Improving Performance of ABAC Security Policies Validation using a Novel Clustering Approach

K.Vijayalakshmi[1]
Vels Institute of Science
Technology and Advanced Studies
Chennai, India

Dr.V.Jayalakshmi[2]
School of Computing Sciences
Vels Institute of Science, Technology and Advanced
Studies, Chennai, India

***Abstract*—Cloud computing offers several services, such as storage, software, networking, and other computing services. Cloud storage is a boon for big data and big data owners. Although big data owners can easily avail cloud storage without spending much on infrastructure and software to manage their data, security is a big issue, and protecting the outsourced big data is challenging and ongoing research. Cloud service providers use the attribute-based access control model to detect malicious intruders and address the security requirements of today's new computing technologies. Anomalies in security policies are removed to improve the efficiency of the access control model. This paper implements a novel clustering approach to cluster security policies. Our proposed approach uses a rule-specific cluster merging technique that compares the rule with the clusters where the probability of similarity is high. Hence this technique reduces the cost, time, and complexity of clustering. Rather than verifying all rules, detecting and removing anomalies in every cluster of rules improve the performance of the intrusion detection system. Our novel clustering approach is useful for the researchers and practitioners in the ABAC policy validation.**

***Keywords*—Anomalies; attribute-based access control model; big data; cloud storage; clustering; intrusion detection system; security policy**

I. Introduction

Cloud storage is one of the most beneficial services to leverage and manage big data efficiently [1]. Large-scale enterprises, governments, and commercial organizations use the cloud to store and manage their big data without spending much on implementing infrastructure. Data privacy and security are essential, and securing the shared big data is a real challenge in today's emerging computing technologies [2] [3]. We have surveyed about challenges, security issues, and existing methodologies for addressing the security requirements [4]. Data breaches, data loss, account hijacking, denial of services, and malicious insiders are some attacks, and various encryption techniques are used to protect the data in the distributed cloud storage [5] [6]. Cloud service providers use various access control models to implement the Intrusion Detection System [7] [8]. The access control model is a function that identifies whether a requested operation on a shared object(resource) is legal or not [9]. In other words, the access control model is a protection technique that categorizes the authorized and unauthorized users and protects the shared resources based on the Access Control List (ACL) or security policies. The access control models use rules to determine which user can get what types of accesses for a shared resource. It manages all access-rights and access-conflicts over the shared resources [10]. The term object refers to the shared resources in a distributed environment. The access control models use the term subject to refer to the process being executed for a single user or an organization, which requests access for the object. Fig. 1 shows the working mechanism of the access control model. Many access control models are developed, and each is good in some situations and gives performance up to their level [11]. In this paper we described the major four access control models Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role Based Access Control (RBAC), and Attribute Based Access Control (ABAC) models.

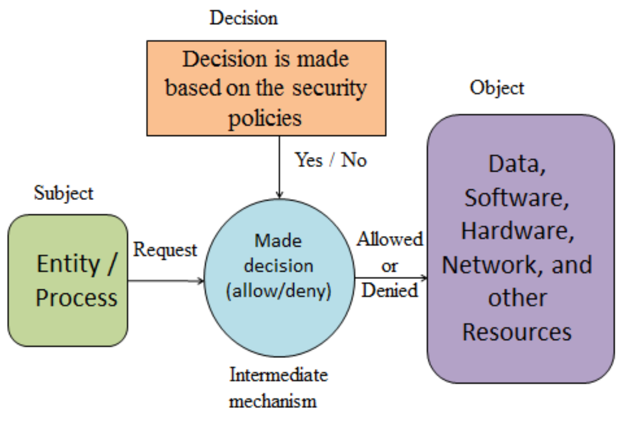

Decision
Decision is made based on the security policies
Yes / No
Object
Data, Software, Hardware, Network, and other Resources
Subject
Entity / Process
Request
Made decision (allow/deny)
Allowed or Denied
Intermediate mechanism

Fig. 1 A Working Mechanism of Access Control Model.

Discretionary Access Control (DAC) allows the owner of the object to specify the access rights of the user for their object. Thus the resource-owner decides and determines the security policies directly and explicitly unlike non-discretionary access control models. In non-discretionary access control models, rather than the resource-owner, the administrator determines the security policies [12]. DAC creates and maintains ACL for every individual resource to determine the access privileges of the user for that resource. ACL of each resource is a table of records and each record specifies the information about the user(or group of users) and the access rights [2]. The process of specifying ACL in DAC is very flexible, easily updatable, and reduces the administrator's work but the level of security is less[6]. The operating systems WINDOWS, LINUX use DAC. DAC meets the security requirements when the number of resources and users are limited but it fails to address the security requirements of today's computing technologies. The delegation feature of DAC has even more benefits like offering cooperation between users and the resource-owner [8]. But it leads to loss of confidentiality, integrity, and availability of resources if any user made a wrong decision while granting access to other users. As DAC requires a resource-owner for each object to determine the access rights, fulfilling this requirement is difficult in new computing technologies like cloud and fog computing [9][10].

Mandatory Access Control (MAC) is a central authority system where the administrator or the authorized person only can set access rights. The access rights cannot be specified or updated by the user like DAC. MAC uses a security label for each resource and subject. There are two pieces of information in a security label. The first piece of information called classification-segment determines the nature (public or private) of the resource or subject. The second segment category specifies the information about the process or application. MAC allows the requested operation on the object if the security labels of the object and subject match. The administrator only can determine the characteristics or attributes of the subject and object. Even the allowed users cannot specify or change the attributes. While DAC fails to give security on sensitive and confidential data, MAC meets these security requirements of the business and government organization. MAC is capable of managing trojan horses to protect the shared resources. Unlike DAC, the integrity and confidentiality of the resource are preserved by MAC. The feature reference monitor of MAC stores all security labels and policies in the database and monitors the access rights of every request for the object. The operating system Security-Enhanced Linux implements MAC to protect the objects. If the communication between the subject and object is high, then managing security labels and policies is a challenging issue. DAC and MAC are efficient when the size of data and the number of users are small [11]. MAC increases the complexity in managing security policies for large applications [2].

RBAC introduces a new concept of assigning roles between users and resources. RBAC (Role-Based Access Control) uses the permission-role and role-subject association to protect shared resources, whereas ABAC(Attribute-Based Access Control) uses security policies [13]. RBAC meets the security requirements of large applications. RBAC determines the security policies or access rights based on the role or job of the subject. Thus this access control model specifies the possible access rights (permissions) to the role. The subject can get access to an object based on his role or job. RBAC performs two assignments thus it first assigns all permissions required for a particular job or role for an object and second it assigns a role to the user or subjects. RBAC provides two efficient features least privilege and separation. The feature least privilege means RBAC assigns access rights or permissions required only for the role or task being processed on the object. As no additional access rights can be availed by the role, it prevents unauthorized access and preserves the integrity of the data. Another feature separation of duties distributes the process into various roles or duties and assigns the duty to the subject. As the subject can perform his/its assigned role only, RBAC prevents the entry of fraud intruders [14]. Hence no subject can perform all permissions or access rights of the shared object rather than the central authority (administrator). It increases the security level and RBAC has the capacity of ensuring whether the end-users can do their permitted transactions. Using the feature of grouping rules reduces the complexity of the administrator's work. Government sectors, military, and private organizations use RBAC to protect databases, web services, and all shared resources [15].

As RBAC performs constant or permanent associations (permission-role and role-subject), it fails to address the security needs of the computing technology that requires the dynamic associations and role-independent security policies. ABAC addresses this issue and is capable of performing dynamic relationships and generating role-independent security policies. ABAC security policies are generated with the attributes of the categories subject (who requests the operation of a resource), object (resource is shared in a distributed environment), and environment (time, the importance of the request, etc.). Both RBAC and ABAC address the security needs of large-scale applications or organizations, but ABAC fulfills the complex security requirements of today's computing technologies [9]. It gives better security even if the communication between the subjects and objects is increasing exponentially [16]. It has many features fine and coarse-grained access, dynamic mapping (subject-object), and flexibility. With the properties efficiency, flexibility, granularity, and security-level [17]. We gave a summarized analysis of the above-described access control models in Table I.

