International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 15, No.1 (Feb-24)

http://dx.doi.org/10.12785/ijcds/150131

HyARBAC: A New Hybrid Access Control Model for Cloud Computing

Okba Houhou[1], Salim Bitam[2] and Ammar Hamida[3]

[1,2,3]*Department of Computer Science, University of Biskra, Biskra, Algeria*

Received 27 Feb. 2023, Revised 6 Jan. 2024, Accepted 21 Jan. 2024, Published 1 Feb. 2024

Abstract: Cloud computing has been among the most critical digital processes and storage technologies in recent years. Nevertheless, the cloud is faced with data security issues due to its vulnerability to weak access control. Access control guarantees accessing cloud data and resources only to authorized users; unauthorized users are then detected and prevented from retrieving these resources. There have been many models of access control presented and used in the cloud domain, such as Attribute Based Access Control (ABAC) and Role Based Access Control (RBAC); however, these schemes suffered from many limitations, such as their difficulty using contextual information such as time, location of the user, type of device or the use of attributes residing in multiple and disparate locations, leading the performance of such approaches to be extremely low. To deal with these problems, we present a novel model of access control applied to Cloud computing called Hybrid attribute and role-based access control for cloud (HyARBAC).
This proposal combines ABAC and RBAC to provide flexible and accurate access control that considers environmental information when controlling access and decreases administrator intervention to manage and control this task. In addition, this model is reinforced by integrating a moving target defense mechanism (MTD), considered an extra layer of security for deterring current and upcoming threats and trying to compromise the original attributes and their corresponding mechanisms of our authorization policies. After an experimental study applied to a healthcare database and comparison against ABAC and RBAC according to several performance features, the effectiveness of HyARBAC was proved to enhance access control.

Keywords: RBAC, ABAC, Cloud computing, Moved Target Defense, Access control

1. Introduction

Cloud computing is one of the critical technologies for sharing hardware and software resources with users. Due to the sensitive and vital information saved in the cloud for consumers, security and privacy are considered essential in cloud computing [1],[2]. Access control is a crucial technology for the security (i.e., availability, confidentiality, and integrity) of all information systems, and its primary purpose is to ensure that authorized users can access data and prevent unauthorized persons from accessing these data [3].

The literature has proposed several models for controlling access to cloud computing in use, such as Discretionary Access Control, Mandatory Access Control, Role Based Access Control, and Attribute Based Access Control. Among these schemes, RBAC and ABAC are the most widely applied for ensuring control over access to data in different enterprises [4]. The fundamental notion of RBAC is as follows: Users are associated with roles, permissions are associated with roles, and users are given permissions by belonging to the appropriate roles [5].

Although RBAC makes policy administration easy and it is also easy to audit, it has several limitations, such as the "Role Explosion", which occurred for fine-grained formulating policies. Due to this situation, several different roles are frequently needed to complete fine-grained permission.

Additionally, RBAC does not enable circumstances in which contextual factors are considered when deciding whether to provide access when the user's time is required [6].

An alternative approach was suggested for these limitations, in [7],[8]. ABAC aims at overcoming RBAC limitations by considering more flexible, context-sensitive information. Specifically, Contextual attributes may be simply used as access control parameters with ABAC. [9]. In terms of administering policies, ABAC is more complicated than RBAC. Moreover, ABAC can lead to a "Rules Explosion", as a system with N attributes could need up to 2^N possible rules [9].

Thus, both models present unique benefits and drawbacks, making their integration a vital research topic [10],

E-mail address: okba.houhou@univ-biskra.dz, s.bitam@univ-biskra.dz, ammar.hamida@univ-biskra.dz

[11],[12]. While both ABAC and RBAC have been extensively utilized independently, their combination in the suggested model is only somewhat groundbreaking. However, the novelty lies in how these models are integrated to provide a more adaptable and precise access control mechanism. By combining the advantages of ABAC with RBAC, this integration aims to overcome the limitations of existing approaches and may provide a more complete solution. Consequently, we propose in this article a new control access approach named: Hybrid attribute and role-based access control for Cloud (HyARBAC). HyARBAC addresses a known weakness of current access control models by considering contextual information such as time, user location, device type, or qualities with various locations.

This creative addition of contextual data enhances the accuracy and effectiveness of the access control mechanism, particularly given the dynamic nature of cloud settings.

Furthermore, HyARBAC introduces a novel security measure to the access control paradigm by incorporating a moving target defense system (MTD). The idea behind MTD is to continuously modify system features, making it more challenging for attackers to exploit vulnerabilities. By creatively enhancing security through the implementation of MTD, our approach strives to bolster the system's defense and mitigate potential risks effectively.

The proposed HyARBAC model demonstrates a considerable amount of originality and creativity by combining ABAC and RBAC, taking into consideration contextual data, including a moving target defense mechanism, and conducting experimental research for validation.

We have introduced a new access control model called Hybrid attribute and role-based access control for cloud (HyARBAC) that addresses the challenges associated with access control in cloud computing. This innovative approach combines ABAC and RBAC to provide flexible and accurate access control while considering environmental information to control access.

This model makes access control management and control more efficient by reducing the need for administrator intervention. Moreover, we reinforce the proposed model by suggesting a Moving Target Defense (MTD) technique [13] that provides an extra layer of security for deterring current and upcoming threats to defense policies. More precisely, the suggested MTD takes as an input an origin authorization to generate the origin attributes listed in the policy and to inspect the attribute bag. As a result, we obtain attributes strongly correlated to the original. Afterward, these recently found attributes are applied to the original strategy in order to modify it by adding new constraints to the ones already there. This modified policy is then applied to the access request in order to enforce it.

