Enhancing Hybrid ACL Based Model by Optimizing Task Scheduling in Cloud

[1]Mojtaba Mohammadi, [2]Keshav Kishore
[1]Post Graduate Student, [2]Associate Professor
[1]Department of Computer Science and Engineering,
[1]AP Goyal Shimla University, Shimla (HP), India

Abstract: Cloud computing is a relatively new technology which is growing rapidly due to its distinctive features. It provides a way to access data from any place at any time. This feature makes cloud computing widely popular, because it reduces the burden of the users. Cloud computing provides various services like infrastructure, platform and software as a service. Due to these features, the efficiency of cloud has been greatly affected as a result of the increasing size of data on cloud. To overcome such a problem, the task security on data appears to be a promising option. Workflow security is a challenging task in cloud computing, considering user requirements and satisfaction. Another issue is the resource utilization due to the nature of cloud computing. In order to maintain and utilize resources in the cloud computing, a security mechanism is required. Many algorithms and protocols are used to manage the parallel processing used to enhance the performance of the CPU in the cloud environment. RBAC and RBAC-ABAC(HYBRID) models use variety of different approaches to improve the security of cloud. Our proposed model however, is based on the optimization of the total execution time as well as the total execution cost. Using intelligence optimization initialized by Pareto distribution, the results are found to be effective in compare to the existing methods. A hybrid model is used to converge the decision of Virtual Machine (VM) migration by its convergence to minimize cost and time as illustrated by the total execution time and total execution cost. It is concluded that our proposed hybrid model performs better in compare to the existing ABAC algorithms.

Keywords: *Role Based Access Control (ABAC), Attribute Based Access Control (ABAC), Context Aware Access Control, HA-RBAC, Role Explosion, Particle Swarm Optimization (PSO), Gray Wolf Optimizer (GWO), Pareto Distribution*

I. INTRODUCTION

The relationship between users and resources is dynamic in the cloud, and service providers and users are typically not in the same security domain. The Identity-based models of security such as mandatory or discretionary access controls cannot be used in an open cloud computing environment, where each resource node may not be familiar, or even do not know each other. The users of the system are usually identified by their characteristics or attributes and not by their predefined identities. There is often a requirement for the control of dynamic access mechanism in order to achieve the authentication on the basis of cross-domain property. In this paper, we will focus mainly on two access control models, the role based and the attribute based. We will review the existing literature of these two models including their approaches, technical applicability, characteristics, pros and cons. Later in this paper, we propose a hybrid model on the basis of ACL by utilizing the impressive features of both the models.

Cloud computing delivers its services in three models including: Software as a Service (SaaS) where the customers of the cloud generally use the provider's applications over the web; Platform as a Service (PaaS) where the customers mainly deploy their self-designed applications types present on a developing platform that a provider of the cloud usually provides; and Infrastructure as a Service (IaaS) where the customers of the cloud rent storage, capacity of the network, and processing from the provider of the cloud service [1]. In a more precise definition, cloud computing is defined as "both the applications delivered as services over the Internet and the hardware and systems software in the data-centers that provide those services." [2] As a low-cost solution, cloud computing provides computation, software, data access, and storage services that are transparent to end users. Its basic characteristics can be briefly summarized as follow: [2, 3].

- On-demand self-service. A consumer can unilaterally provision computing capabilities as needed with little or no human intervention.
- Broad network access. Resources can be accessed over the network using existing mechanisms (e.g., current Internet protocols).
- Resource pooling. Resources are pooled to provide services to multiple consumers. Physical and virtual resources can be dynamically assigned and reassigned according to consumers' demand.

1.2 Role Based Access Control

RBAC is a secure method of restricting account access to authorized users. This method enables account owners to add users and assign roles. Each role has specific permissions that Rack space has defined. RBAC gives customers a greater degree of control over cloud resource use, with an additional layer of system security [4]. RBAC enables users to perform actions based on the scope of their assigned roles. Users: RBAC has the following types of users:

- *Account user* - The account user is a user who has been added by the account owner and has been assigned to specific product or account roles.
- *Account owner* - The account owner is the primary contact for the account and has full permissions to execute all capabilities for every product available. Each account has a single account owner.

RBAC has the following roles as stated below:

1. *Multiple-product roles:*
   Multiple-product roles grant access to resources associated with multiple products. RBAC has the following multiple-product roles:
   - *Full access* - The full access role has permissions to create, read, update, and delete resources within multiple designated products.
   - *Read-only access* - The read-only access role has permissions to view resources within multiple designated products.

2. *Custom roles:*
   Custom roles enable account owners to assign users different permissions for different products. After a user is assigned custom roles, those roles can only be changed on a per-product basis. RBAC has the following custom roles:
   - *Product: admin* - The product admin role has permissions to create, read, update, and delete resources for the designated product.
   - *Product: creator* - The product creator role has permissions to create, read, and update resources for the designated product. The creator role cannot delete a resource. (Any destructive actions are prohibited.
   - *Product: observer* - The product observer role has permissions to read given resources for the designated product. This role is read-only.

3. *Account roles:* Assign the following account roles to the users who manage your Rack space customer account.
   - *Billing: admin* - The billing admin role has permissions to create, read, update, and delete billing and payment resources for the designated product.
   - *Billing: observer* - The billing observer role has permissions to read billing and payment resources for the designated product. This role is read-only.

The RBAC model greatly simplifies the management of user permissions on large systems and ensures that administrators can enforce the security principles of least privilege and separation of duty/privilege [6]. RBAC also simplifies the problem of ensuring that users are given correct access rights to a system. As users are assigned roles which map to permissions that in turn map to abstract operations on an information system, an administrator would only need to check that a user has been assigned the correct roles to ensure they have the correct access rights (assuming the roles and permissions were created correctly). Similarly, changing the level of access a role is given to match changes in organizational 8 policy or structure is trivial in RBAC as only the role-permission assignment would need to be altered to affect the access rights of every user assigned to the role.

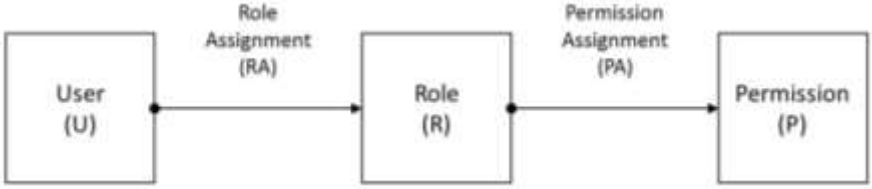

Fig.1 Basic RBAC model [34]

In a role-based access control (RBAC) model, the role of a user is assigned based on the least privilege concept – i.e. the role with the least amount of permissions or functionalities that is necessary for the job to be done. Task Role-based access control model (TRBAC) [8] has been considered a viable model for cloud computing environments [9] wherein the traditional static access control models such as discretionary, mandatory or simple role-based models cannot be employed. TRBAC can dynamically validate access permissions for users based on the assigned roles and the task the user has to perform with the assigned role. Tasks could be classified as workflow tasks (those that need to be completed in a particular order) that require active access control and non-workflow tasks (those that can be completed in any order) that require passive access control. Workflow tasks driven active role-based access control is time sensitive and the access permissions assigned for users performing these tasks change dynamically with time, depending on the order in which the tasks are to be executed. Care should be taken to ensure that a user has the minimum required privileges to perform a task under a particular role, and that no role can be assigned to two or more tasks at the same time. Another variant of role based access control proposed for cloud computing environments is the Attribute-role-based access control (ARBAC) model [10], wherein the data object to be protected are assigned certain attributes and values; a user with a specific role has to submit the appropriate values for these attributes, and are given access to the objects after proper validation by the service provider. A fine-grained key based ARBAC model has been proposed in [11], where users are assigned the private keys or symmetric keys that are used to encrypt/decrypt the values of the attributes defined for the data objects whose privacy needs to be protected. The proposed temporal-RBAC (TRBAC) model that enables and disables a role at run-time depending on user requests, Generalized TRBAC (GTRBAC) model that advocates for role activation instead of role enabling. A role is said to be activated if at least one user assumes that role. GTRBAC supports the enabling and disabling of constraints on the maximum active duration allowed to a user and the maximum number of activations of a role by a single user within a particular interval of time. In [14], the authors present an XML-based RBAC policy specification framework to enforce access control in dynamic XML-based web services. However, both GTRBAC and X-RBAC cannot provide trust and context-aware access control (critical for dynamic web services, characteristic of cloud computing environments), and rely solely on identity or capability-based access control. In [10], the authors propose an enhanced hybrid version of the X-RBAC and GTRBAC models, called the X-GTRBAC model. XGTRBAC relies on the certification