TABLE I. ANALYSIS OF ACCESS CONTROL MODELS

Access control model	Granularity (accuracy level of the model)	Flexibility (how the security policies are generated, expressed, and updated)	Efficiency (how quickly and correctly a decision is performed on the subject's request)	Security level (how the shared resources are protected)
DAC	Good at small scale applications	Good	Poor	Low
MAC	Good at small scale applications	Good	Poor	Better than DAC
RBAC	Good at large scale applications	Good	Good	Good
ABAC	Good at today's computing technologies and big data	Good	Good	Good

Most of today's computing technologies cloud, fog, edge, and IoT use ABAC to meet the security requirements. But the anomalies in the ABAC security policies dilute the reliability and efficiency of the mechanism [18]. Detection and elimination of all possible anomalies in security policies or rules improve the efficiency and accuracy of security. We can improve the performance of the detection mechanism by detecting anomalies in every cluster of similar rules, instead of detecting in every rule. This paper implemented a new approach to cluster similar ABAC security rules. The scopes of our approach are:

- Tool to generate ABAC Policies.
- Ability to measure the similarity of rules.
- Reduces the generation of more clusters.
- Avoids avoidable-redundancy (the same rule is contained in many clusters).
- Avoids the conflict clusters (not all rules in clusters are similar).
- Less time-complexity.
- Ease of implementation.

We use Section II to describe some clustering approaches already used in various research works, and Section III presents the fundamental concepts of the ABAC model and our proposed clustering approach. We use Section IV to describe the system architecture of our implementation. We analyzed output and discussion about the outcome of our approach in Section V. Finally, we concluded in Section VI.

II. Related Work

Bhatia and Vandana surveyed Nearest Neighbor techniques (NN). NN technique is simple, effective, and robust to noise. This approach generates clusters of nearest neighbors where the nearest neighbor is identified by some calculated value. The KNN(K-Nearest Neighbor) is an unsupervised clustering technique and determines the nearest neighbor based on the k-value. They categorized the NN techniques into two groups; Structure less NN techniques and structure-based NN techniques. Structure less NN determines the nearest neighbor from the training and sample data. In structure-based NN, the nearest neighbors are determined based on the structure of data such as top-down or bottom-up like a k-d tree or bell tree. Both categories are the extended or improved version KNN techniques. They conclude that researchers can improvise the NN technique based on their research [19]. Ahalya and Pandey analyzed various clustering algorithms such as the K-means algorithm, Hierarchical algorithm, self-organizing map (SoM) algorithm, and expectation maximization(EM) algorithm. They described some tools used to implement the clustering algorithms. They compared and analyzed the above algorithms based on the type and size of the data set, the number of clusters, implementation tools, and accuracy. They reported that k-means and EM give better performance than other approaches and the SoM algorithm gives more accuracy than others [20].

MaryemAit El Hadj, Mohammed Erradi, and their team proposed an approach to cluster the security policies and additionally used the information of access log to detect the fraud intruders. They used the KNN algorithm to cluster the security rules and applied the rule-sub-module-reduction technique to minimize the count of rules in each cluster [21]. MaryemAit El Hadj and his team proposed a clustering approach to cluster XACML (eXtensible Access Control Markup Language) policies. XACML is an efficient markup language to express ABAC policies. They cluster the rules based on their similarity. Every pair of rules is clustered if the similarity value of the pair or rules is greater than the threshold value of 0.8 [23]. This method ensures that each rule must be clustered once, and the same rule may be contained in more than one cluster. There must be non-empty clusters only. The above approach may produce a maximum number of clusters and clusters with a maximum number of rules [22]. In contrary to the above research, we proposed and implemented a novel approach that our approach uses the basic technique of hierarchical clustering algorithm [24]. Rather than using the distance between the new rule (data point) and the cluster center, we use the similarity value of the new rule and the existing rules in the previous cluster. We use a rule-specific-cluster-merging approach instead of using the rule-sub-module-reduction method. Thus if the similarity value of two rules Rule-1 and Rule-2 is above the threshold value, then we create a new cluster if and only if Rule-1 and Rule-2 are new rules that are not yet clustered. If Rule-1 (or Rule-2) is already clustered and Rule-2 (or Rule-1) is not clustered and matched with all rules of the existing cluster, which contains Rule-1 (or Rule-2), then Rule-2 (or Rule-1) is merged with that existing cluster. Our enhanced approach reduces the generation of more clusters, redundancy (the same rule is contained in many clusters), and avoids the conflict-clusters (not all rules in a clusters are similar). Table II summarizes our approach and the researches done in previously.

TABLE II. SUMMARIZATION OF PREVIOUS RESEARCHES TOWARDS CLUSTERING INCLUDING THIS PAPER

References	Proposed work
[19]	Surveyed Nearest Neighbor (NN) clustering techniques. The nearest neighbor is identified by the calculated value. Concluded that the NN technique is efficient, simple and robustness and researchers can use and improvise the NN techniques based on their research.
[20]	Compared and analyzed various clustering algorithms such as the K-means algorithm, Hierarchical algorithm, self-organizing map (SoM) algorithm, and expectation maximization(EM) algorithm based on the type and size of the data set, the number of clusters, implementation tools, and accuracy. Reported that k-means and EM give better performance than other approaches and the SoM algorithm gives more accuracy than others.
[21]	Proposed an approach to cluster the security policies using the KNN algorithm and additionally used the information of access log to detect the fraud intruders. Applied rule-sub-module-reduction technique to minimize the count of rules in each cluster.
[22]	Proposed a clustering approach to cluster XACML (eXtensible Access Control Markup Language) policies based on the similarity value. Pair of rules are clustered if they are similar rules. This approach generates an increased number of clusters and also the cluster contains an increased number of rules. A rule may be contained in more than one cluster (redundancy) and not all the rules in a cluster are similar (conflict-cluster).
This paper	Proposed novel approach that uses the basic technique of hierarchical clustering algorithm. We use a rule-specific-cluster-merging approach instead of using the rule-sub-module-reduction method. Our enhanced approach reduces the generation of more clusters, avoidable redundancy (the same rule is contained in many clusters), and avoids the conflict clusters (not all rules in clusters are similar).

III. PRELIMINARIES

The fundamental concept of the ABAC model is described in this section. This also section describes how the ABAC rules are expressed and how the similarity value of pair of rules is calculated.

A. ABAC Model

This section describes the basic concepts of the ABAC model and the simple authentication process. The common simple authentication process is making a decision (allow or deny) by using the information like username and password of the subject or user for the request to access the shared resources such as files, databases, software. This simple process of checking the identity of the subject is not sufficient to meet the security needs of today's emerging computing technologies. RBAC and ABAC are the models used to address the complex security requirements of new computing technologies like cloud and fog computing. While RBAC fails to generate dynamic mappings between subject and object and also it is a role-independent model, ABAC or the combined features of RBAC and ABAC can be used for the protection of shares resources [25]. In this paper, we use the ABAC model and we aim to detect and eliminate the anomalies in ABAC security policies. This paper proposes an enhanced approach for clustering the security policies to simplify the process of detection and removal of anomalies. Fig. 2 and Fig. 3 show the simple standard authentication system and the ABAC model.

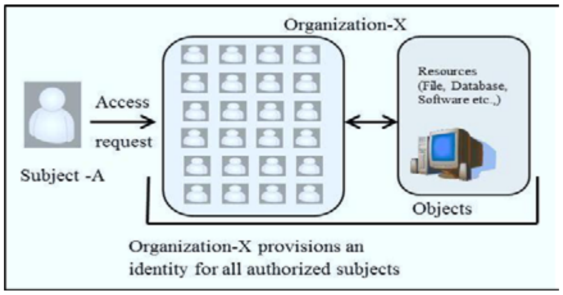

Organization-X
Access
request
Subject -A
Resources
(File, Database, Software etc.,)
Objects
Organization-X provisions an identity for all authorized subjects

Fig. 2 Traditional Authentication System.

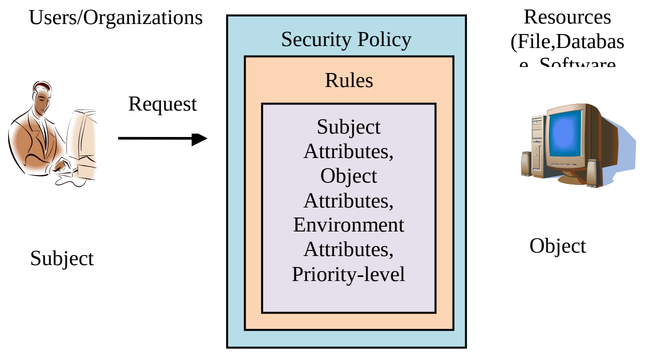

Users/Organizations
Request
Subject
Security Policy
Rules
Subject Attributes, Object Attributes, Environment Attributes, Priority-level
Resources (File,Databas e, Software
Object

Fig. 3 The Architecture of ABAC Moel.

ABAC model has a set of determined security policies where each security policy consists of security rules. In general, the security rule of the ABAC model is expressed with the decision (allow or deny), operations (read, write, print, etc.), and the attributes of subject (who requests the access), object (shared resources), and environment conditions (time, the importance of the operation, etc.). The decision either allowing or denying the subject's request is made based on the attributes of the subject, object, and environmental conditions specified in the rules. In our research work additionally, we added one parameter Priority-level to each security rule to avoid the demand on a single object. The priority level is assigned based on the subject's attributes and the importance of the operation on the object. The most common jargons used in the ABAC model are:

1) *Subject and its attributes:* The term subject refers to a single or group of users or organizations or the process that requests the resource. The attributes of subjects are the important credential information about the subject like name of user or process, designation, department, affiliation, etc.