The rest of the paper is structured in the following order: We present basic concepts and related work on similar access control approaches proposed for cloud in section 2. In Section 3, we introduce in detail our proposal. After that, we introduce in Section 4 the experimental study and the results obtained. Finally, we highlight our concluding remarks and futures in Section 5.

2. State of the Art

A. Background and Basic Concepts

- Role Based Access Control (RBAC) The RBAC [14],[15] is an access control that defines a set of permissions that are associated with a set of roles. Additionally, in RBAC, various users are assigned to appropriate roles, and users obtain permissions as members of those roles. That makes permission management much more accessible.

For the various organizational functions, roles are created, and users are then assigned roles according to their responsibilities and qualifications. Additionally, roles may be given additional permissions as novel applications and systems are added. Permissions can also be removed from roles as required.

In [15], Sandhu et al. defined a set of four conceptual (RBAC) models. The relationship between these models is illustrated below. RBAC0, the basic model, is in the entity, stating that this is a necessary condition for any system to implement RBAC. RBAC0 is included in both RBAC1 and RBAC2, however each has unique functions.

RBAC1 includes RBAC0 and includes the notion of role hierarchies, wherein roles can inherit authorization from others. RBAC2 incorporates RBAC0 and introduces constraints that restrict the acceptable configurations of the various components of RBAC. RBAC1 and RBAC2 are incomparable to each other. Finally, the RBAC3 model is a combination of RBAC1 and RBAC2.

- Attribute Based Access Control (ABAC)

ABAC was presented in [8], where ABAC's access decisions are based on user, object, and environment attributes. Authorization rules in the ABAC system are based on attributes [16].

- Moving Target Defenses (MTD)

MTD is a paradigm emerging around the idea of proactive modification, for example, moving a protected system's settings. In order to deter potential attacks, for example, it complicates the reconnaissance process in which the attacker gathers information about the victim's existing setups or deters ongoing attacks designed based on previously discovered patterns. [17]

MTD is designed to give defenders the upper hand over attackers by continually shifting resource locations in a way that is only known to defenders, making the information used to identify attackers quickly out-of-date and erroneous. This strategy increases the cost for adversaries to exploit and identify potentially vulnerable systems and

allows defenders to limit the effectiveness of recognizing and exploiting attacks.

B. Related Work

Various systems combine the RBAC and ABAC in different ways, so many models are proposed in the literature. We will discuss more essential models in detail in this section.

Kuhn et al. [9] have suggested that combining the best characteristics of RBAC and ABAC can offer efficient access control for rapidly evolving distributed systems. The authors compared the pros and cons of RBAC and ABAC and then presented possible approaches and options for the combination of attributes with RBAC.

Kuhn et al. identified three principles for schemas to manage the relationship between attributes and roles.

The first concerns dynamic roles, in which the front end uses attributes, such as the time of day, to define the subject's role. The second principle proposes that attributes and roles can be integrated in an attribute-centric way. Roles are treated as one of many attributes; roles are not associated with permissions.

The primary weakness of this model is the loss of the administrative simplicity of RBAC as attributes are added. The third role-centric concept uses attributes to limit RBAC. Constraint rules that embed attributes only restrict the permissions available to the user, not extend them, identified this strategy as a direction for research in the future [18].

Jin et al. [11] presented an approach for combining roles and attributes based on the role-centric model identified by Kuhn et al. [9]. Jin et al. suggested a hybrid access control approach. This model addresses the issue of role explosion. The authors of [11] applied a component called "Permission Filtering Policy" that limits the collection of available permissions based on object attributes and user; however, this approach does not incorporate environmental attributes.

In [18], Rajpoot et al. presented An approach that integrates the ABAC and RBAC models and provides a precise access control system that takes the current context into account when making control decisions. The authors of this study consider that the model presented by Jin et al. in [11] did not include environmental attributes and was not designed for systems with frequently varying attributes, for example, time and location.

In addition, the authors make a significant change to RBAC through the use of object attributes in authorization, solving the problem of role-permission explosion.

Aftab et al. [19] presented a combined RBAC and ABAC based access control model that implements the concept of roles between objects and object attributes as well as between users and user attributes. In this proposal, several attributes are assigned with roles. These roles are associated with particular users and different objects. Note that this model reduces the burden on the administrator and provides the least privilege.

Barkha and Sahani [20] proposed an access control scheme called the "Flexible attribute enriched RBAC model." It provides the integration of ABAC and RBAC; they used contextual attributes for the role activation mechanism, which checks the related conditions to activate the roles. Also, this scheme adds revocation of permission by checking the condition related to permission. In this way, this approach decreases the control's computational complexity. However, this model does not handle the inherited role that allows for simplifying role engineering tasks.

In the access control in cloud computing state of the art, the reader can find [21], in which Hou et al. introduced an enhanced grain access control mechanism for data security to protect data during access to edge computing. This technique has been combined with RBAC and allocates roles depending on user group credibility. In addition, this model checks user access based on matching attributes in order to obtain satisfactory data protection and decrease the danger of insider attacks.

Huang et al. [12] proposed an approach to combine ABAC and RBAC where this model was developed at two levels: the former is an above-ground level, which is a standard RBAC scheme extended with contextual constraints. The latter is an underground level, representing the knowledge building of the RBAC model as attribute policies. These policies are used to create the simple RBAC at the above-ground level automatically. This proposed model has tackled the problem of assigning permissions for large-scale applications. However, it still has a limitation on the problem of role explosion for fine-grained control of access.

In [22], Belhadaoui et al. presented a new access control paradigm for the Hadoop Ecosystem. This paper offers the benefits of the ABAC and the RBAC models. Depending on user roles and attributes, this hybrid paradigm applies access control policies dynamically. In addition, Belhadaoui et al.[22] protected sensitive data in Hadoop and enforced access control policies. The suggested model offers a flexible and fine-grained access control mechanism. This paper discusses the architecture, components, and integration of the model within Hadoop, with a focus on Hadoop MapReduce and Hadoop Distributed File System (HDFS).

