the conflict. Even the two rules are similar, making the decision (Allow, Deny) may vary depending on the priority-level. Our approach solves the conflict that occurs the simultaneous access of the same request of operation (read, write) on the same object(file, database). In the above example, The similarity value of rules R1 and R2 is 0.91which is greater than the threshold value 0.8. But the Priority-level of R1 and R2 are not equal (1! =2). Hence R1 and R2 are not grouped as the same cluster. If the similarity value of two rules is above 0.8 and the Priority-level of the two rules are equal, then those two rules can be contained in the same cluster.

TABLE I. SIMILARITY VALUES AND PRIORITY DIFFERENCE OF RULES

| Pair of Rules | Similarity value | The absolute difference of Priority-level |
|---|---|---|
| $(R_1, R_2)$ | 0.9 | 2 |
| $(R_1, R_3)$ | 0.93 | 0 |
| $(R_1, R_4)$ | 0.5 | 1 |
| $(R_1, R_5)$ | 0.4 | 0 |
| $(R_2, R_3)$ | 0.5 | 1 |
| $(R_2, R_4)$ | 0.5 | 1 |
| $(R_2, R_5)$ | 0.7 | 0 |
| $(R_3, R_4)$ | 0.6 | 0 |
| $(R_3, R_5)$ | 0.52 | 1 |
| $(R_4, R_5)$ | 0.9 | 0 |

Consider the set Security Policy $SP=\{R_1,R_2,R_3,R_4,R_5\}$. Our proposed algorithm makes ten combinations of pair of rules. The above table Table-1 shows the similarity value and priority-level of each pair of rules. The proposed algorithm made the clusters $C_1=\{R_1, R_3\}$, $C_2=\{R_2\}$, and $C_3=\{R_4, R_5\}$ for the above example. The set of pairs of rules $\{(R_1, R_2), (R_1, R_3), (R_4, R_5)\}$ has the similarity value above the threshold value 0.8. But the Priority-levels of $(R_1, R_2)$ are not equal; they are not clustered as the same group. The two pair of rules $(R_1, R_3)$ and $(R_4, R_5)$ is clustered because their Priority-levels of these pair of rules are equal.

### *B. Flowchart of our proposed algorithm*

Figure Fig. 2 shows the flowchart of our proposed algorithm.

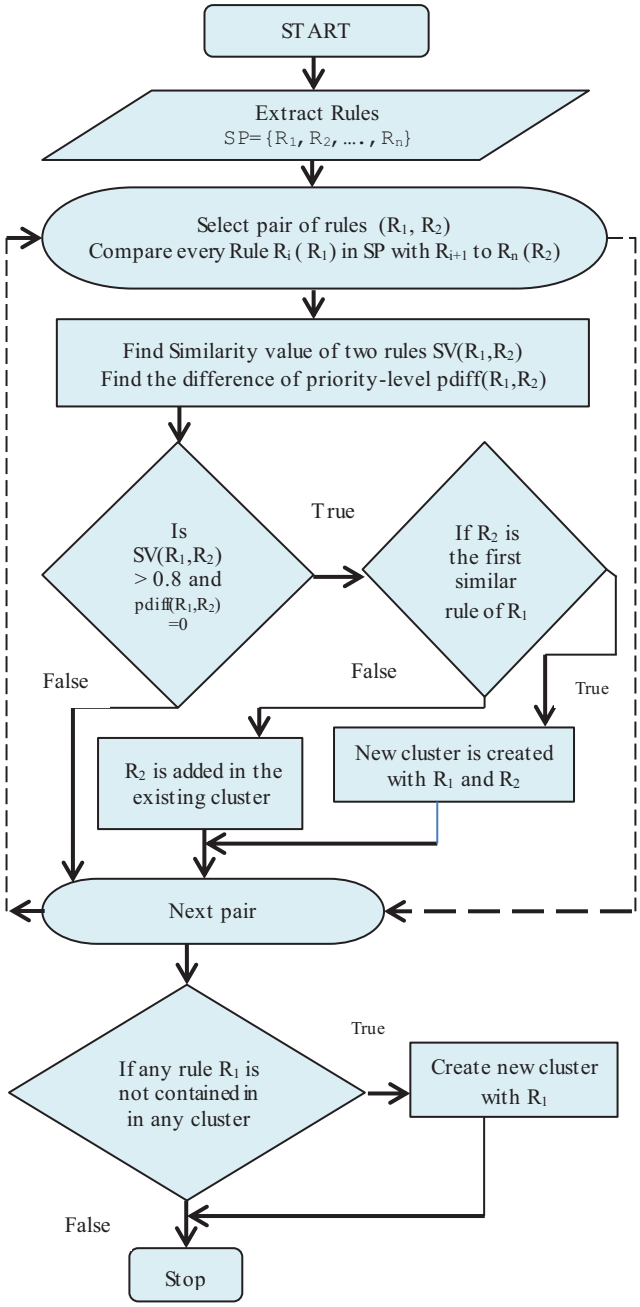


Fig. 2. Flowchart for Priority-Similarity-based Algorithm

### *C. Trace out of our proposed approach*

Consider the following the security policy SP which contains three rules SP={$R_1$, $R_2$, $R_3$}