provided by trusted third parties (such as any PKI Certification Authority) to assign the roles to users. X-GTRBAC also considers the context (such as time, location, or environmental state at the time the access requests are made) to directly affect the level of trust associated with a user (as part of user profile), and incorporates it in its access control decisions. The access privileges for a user/role are based on the threshold (i.e. the trust level) established based on the requestor's access patterns; if the user appears to deviate from his/her usual profile, then the trust level for the user is automatically reduced to prevent potential abuse of privileges. Such a real-time feature of X-GTRBAC suits to the web-based cloud computing environments with diverse customer activity profiles.

1.3 Cloud Problems

While cloud computing may offer potential cost savings and dynamically scalable infrastructure, it also brings with it new security and privacy issues that need to be addressed:

- Confidentiality: Protecting cloud based storage and network transmissions from possible unwanted access by cloud providers or data leakage to other cloud users.
- Legal: Complying with data privacy laws that may be in effect in given geographical regions.
- Auditability: Maintaining logs of users' actions with the system and ensuring that no part of the system has been tampered with or compromised.
- Security: Preventing user credentials, which may be used for multiple services on and off the cloud, from being obtained by untrusted parties including the cloud provider or other cloud users.

1.4 RBAC: Standards

Several role based access control models have been developed in recent years that have expanded on and standardized the core ideas behind RBAC. One of the most notable is the effort by the National Institute of Standards and Technology (NIST) to create a standardized template for which the majority of RBAC implementations can be based and expanded upon. The NIST model integrates several previously published RBAC models/frameworks into the standard for RBAC adopted as ANSI INCITS 359-2004 (Inter National Committee for Information Technology Standards, 2004). This model divides RBAC into four functional components: Core RBAC and three optional components (Hierarchical RBAC, Static Separation of Duty Relations and Dynamic Separation of Duty Relations), which may be combined to create the basis for implementing an RBAC package [6]. Core RBAC maps together the five basic elements of a role based access control system (users, roles, objects, operations and permissions) to assign users to roles, and roles to permissions, where permissions are in turn mapped between operations and objects. User sessions are also present in the Core RBAC model, allowing users to activate a subset of roles they have been assigned in a given session. Figure 2 illustrates the many-to-many mapping between these basic elements.

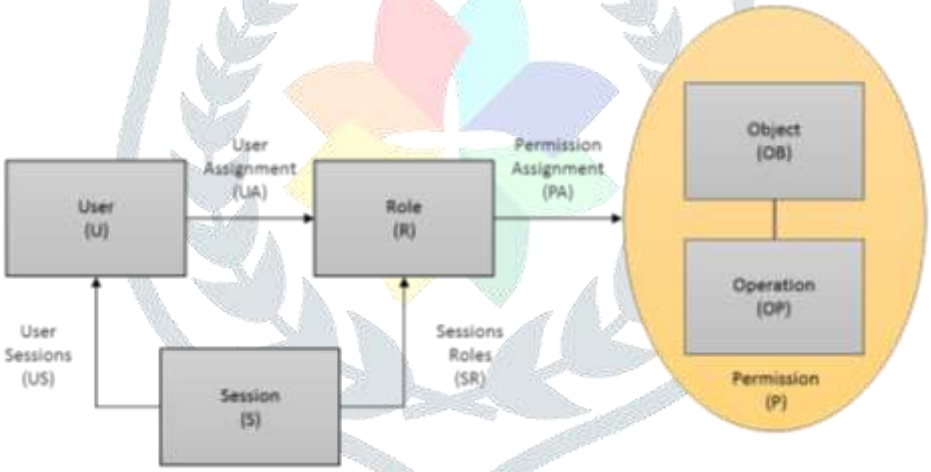

Fig.2 Core RBAC Model [15]

Hierarchical RBAC extends the Core RBAC model to add a hierarchy of roles which inherit their parent's permissions. Support for both a limited inheritance (roles being limited to one descendent) and a general inheritance (roles with any number of descendants and ascendants) are given in the NIST model. Hierarchical roles allow for simplified management of permission assignments and more closely model the relations between roles in real organizations. Two of the RBAC extension have been explained as follows:

1. Static Separation of Duty (SSD): relations extend the Core RBAC model to enforce simple separation of duty policies during user role assignment. Sets of two or more conflicting roles and the maximum cardinality of the intersection of users' roles with such a set are maintained in the system to represent an organization's SSD policy.

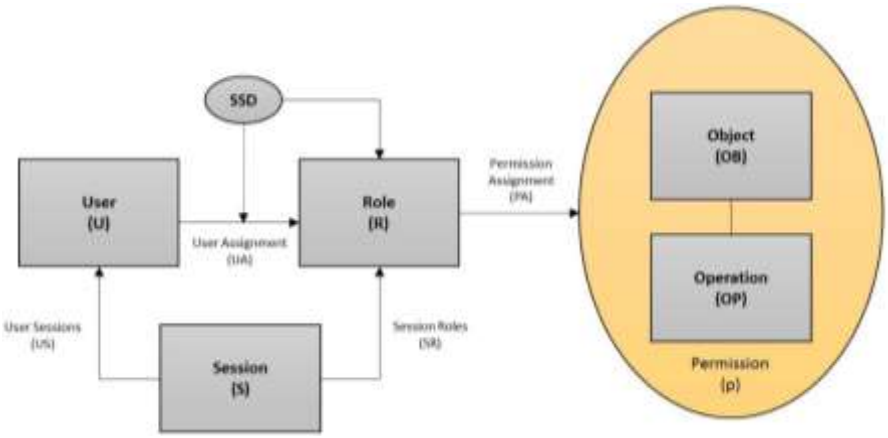

Fig.3 SSD RBAC [15]

2. Dynamic Separation of Duty (DSD): It also extend the core RBAC model to enforce an organization's separation of duty policies. However, unlike SSD, DSD constrains the active permissions a user may be indirectly assigned rather than their role assignment by limiting the roles a user may have active together in a single session. As with SSD, sets of conflicting roles and maximum cardinalities are used to represent the organization's DSD policy. However, in this case the intersection is between the conflicting roles and the user's active roles in a session rather than their overall role assignment.

1.4.1 OASIS RBAC:

OASIS role-based access control provides an architecture and model for secure service authentication and access with an open distributed environment. Unlike most role based access control models, OASIS does away with the traditional role hierarchy and does not use role delegation but instead an appointment process. Rather than relying on a centralized role administrator to delegate roles, the OASIS model uses a credential-based system for role activation whereby users may activate roles based on the current credentials they possess and conditions relating to the systems environment (e.g. current time, current task a user is performing, etc.). Credentials are granted through appointments initiated by any user who is a member of some role which grants the appointment ability for the given credential [9] [11]. For example, a user may be granted a new credential upon obtaining a professional qualification which will in turn grant access to a new set of roles in the system during role activation. The OASIS model supports separation of duty constraints at the role activation level (similar to dynamic separation of duty in the ANSI model) as a role activation rule.