Although this study examines access control within the framework of the Hadoop ecosystem, it may need to be more detailed regarding possible security vulnerabilities or issues related to the suggested model access policies.

In [23], Alayda et al. proposed a combination method

for access control in cloud environments called RAAB-AC, which combines Roles Based Access Control and Attributes Based Access Control models to overcome their limitations and create a flexible, simple, and robust access control model for cloud computing.

The paper describes various approaches to combining the models, highlights the advantages of the suggested model, and emphasizes cost reduction and adherence to the concept of least privilege. This paper needs empirical evaluation or experimental results to validate the effectiveness of the proposed approach. Furthermore, it does not discuss potential trade-offs or drawbacks associated with combining RBAC and ABAC models, such as the "role explosion problem" in RBAC and the scalability issue.

Sahani et al.in [24] extended the RBAC model for large-scale applications and developed a framework based on attributes to automate the user-role assignment procedure. The role activation attributes make it possible to implement context-aware and fine-grained access control.

In [13], Rubio et al. presented a new MTD-inspired technique that permits companies to proactively collect the attributes of different entities involved in the access request process. After collection, the authors attempted to choose the attributes that would specifically identify the entities above and to modify the original accesses randomly over time by inserting additional policies built from the reidentified attributes.

This approach can provide an extra layer of security for deterring current and upcoming threats from compromising original attributes. However, this approach makes it possible to prevent attackers from compromising the created and transmitted attributes by using hacking methods, for example, tampering, in order to circumvent the policies of authorization and their corresponding technique and thus obtain involuntary access to sensitive data.

Through this analysis study of the related work of the access control techniques proposed in the literature for Cloud computing, we discovered that many researchers overlook crucial elements such as scalability, revocation permission, explosion problem, inherited role, and access policy security. To deal with these issues, we designed our HyARBAC model, ensuring scalability, efficient management, context awareness, fast computation, auditability, inherited role, permission revocation, and a secure solution for all users.

3. Hybrid Role and Attribute Based Access Control for Cloud Computing: Our Proposal

In this section, the proposed HyARBAC model is illustrated. We start by explaining the proposed model. After that, the assignment permission principle for each role through attributes-based rules and constraints is presented, which is considered the main idea of this proposal. To enhance this access control in the cloud, an MTD scheme is conceived, which is explained in this part.

A. Proposed Model

As explained above, we propose to combine the access control oriented by the role principle with ABAC in order to ensure the effectiveness of this process control, aiming to apply and integrate environmental data to identify authorized cloud resource users.

In particular, the proposed HyARBAC system described in Fig.1 uses the attributes of the environment during the session to enable roles in which a user can be attributed to many roles but can be activated at a time for only one role, and the constraint role-permission is introduced to surmount the issue of role explosion and obtain fine-grained results.

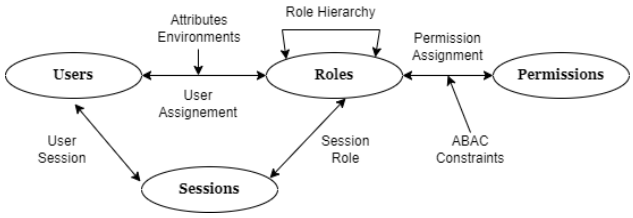

Figure 1. Proposed Model HyARBAC

According to the proposed model, when a user sends a request, his session is then started, but before activating the role, the environment condition is checked (satisfied or not).

The role is set whenever the environment condition is satisfied; otherwise, the role is disabled. After activating the role, a permission set is appropriate for that role.

Thus, the admin needs to assign the roles once, after which these roles can be assigned to many users and may be created with the specification of the ABAC rules depending on the object's security level.

Moreover, the administrator may set two security levels according to the sensitivity of objects; he uses the RBAC technique if an object is less sensitive; otherwise, he will enforce more security by adding ABAC attributes and the MTD technique.

The flowchart of the proposed model is presented in Fig.2.

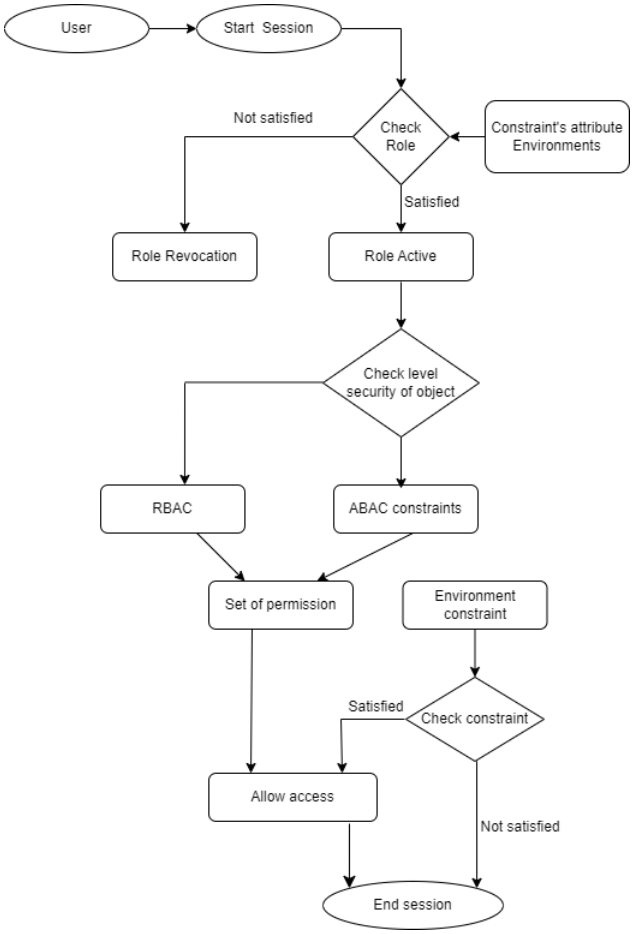

Figure 2. Flowchart of HyARBAC

B. *Assignment Role-Users*

In our model, each user could have one or many roles, and many users may be given access to the same role. So, initially, the administrator statically assigns each user with their roles.