$R_1$={$allow_{read}$ | Designation = {Chief Doctor, Duty Doctor}, Departmnet = {Pediatric}, FileName={Blood-Reports}, Time={08.00-18.00}^Priority-level=1}

$R_2$={$allow_{read}$ | Designation ={Duty Dcotor}, Departmnet={Pediatric}, FileName={Blood-Reports}, Time={08.00-18.00}^Priority-level=1}

$R_2$={$allow_{read}$ | Designation ={Duty Doctor, Nurse}, Departmnet={Pediatric}, FileName={Blood-Reports}, Time={08.00-18.00}^Priority-level=2}

The following tables Table II, Table III and Table IV show the similarity value of each attribute, similarity value of the each category and the similarity value of rules respectively. The similarity value of each attribute is calculated using the fomula-4. The similarity value of each category is measured using formula-3 and the similarity value of two rules is determined using the formula-2.

TABLE II. SIMILARITY VALUE OF ATTRIBUTES

| Pair of Rules | $R_1$, $R_2$ | $R_1$, $R_3$ | $R_2$, $R_3$ |
|---|---|---|---|
| Similarity value of the attribute Designation | 1/2 | 1/3 | 1/2 |
| Similarity value of the attribute Department | 1 | 1 | 1 |
| Similarity value of the attribute File | 1 | 1 | 1 |
| Similarity value of the attribute Time | 1 | 1 | 1 |

TABLE III. SIMILARITY VALUE OF CATEGORIES

| Pair of Rules | $R_1$, $R_2$ | $R_1$, $R_3$ | $R_2$, $R_3$ |
|---|---|---|---|
| Similarity value of the category Subject | 0.75 | 0.66 | 0.75 |
| Similarity value of the category Object | 1 | 1 | 1 |
| Similarity value of the category Environment | 1 | 1 | 1 |

TABLE IV. SIMILARITY VALUE OF RULES

| Pair of Rules | Similarity value of Rules | Difference between Priority-level of Rules | Clusters formed in existing approaches | Clusters formed in our proposed approach |
|---|---|---|---|---|
| $R_1$, $R_2$ | 0.91 | 0 | C1={ $R_1$, $R_2$} | C1={$R_1$, $R_2$} |
| $R_1$,$R_3$ | 0.88 | 1 | C1={ $R_1$, $R_2$, $R_3$} | - |
| $R_2$,$R_3$ | 0.91 | 1 | C2={ $R_2$,$R_3$ } | C2={ $R_3$ } |

Our proposed algorithm verifies the similarity value and Priority-level of each pair of rules ($R_1$, $R_2$ ), ($R_1$, $R_3$ ) and ($R_2$, $R_3$). As the similarity of ($R_1$, $R_2$ ) is 0.91(above the threshold value 0.8) and the difference of priority-level is zero, these two rules are grouped as a new cluster C1={ $R_1$, $R_2$}. Even the similarity of ($R_1$, $R_3$ ) is 0.9, the difference of priority-level is not zero, they are not grouped. Then the rule $R_2$ is compared with the rule $R_3$; These rules are also not clustered according to our proposed clustering-criteria. Then finally, the last rule $R_3$ is not contained in any cluster which is notified by the boolean array clustered[n]. The rule $R_3$ is stored in a new cluster C2={ $R_3$}. Our approach makes two cluster C1={ $R_1$, $R_2$}, and C2={ $R_3$}. If the previous approach is used to cluster the above rules, it forms several clusters that require more processing time. Our approach with the use of priority-level reduces the redundant clusters and also avoids the conflict on demand.

## V. ANOMALIES

This paper classifies the anomalies into three categories, namely Redundancy-Anomaly, Conflict-Decision-Anomaly, and Conflict-Demand-Anomaly.

Redundancy-Anomaly will occur when the request of operation (on the same object) is the same (X=Y) in both rules ($R_i$, $R_j$), and $R_i$ is a subset of $R_j$ or $R_j$ is a subset of $R_i$.

$$\text{Redundancy}(R_i, R_j) = \begin{cases} \text{true} \mid R_i \text{ is a subset of } R_j \text{ or } R_j \text{ is a subset of } R_i \text{ and } X=Y \\ \text{false} \mid \text{Otherwise} \end{cases}$$

Conflict-Decision-Anomaly will occur when the decision made for two rules are varied if the two rules are similar.

$$\text{Conflict-Decision-Anomaly}(R_i, R_j) = \begin{cases} \text{true} \mid X_{op}(R_i) \neq X_{op}(R_j) \text{ and } R_i \text{ is a subset of } R_j \text{ or } R_j \text{ is a subset of } R_i, \\ \text{false} \mid \text{Otherwise} \end{cases}$$

Conflict-Demand-Anomaly will occur when more than one request of the same operation upon the same object is made. This conflict will occur if there is more demand for the limited object.