1.4.3 RBAC: As a Service:

This section presents a new model for role based access control which aims to fulfil the required qualities of an RBAC system for the cloud. That is, be scalable, distributed, auditable, enforced confidentiality, provide simple administration and be reliable. This is accomplished by dividing the traditionally centralized RBAC system among multiple system domains. Each domain is enabled to create and assign roles to their users, create and assign their own permissions, and more importantly, extend the roles and permissions of other domains via inheritance in the role and/or permission hierarchies [4] [12]. Additionally, a system for conditional permissions and user groups is introduced. Users activating roles with conditional permissions are only granted the permission if set conditions are met at the time of role activation and the permissions use. Conditional groups only allow membership to a user if they meet a set condition at the time of authentication (e.g. a condition may require that a user be authenticating from a given network or IP range). The process of user parameterization allows users to be assigned parameters which may be checked in these conditions. For example, a parameter may be created which indicates that a user has been granted some safety certificate and a conditional group may be created to only allow users with the corresponding parameter to activate a given role (e.g. a role which has permissions for accessing and using equipment in a lab).

1.5 RBAC: Architecture

This section presents an overview of the proposed Attributes Enhanced Role-Based Access Control model (AERBAC). Figure 4 depicts our access control model and its components. The entities users, roles, objects and operations have the same semantics as in RBAC. Users and objects in our model are associated with attributes too [7] [12]. We also incorporate the environment attribute to fully capture the situation in which access needs to be authorized. The dotted-box in Figure 4 represents the modules of the architectural design to enforce this model. Below, we first describe the attributes and then discuss semantics of different components involved in AERBAC, including permissions, conditions, sessions and request evaluation.

1.5.1 RBAC Proposed Model

1. Attributes: Attributes capture the properties of specific entities (e.g. user). We define an attribute function for each attribute that returns the value of that attribute. Each attribute is represented by a range of finite sets of atomic values. For example, the range of branch attribute is a set of names of branches semantically relevant for the application domain. User attributes capture the properties of the user who initiates an access request. Examples of user attributes are title, specialization, location, security clearance etc.

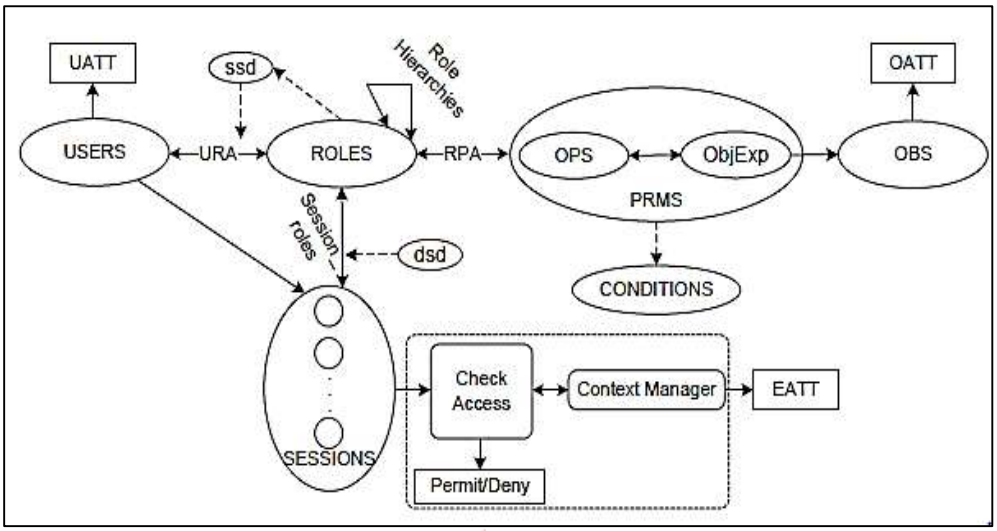

Fig.4 Attributes Enhanced Role Based Access Control (AERBAC) Model

Object attributes are used to define the properties of the resources protected by the access control policy. Example of object attributes include type, status, location, time of object creation etc. Environment attributes capture external factors of the situation in which the access takes place. Temperature, occurrence of an incident, system mode or other information which not only pertains to a specific object or user, but may hold for multiple entities, are typically modelled as environment attributes. An attribute may be either static or dynamic. The values of static attributes rarely change e.g. designation, department, type etc. On the other hand, dynamic attribute values may change frequently and unpredictably, so they may well change during the lifetime of a session. Examples of such attributes include command, location, occurrence of an incident etc. They are also referred to as contextual attributes in the literature [6].

2. Permissions and conditions: In contrast to the traditional approaches in RBAC, the permissions in AERBAC refer to objects indirectly, using their attributes. A permission refers to a set of objects sharing common attributes, e.g. type or branch, using a single permission, in contrast to separate permissions for each unique object. This is particularly relevant in those domains where several objects share common attribute values. This helps in significantly reducing the number of permissions associated with a role, while increasing the expressiveness and granularity of access control in a role-centric fashion. In our proposed model, a permission consists of an object expression and an authorized operation on the object set denoted by the expression. Object expressions are formed using the attributes of objects. Each permission is associated with one or more conditions, which must be evaluated to be true in order for the user to exercise that permission.

3. Session: A session contains a list of permissions associated with the roles activated by the user. As described earlier, the permissions are different from standard RBAC permissions in terms of referring to the objects using their attributes and being tied with the conditions that are evaluated every time a permission is to be exercised. Hence, the Check Access function needs to be re-defined.

4. Access request: An important consideration, in environments motivating the proposed approach, is that the user's request may also be based on the attributes of the objects. For a user request to be granted, there must exist an object expression in the user's session that denotes the requested objects, and the condition tied to that object expression must be evaluated to be true. There are different possibilities in which such a request may be evaluated.

1.6 RBAC: Model Analysis

Attribute-based access control (ABAC) and role-based access control (RBAC) are currently the two most popular access control models. Due to this fact, integration of RBAC and ABAC has recently emerged as an important area of research.

1.6.1 ABAC Analysis:

ABAC models can be broadly classified into two main categories; pure ABAC models and Hybrid models.

1. Pure ABAC Models: Recent efforts have aimed to take the first steps toward creating foundational models of "pure" ABAC (i.e., ABAC models that are not simply extensions to existing models, e.g., RBAC, but new attribute-based models that can be seen as a generalization of traditional models). These efforts can be subdivided into two categories, "general" and "domain specific." "Domain specific" models aim to provide ABAC for specific use cases such as cloud computing, web services, and the like, while "general" models aim to provide an ABAC solution that may be applied to any situation where access control is desired.

2. ABAC Hybrid: Hybrid models of ABAC aim to combine attributes into existing models of access control or to extend the traditional models with identity-less or policy-based access control concepts [3, 24]. This includes both early attempts at adding parameterized roles and permissions to RBAC as well as more modern efforts to unify ABAC with alternative access control models such as relationship-based access control (ReBAC) and behavior-based access control (BBAC) describe a number of hypothetical strategies for adding attributes to RBAC:

- **Dynamic Roles:** Roles are assigned dynamically based on the user's and environment's attributes, providing identity-less access control for RBAC-based systems.
- **Attribute-Centric:** Roles are considered to be just another attribute of a user. No role-permission relation is created and permissions are assigned through policies. If no special consideration for roles is provided in an Attribute-Centric model this could be seen simply as "pure" ABAC modelling RBAC. As this can be seen as equivalent to "pure" ABAC in most

cases, it is deprived of the advantages of RBAC (simple administration, auditability, straightforward separation of duties, etc.).

- **Role-Centric:** The maximum permission set available in a given session is constrained by attribute-based rules. Constraint rules are used only to reduce permissions available to the user and never expand them (differentiating it from role parameterization).