C. *Assignment Permission–Role*

For this model, each permission could have one or more roles associated with it, and each role could be associated with one or more permissions.

Suppose the security level of the object is less. In that case, the RBAC technique is used(in the case where the data is not sensitive), in which each role has its permissions, and each permission is attributed to its roles statically; otherwise (in the case where the data is sensitive), we assign roles to permissions through ABAC rules (constraints) using the attributes of users, objects, or environments, users will be able to access the set of permissions represented by the intersection of Ps and Rs, where Ps represents the set of permissions associated with the current roles and Rs represents the set of permissions defined by the relevant ABAC rules.

D. *Access Decision*

Once the user sends an access request, the session is started; enabling the role depends on checking the context condition. If the contextual condition is true, the role will be activated; otherwise, the role activation will be revoked.

After activating the role, if the level of security of the object is less than a threshold (fixed at 1), the system enforces the RBAC technique, where a collection of permissions is connected to the role; otherwise, the system assigns a set of permissions to the role through ABAC rules (constraints) using user, object, and/or environment attributes.

In this step, an environment constraint check is performed; if satisfied, access to data is allowed, and otherwise, the role revocation and session are ended. Then, while accessing the object, that condition is checked; if the environment condition becomes unchecked, this authorization is removed from the last available authorization, which the MTD system reinforces the ABAC method [13], where this system represents an additional layer of protection for a future attack on policies. The set of flows from the proposed system is shown in Fig.2.

In this step, an environment constraint check is performed; if satisfied, access to data is allowed, and otherwise, the role revocation and session are ended.

E. *Moving Target Defenses (Policy Mutation)*

This technique is inspired by the approach presented by Rubio et al. in [13]

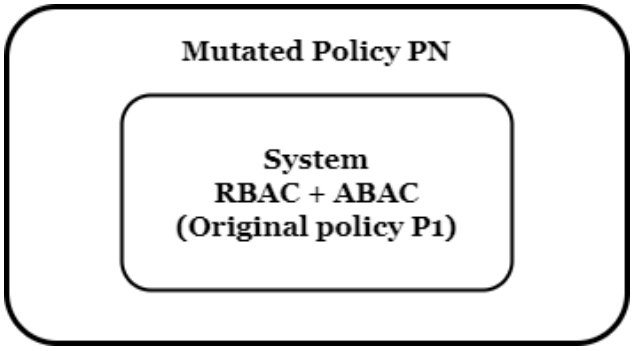

Figure 3. A graphical representation of our approach reinforced by technique MTD.

Initially, in Fig.3, we model an RBAC+ABAC (Original policy Pl) access policy as a set of constraints-based rules Ru = {ru1, ru2, ru3,..., run}.

For every rule ru ∈ Ru, we present the collection of original attributes (Sr). Additionally, we represent the attribute bag (attributes set A) as A∩Sr≠Φ that with each rule.

The strategy then seeks to develop a modified policy Pn by the initial policy Pl as the next: for every rule ru∈Ru,

we find the collection of attributes Ct = {c1, c2, ... cp} ⊆A, Ct≠Sr who correlate with the set of Sr attributes.

The rule ru can then be changed into a new rule r*ù* by arbitrarily selecting a subset c∈Ct such that S*ŕ* = Sr⋃c.

The modified rule set R*ù* = {r*ù*1, r*ù*2, ...r*ù*n} is later combined to generate the novel modified policy Pn. Thus, Pn is said to extend the original policy Pl by including rules that describe additional correlated attributes.

We periodically repeat the procedure above to generate many distinct political mutations.

In order to achieve this effect, an interaction technique can generate rules that are changed at random, as mentioned above, by only choosing a subset of the set Ct of attributes that are correlated at a time, where the resultant rules may differ now and then.

We summarize this description with the following algorithm:

Algorithm Policy Mutation

Require: An original policy Pl, an attribute bag A

Ensure: A mutated policy Pn

Pn← Φ

Ru ←getRulesOfPolicy(Pl)

for all rui ∈ Ru do

Sr←getAttributesOfRule(rui)

Ct ←locateCorrelatedAttributes(Sr , A)

r*ù*i ←getRandomMutatedRule(rui , Ct)

Pn←Pn ⋃ r*ù*i

endfor

returnPn.

4. Experimental Scenario : Case Study of Access Control of Hospital Information System

In our proposed implementation of the HyARBAC model, we have taken great care to include authorization rule definitions and XACML[25] policies for access control, utilizing the Axiomatic Language for Authorization (ALFA) [26], a highly effective domain-specific language for defining XACML policies.

Our goal was to create a simple strategy for developers to utilize, and we achieved this by incorporating the ALFA plug-in for the Eclipse IDE to generate and edit XACML policies.

To evaluate and validate this approach, we utilized the open-source WSO2 Identity Server [27], a proven solution that has helped us offer experimental proof for the effectiveness of our proposed strategy.

We also developed a series of tests evaluated and validated using the WSO2 Identity Server, reinforcing our proposal. We aim to provide a practical solution that can be easily implemented, and our proposed approach, backed by real-world testing, is a testament to its effectiveness.