2) *Object and its attributes*: Shared resources (operating system, network, file, software, and database) are named as objects. The information like name, type, owner, and date of creation are the attributes of objects.

3) Operation: The task (read, write, etc.) on the object is requested.

4) Environmental conditions: Other information such as date of request, current time, waiting period, etc.

5) Rule and decision: Constructed with the above three categories subject, object, and environmental conditions. In our proposed approach, we included the additional parameter priority-level assigned based on the attributes of the subject. The rule takes the decision (allow or deny) based on the attributes of the subject, object, and environmental conditions.

6) Policy: It is a set of rules established for the protection of objects in an organization.

B. Expression of ABAC Security Policy

The security policies of ABAC are constructed with a set of rules. Every rule contains the attributes of the subject, objects, and environment. The decision (allow or deny) is made based on the values of the attributes specified in the rule [26]. We added one additional parameter priority-level in every rule to avoid the anomaly conflict-demand. Thus more requests on a limited object are referred to as conflict-demand. The parameter priority-level is a non-negative integer. The demand for a limited object is handled based on the value of the priority level. Thus the rule or request which has the highest priority value will be allowed to get access. The pair of rules is not clustered if the priority-level of the two rules are not equal. The security policy set of the ABAC model is expressed by JavaScript Object Notation (JSON) as follows:

```
Policy-Set: [ {
Policy: <policyID>,
Rules: [{
<ruleID>,
Action: <actionName>,
Operation :<operationName>,
Subject: <subjectID>,
Resource: <resourceName>
Environmental :<environmentID>},
        }]
    }]
```

Extensible Access Control Markup Language (XACML) is the most common acceptable and widely used language to express the security policies. Each attribute is expressed with the pair key and value using a markup language. Key is the attribute name (eg. Department) specified as the tag and value (eg. Urology) is the value of the attribute name specified within the key-tag. The ABAC policy set can be expressed by XACML as follows:

```
<PolicySet>
 <Policy PolicyID=”P1”>
   <Rule RuleID=”R1” Decision=”Allow” >
     <Operation>
          <Operation-1>read</Operation-1>
          <Operation-2>write</Operation-2>
     </Operation>
     <Subject>
          <Department>urology</Department>
          <Designation>surgeon<Designation>
     </Subject>
     <Object>
                    <ResourceName>

   PatID_005_Urine_Culture_Report
          </ ResourceName >
     </Object>
     <EnvironmentalCondition>
          <Duration>8:18</Duration>
     </EnvironmentalCondition>
   </Rule>  ………// more rules can be specified
 </Policy>   ………// more policies can be specified
</PolicySet>
```

In the above example, rule R1 states that security policy allows the surgeon in the department of urology to read and write the file ‘PatID_005_Urine_Culture_Report’ during the time 8:18 hours. The following standard is usually used to write the rules.

Rule$_1$= {Allow$_{read}$ | Designation = {duty-doctor, surgeon}, Department = { hematology}, File-Name = {pat_007_blood_report}, Time= {8:18}, PriorityLevel=2}

Rule$_2$ = {Allow$_{read}$ | Designation = {head-nurse}, Department = {hematology}, File-Name = { pat_007_blood_report }, Time= {8:18}, PriorityLevel=1}

The above rules Rule$_1$ and Rule$_2$ are security rules represented including the parameter PriorityLevel. The attributes Designation and Department are the categories of subject, File-Name is the attribute of category object, and the attribute Time is the category of environment. The rules state that the file ‘pat_007_blood_report’ can be read by the users ‘duty doctor, ‘surgeon’ and ‘head nurse’ belong to the department ‘hematology’. When all these three subjects are trying to get access to the file, ‘duty doctor, ‘surgeon’ will get access first due to the highest priority. The relationship diagram of security policy and working principle of ABAC model is shown in Fig. 4. This security system provides two stages of authorization. The first stage performs the common traditional authorization process where the second stage completes the ABAC mechanism to increase the level of protection.

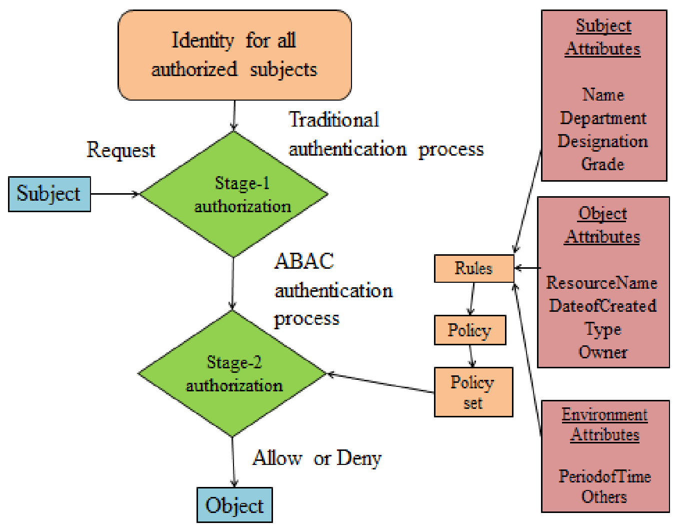


Fig. 4 The Security System Provides Two Stages of Authorization.

C. *Measurement of Similarity Value*

We used the following formulae to measure the similarity value of the pair of rules[27]. The similarity value (S) of the pair of rules R_1 and R_2 for an attribute ‘atr’ is calculated using formula-1.

$$S_{atr}\ (R_1, R_2) = \frac{NSV}{NDV} \qquad (1)$$

Where NSV (Number of Same Values) is the number of the same values of the attribute ‘reappeared in both the rules R_1 and R_2. The variable NDV (Number of Distinct Values) is the number of distinct values of the attribute ‘atr’ in the rules R_1 and R_2. The similarity-value (S) of two rules for category C (Subject, Object, and Environment) is measured using the formula-2.

$$S_C\ (R_1, R_2) = \sum_{atr\, \in\, \{ATS\,(R1)\, \cap\, ATS\,(R2)\}} P_{atr}\, S_{atr}\ (R_1, R_2) \qquad (2)$$

where ATS(R_1) and ATS(R_2) are a set of attributes of R_1 and R_2, respectively. ‘P_{atr}’ is the probability of the attribute (P_{atr} =1/number of the common attribute in both R1 and R2). The variable ‘atr’ is the set of attributes that appeared in both R1 and R2 under the category C. The similarity-value (S) of the pair of the rule is measured by the formula-3.

$$S\ (R_1, R_2) = \left\{ \begin{array}{l} P_{subject}\, S_{subject}(R_1, R_2) + \ P_{object}\, S_{object}(R_1,R_2) + \\ P_{environment}\, S_{environment}(R_1,R_2) \end{array} \right. \qquad (3)$$

The variable P is the probability of the category (subject, object, and environment). As equal probability is applied, the probability of each category is 1/3. The similarity value of the above-mentioned rules $Rule_1$ and $Rule_2$ is calculated as,

$$\boxed{\begin{array}{l} S(Rule1,Rule2) = \frac{1}{3}S_{subject}\ (R_1,\ R_2) + \frac{1}{3}S_{object}(R_1,R_2) + \\ \qquad \frac{1}{3}S_{environment}(R_1,R_2) \\ \qquad = \frac{1}{\ }\times (0.25+0.5) + \frac{1}{\ }\times 1 + \frac{1}{\ }\times 1 \end{array}}$$

IV. System Architecture

Fig. 5 shows the system architecture of our proposed method. It contains the rules-clusters storage and management module, rule generation module, and rule clustering module. The functional operation, task, requirements of each module, and relationship among the modules are described in the following sections.