1.6.2 Formal AERBAC:

In this section, we propose the formal model that incorporates the attributes of the user, object and environment into RBAC in a role-oriented fashion. We define the sets and functions used in AERBAC given as follows:

- Sets and Functions used in AERBAC
- {USERS, ROLES, OBS, and OPS (users, roles, objects and operations respectively
- {UATT, OATT and EATT} represent finite sets of user, object and environment attribute functions respectively.

The sets and functions defined in NIST RBAC which are also applicable to AERBAC. We provide further sets and functions needed for AERBAC in the lower part of the table. UATT, OATT and EATT represent sets of attribute functions for users, objects and environment, respectively. The notion we used for attribute representation is adapted [13].

1.6.3 HA-RBAC Model Analysis:

The HA-RBAC model proposed in this section introduces the concept of attribute, through which the static-attribute-based roles and dynamic-attribute-based rules are built. The essential idea of HA-RBAC is that roles are a collection of static attributes, while rules are a collection of dynamic attributes. As illustrated in Figure 5, the components of a HA-RBAC model are as follows: , , , , , , and , which refer respectively to the set of users, roles, permissions, sessions, static roles, dynamic rules, static permissions and dynamic permissions. In addition, we propose two layers of roles which are referred () Usersu () Rolesr () PermissionsPerms() Sessionss() SRsr() DRdr() SPsp() DPdp as static-attribute roles ( ) and dynamic-attribute roles ( ), where role hierarchy ( ) in HA-RBAC are represented as SR DR RH SRH DRH.

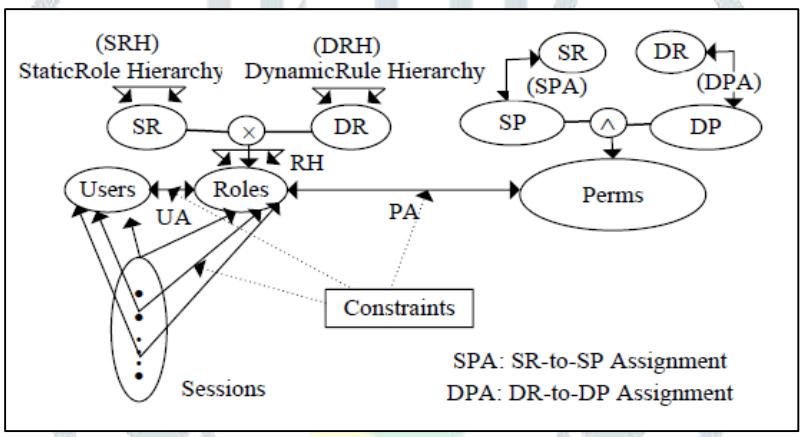

Fig.5 Model Structure of HA-RBAC

1.7 RBAC Model Benefits

We believe several limitations of the RBAC and ABAC approaches may be overcome using the approach we proposed. Below, we enlist some of these limitations and discuss how our approach overcomes these problems.

1. Fine-grained Access Control: Using the proposed approach, we may provide a near-grained access control mechanism without creating a large number of roles.

2. Context-aware Access: RBAC cannot easily handle dynamically changing attributes [7]. It typically does not support making contextual decisions unless many similar roles are created causing role-explosion problem. We provide a mechanism to incorporate these dynamically changing attributes in a role-centric manner yet without requiring to create a large number of roles.

3. Easy Auditing: Our approach makes it simpler to audit what permissions may be granted to a user because of being role-centric while adding the flexibility and access features offered by ABAC. When auditing for a particular position or employee, we need to consider only the policy rules given in the roles assigned to that position or employee.

4. Policy Modification Visualization: In our approach, it is relatively easy to visualize what the impact of adding is or removing a policy since policy specification is at the level of role. Therefore, a change in policy can elect only those users who are assigned to a role being modified.

II. Related Work

III. Zhihua, et.al [1] proposed a scheme using preserved-privacy content-based image allowing the owner of the data to perform image data-base outsourcing and cloud CBIR service without discussing the real data-base content. In order to represent the image, the local features are utilized and the EMD i.e. earth mover's distance evaluates the image similarity. The computation done by EMD is usually an issue of linear programming. The proposed scheme transformed the issue of EMD such that the cloud server can resolve it without using the mechanism of sensitive learning. Additionally, LSH i.e. Local Sensitive Hash was utilized to improve the efficiency of searching. The results have shown the tremendous analysis of efficiency and security of the system. Gang Liu, et.al [2] added the context to privacy information access, proposes a new method to realize context constraints, this method rates the context in which the subject is located, determines whether the access is allowed by judging whether the level satisfies the conditions. By adding context constraints to the access, the risk of privacy disclosure is greatly reduced and privacy protection is realized. Dariush Abbasinezhad-Mood, et.al [3] introduced the security goals that should be considered in an efficient trust-based system. The researchers proposed a novel trust and reputation based RBAC model that not only can properly withstand the security threats of trust-based RBAC models, but also was scalable as it has reasonable execution time. Third, we evaluated the proposed model using the famous trust network

of dataset. Eventually, we compare the proposed model with recently-published ones in terms of mean absolute error, execution time of indirect trust computation, and provided features. The achieved results are indicative of the priority of the proposed model to be employed in real cloud environments.

IV. Joseph K. Liu, et.al [4] provided a comprehensive survey of access control of users' data in the environment of fog computing with the aim of highlighting security problems and challenges. It discusses the definition, architecture, and characteristics of fog computing, based on which typical requirements and essential models of access control are addressed. Finally, it highlights known access control schemes in the environment of fog computing, and identifies existing unresolved problems as future directions. Aissam Outchakoucht, et.al [5] contributed to reinforcing the security of Big Data platforms by proposing a block chain-based access control framework. We define the concept of block chain and breakdown the mechanism and principles of the access control framework. Adopting the block chain technology to handle access control functions is not straightforward. It has successfully achieved the following objectives; Distributed nature and the lack of a central authority. User-driven and transparency; Light weightiness; Fine-granularity; Pseudonymise and Unlink ability. The researchers have implemented the Adaptive Dynamic Sharing and Privacy-aware Role Based Access Control (Adaptive DySP-RBAC) model which provides user's information privacy to dynamically adapt the changes occurring in the system at any time. The proposed model has been implemented as a prototype and tested. Results have shown that our system efficiently and effectively adapts access rules according to the changes happening in a CWE along with preserving the user's information privacy in the system. Muhammad Anwar, et.al [6] implemented the Adaptive Dynamic Sharing and Privacy-aware Role Based Access Control (Adaptive DySP-RBAC) model which provides user's information privacy to dynamically adapt the changes occurring in the system at any time. The proposed model has been implemented as a prototype and tested. Results have shown that our system efficiently and effectively adapts access rules according to the changes happening in a CWE along with preserving the user's information privacy in the system. Antonis Michalas, et.al [7] outlined significant security challenges presented when migrating to a cloud environment and propose PaaS word – a novel holistic framework that aspires to alleviate these challenges. Specifically, the proposed framework involves a context-aware security model, the necessary policies enforcement mechanism along with a physical distribution, encryption and query middleware. Ahmad Kamran, rt.al [8] implemented the Adaptive Dynamic Sharing and Privacy-aware Role Based Access Control (Adaptive DySP-RBAC) model which provides user's information privacy to dynamically adapt the changes occurring in the system at any time. The proposed model has been implemented as a prototype and tested. Results have shown that our system efficiently and effectively adapts access rules according to the changes happening in a CWE along with preserving the user's information privacy in the system. Sneha Warang, et.al [9] presented in literature to control unauthorized access to data. One such technique is RBAC (Role Based access Control) model. RBAC method controls the access to data based on roles given to individual users within an organization. RBAC model provides flexible control and management using two simple mappings first is User to their role in organization and second is Roles to accessible data to that Role. We propose RBE (Role Based Encryption) scheme which combines encryption technique with traditional RBAC model. In this system role hierarchy is used for efficient user management. David Mun-Hien Choy, et.al [10] Embodiments of the present invention provide an efficient and Scalable scheme for role-based access control to resources. The resources are assigned a protection class. Resources in the same protection class share the same access control policy. Permissions granted to various roles are then defined based on privilege sets and protection classes. Accordingly, the permissions of a role can be dynamically determined at runtime. Furthermore, as new resources are added, they can be assigned to a pre-existing protection class. The new resource may thus automatically inherit the various permissions and roles attached to the protection class. Jian Zheng, et.al [11] deeply proposed a research the mapping relationships of roles and attributes, propose a combination of static-attribute-based roles and dynamic-attribute-based rules to simplify the management of access control policy and downsize the access control system, while we formally define the construction of HA-RBAC model and access control algorithm. Comparative analysis and simulation experiments showed that this model can not only adapt to the role of fine-grained division and simplify policy management, but also improved the efficiency of access control, and reduced system consumption. James BD Joshi, et.al [13] proposed a comprehensive security framework for cloud computing environments. The researchers also discussed the challenges, existing solutions, approaches, and future work needed to provide a trustworthy cloud computing environment. We have presented a comprehensive security framework for cloud computing environments. We have described its components, discussed existing solutions and identified possible approaches to deal with different security issues related to the cloud. Jorge Lobo, et.al [14] proposed an extended role-based access control (RBAC) model, called Privacy-Aware Role-Based Access Control (PARBAC) model, for enforcing privacy policies within an organization. The PARBAC model combines RBAC, Domain-Type Enforcement, and privacy protection by modelling business purposes and data policies. Consented consumer privacy preferences are recorded as data policies, which govern how to use actual consumer data. One of the key elements in a privacy policy was purpose. The actual purpose of a business operation to consumer data must be consistent with the purpose consented by the consumer. This is the so-called purpose binding privacy requirement. This paper focuses on enforcing this requirement. Privacy enforcement mechanism with the PARBAC model was then discussed and a privacy scenario is illustrated to describe its application.