A. Studied Scenario

Our research involved a thorough examination of HyARBAC, a system designed to address access control challenges in hospital information systems placed on a cloud computing platform.

To evaluate the performance and effectiveness of HyARBAC, we conducted experiments on a comprehensive set of case studies that covered a wide range of access control scenarios.

The case studies featured multiple users, including patients, doctors, nurses, and other medical staff, each with their own specific roles and responsibilities. We formulated policies for each role, using XACML language with ALFA to specify the policies and WSO2 IS to validate them.

The case studies featured multiple users across various departments, including patients, doctors, nurses, and others, each with their own roles and corresponding policies. Specifically, we created three policies: two for doctors (policy2 and policy3) and one for a nurse (policy1).

Our experimental results and analysis provide a comprehensive understanding of the technical aspects of our approach and demonstrate its effectiveness in addressing access control challenges in a hospital information system placed on cloud computing.

Policy1: ”The nurse can active her role only during working duration.”

Policy2: ”The doctor is capable of reading and writing patient medical records that are exclusive to his department.”

Policy3: ”the primary doctor can read the patient’s medical records and can add information for patients who are exclusively part of his department.”

-Role explosion problems and fine-grained access:

When implementing access control policies in RBAC, fine-grained control is often necessary to ensure that users have access only to the resources they need.

However, implementing such policies often requires a large number of roles, which can lead to a role explosion problem. For instance, in the case of policy2, a doctor would be allowed to read or write the medical records of

their patients within their department. To achieve this level of fine-grained control, we might need to create multiple roles such as Doctor Cardiology, Doctor Brain, and Doctor Orthopedic.

To address this problem, we can use attribute constraints role-permission instead of creating multiple roles. By doing so, we can define permissions based on attributes of users (such as their department) rather than creating a new role for each possible combination of attributes.

This approach reduces the number of roles required and simplifies the management of access control policies. For example, in Table II, we only need one role (Doctor) to achieve the same policy that requires three roles in Table I.

Role	Cardiology patient record	Brain patient record	Orthopedic patient record
Doctor Cardiology	Read and Write	Read	Read
Doctor Brain	Read	Read and Write	Read
Doctor Orthopedic	Read	Read	Read and Write

TABLE I. ROLE EXPLOSION PROBLEM IN RBAC(NEED OF 3 ROLES)

Role	Department	Resource-department	Action
Doctor	Cardiology	Cardiology Brain Orthopedic	Read/Write Read Read
Doctor	Brain	Cardiology Brain Orthopedic	Read Read/Write Read
Doctor	Orthopedic	Cardiology Brain Orthopedic	Read Read Read/Write

TABLE II. CONSTRAINT'S ROLE-PERMISSION (ONLY ONE ROLE IS NEEDED)

1) In order to apply policy1 (concerning the nurse), we generate the eXtensible Access Control Markup Language (XACML) code corresponding to this policy; this code is presented in Fig.4.

Figure 4. XACML Activation role policy

1.1 Test case 1 for the activation role nurse: In Table III, the nurse wants to activate its role at 7:00 a.m. and then at 11:30 a.m.

Role	Duration of role activation	Current time	Result
Nurse	08:00:00-16:00:00	11:30:00	Permit
Nurse	08:00:00 - 16:00:00	07:00:00	Deny

TABLE III. TEST CASE1 FOR ACTIVATION ROLE

Fig.5 shows the XACML request1 code of the nurse's role activation.

-Request1 for activation role: Case permit

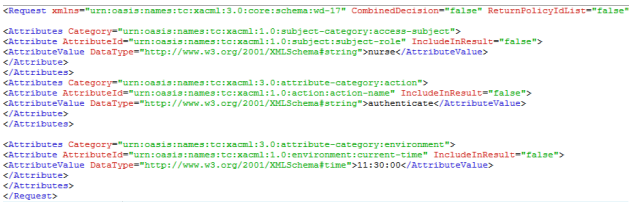

Figure 5. XACML Request1 activation role

Fig.6 shows the generated response using the XACML request1 code of the nurse's role activation in working time (11:30 h), which the decision permits.

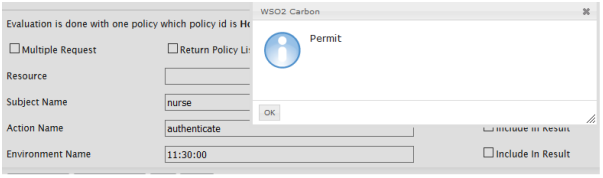

Figure 6. Generate response for request1

Fig.7 shows the XACML request2 code of the nurse's role activation outside working duration (at 07:00h).

- Request2 for activation role: Case deny

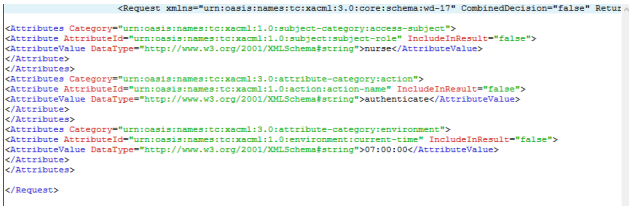

Figure 7. XACML Request2 activation role in outside working duration

Fig.8 depicts the generated response by the XACML request2 code of the nurse's role activation outside working duration (at 07:00 h), where the decision is denied.

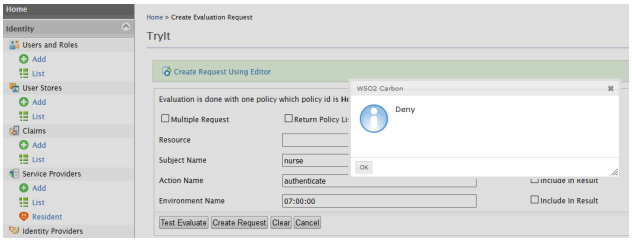

Figure 8. Generate response for request2

As indicated in the activate role policy, it is enabled based on context (Time), so the role is activated during working hours, and hence, it is switched off outside working hours.