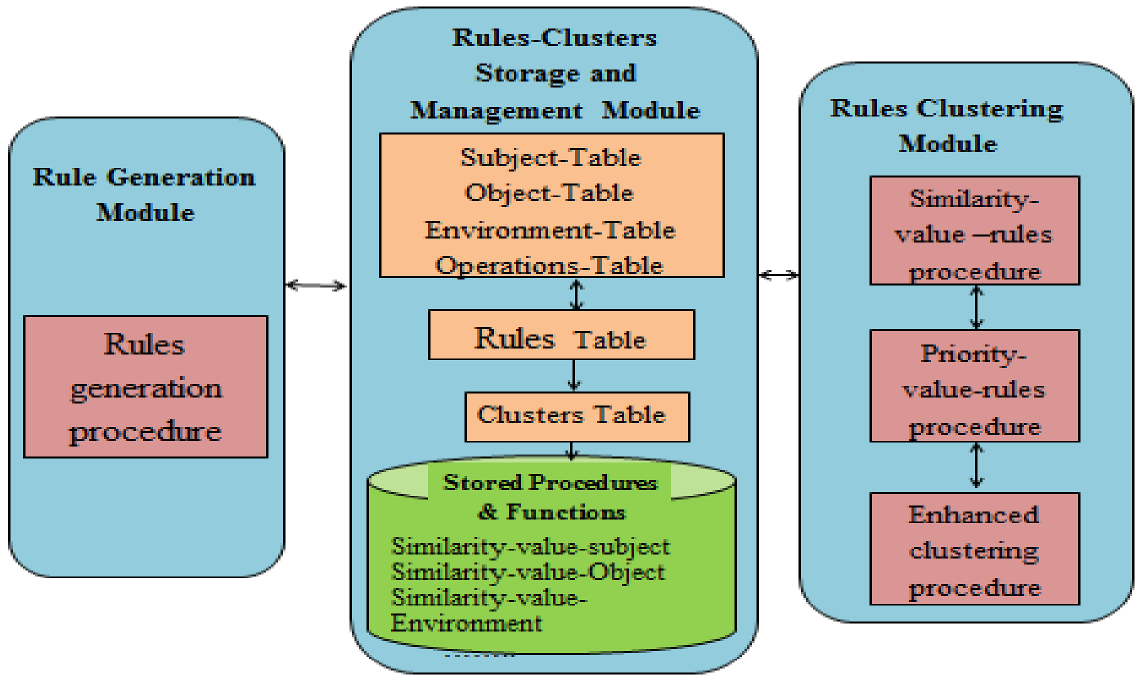

Rule Generation Module
Rules generation procedure
Rules-Clusters Storage and Management Module
Subject-Table
Object-Table
Environment-Table
Operations-Table
Rules Table
Clusters Table
Stored Procedures & Functions
Similarity-value-subject
Similarity-value-Object
Similarity-value-Environment
Rules Clustering Module
Similarity-value –rules procedure
Priority-value-rules procedure
Enhanced clustering procedure

Fig. 5 The System Architecture of Our Approach.

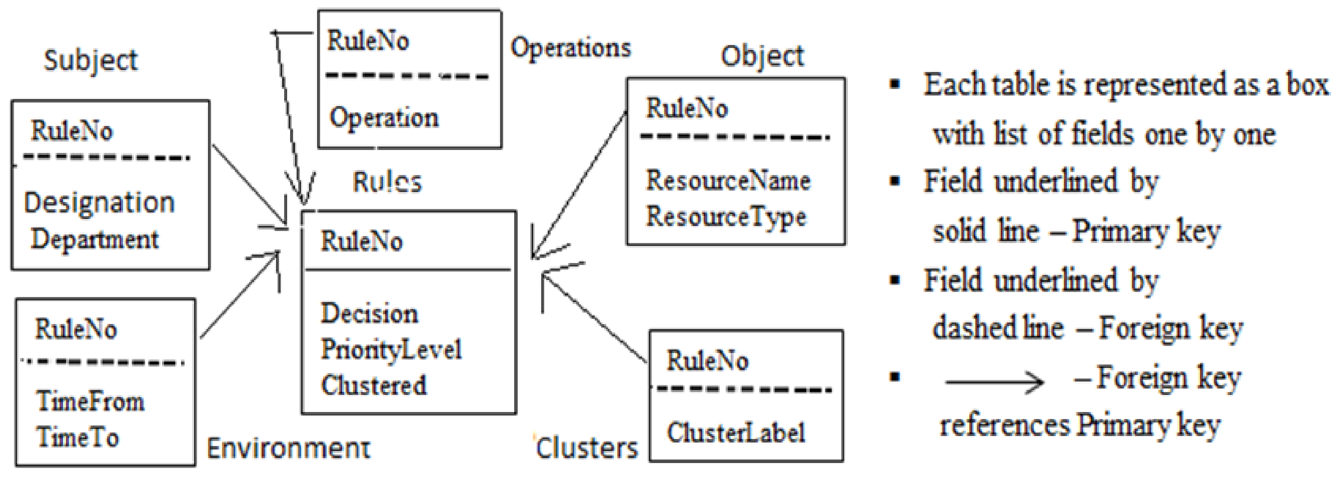


Fig. 6 Schema Diagram of Specification of ABAC Policies and Clusters.

A. *Rules-Clusters Storage and Management Module (RCSMM)*

We created relations (tables) in Oracle11 (Oracle 12c is for an in-memory database environment). The schema diagram of RCSMM is shown in Fig. 6. The relation 'Subject' is used to store and manage the attributes and values of the subject. Likely the relation 'Object' and the relation 'Environment' store and manage the attributes of the resources and the environmental conditions respectively. The generated security policies are stored and maintained in the table 'Rules'. The information about the request of operation for a resource is maintained in the relation 'Operations'. The generated security rules are clustered and stored in the relation 'Clusters'. Table III, Table IV, Table V, Table VI show the sample records of the relations Subject, Object, Environment, and Clusters.

Table VI illustrates that rules 1 and 2 are clustered in cluster-1 and cluster-11 consists of rules 32 and 33. We created a relation for each category (Subject, Object, and Environmental Conditions) to store and manage the value of all attributes. We have written stored functions to measure the similarity value of pair of rules for an attribute (e.g. Department), the similarity value of pair of rules for each category (e.g. Subject) and finally to measure the similarity value of pair of rules.

TABLE III. ATTRIBUTES OF SUBJECT

Relation: Subject		
RULENO	RULENO	RULENO
9	9	9
35	35	35
501	501	501
720	720	720

TABLE IV. ATTRIBUTES OF OBJECT

Relation: Object	
RULENO	RESOURCENAME
9	Pat00005_blood_report
35	Pat00007_urine_report
501	Pat00008_thyroid_report
720	Pat00039_blood_report

TABLE V. ATTRIBUTES OF ENVIRONMENT

Relation: Environment		
RULENO	TIMEFROM	TIMETO
9	8	18
35	6	14
501	8	18
720	7	19

TABLE VI. CLUSTERS OF RULES

Relation: Clusters	
RULENO	CLUSTERLABEL
1	1
2	1
32	11
33	11

The above-stored function measures the similarity value of pair of rules for the attribute Department. The stored function call nsv_department(r1,r2) returns the number of values that are the same for the attribute 'Department' in both r1 and r2. The stored function call ndv_department(r1,r2) returns the number of distinct values for the attribute 'Department' in both r1 and r2.

Stored function sv_department(r1 number,r2 number)

```
create or replace function sv_department(r1 number, r2 number)
return float is
sv float;
nsv number(3);
ndv number(3);
begin
nsv:=nsv_department(r1,r2);
ndv:=ndv_department(r1,r2);
if nsv=0 or ndv=0 then
sv:=0;
else
sv:=nsv/ndv;
end if;
return sv;
end sv_department;
```

We have created stored functions to measure the similarity values of all attributes. The stored function sv_rules(r1,r2) measures the final similarity value of two rules r1 and r2. In this procedure p is the probability where we use equal probability of all the categories (Subject, Object, and Environment) and sv_subject(r1,r2), sv_object(r1,r2), sv_environment(r1,r2) are the similarity values of Subject, Object and Environment.

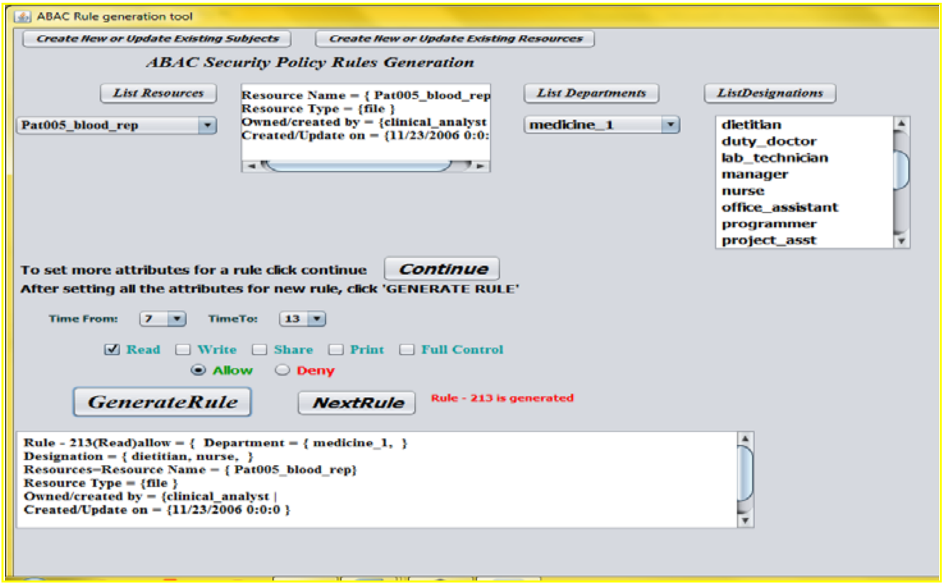
Stored procedure function sv_rulet(r1 number,r2 number)

```
create or replace function sv_rule(r1 number,r2 number)
return float is
sv float;
p float;
sv1 float;
sv2 float;
sv3 float;
begin
p:=1/3;
sv1:=sv_subject(r1,r2);
sv2:=sv_object(r1,r2);
sv3:=sv_environment(r1,r2);
sv:=(p*sv1)+(p*sv2)+(p*sv3);
return sv;
end sv_rule;
```

