An access control and authorization model with Open stack cloud for Smart Grid

Yagnik A. Rathod[1]
Dr. Chetan B. Kotwal[2]
Dr. Sohil D. Pandya[3]
Divyesh R. Sondagar[4]
[1] *Research Scholar, Computer/IT Engineering, Gujarat Technological University, Ahmedabad, Gujarat, India*
[2] *Professor& Head, Electrical Engineering Department, SVIT, Vasad, Gujarat, India*
[3] *Asst. Professor& Head, MCA Department, SVIT, Vasad, Gujarat, India*
[4] *UG Student, Computer Engineering Department, Government Engineering College, Dahod*
[1] *rathod.yagnik@gmail.com,* [2] *chetan.kotwal@gmail.com,* [3] *sohilpandya@gmail.com,*
[4] *divyeshsondagar135@gmail.com*

Abstract— Access control and authorization are always an area of interest for researchers to enhance the security of critical assets for many decades now. Our prime focus and interest are in the field of access control model based on Attribute-Based Access Control (ABAC). The realization of Smart Grid demands that critical infrastructures like the traditional electrical grid open ups to modern information and communication technology for getting the benefit in terms of efficiency, scalability, accessibility, and transparency having a greater level of bi-directional interaction among stakeholders like a customer, generation units, distribution units, and administrations. Cloud computing has proven to be the most efficient approach for a smart grid domain to accommodate the security requirements of participating legacy systems. The proposed approach integrates the ABAC model to OpenStack cloud with its default RBAC approach for achieving a more elegant level of granularity in access policies for a Smart Grid domain. The proposed method extends that default supports and integrates multiple access control policies in making authorization decisions. For Experimental analysis, a case study of the smart grid domain has opted that requires the support of multiple access policies (like RBAC, ABAC, DAC, etc.) with our model for access control and authorization.

Keywords*—Smart grid, Cloud Computing, ABAC, access control, authorization*

I. Introduction

In the early day of computing, access control is all about to support access rules that apply to various users to define the access permission in terms of read and write operations to the specific resource. Most of the domains confine to independent and individual user entities and which are mainly the secluded system by nature. Gradually, computing starts to become modern and enhances its capability through the technological advancement made towards the real distributed systems. Computing solutions based on some superior and an inter-reliant system becomes the first choice of relevant domains. Today's modern era demands that applicable policies imposed within and among the services which support the massive amount of heterogeneous data, record management, and work. Services requesting to access the resource with the type of the target resource confirm the applicable policies which can impose for that request. These policies need to be associated with the participating entities. Today's policy enforcement needs to contend with a large variety of operations like read/write, send, review, approve, insert, and copy/cut-paste, etc. Policy enforcement becomes more challenging as these operation types apply to a large variety of data types like files, messages, attachments, work items, records, fields, and clipboards. These kinds of operations and actions are executable under the control of different arrangements where applications are often running

concurrently within co-existence and interrelations, therefore causing the circumstances much more difficult to handle. The capability of an institute to impose the access control policies affects its abilities to achieve the security intention decently by formative the level to which its amount of data may be confined and shared among its user group. The policy is an absolutely crucial component to deliver efficient authorization, and due to that different characteristic of policies should be identified carefully for delivering the appropriate security system. The policy characteristics that become very important are listed below.

- Flexible policy capabilities that virtually accumulate every policy for configuration and enforcement.
- The capabilities to combine the policies for effectively securing the resources.
- The Scope of policies should imply the entire organization level.
- The comprehensive nature of policy imposition for the access requests of each user and its task.
- Capabilities to manage all traversal of bidirectional data among applications among the processes.
- Consistent in handling the information migrating outside the control of the decision system to prevent the concerned objects and provide assurance for competence.