III. THE PROPOSED METHOD

3.1 Proposed Methodology

In order to create a hybrid model and apply the features mentioned earlier we are required to create a private cloud. We could use either OpenStack or Own Cloud as a tool for building either a public cloud or a private cloud. OpenStack or Own Cloud not only enables us to create cloud computing platforms but they also provide a dashboard through which one can customize and manage the cloud. Since OpenStack is an open source software, we could easily modify the access control policy within the cloud. Another tool we are going to use is a policy language called "XACML". XACML stands for "eXtensible Access Control Markup Language". Using this language, we can define a declarative fine-grained, attribute-based access control policy language. Abbreviated Language for Authorization (ALFA) is a pseudo code language which maps directly into the eXtensible Access Control Markup Language (XACML) and contains the same structural elements as XACML.

3.2 Proposed Methodology: Flowchart

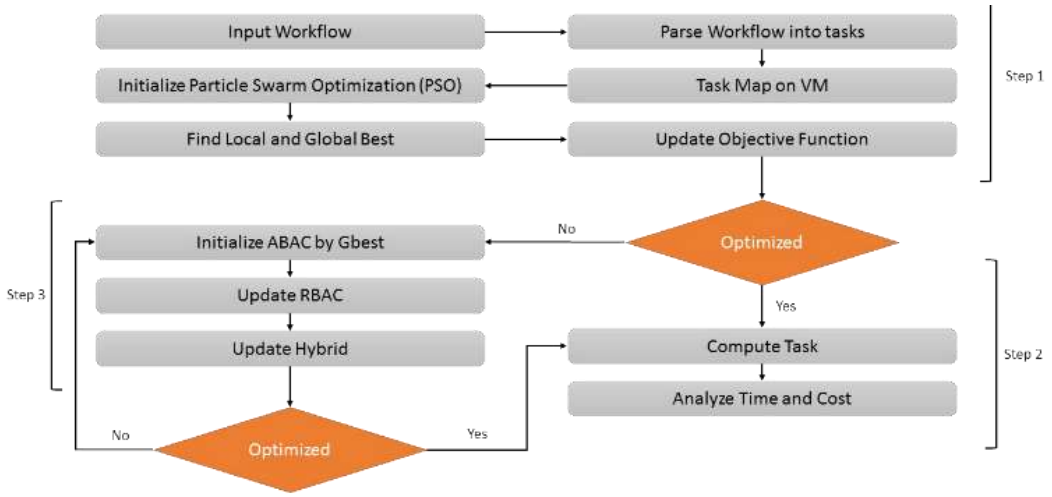

Fig.6 Flow Diagram of Proposed Methodology

IV. Result Analysis

4.1 Simulator Used:

The proposed work is done on the CloudSim simulator for using virtually cloud environment workflows. CloudSim provides a platform for modeling, simulation and experimentation of cloud computing. This simulator provides the user cloud system and investigated without concerning the low level details. CloudSim is basically a library for the simulation of cloud scenarios. It provides the feature in which useful classes are available for describing the data centers, virtual machines, applications and users. CloudSim also provides the facilities of scheduling and provisioning to manage the cloud resources. It is also used as a perspective of cost, execution time and application.

4.2 Scientific Workflows Used:

- SIPHT: It is used to automate the search for untranslated RNA for bacterial replicons in the NCBI database.
- Montage: It is created by NASA/IPAC together input image to create custom mosaics of the sky.
- CyberShake: This workflow is used by the California to characterize the earthquake hazards in a region.

4.3 Result Analysis:

The proposed methodology is implemented with the help of CloudSim and Eclipse. CloudSim is the library that provides the cloud computing simulation environment and also provides core classes that describe virtual machines, data centers, users and applications.

4.3.1 Comparison of RBAC, ABAC and HYBRID Using SIPHT:

Table 1 demonstrates the behavior of SIPHT workflow application with respect to virtual machines of different size. The results show the RBAC and hybridization of particle swarm optimization and GWO on total execution time, total execution cost and response time.

Table.1 TET-TEC Comparison Table of RBAC, ABAC and HYBRID Using SIPHT

	RBAC		ABAC		Hybrid	
VM	Time	Cost	Time	Cost	Time	Cost
2'	33.09	9695.465	23.06	5620.855	5.59	4550.063
4'	43.06333333	12329.64333	41.16	10009.35	12.26	8603.69
6'	58.54666667	14839.89	38.05	13456.19	14.13	12125.43
8'	66.69666667	17173.36	52.98	13523.39	34.83	14265.19
10'	66.58333333	19887.44333	87.61	17540.09	29.77	16958.17
12'	58.11666667	22636.07667	62.5	20456.6	32.34	19965.58
14'	68.38333333	26978.06333	52.64	21665.64	33.06	22432.37
16'	74.85	31127.18	62.21	25785.99	55.59	30112.12
18'	81.67	33797.775	93.3	33482.56	75.36	33230.s33
20'	71.04	34112.99	72.04	34112.99	62.55	33012.34

4.3.1.1 TET Comparison Graph of RBAC, ABAC and HYBRID Using SIPHT:

In figure 7, X-axis represents the number of virtual machines, and Y-axis indicates the total time (in ms) of execution in different models. Using 2 VMs, the execution time for RBAC is 33ms, for ABAC is 23ms and for HYBRID is 5.5ms. Using 10 VMs, the execution time for RBAC is 66ms, for ABAC is 87ms and for HYBRID is 29.7ms. Using 20 VMs, we get 71ms for RBAC, 72ms for ABAC and 62ms for Hybrid. Overall, we can see that the total execution time of our hybrid model is less than the other two approaches. This is because the hybrid searching time is first decided by PSO and if not optimized it will be determined by GWO. In addition, the VM task migration in the hybrid model depends on the transient state and relies on Pareto distribution which depends on normal distribution.