2) In order to apply policy2 concerning the doctor, we generate the XACML code corresponding to this policy. It is presented in Fig.9.

Using the constraints of role-permission, this proposal allows for fine-grained access and resolves the 'Role explosion' issue, as shown in the example presented in TableII.

Policy2 can be translated as follows:

Subject-role = doctor and subject-department=Brain ;

resource-name = medicalRecord and resource-department= Brain;

-Constraint role- permission:

If subject-department=resource-department then decision=permit

else decision=deny

Figure 9. XACML Doctor policy2

- Test for constraints role-permission: Case Permit / Deny.

Figure 10 shows the request for constraints role-permission (subject-department=resource-department);

We have two decisions: the case to permit a doctor to access a resource (medicalRecord) and the case to deny a doctor access to this resource.

Subject	Resource	Action	Condition	Decision
subject-department = brain & subject-role = doctor	resource-department = brain & resource-name = medicalRecord	action-type = write	subject-department=resource-department = True	Permit
subject-department = brain & subject-role = doctor	resource-department = cardiology & resource-name = medicalRecord	action-type = write	subject-department=resource-department = False	Deny

Figure 10. TEST CONSTRAINTS ROLE-PERMISSION

Figure 11 shows the response to the test decision to permit or deny a doctor access to resources (medicalRecord).

Subject	Resource	Action	Environment	Condition	Decision	Verification Result
subject-department = brain & subject-role = doctor	resource-department=brain & resource-name = medicalRecord	action-type = write	Environment = Any Value	subject-department=resource-department = True	Permit	TRUE
subject-department = brain & subject-role = doctor	resource-department = cardiology & resource-name = medicalRecord	action-type = write	Environment = Any Value	subject-department=resource-department = False	Deny	TRUE

Figure 11. RESPONSE TEST CONSTRAINTS ROLE-PERMISSION

3) In order to apply policy3 concerning the doctor(inherited role) :

We generate the XACML corresponding to this policy, which permits simplifying role engineering tasks using the inheritance of one or more parent roles in our policy, as shown in Fig.12.

Policy3 is as follows:" The primary doctor can read the patient's medical records and add information for patients who are exclusively part of his department."

We need two rules to apply the policy; we create the first rule (the original one) and the second (inherited), which will be created automatically by the inherited role, as shown below (see Fig 13).

Figure 12. XACML inherited role policy

3.1)Test case3 for inherited roles:

Figure 14 presents the request for an inherited role, which permits or denies the primary doctor access to a resource.

Figure 15 presents the response to the request for inherited roles and the verification result of the decision to access resources.

Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
Subject-role = PrimaryDoctor & Subject-department = Brain	Resource-name = MedicalRecord & Resource-department = Brain	action-type = Write	Environment = Any Value	subject-department=resource-department = True	Permit	Inherited
Subject-role = PrimaryDoctor	Resource-name = MedicalRecord	action-type = Read	Environment = Any Value	Condition = Any Value	Permit	Originated

Figure 13. TWO RULES ACCESS

Subject	Resource	Action	Environment	Condition	Decision
Subject-role = PrimaryDoctor & Subject-department = Brain	Resource-name = MedicalRecord & Resource-department = Brain	action-type = Write	Environment = Any Value	subject-department=resource-department = True	Permit

Figure 14. TEST CASE3 INHERITED ROLE

Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Subject-role = PrimaryDoctor & Subject-department = Brain	Resource-name = MedicalRecord & Resource-department = Brain	action-type = Write	Environment = Any Value	subject-department=resource-department = True	Permit	TRUE

Figure 15. GENERATE RESPONSE TEST CASE3

B. HyARBAC Security Aspects Analysis and Features

- Context-aware: We have a context-aware system. We utilize environmental conditions in our system.

-Auditability: - In order to make the system simple to use, we maintain the RBAC model, where the role assigned to a user is static and the permission is associated with the role.

- Revocation of the permission: The revocation of the permission based on the condition of the context is ensured by the system. When accessing certain objects, we constantly check the context constraint, and when certain conditions fail, we immediately repeal the user's access.

- Inherited role: The inherited role permits simplifying role engineering tasks using the inheritance of one or more parent roles.

- Computationally fast: Our system assures the activation of the role according to context conditions because the role can only be activated if the conditions for activating the role are satisfied, which makes the system faster.

- Ease management: The proposed model allows for easier administration of policies, such as assigning roles to users and assigning objects to roles with constraints attributes (object, user, and environment). This makes it simpler to manage policies.

- Scalability: Since the role explosion problem is solved (by constraints role-permission), we can increase the number of authorized users, and the system can operate effectively. As a result, the number of permitted users cannot affect our system.

- Security: The security of our system is dependent on the sensitivity of our resources. The data is secured by RBAC and ABAC access control techniques; in addition, our access policy (attributes, rules, etc.) is secured against attackers who want to hack the policy (by modifying attributes, rules, or other) by using the Moved Targeted Defense technique, which constitutes a barrier of protection against unauthorized access to our sensitive data and to deter current and future attacks.

Hence, our system of access control overcomes the limitations of both ABAC and RBAC and has the following features:

-For the purpose of solving the problem of RBAC role explosion and to provide fine access control in RBAC, we use the constraints role-permission.

- The system is simple to audit because we use the RBAC approach, in which a user's role assignment is static, and the permission is associated with the role, so it is easy to audit.

- The system is easy to manage and update.

- Our system is faster because activation of the role is based on checking the environment attribute (after and before activation).