The tiresome work regarding the configuration of the access control and the computational time it takes remained conventionally a primary reason for disappointment at the users' end and system administrators alike. This disappointment and dislikes becomes the prime culprit of severe security vulnerabilities due to improper configuration. Administrators must autonomously manage and regulates an extensive volume of accounts attached to each user and harmonize the access-control policies across different data services integrated through various interfaces. Users must have authorization through different authorization schemes for the sake of exercising their endorsed capabilities through various data services. Any security mechanisms must be implemented within the applications with a vision towards a greater revelation for attack and get around. Implementation in a cloud environment (where data services, users, data objects, and access control policies easily accommodate and managed) supports fulfilling the aspirations of the subscriber by tendering him the capabilities about filling an essential gap of authorization and distribution system like Smart Grid. Definement of smart grid as per US NIST is "a modernized grid that enables bidirectional flows of energy and uses two-way communication and control capabilities that will lead to an array of new functionalities and applications" [1]. Figure 1 provides a conceptual representation of the Smart Grid[2].

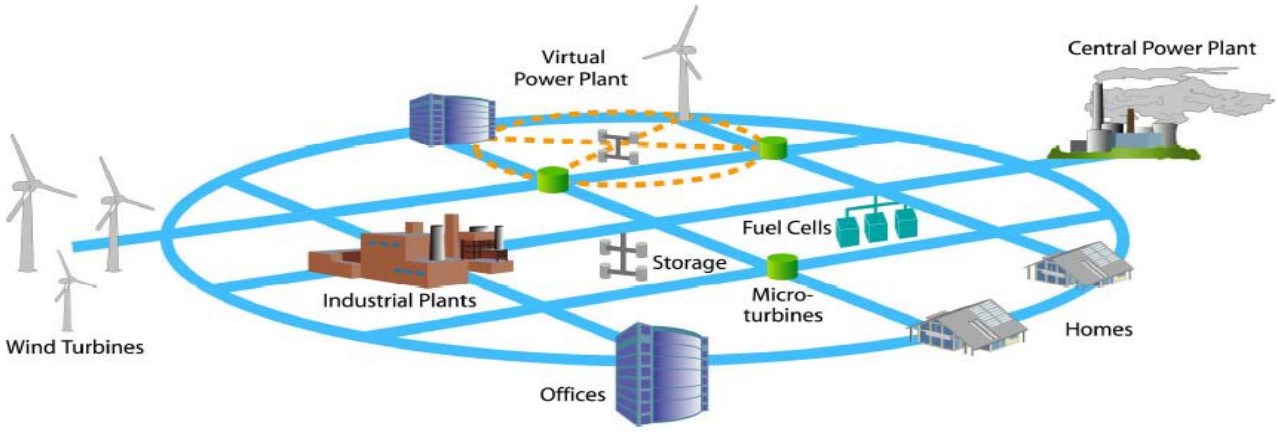

Figure 1 Smart Grid Structure

New Grid Components identifies to actively participate in the realization of the concept of smart grids are Distributed Generation, Plug-in Hybrid Electrical Vehicles, Micro turbines and clean energy generation, Sensing and Control Devices at the physical level.. ICT incorporation is much needed to get the essential supports like Communications Infrastructure, Automation & IT Backend services, Advanced Analytic features concerning Self-Healing & Adaptive capabilities, National Integration of the grid, Bi-directional Interaction during active participation of customers & energy markets, enhanced the Cyber Security, Upgraded Quality of Power, efficient integration of a Wide Diversity of Generation types, and Increase transparency for the realization of the Smart Grid [2]. Access to electricity is recognized to be fundamental human rights, which becomes a critical component for the prosperity, safety, and general well-being of every human being as per the report of The National Association for the Advancement of Colored People