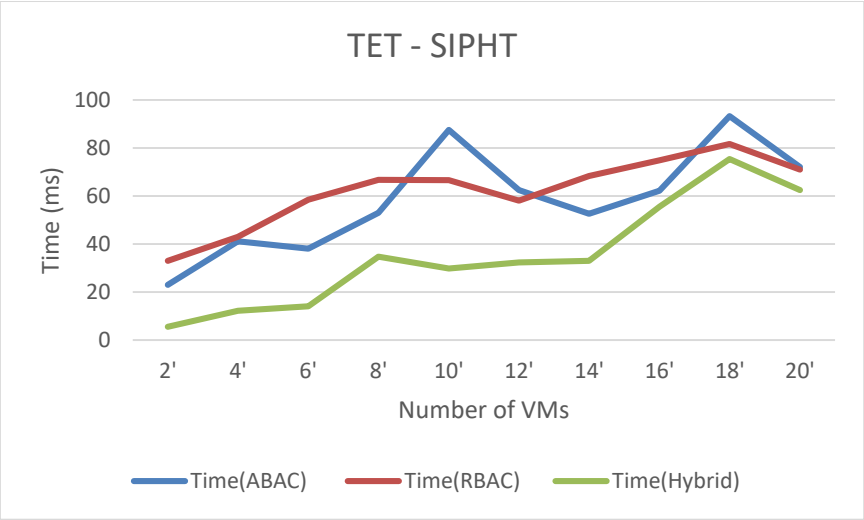

Fig.7 TET Comparison graph of RBAC, ABAC and HYBRID using SIPHT

4.3.1.2 TEC Comparison Graph of RBAC, ABAC and HYBRID using SIPHT:

In figure 8, X-axis represents the number of virtual machines and Y-axis indicates the total execution cost (in Rupees which are converted into USD) of the three models. Using 2 VMs, the total execution costs of RBAC, ABAC and the hybrid model are 135$, 78$ and 63$ respectively. Using 10 VMs, we get 278$, 245$ and 237$ for RBAC, ABAC and hybrid respectively. Similarly, if we use 20 VMs, we get 478$, 478$ and 462$ for RBAC, ABAC and hybrid respectively. Based on these results, we can therefore conclude that our proposed HYBRID model performs better as the cost has been reduced.

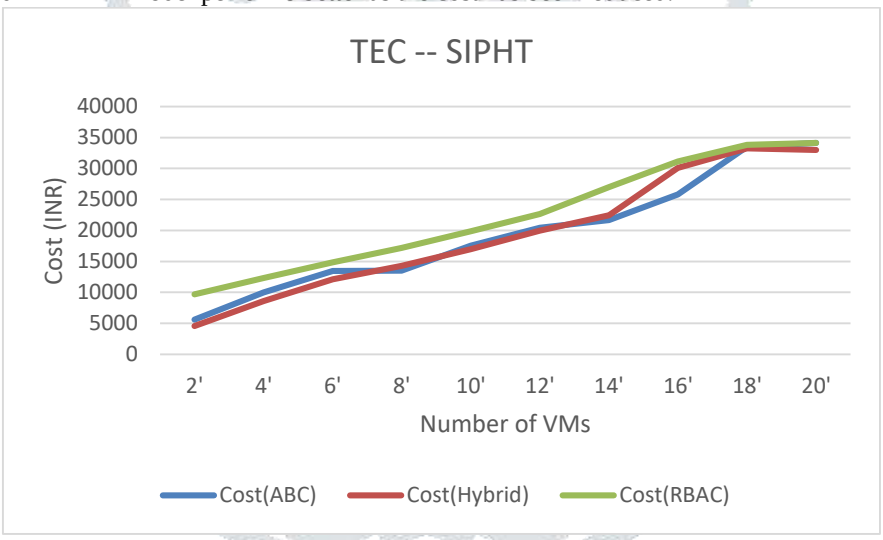

Fig.8 TEC Comparison Graph of RBAC, ABAC and HYBRID using SIPHT

4.3.2 Comparison of RBAC, ABAC and HYBRID Using Montage

Table 2 illustrates the behavior of Montage workflow application with respect to virtual machines of different size. The results show the total time and cost execution of RBAC, ABAC and HYBRID models.

Table.2 Comparison table of RBAC, ABAC and HYBRID using Montage

	RBAC		ABAC		HYBRID	
VM	Time	Cost	Time	Cost	Time	Cost
2'	10.92666667	358.55887	0	0	0	0
4'	17.91	602.22177	15.09	447.3718	1.73	299.958
6'	21.37	817.50017	17.69	628.3048	1.82	356.9556
8'	24.77	947.18723	20.95	730.9887	4	496.4608
10'	28.17	1146.799	25.47	1093.207	4.8	496.4608
12'	30.41333333	1211.9447	27.89	1017.366	6.84	1147.339
14'	32.80666667	1385.6223	31.15	1329.824	7.61	1131.746
16'	34.52333333	1415.3067	32.2	1288.644	9.52	1480.906
18'	35.685	1478.638	35.07	1538.399	8.66	1278.957
20'	36.3	1418.877	36.3	1418.877	11.4	1745.794

4.3.2.1 TET Comparison Graph of RBAC, ABAC and HYBRID Using Montage

In figure 9, X-axis represents the number of virtual machines, and Y-axis indicates the total execution time. Using 2 VMs, the execution time of RBAC, ABAC and HYBRID model is 10.9ms, 0ms and 0ms respectively. Using 10 VMs, we get 28ms, 25ms and 4.8ms for RBAC, ABAC and HYBRID respectively. Similarly, if we use 20 virtual machines, the total execution time will be 36ms, 36ms and 11ms for RBAC, ABAC and HYBRID respectively. Based on the obtained results, we can conclude that these HYBRID parameters work well in term of time parameter due to the search time of colon ants by adaptive pheromones.

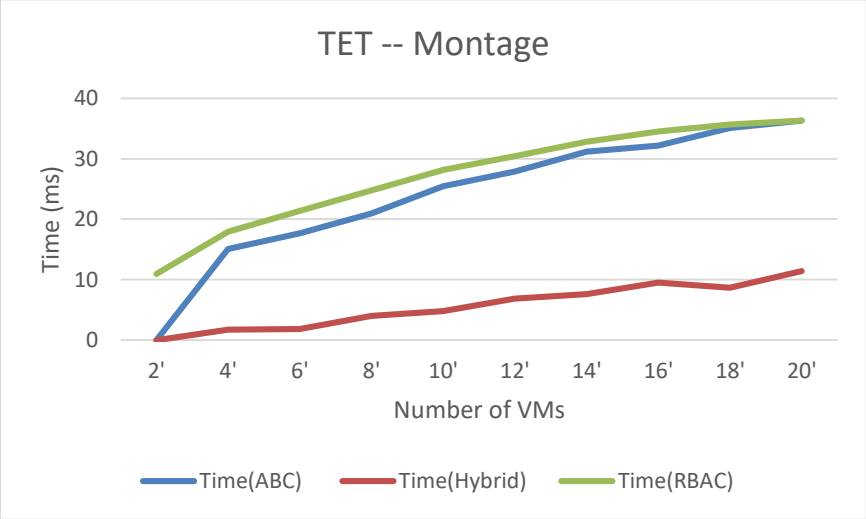

Fig.9 TET Comparison Graph of RBAC, ABAC and HYBRID Using Montage

4.3.2.2 TEC Comparison Graph of RBAC, ABAC and HYBRID Using Montage

In figure 10, the X-axis represents the number of virtual machines and the Y-axis indicates the total execution cost of each model. Using 2 VMs, the execution cost of RBAC, ABAC and HYBRID is 4.9 $, 0 $, 0 $ respectively. If we use 10 VMs, then the execution cost will be 15.8 $, 15 $, and 6.8 $ for RBAC, ABAC and HYBRID respectively. Similarly, if we use 20 VMs, then we get 19.5 $, 19.5 $ and 24 $ as the execution cost for RBAC, ABAC and the HYBRID respectively. Based on the obtained result, we can conclude that the total average of the execution cost of the HYBRID model is less than the other two models. This is because the ant colony searching time is decided by adaptive pheromones. In this scenario, the VM task migration depends on the Transient status and the algorithm depends on the candidate solution which is again static. In our HYBRID model, the initialization depends on the Pareto distribution which depends on normal distribution.