B. Rule Generation Module (RGM)

We designed the module RGM in java for automated rule generation. RGM is an admin interface in java to generate or specify the security policies for all shared resources. The admin can easily set the rules for a resource with the attributes of the categories Subject, Object, and Environment. This module is flexible for the admin to update the rules easily. This module allows the admin to create, or delete or update the entities of the subject, object, and environmental conditions. The priority of the rule is determined based on the role of the subject. The subject-attribute 'grade' is assigned with the value (low or high or middle) based on the attribute designation. The priority of the rule or request is determined based on the attribute 'grade'. We have written a rule generation procedure in Java to generate ABAC rules.

Fig. 7 shows the interface of rule generation. The generated policies are stored in the relations Rules, Subject, Object, Environment, and Operations. Fig. 8 shows the sample of generated rules.


ABAC Rule generation tool
Create New or Update Existing Subjects
Create New or Update Existing Resources
ABAC Security Policy Rules Generation
List Resources
List Departments
ListDesignations
Continue
GenerateRule
NextRule

Pat005_blood_rep

```
Resource Name = { Pat005_blood_rep
Resource Type = {file }
Owned/created by = {clinical_analyst
Created/Update on = {11/23/2006 0:0:
```

medicine_1

dietitian
duty_doctor
lab_technician
manager
nurse
office_assistant
programmer
project_asst

To set more attributes for a rule click continue
After setting all the attributes for new rule, click 'GENERATE RULE'

Time From: 7 TimeTo: 13

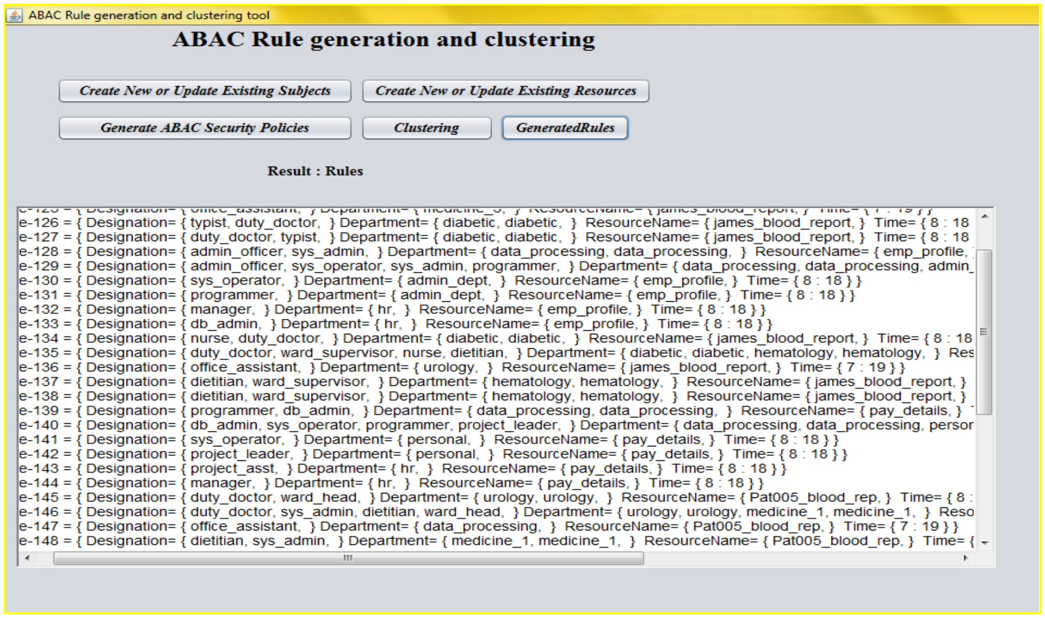
[✓ Read] Write Share Print Full Control

[✓ Allow] Deny

Rule - 213 is generated

```
Rule - 213(Read)allow = {  Department = { medicine_1,  }
Designation = { dietitian, nurse,  }
Resources=Resource Name = { Pat005_blood_rep}
Resource Type = {file }
Owned/created by = {clinical_analyst |
Created/Update on = {11/23/2006 0:0:0 }
```

Fig. 7 ABAC Rule Generation Tool.


ABAC Rule generation and clustering tool
ABAC Rule generation and clustering
Create New or Update Existing Subjects
Create New or Update Existing Resources
Generate ABAC Security Policies
Clustering
GeneratedRules
Result : Rules

```
e-126 = { Designation= { typist, duty_doctor,  } Department= { diabetic, diabetic,  }  ResourceName= { james_blood_report, }  Time= { 8 : 18
e-127 = { Designation= { duty_doctor, typist,  } Department= { diabetic, diabetic,  }  ResourceName= { james_blood_report, }  Time= { 8 : 18
e-128 = { Designation= { admin_officer, sys_admin,  } Department= { data_processing, data_processing,  }  ResourceName= { emp_profile,
e-129 = { Designation= { admin_officer, sys_operator, sys_admin, programmer,  } Department= { data_processing, data_processing, admin_
e-130 = { Designation= { sys_operator,  } Department= { admin_dept,  }  ResourceName= { emp_profile, }  Time= { 8 : 18 } }
e-131 = { Designation= { programmer,  } Department= { admin_dept,  }  ResourceName= { emp_profile, }  Time= { 8 : 18 } }
e-132 = { Designation= { manager,  } Department= { hr,  }  ResourceName= { emp_profile, }  Time= { 8 : 18 } }
e-133 = { Designation= { db_admin,  } Department= { hr,  }  ResourceName= { emp_profile, }  Time= { 8 : 18 } }
e-134 = { Designation= { nurse, duty_doctor,  } Department= { diabetic, diabetic,  }  ResourceName= { james_blood_report, }  Time= { 8 : 18
e-135 = { Designation= { duty_doctor, ward_supervisor, nurse, dietitian,  } Department= { diabetic, diabetic, hematology, hematology,  }  Res
e-136 = { Designation= { office_assistant,  } Department= { urology,  }  ResourceName= { james_blood_report, }  Time= { 7 : 19 } }
e-137 = { Designation= { dietitian, ward_supervisor,  } Department= { hematology, hematology,  }  ResourceName= { james_blood_report, }
e-138 = { Designation= { dietitian, ward_supervisor,  } Department= { hematology, hematology,  }  ResourceName= { james_blood_report, }
e-139 = { Designation= { programmer, db_admin,  } Department= { data_processing, data_processing,  }  ResourceName= { pay_details, }
e-140 = { Designation= { db_admin, sys_operator, programmer, project_leader,  } Department= { data_processing, data_processing, persor
e-141 = { Designation= { sys_operator,  } Department= { personal,  }  ResourceName= { pay_details, }  Time= { 8 : 18 } }
e-142 = { Designation= { project_leader,  } Department= { personal,  }  ResourceName= { pay_details, }  Time= { 8 : 18 } }
e-143 = { Designation= { project_asst,  } Department= { hr,  }  ResourceName= { pay_details, }  Time= { 8 : 18 } }
e-144 = { Designation= { manager,  } Department= { hr,  }  ResourceName= { pay_details, }  Time= { 8 : 18 } }
e-145 = { Designation= { duty_doctor, ward_head,  } Department= { urology, urology,  }  ResourceName= { Pat005_blood_rep, }  Time= { 8 :
e-146 = { Designation= { duty_doctor, sys_admin, dietitian, ward_head,  } Department= { urology, urology, medicine_1, medicine_1,  }  Reso
e-147 = { Designation= { office_assistant,  } Department= { data_processing,  }  ResourceName= { Pat005_blood_rep, }  Time= { 7 : 19 } }
e-148 = { Designation= { dietitian, sys_admin,  } Department= { medicine_1, medicine_1,  }  ResourceName= { Pat005_blood_rep, }  Time= {
```

Fig. 8 Sample Generated Rules.

C. Rule Clustering Module (RCM)

We implemented our enhanced clustering algorithm in java to cluster the security policies. In our previous work [28] [29], we applied a direct clustering approach that every rule R_i (from top to bottom) is paired with the rule R_j (from the next successive rule of Ri to the last rule in the database). The pair of rules are clustered if the similarity value of the pair of rules is above the threshold value and the difference between the priority-level of those rules is zero (priority-level of two rules are equal). The introduction of the parameter priority-level reduces the size of the clusters [30].