$$\text{Conflict-Demand-Anomaly}(R_i, R_j) = \begin{cases} \text{true} \mid \text{Operation}(R_i) = \text{Operation}(R_j) \text{ and } \text{Object}(R_i) = \text{Object}(R_j) \text{ and } R_i \neq R_j \\ \text{false} \mid \text{Otherwise} \end{cases}$$

## VI. ANALYSIS OF PRIORITY-SIMILARITY-BASED ALGORITHM

We used JAVA to implement our proposed algorithm with various sizes of security policies. The rules consist of attributes of categories Subject, Object, Environmental Conditions, and an additional parameter Priority-level. The time complexity of the existing algorithm ABAC-PC is O (n2). The time complexity of the Priority-Similarity-based algorithm is O (nm), where n is the number rules, and m is n-1. Each rule Ri is paired with the rule Rj which is not already combined with Ri. Each rule Ri is combined with rules $R_{i+1}$ to $R_n$. Hence the redundant combination of two rules is avoided to improve the performance of the algorithm. Clustering is made not only based on similarity value, but it also involves the Priority-level to avoid the conflict of demand

of the same resource.

## VII. CONCLUSION

In the emerging digital world, data generation is increased exponentially. Data and all resources are outsourced and distributed to make available to everyone who needs them. Developing an efficient access control model to secure the outsourced data in a large distributed environment is an essential and ever going challenging task due to various security threats. Detecting anomalies in the security policies made the access control model to provide high security of data. Our proposed approach collects all rules of policy and group similar rules based on similarity value and Priority-level. Our approach avoids the conflict of demand on limited objects by using additional parameter Priority-level. Detecting anomalies only in clusters rather than all rules improves the performance of the approach. This paper represents only three classifications of anomalies, and we aim to classify more possible anomalies. Our future work is to use Data aggregation and Map_Reduce techniques to make our approach to provide security of data in a large distributed environment.

## REFERENCES

[1] A. Praveena and D. S. Smys, "Anonymization in Social Ne tworks: A Survey on the issues of Data Privacy in Social Network Sites," *Int. J. Eng. Comput. Sci.*, vol. 5, no. 3, pp. 15912–15918, 2016, doi: 10.18535/ijecs/v5i3.07.

[2] D. K. Anguraj and S. Smys, "Trust-Based Intrusion Detection and Clustering Approach for Wireless Body Area Networks," *Wirel. Pers. Commun.*, vol. 104, no. 1, 2019, doi: 10.1007/s11277-018-6005-x.

[3] Y. Imine, A. Lounis, and A. Bouabdallah, "AC SC," 2018, doi: 10.1016/j.jnca.2018.08.008.

[4] E. Sahafizadeh, "Survey on Access Control Models," pp. 1–3, 2010.

[5] H. Qi, X. Di, and J. Li, "Journal of Information Security and Applications Formal definition and analysis of access control model based on role and attribute," vol. 43, pp. 53–60, 2018, doi: 10.1016/j.jisa.2018.09.001.

[6] V. C. Hu *et al.*, "Guide to attribute based access control (abac) definition and considerations," *NIST Spec. Publ.*, vol. 800, p. 162, 2014, doi: 10.6028/NIST.SP.800-162.

[7] J. D. Moffett and M. S. Sloman, "Policy conflict analysis in distributed system management," *J. Organ. Comput.*, vol. 4, no. 1, pp. 1–22, 1994, doi: 10.1080/10919399409540214.

[8] M. A. El Hadj, M. Ayache, Y. Benkaouz, A. Khoumsi, and M. Erradi, "Clustering-based approach for anomaly detection in XACML policies," *ICETE 2017 - Proc. 14th Int. Jt. Conf. E-bus. Telecommun.*, vol. 4, no. Icete, pp. 548–553, 2017, doi: 10.5220/0006471205480553.

[9] M. Ait, E. Hadj, M. Erradi, and A. Khoumsi, "Validation and Correction of Large Security Policies : A Clustering and Access Log Based Approach," *2018 IEEE Int. Conf. Big Data (Big Data)*, no. 1, pp. 5330–5332, 2018, doi: 10.1109/BigData.2018.8622610.

[10] F. Deng *et al.*, "Establishment of rule dictionary for efficient XACML policy management," *Knowledge-Based Syst.*, vol. 175, pp. 26–35, 2019, doi: 10.1016/j.knosys.2019.03.015.

[11] C. D. P. K. Ramli, H. R. Nielson, and F. Nielson, "The logic of XACML," *Sci. Comput. Program.*, vol. 83, pp. 80–105, 2014, doi: 10.1016/j.scico.2013.05.003.

[12] D. Lin, P. Rao, R. Ferrini, E. Bertino, and J. Lobo, "A similarity measure for comparing XACML policies," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 9, pp. 1946–1959, 2013, doi: 10.1109/TKDE.2012.174.

[13] G. Ahalya and H. M. Pandey, "Data clustering approaches survey and analysis," *2015 1st Int. Conf. Futur. Trends Comput. Anal. Knowl. Manag. ABLAZE 2015*, pp. 532–537, 2015, doi: 10.1109/ABLAZE.2015.7154919.

[14] S. Guo, "Analysis and Evaluation of Similarity Metrics in Collaborative Filtering Recommender System," 2014.