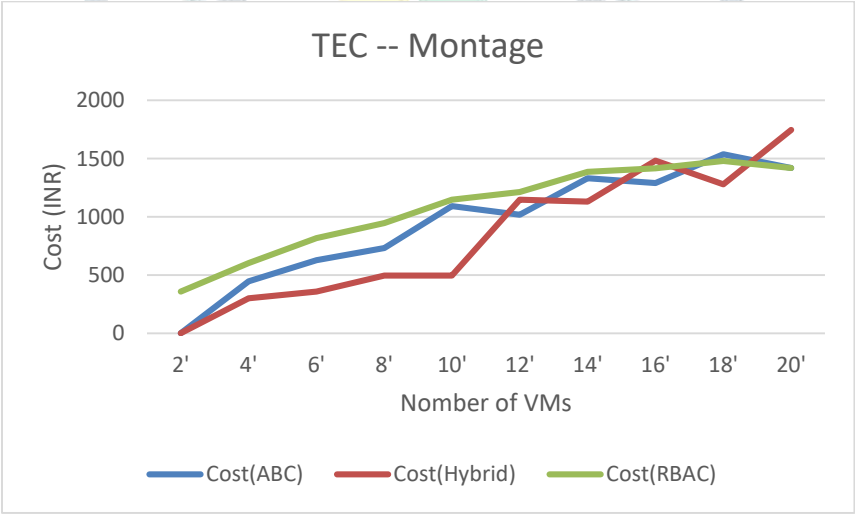

Fig.10 TEC Comparison Graph of RBAC, ABAC and HYBRID Using Montage

4.3.3 Comparison of RBAC, ABAC and HYBRID using CyberShake

Table 3 illustrates the behavior of CyberShake workflow application with respect to virtual machines of different size. The results demonstrate the optimization of the ant colony and the optimization of the genetic algorithm with respect to the total execution of time, the total execution of cost as well as the response time.

Table.3 TET-TEC Comparison Table of RBAC, ABAC and HYBRID Using CyberShake

	RBAC		ABAC		HYBRID	
VM	Time	Cost	Time	Cost	Time	Cost
2'	14.27333333	653.215167	0	0	0	0
4'	22.95666667	1168.53583	18.57	822.6585	2.4	687.9981
6'	26.71666667	1522.043	24.25	1136.987	3.36	950.981
8'	28.89333333	1719.51067	26.05	1545.962	6.92	1518.366
10'	32.47666667	2047.61233	29.85	1883.18	4.5	1689.733
12'	33.97333333	2122.101	30.78	1729.39	8.08	1724.117
14'	36.30333333	2316.05533	36.8	2530.267	10.96	2463.681
16'	37.20333333	2319.93567	34.34	2106.646	9.25	2731.877
18'	38.635	2426.5805	37.77	2311.253	11.58	1989.96
20'	39.5	2541.908	39.5	2541.908	11.42	1982.903

4.3.3.1 TET Comparison Graph of RBAC, ABAC and HYBRID Using CyberShake

In figure 11, the X-axis represents the number of virtual machines and the Y-axis indicates the total time of execution of the three models. Using 2 VMs, the execution time is 14ms for RBAC, 0ms for ABAC and 0ms for the HYBRID model. Using 10 VMs, the execution time changes to 32ms for RBAC, 29.8ms for ABAC and 4.5ms for the HYBRID model. Similarly, if we increase the number of virtual machines to 20, then the execution time will change to 39.5ms, 39.5ms and 11.42ms for RBAC, ABAC and the HYBRID model respectively. Based on the results, the HYBRID performs considerably better with respect to the time parameter, because the optimization time of particle swarm is decided by the adaptive pheromones. The migration of the VM tasks depends on the Transient status where in genetic algorithm, it depends on the candidate solution which is static. The initialization of HYBRID model however, depends on the Pareto distribution which depends on normal distribution.

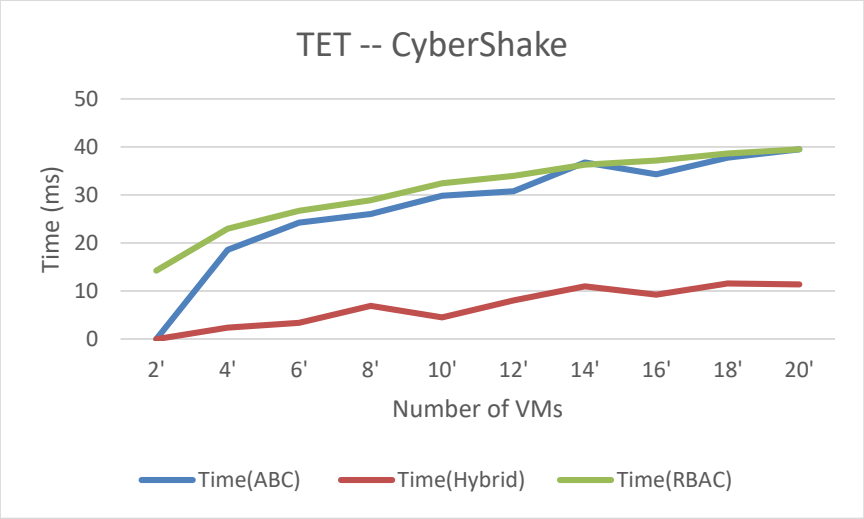

Fig.11 TET Comparison Graph of RBAC, ABAC and HYBRID Using CyberShake

4.3.3.2 TEC Comparison Graph of RBAC, ABAC and HYBRID Using SyberShake

In figure 12, the X-axis represents the number of virtual machines and the Y-axis indicates the total cost of execution of the three models. Using 2 VMs, we get the execution cost as 9$ for RBAC, 0$ for ABAC and 0$ for the HYBRID model. Increasing the number of VMs to 10, we get the execution cost as 28$ for RBAC, 26$ for ABAC and 23$ for the HYBRID model. Similarly, if we again increase the number of VMs to 20 then we get the execution cost of 35$, 35$ and 27$ for RBAC, ABAC and the HYBRID model respectively. We can again conclude that the HYBRID model performs better than the other two models when it comes to the total execution time.