Although the previous approach clusters the rules efficiently, it produces more clusters, and the same rule is clustered in multiple clusters. In our enhanced approach, we follow the top-to-bottom approach thus we start from the first rule to last (1,2,3,..,n) and for each rule R_i we made the pairing of the rule R_i with every R_j (j=i+1,i+2,..,n}. If two rules R_i and R_j are similar rules based on the condition (similarity-value(R_i, R_j)>0.8 and priority-level(R_i) is equal to priority-level(R_j)), then we follow the following criteria to cluster the pair of rules R_i and R_j.

- The rules R_i and R_j are stored in a new cluster if R_i is not yet clustered. (first similar rule of R_i)
- In the case of the rule, R_i is already clustered in C_k, and R_j is not clustered: (rule-specific-cluster-merging technique)
  - the rule R_j is merged with the cluster C_k if R_j is similar to all the rules in the cluster C_k,
  - otherwise, R_j and R_i are stored in a new cluster.
- Every rule should be clustered, and any cluster should not be empty.

The enhanced clustering algorithm is written as follows:

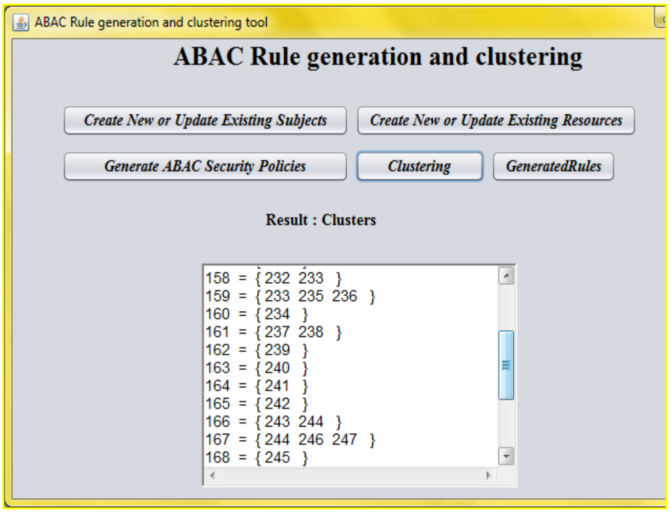
Algorithm: Rule-Specific-Cluster-Merging Approach (RSCA)

```
Input : Security rules $R_1$, $R_2$,…, $R_n$ , // n is number of rules
Output: Cluster of Rules $C_1$, $C_2$,.., $C_k$ // k is the number of clusters
1.  K=0; // K is the number of clusters
2.  T={$R_1$, $R_2$,……$R_n$} /* T is the table of records and each record
    $R_i$ contains information of single rule */
3.  For each rule $r_1$ in $R_1$ to $R_n$ loop   //form first record to last
    record
4.      For each rule $r_2$ in $R_{i+1}$ to $R_n$ loop
5.          Compute Similarity-value($r_1$ , $r_2$)
6.          If Similarity-value ($r_1$ ,$r_2$)>0.8 and priority-level($r_1$) =
              priority-level($r_2$) then
7.              If $r_1$ is not yet clustered then
8.                  K=K+1; $C_k$={ $r_1$, $r_2$}
9.              Else if $r_1$ is already clustered and $r_2$is not clustered
                then
10.                 Found the cluster $C_F$ such that $r_2$ is similar to
                    all the rules in $C_F$ where $r_1$ is the member of
                    this cluster
11.                 If such cluster $C_F$ is found then
12.                     $C_F$= $C_F$ U {$r_2$}
                    // is merged with the existing cluster.
13.                 Else If such cluster $C_F$ is not found then
14.                     K=K+1;$C_k$={ $r_1$, $r_2$}
                    // new cluster is created to store $r_1$ and $r_2$
15.                 End If
16.             End If
17.         End If
18.     End Loop
19.     If $r_1$ is not yet clustered then
          // none of the rules is similar to $r_1$
20.         K=K+1;$C_k$={ $r_1$} // $r_1$ is clustered in a new cluster
21.     End If
22. End Loop
23. End
```

The implementation of our enhanced clustering approach clusters the given set of rules. Fig. 9 shows the results of our implementation. We managed six tables Rules, Operations, Subject, Object, Environment, and Clusters to store the detail of rules, operations on objects, subjects, objects, environments, and clusters respectively. The creation of an individual table for every entity avoids the redundancy of data and null values.

Rule clustering module (RCM) proposes an approach to cluster ABAC Policies before policy validation to improve the performance of the model (Resolving the policy errors in every rule is a time-consuming and complex process). We used only the basic technique of the hierarchical clustering algorithm. We introduce a novel clustering (rule-specific-cluster-merging) and resolved the two major problems of clustering: avoidable-redundancy (the same rule is contained in many clusters), and avoids the conflict clusters (not all rules in clusters are similar). The scopes of our approach are:

- We used a rule-specific-cluster-merging approach (instead of using the rule-sub-module-reduction method or cluster-merging method). (time complexity is low).
- If two rules are already clustered, our approach will not entertain the clustering once again (reduces the number of clusters and a rule is not stored in multiple clusters unnecessarily).
- If a pair of rules are similar and anyone rule is already clustered, then we are seeking to cluster the next one with the specific clusters where the clustered rule is a member. (improves the performance of the process).
- A rule is merged with the existing cluster if all the rules of the existing clusters are similar only (avoids conflict-clusters-not all rules in a cluster are similar).


ABAC Rule generation and clustering tool
ABAC Rule generation and clustering
Create New or Update Existing Subjects
Create New or Update Existing Resources
Generate ABAC Security Policies
Clustering
GeneratedRules
Result : Clusters

```
158 = { 232 233 }
159 = { 233 235 236 }
160 = { 234 }
161 = { 237 238 }
162 = { 239 }
163 = { 240 }
164 = { 241 }
165 = { 242 }
166 = { 243 244 }
167 = { 244 246 247 }
168 = { 245 }
```

Fig. 9 Sample Result of Clustering.

V. Results and Discussion

We described and compared our approach with the hierarchical clustering algorithm. Fig. 10 illustrates the example of the hierarchical clustering algorithm.

We took eight rules {R1,R2,R3,R4,R5,R6,R7,R8} for clustering. In the first step, the hierarchical algorithm considers each rule as a cluster. At each iteration, every cluster is compared with other clusters and the similar clusters are merged. This is a continuous process while there are comparable clusters. Let we consider.

C is the cluster to be compared,

k is the number of current clusters,

k-1 is the number of clusters excluding C,

r1 is the number of rules in the i^{th} cluster ($1\leq i\leq k-1$) and

r2 is the number of rules in C.

The comparison of every cluster C, this approach performs k-1 × r1 × r2 iterations. Thus each cluster is compared with all other clusters at every iteration and similar clusters are only merged. The clusters {R1} and {R2} are merged. The cluster {R3} and {R6} are not similar with all other clusters. The clusters {R1} and {R4} are merged. The clusters {R5}, {R7} and {R8} are merged in a cluster. The generated clusters are {R1,R2}, {R3}, {R1,R4}, {R5,R7,R8}, and {R6}.

Fig. 11 illustrates the same example of clustering eight rules with our proposed approach. In the first step of our approach, we compare every rule R_i ($1\leq i\leq n-1$, n is the number of rules) with Rj ($i+1 \leq j \leq n$). At each iteration, the un-clustered rule is compared only with the specific clusters, not all the clusters. Thus if R1 and R2 are similar rules, R1 is clustered and R2 is yet to clustered, then R2 is compared with the clusters that contain R1. If k is the number of specific clusters (where the feasibility of similarity occurs for the un-clustered rule) and t is the number of rules in the k^{th} cluster, then our approach requires k×t comparisons only. Hence our approach reduces the number of comparisons and consumes less time than the other approaches. For the discussion of our proposed approach, we took five rules as a sample (listed in Table VII) to make the discussion easy and understandable.