- Additionally, in this model, two security levels may be defined according to the sensitivity of objects.

- Layer of additional protection for deterring current and future attacks against policies.

As shown in Table IV, our proposed HyARBAC model surpasses other access control schemes found in the literature. While most research does not address crucial challenges such as scalability, revocation permission, explosion problem, inherited role, and access policy security, our model was specifically designed to overcome these obstacles and deliver outstanding performance. HyARBAC offers efficient management, fast computation, auditability, context

awareness, inherited role, permission revocation, and policy security.

Our study results demonstrate that HyARBAC outperforms other access control models, making it the most promising solution for various settings. Choose HyARBAC for an access control model that delivers exceptional performance, scalability, and security.

Features	Jin et al. [10]	Rajp et al. [15]	Barkha et al. [17]	Alayda et al. [24]	Our model
Audit-ability	Yes	Yes	Yes	Yes	Yes
Fine-grained access control	Yes	Yes	Yes	Yes	Yes
Scalability	Yes	No	No	No	Yes
Revocation permission	No	No	Yes	No	Yes
No role exlosion problem	No	Yes	No	No	Yes
Context-aware	No	Yes	Yes	Yes	Yes
Inheritd role	No	No	No	No	Yes
Ease management	Yes	Yes	Yes	Yes	Yes
Dynamicity	Yes	Yes	Yes	Yes	Yes
Fast comput-ationally	No	Yes	Yes	Yes	Yes

TABLE IV. COMPARISON OUR MODEL AGAINST THE STATE-OF-THE-ART ACCESS CONTROL SCHEMES

5. Conclusions and Future Work

This article presents a new approach to access control for cloud computing, called Hybrid Role and Attribute Based Access Control for cloud computing (HyARBAC). This model combines ABAC and RBAC to provide flexible and fine-grained access control that considers environmental information, reducing the need for administrator intervention. Additionally, the model is strengthened by the Moving Target Defense technique, adding an extra layer of security to our policy.

This approach brings the benefits of ABAC and RBAC while overcoming their drawbacks. In our model, the activation of the role is dependent on an environmental attribute; it depends on the constraint's role permission to fix the problem of role explosion. Also, several attribute-based rules are suggested for any permission to achieve fine-grained access. HyARBAC guarantees dynamicity, auditability, scalability, fine-grained access control models without role explosion, and more security.

In future research, we can consider leveraging machine learning algorithms to improve the accuracy of access control decisions in cloud computing.

Furthermore, we can explore the potential benefits of integrating blockchain technology into access control systems, as this could enhance transparency and traceability of access decisions.

1- We suggest integrating blockchain systems in cloud computing to increase the security of sensible data in the access process. Moreover, blockchain could help to:

-Increase security: Blockchain technology's decentralized structure and cryptographic methods make access extremely secure.

-Increase transparency: Organizations can use blockchain systems to generate an unchangeable, transparent record of who has access to what resources. That can lessen the chance of unauthorized access and guarantee that the right persons are given the proper amount of access.

-Automation: Access control in cloud computing can be automated using blockchain technology. That implies that access can be provided or denied automatically. That can lower the possibility of human error and guarantee that access is granted and canceled as soon as possible.

2- We propose to integrate AI technology into cloud computing to create more secure, adaptable, and efficient cloud environments for users and organizations. In this direction:

-We should concentrate on developing AI algorithms that can dynamically combine the benefits of ABAC and RBAC models based on user attributes and contextual elements to create a more effective and flexible access control system.

-We have to concentrate on creating adaptive access control systems that are AI-powered. These systems can dynamically modify access permission based on user behavior, context, and risk assessments by utilizing machine learning algorithms.

This adaptability will enhance security and usability, ensuring that access decisions align with the dynamic nature of cloud environments.

References

[1] M. A. AlZain, B. Soh, and E. Pardede, "A Survey on Data Security Issues in Cloud Computing: From Single to Multi-Clouds," *JSW*, vol. 8, no. 5, pp. 1068–1078, May 2013. [Online]. Available: http://ojs.academypublisher.com/index.php/jsw/article/view/8330

[2] N. H. Hussein and A. Khalid, "A survey of Cloud Computing Security challenges and solutions," vol. 14, no. 1, p. 5, 2016.

[3] S. Ruj, "Attribute based access control in clouds: A survey," in *2014 International Conference on Signal Processing and Communications (SPCOM)*. Bangalore, India: IEEE, Jul. 2014, pp. 1–6.

[4] S. Das, B. Mitra, V. Atluri, J. Vaidya, and S. Sural, "Policy engineering in rbac and abac," *From Database to Cyber Security: Essays Dedicated to Sushil Jajodia on the Occasion of His 70th Birthday*, pp. 24–54, 2018.

[5] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," *ACM Trans. Inf. Syst. Secur.*, vol. 4, no. 3, pp. 224–274, Aug. 2001. [Online]. Available: https://dl.acm.org/doi/10.1145/501978.501980

[6] M. P. Singh, S. Sudharsan, and M. Vani, "ARBAC: Attribute-Enabled Role Based Access Control Model," in *Security and Privacy*, S. Nandi, D. Jinwala, V. Singh, V. Laxmi, M. S. Gaur, and P. Faruki, Eds. Singapore: Springer Singapore, 2019, vol. 939, pp. 97–111, series Title: Communications in Computer and Information Science. [Online]. Available: http://link.springer.com/10.1007/978-981-13-7561-3_8

[7] X. Jin, R. Krishnan, and R. Sandhu, "A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC," in *Data and Applications Security and Privacy XXVI*, N. Cuppens-Boulahia, F. Cuppens, and J. Garcia-Alfaro, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, vol. 7371, pp. 41–55, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-642-31540-4_4

[8] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to Attribute Based Access Control (ABAC) Definition and Considerations (Draft)," *NIST Special Publication*, p. 54.