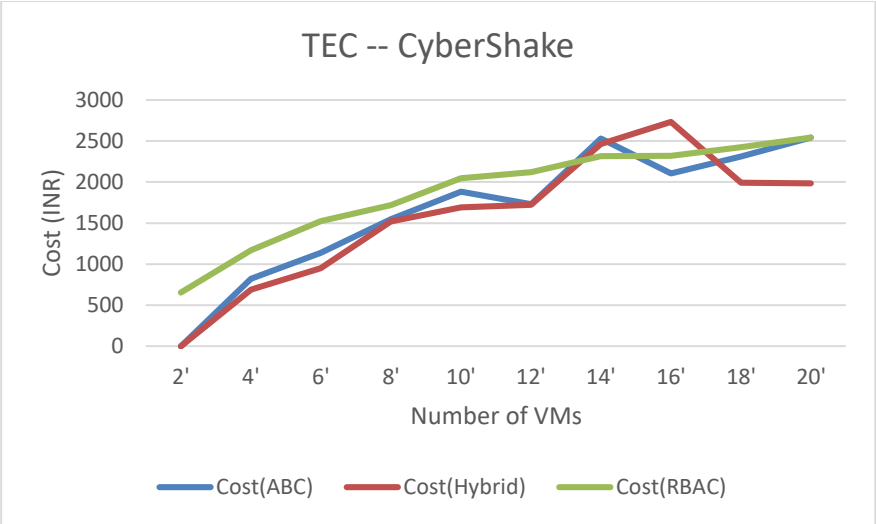

Fig.12 TEC Comparison Graph of RBAC, ABAC and HYBRID Using Cybershake

V. CONCLUSION

In this work, we proposed the scheduling mechanism for execution of the sensible forms on the IaaS clouds. The main issue in the cloud computing while minimizing makespan is the execution cost. This issue is solved by using Hybrid PSO with GWO. The tests were directed by mimicking three surely understood work processes (SIPHT, Montage and CyberShake) on Cloudsim, which demonstrates that our answer is generally more beneficial in terms of execution than other existing algorithms.

The given graphs and tables represented a comparative analysis of TET and TEC parameters on the basis of Bio inspired Optimization (RBAC) and Particle Swarm optimization (PSO) with Grey Wolf Optimization (GWO). In our experiment, we used workflow scheduling in cloud environment with the utilization of different types of scientific workflows.

In our analysis, the total cost and execution time which depend on the initializing factors are improved by optimization. In the proposed approach, we use Pareto distribution instead of random distribution.

If random distribution is used, more time will be required to converge. The convergence however, is sometimes enforced by iteration but that will increase the computation and execution time, therefore does not meet the deadline condition.

Thus, task initialization is of crucial factor as defined in this paper. Additionally, we observed that our proposed HYBRID model performs better that the other two models for the reduction of cost and time as a result of random crossover.

The worthy results are achieved as a result of using two optimization algorithms, PSO (Particle Swarm Optimization) and GWO (Grey Wolf Optimizer) as they play a key role in the global optimization. With the proposed approach, the work processes can be delivered more efficiently as both the time and cost have been reduced.

VI. FUTER WORK

In future, we might need to research distinctive choices for the preliminary resource pool as it is being influenced by the execution of the figuring. Various optimization approaches and algorithms need to be examined for their execution. Another future task would be analyzing the data exchange between the servers to show that virtual machines may use in various areas. Last but not least, we need to actualize our approach in a work process motor so that it may be used for applications.

VII. ACKNOWLEDGMENT

I would like to place on record my deep sense of gratitude to Mr. Keshav Kishore, Associate Professor, School of Technology and Science, AP Goyal Shimla University, Shimla, for his stimulating guidance, continuous encouragement and supervision throughout the course of present work.

REFERE NCES

[1] Xia, Zhihua, Yi Zhu, Xingming Sun, Zhan Qin, and Kui Ren. "Towards privacy-preserving content-based image retrieval in cloud computing." IEEE Transactions on Cloud Computing 6, no. 1 (2018): 276-286.

[2] Wu, Zili, Xiaoqian Qi, Gang Liu, Lu Fang, Jiayu Liu, and Juan Cui. "An extend RBAC model for privacy protection in HIS." In Digital Forensic and Security (ISDFS), 2018 6th International Symposium on, pp. 1-6. IEEE, 2018.
Ghafoorian, Mahdi, Dariush Abbasinezhad-Mood, and Hassan Shakeri. "A Thorough Trust and Reputation Based RBAC Model for Secure Data Storage in the Cloud." IEEE Transactions on Parallel and Distributed Systems (2018).

[4] Zhang, Peng, Joseph K. Liu, F. Richard Yu, Mehdi Sookhak, Man Ho Au, and Xiapu Luo. "A Survey on Access Control in Fog Computing." IEEE Communications Magazine 56, no. 2 (2018): 144-149.

[5] Es-Samaali, Hamza, Aissam Outchakoucht, and Jean Philippe Leroy. "A blockchain-based access control for big data." International Journal of Computer Networks and Communications Security 5, no. 7 (2017): 137.

[6] Malik, Ahmad Kamran, Muhammad Anwar, Abdul Mateen, Wajeeha Naeem, Yousra Asim, Malik Ahsan Ali, and Basit Raza. "Rule Adaptation in Collaborative Working Environments using RBAC Model." International Journal of Advanced Computer Science and Applications 8, no. 3 (2017): 452-457.

[7] Verginadis, Yiannis, Antonis Michalas, Panagiotis Gouvas, Gunther Schiefer, Gerald Hübsch, and Iraklis Paraskakis. "Paasword: A holistic data privacy and security by design framework for cloud services." Journal of Grid Computing 15, no. 2 (2017): 219-234.

[8] Malik, Ahmad Kamran, Muhammad Anwar, Abdul Mateen, Wajeeha Naeem, Yousra Asim, Malik Ahsan Ali, and Basit Raza. "Rule Adaptation in Collaborative Working Environments using RBAC Model." International Journal of Advanced Computer Science and Applications 8, no. 3 (2017): 452-457.

[9] Sneha Warang, Tabassum Maktum, "Role Based Access Control Model for Cloud Computing Using RBE Scheme" International Journal of Advanced Research in Computer Science and Software Engineering. Vol. 6, no. 10, 2016

[10] Beedubail, Ganesha, David Mun-Hien Choy, Hui-I. Hsiao, Sriram Raghavan, and Ganesh Vaideeswaran. "System and method for role-based access control in a content management system." U.S. Patent 9,455,990, issued September 27, 2016.

[11] Cai, Ting, Jian Zheng, and Xing Du. "A Hybrid Attribute based RBAC Model." International Journal of Security and Its Applications 9, no. 7 (2015): 317-328.

[12] Daniel Servos. (2012) A Role And Attribute Based Encryption Approach To Privacy And Security In Cloud Based Health Services. Lakehead University

[13] Takabi, Hassan, James BD Joshi, and Gail-Joon Ahn. "Securecloud: Towards a comprehensive security framework for cloud computing environments." In Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual, pp. 393-398. IEEE, 2010.

[14] Ni, Qun, Elisa Bertino, Jorge Lobo, and Seraphin B. Calo. "Privacy-aware role-based access control." IEEE Security & Privacy 7, no. 4 (2009).

[15] General inheritance Hierarchical RBAC. Accessed on: Oct 27, 2018. [Online]. Available:https://www.researchgate.net/figure/General-inheritance-Hierarchical-RBAC_fig11_310798876

Mojtaba Mohammadi is a post graduate student (M.Tech) in computer science and engineering at AP Goyal Shimla University, where his area of research revolves around information security as well as cloud computing. He graduated from American University of Afghanistan (AUAF), with a bachelor degree in computer science and information technology in 2014. His interest areas are information security, cloud computing and access control mechanism for clouds. Currently he is doing research on cloud computing security.

Keshav Kishore is currently serving as trainer at AP Goyal Shimla University, Shimla (H.P.). He is also serving as Associate Professor in the Department of Computer Science & Engineering. He has played vital role in the implementation of New IT-Labs and Web application Development for the various projects of the University. He is having 4 years of Industry experience at various levels. He is expert in FOSS; Application Development & Security Management. He is having more than 5 Years of academic and administrative experience in the field of Computer Science & Information Technology. He has published more than 14 research manuscripts in various International & National journals &conferences. He has also presented papers in International and National conferences. He is the member of various International & National professional & academic bodies. In addition to this, he is the member of various committees and BOS (Board of Studies). He has guided the students in their M. Tech dissertations. He is an active participant of various Seminars, Induction and faculty development programs.