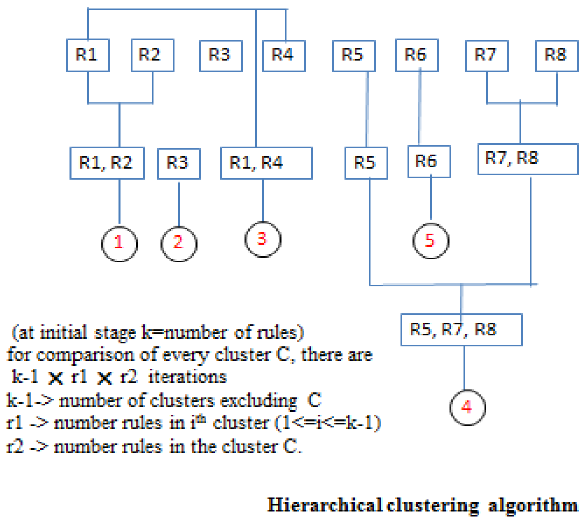

R1
R2
R3
R4
R5
R6
R7
R8
R1, R2
R3
R1, R4
R5
R6
R7, R8
1
2
3
5
R5, R7, R8
4
(at initial stage k=number of rules)
for comparison of every cluster C, there are
k-1 × r1 × r2 iterations
k-1-> number of clusters excluding C
r1 -> number rules in ith cluster (1<=i<=k-1)
r2 -> number rules in the cluster C.
Hierarchical clustering algorithm

Fig. 10 Example of the Hierarchical Clustering Approach.

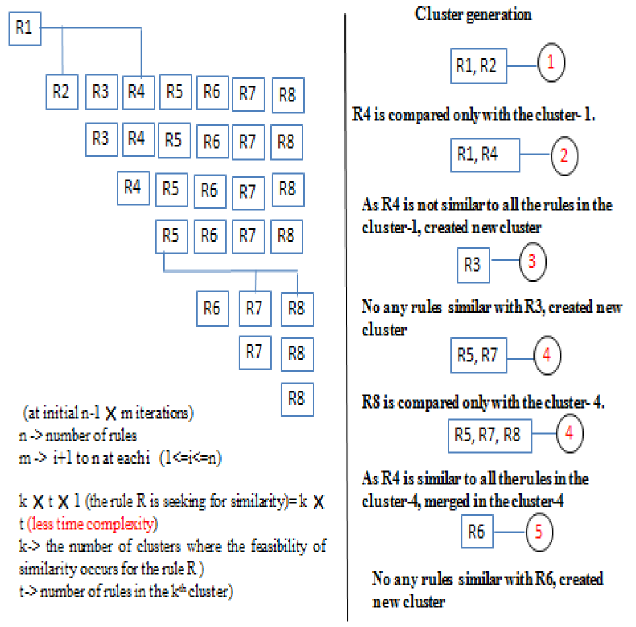

R1
R2 R3 R4 R5 R6 R7 R8
R3 R4 R5 R6 R7 R8
R4 R5 R6 R7 R8
R5 R6 R7 R8
R6 R7 R8
R7 R8
R8
(at initial n-1 X m iterations)
n -> number of rules
m -> i+1 to n at each i (1<=i<=n)
k X t X 1 (the rule R is seeking for similarity)= k X t (less time complexity)
k-> the number of clusters where the feasibility of similarity occurs for the rule R)
t-> number of rules in the kth cluster)
Cluster generation
R1, R2 1
R4 is compared only with the cluster- 1.
R1, R4 2
As R4 is not similar to all the rules in the cluster-1, created new cluster
R3 3
No any rules similar with R3, created new cluster
R5, R7 4
R8 is compared only with the cluster- 4.
R5, R7, R8 4
As R4 is similar to all the rules in the cluster-4, merged in the cluster-4
R6 5
No any rules similar with R6, created new cluster

Fig. 11 Example of Our Proposed Clustering Approach.

The similarity value of each pair of rules are (1,2)= 0.83, (1,3)= 0.33, (1,4)= 0.33, (2,3)= 0.33, (2,4)=0.83, (2,5)=0.83, (3,4)=0.33, (3,5)=0.33, (4,5)=1. The workflow and result of the above sample rules are described in Table VIII. Our novel clustering approach produces three clusters C1={2,5,4}C2={1,2} C3={3} for the above five rules. In our proposed approach, all rules in every cluster are similar. Thus a rule is merged if it is similar to all the rules already exist in that cluster. In some previous researches, the number of clusters is very high. Also, only minimum rules are similar in a cluster, and this leads to complexities in detecting and removing anomalies [22]. Unlike the rule-sub-module-reduction method [21], we merge the cluster at the time of the creation of the cluster itself, which increases the performance of the approach. In our previous work [28], we introduced the parameter priority-level to avoid the anomaly conflict-demand and reduce the more number of clusters. Although our previous research reduced the number of clusters, this enhanced cluster with the cluster-merging technique decreases the creation of more clusters and also maintains the constraints that all rules in a cluster are similar only.

Fig. 12 illustrates the comparison of our novel rule-specific clustering approach (RSCA) with the previous approaches 'Cluster-based approach' (CBA) and 'Log-based clustering approach' (LBCA) [21][22] (Maryem Ait El Hadj, Mohammed Erradi). CBA generates more clusters than LBCA and RSCA. Comparing to the hierarchical clustering algorithm(HCA) and other discussed previous approaches, our proposed clustering approach reduces the generation of more clusters and the number of comparisons, hence this result shows that our proposed approach consumes less time and increases the performance of the clustering technique and this helps to improve the efficiency of the ABAC model. Fig. 13 gives the comparative analysis of the existing approaches with our proposed approach based on the time complexity (Time is represented as Nanoseconds).

Table IX shows the qualitative analysis of clustering approaches. We used rule-redundancy-clusters (the same rule is clustered in multiple clusters), conflict-clusters (not all the rules in a cluster are similar), high-cluster-generation (more number of clusters are generated), and less time-consuming (measured based on the number of comparisons and iterations) to perform this qualitative analysis.

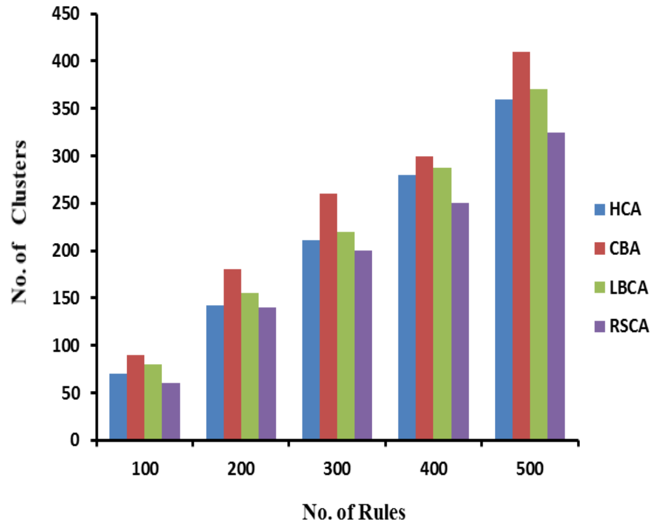


Fig. 12 A Comparative Study based on the Size and Number of Clusters.

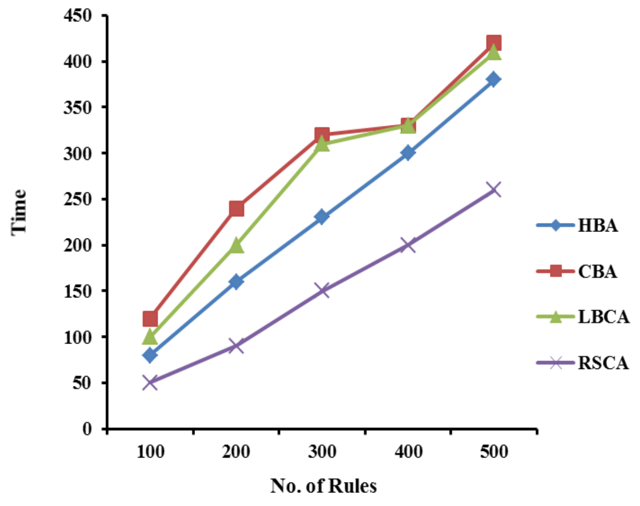


Fig. 13 A Comparative Study based on Time Complexity.

TABLE VII. SAMPLE ABAC RULES

RuleNo	Decision	PriorityLevel	Designation	Department	ResourceName	Time
2	$allow_{read}$	1	db_admin, typist, sys_asst, duty_doctor	diabetic, hematology	james_blood_report	8:18
3	$allow_{read}$	0	Office assistant	medicine_3	james_blood_report	7:19
5	$allow_{read}$	1	typist	diabetic	james_blood_report	8:18
1	$allow_{read}$	1	sys_asst, db_admin	hematology	james_blood_report,	8:18
4	$allow_{read}$	1	typist, duty_doctor	diabetic	james_blood_report	8:18

TABLE VIII. RESULT OF OUR PROPOSED CLUSTERING APPROACH WITH A SAMPLE OF THE ABOVE FIVE SECURITY RULES

Pair of rules	Clusters generated	Discussion
(2,3) (2,5) (2,1) (2,4)	C1={2,5} C2={2,1} C1={2,4,5}	(2, 3) is not similar. (2,5) is similar and clustered in C1. (2,1) is similar and tried to merge with the existing clusters (C1) where rule 2 is a member. But the rule 1 is not similar to all the rules in C1, so (2,1) is stored in a new cluster C2={2,1}. As (2,4) is similar and matched with all the rules in the existing cluster C1where rule 2 is a member, and rule 4 is merged with C1. C1={2,5,4}
(2,3) (2,4) (2,5)	C1={2,5,4} C2={1,2}	(2, 3) is not similar. (2,4) and (2,5) are similar pairs but already both are clustered
(3,4) (3,5)	C1={2,5,4}C2={1,2} C3={3}	(3,4) and (3,4) are not similar pairs. Hence 3 is clustered in new cluster C3={3}
(4,5)	C1={2,5,4}C2={1,2} C3={3}	(4,5) is similar but already it is clustered.