[9] D. R. Kuhn, E. J. Coyne, and T. R. Weil, "Adding Attributes to Role-Based Access Control," *Computer*, vol. 43, no. 6, pp. 79–81, Jun. 2010. [Online]. Available: http://ieeexplore.ieee.org/document/5481941/

[10] E. Coyne and T. R. Weil, "ABAC and RBAC: Scalable, Flexible, and Auditable Access Management," *IT Prof.*, vol. 15, no. 3, pp. 14–16, May 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6519249/

[11] X. Jin, R. Sandhu, and R. Krishnan, "RABAC: Role-Centric Attribute-Based Access Control," in *Computer Network Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, vol. 7531. [Online]. Available: http://link.springer.com/10.1007/978-3-642-33704-8_8

[12] J. Huang, D. M. Nicol, R. Bobba, and J. H. Huh, "A framework integrating attribute-based policies into role-based access control," in *Proceedings of the 17th ACM symposium on Access Control Models and Technologies - SACMAT '12*. Newark, New Jersey, USA: ACM Press, 2012, p. 187.

[13] C. E. Rubio-Medrano, J. Lamp, A. Doupé, Z. Zhao, and G.-J. Ahn, "Mutated Policies: Towards Proactive Attribute-based Defenses for Access Control," in *Proceedings of the 2017 Workshop on Moving Target Defense*. Dallas Texas USA: ACM, Oct. 2017, pp. 39–49.

[14] D. Ferraiolo, J. Cugini, D. R. Kuhn *et al.*, "Role-based access control (rbac): Features and motivations," in *Proceedings of 11th annual computer security application conference*, 1995, pp. 241–48.

[15] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-Based Access Control Modelsyz," p. 22.

[16] S. Jha, S. Sural, V. Atluri, and J. Vaidya, "Specification and Verification of Separation of Duty Constraints in Attribute-Based Access Control," *IEEE Trans.Inform.Forensic Secur.*, vol. 13, no. 4, pp. 897–911, Apr. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8101537/

[17] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving target defense: creating asymmetric uncertainty for cyber threats*. Springer Science & Business Media, 2011, vol. 54.

[18] Q. M. Rajpoot, C. D. Jensen, and R. Krishnan, "Integrating attributes into role-based access control," in *Data and Applications Security and Privacy XXIX: 29th Annual IFIP WG 11.3 Working Conference, DBSec 2015, Fairfax, VA, USA, July 13-15, 2015, Proceedings 29*. Springer, 2015, pp. 242–249.

[19] M. U. Aftab, Z. Qin, S. F. Quadri, Zakria, A. Javed, and X. Nie, "Role-Based ABAC Model for Implementing Least Privileges," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*. Penang Malaysia: ACM, Feb. 2019, pp. 467–471.

[20] P. Barkha and G. Sahani, "Flexible attribute enriched role based access control model," in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*. Indore: IEEE, Aug. 2017, pp. 1–6.

[21] Y. Hou, S. Garg, L. Hui, D. N. K. Jayakody, R. Jin, and M. S. Hossain, "A Data Security Enhanced Access Control Mechanism in Mobile Edge Computing," *IEEE Access*, vol. 8, pp. 136 119–136 130, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9146646/

[22] H. Belhadaoui, R. Filali, O. Malassé *et al.*, "A role-attribute based access control model for dynamic access control in hadoop ecosystem." *IAENG International Journal of Computer Science*, vol. 50, no. 1, 2023.

[23] S. Alayda, N. A. Almowaysher, M. Humayun, and N. Jhanjhi, "A novel hybrid approach for access control in cloud computing," *Int. J. Eng. Res. Technol*, vol. 13, no. 11, pp. 3404–3414, 2020.

[24] G. J. Sahani, C. S. Thaker, and S. M. Shah, "Scalable rbac model for large-scale applications with automatic user-role assignment," *International Journal of Communication Networks and Distributed Systems*, vol. 28, no. 1, pp. 76–102, 2022.

[25] O. Standard, "extensible access control markup language (xacml) version 3.0," *A:(22 January 2013). URl: http://docs. oasis-open. org/xacml/3.0/xacml-3.0-core-spec-os-en. html*, 2013.

[26] *ALFA Plugin for Eclipse User's Guide,by Axiomatics AB*, 2013.

[27] URl: http://wso2.com.

Okba Houhou is a Ph.D. student in Computer Science at Department of Computer Science, University of Biskra, Algeria. His current research is Access control and data mining. He is currently an assistant professor at the University of Biskra.

Salim Bitam (s.bitam@univ-biskra.dz) is a full professor of Computer Science and a vice rector responsible of post-graduation training and scientific research at the University of Biskra, Algeria. He received an Engineering degree in computer science from the University of Constantine, Algeria, his Master's and Ph.D. in computer science from the University of Biskra, and a Doctorate of Sciences (Habilitation) diploma from the Higher School of Computer Science - ESI, Algiers, Algeria. His main research interests are vehicular ad hoc networks, cloud computing, and bio-inspired methods for routing and optimization. He has to his credit more than 40 publications in journals, books, and conferences, for which he has received two best paper awards. He has served as an editorial board member and a reviewer of several journals for IEEE, Elsevier, Wiley, and Springer, and on the technical program committees of several international conferences (IEEE GLOBECOM, IEEE ICC, IEEE/RSJ IROS, and others).

Ammar Hamida is a Ph.D. student in Computer Science at Department of Computer Science, University of Biskra, Algeria. His current research is on the Internet of Things. He is currently an assistant professor at the University of Biskra.