TABLE IX. QUALITATIVE ANALYSIS OF EXISTING ABAC POLICIES CLUSTERING AND OUR APPROACH

References	Technique	Reduces rule-redundancy-clusters	Avoids Conflict-clusters	Reduces high cluster-generation	Less time-consuming
M. A. El Hadj et al. [22]	A rule is clustered with all similar rules, comparable clusters are merged.	No	No	Yes	No
M. A. El Hadj et al. [21]	Enhanced approach and used rule-merge sub-module clustering	No	No	Yes	No
Hierarchical Clustering [24]	No centroid. All elements are considered clusters. At each cycle, comparable clusters are merged	Yes	Yes	Yes	No
Our approach	Rule-specific-cluster-merging approach	Yes	Yes	Yes	Yes

VI. CONCLUSION

Cloud service providers use many access control models in the implementation of the intrusion detection system to secure big data and other shared resources in the distributed environment. ABAC model meets the security requirements of advanced computing technologies like cloud computing, fog computing, and the Internet of Things (IoT). Detection and Removal of Anomalies (DRA) in the policies improve the efficiency and reliability of the model. Applying DRA in every cluster of rules rather than in every rule enhances the performance of the approach highly. Existing DRA approaches have the inability to prove the efficiency of the model, due to applying the poor clustering approach or the approach which not properly detect and resolve all considerable anomalies. Our research contribution consists of three modules: 1) developing a tool to generate ABAC policies; 2) Storing and managing ABAC policies in the database; 3) applying a novel rule-specific-cluster-merging algorithm to cluster ABAC policies. In previous researches, clustering methods produce more clusters and cluster a rule in multiple clusters. The previous approaches generate conflict clusters thus not all the rules in a cluster are similar, which results that in increases the complexity of the approach and degrades the performance too. Our novel approach with the rule-specific-cluster-merging technique reduces the generation of more clusters, avoids the clustering of the same rule in multiple clusters and conflict clusters. With our RSCA algorithm, we decrease the number of comparisons. Hence our approach gives high performance and reduces the complexity in applying DRA. The additional parameter priority-level in every rule avoids the anomaly conflict-demand. This is only our part of research towards improving the performance and efficiency of the ABAC model. Our future work is to detect and resolve the rule redundancy at the time of clustering to decrease the complexity in the clustering mechanism. In the future, we will propose an approach to detect and resolve all considerable policy errors to improve the efficiency of the ABAC model and intrusion detection system.

REFERENCES

[1] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," J. Internet Serv. Appl., vol. 1, no. 1, pp. 7–18, 2010, doi: 10.1007/s13174-010-0007-6.

[2] P. B. Tarigan, "Encyclopedia of Cryptograpghy and Security, Second Edition, Springer," Journal of Chemical Information and Modeling, vol. 53, no. 9. pp. 1689–1699, 2013, doi: 10.1017/CBO9781107415324.004.

[3] C. Liang et al., "Intrusion detection system for the internet of things based on blockchain and multi-agent systems," Electron., vol. 9, no. 7, pp. 1–27, 2020, doi: 10.3390/electronics9071120.

[4] C. Prakash and S. Dasgupta, "Cloud computing security analysis: Challenges and possible solutions," Int. Conf. Electr. Electron. Optim. Tech. ICEEOT 2016, pp. 54–57, 2016, doi: 10.1109/ICEEOT.2016.7755626.

[5] M. Mukherjee et al., "Security and Privacy in Fog Computing: Challenges," IEEE Access, vol. 5, pp. 19293–19304, 2017, doi: 10.1109/ACCESS.2017.2749422.

[6] R. Zhang, H. Ma, and Y. Lu, "PT US CR," J. Syst. Softw., 2016, doi: 10.1016/j.jss.2016.12.018.

[7] E. Conrad, S. Misenar, and J. Feldman, "Domain 5: Identity and Access Management (Controlling Access and Managing Identity)," CISSP Study Guid., pp. 293–327, 2016, doi: 10.1016/b978-0-12-802437-9.00006-0.

[8] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, 2019, doi: 10.1186/s42400-019-0038-7.

[9] K. Vijayalakshmi and V. Jayalakshmi, "Shared Access Control Models for Big data : A Perspective Study and Analysis," I Pandian A.P., Palanisamy R., Ntalianis K. Proc. Int. Conf. Intell. Comput. Inf. Control Syst. Adv. Intell. Syst. Comput. vol 1272. Springer, Singapore. https//doi.org/10.1007/, 2021.

[10] A. Markandey, P. Dhamdhere, and Y. Gajmal, "Data access security in cloud computing: A review," 2018 Int. Conf. Comput. Power Commun. Technol. GUCON 2018, pp. 633–636, 2019, doi: 10.1109/GUCON.2018.8675033.

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

[12] S. Salloum, J. Z. Huang, and Y. He, "Random Sample Partition: A Distributed Data Model for Big Data Analysis," IEEE Trans. Ind. Informatics, vol. 15, no. 11, pp. 5846–5854, 2019, doi: 10.1109/TII.2019.2912723.

[13] R. S. Sandhu et al., "Role Based Access Control Models," IEEE, vol. 6, no. 2, pp. 21–29, 1996, doi: 10.1016/S1363-4127(01)00204-7.

[14] K. Soni and S. Kumar, "Comparison of RBAC and ABAC Security Models for Private Cloud," Proc. Int. Conf. Mach. Learn. Big Data, Cloud Parallel Comput. Trends, Prespectives Prospect. Com. 2019, pp. 584–587, 2019, doi: 10.1109/COMITCon.2019.8862220.

[15] M. B. and A. C. A. Nascimento, "Information Security Example Policy," "Uni-ARBAC A Unified Adm. Model Role-Based Access Control. Int. Publ., vol. 1, pp. 218–230, 2016, doi: 10.1007/978-3-319-45871-7.

[16] E. Franco and D. C. Muchaluat-saade, "ACROSS : A generic framework for attribute-based access control with distributed policies for virtual organizations," Futur. Gener. Comput. Syst., vol. 78, pp. 1–17, 2018, doi: 10.1016/j.future.2017.07.049.

[17] C. Morisset, T. A. C. Willemse, and N. Zannone, "A framework for the extended evaluation of ABAC policies," Cybersecurity, vol. 2, no. 1, 2019, doi: 10.1186/s42400-019-0024-0.

[18] M. Yahiaoui, A. Zinedine, and M. Harti, "Deconflicting Policies in Attribute-Based Access Control Systems," Colloq. Inf. Sci. Technol. Cist, vol. 2018-Octob, pp. 130–136, 2018, doi: 10.1109/CIST.2018.8596576.

[19] N. Bhatia and Vandana, "Survey of Nearest Neighbor Techniques," vol. 8, no. 2, pp. 302–305, 2010.

[20] 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.

[21] 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.

[22] 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.

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

[24] M. S. Yang and K. L. Wu, "A similarity-based robust clustering method," IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 4, pp. 434–448, 2004, doi: 10.1109/TPAMI.2004.1265860.

[25] M. Pratap, S. Sural, and J. Vaidya, "Managing attribute-based access control policies in a unified framework using data warehousing and in-memory database," Comput. Secur., vol. 86, pp. 183–205, 2019, doi: 10.1016/j.cose.2019.06.001.

[26] 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.

[27] 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.

[28] V. Vijayalakshmi, K. and Jayalakshmi, "A Priority-based Approach for Detection of Anomalies in ABAC Policies using Clustering Technique," no. Iccmc, pp. 897–903, 2020, doi: 10.1109/iccmc48092.2020.iccmc-000166.

[29] K. Vijayalakshmi and V. Jayalakshmi, "Analysis on data deduplication techniques of storage of big data in cloud," Proc. 5th Int. Conf. Comput. Methodol. Commun. ICCMC 2021, IEEE, pp. 976–983, 2021.

[30] K. Vijayalakshmi and V. Jayalakshmi, "Identifying Considerable Anomalies and Conflicts in ABAC Security Policies," Proc. Fifth Int. Conf. Intelligent Computing and Control Systems. ICICCS 2021, IEEE, pp. 1286–1293, 2021.