(NAACP). Significant challenges that can constraint the Smart Grid are recognizes as the integration of various types of generation facilities ranging from traditional generation (thermal, coal, hydro, etc.) to clean energy generation (Wind, Solar, etc.). It's very crucial to enhance effectiveness by the best way of utilizing available resources on hand and also minimizing the loss at different process levels without overlooking the scrupulous environmental rules. The most promising solutions among the many solutions to win despite having the limitations and challenges of the available electrical grid can be an intelligent grid, referred to as a smart grid. The smart grid is accepted as a prominent solution due to its adaptability by the real world because of the technological enhancement extending the integration of ICT with the power system domain. The integration of various stakeholders makes smart grids a complex architecture that needs to be effectively supported and facilitated by the ICT systems and that integration can become a game-changer for realizing the concept of the Smart Grid[3]. The crucial involvement and integration of ICT to provide sound and effective access mechanism through high bandwidth bi-directional communication technologies strongly requires the modern, well suited, and customizable security solutions for such a large, distributed, and wide-area communication networks like smart grid[4]. The Cyber Security challenges for the Smart Grids can be catalogs as connectivity among stakeholders, appropriate new trust models, security administration and management, Software vulnerabilities – Malware attacks, Consumers' privacy, and Human factors like cultural differences and lack of technological understanding[5]. At present, Cloud computing becomes a choice and preference for numerous users because it facilitates efficient data storage at various data centers provides by cloud services, and supplement to that SCA property of the cloud makes it the superior candidate for smart grid solutions[6]. OpenStack is a cloud operating system that manages an ample number of stakes like resources for computation and data storage, networking facilities all over a datacenter that all controlled, handled, and made available through APIs with universal authentication means. Additionally, a dashboard facility is available to give administrators the power of managing, controlling the resources as well as giving authority to their users to obtain available the resources via a web interface. as per the customized needs of the requirement. The prime reason for adopting an OpenStack cloud is it's an open-source platform. The open-source platform has a collection of virtual resources to construct and deal with various types of clouds where anybody can have the right to access the source code, formulate any changes, or customize the modifications as per their preferences and generously distribute these changes to the community. OpenStack is fundamentally a succession of instructions referred to as programmed scripts that are combined and presented as one unit commonly referred to as projects and executed it to setups the cloud environments. These all features of OpenStack make it our prime choice of cloud. OpenStack has six stable core services and that are named as compute, networking, storage, identity, and images are Nova, Neutron, Swift, Cinder, Keystone, and Glance. Keystone is an OpenStack service that provides API client authentication, service discovery, and distributed multi-tenant authorization by implementing OpenStack's Identity API[7]. The OpenStack Identity service (keystone) is flexible to numerous ways of authentication, including the most commonly used user name & password combination, LDAP. Access control through keystone makes it an ideal candidate for experimenting with additional customized security constraints. Keystone provides an authorization token to the user for the successful authentication, and it's then useful for all subsequent access requests to the services. With OpenStack, Role-Based Access Control (RBAC) is preferred by all services across the cloud to manage and control the resource against the request permission for resources. The access request is approved or permitted only if a requester has the specified role as per the RBAC mechanism adopted by the domain to carry out an asked operation on the resource. The security mechanism is responsible for validating legitimate users who can have access to each of the available resources and cannot be accessed by illegitimate users. In recent times there is a shift of paradigm towards the attribute-based access control (ABAC) due to its support for fine-grained policies and flexibility in provisions of access control compares to RBAC that has its own limitation. Our prime focus is on providing access control and authorization that is ABAC-based and compatible with cloud platforms like OpenStack for Smart grid.

II. Related WORK

Researchers have shown incredible interest and contributed a lot in present days towards providing various security solutions and specifically towards access controls mechanism of security provisions. The journey of access control started from traditional access control mechanisms like DAC, MAC, RBAC and

now made significant progress by proposing many promising approaches like ABAC, NGAC, etc. The journey is quite noteworthy and impressive that providing improved and enhanced access control mechanisms adopted by the real world. A very casual way to state the ABAC is the access control model that makes access decisions based on participating entities' attributes is commonly known as ABAC. Access control is growing from its conventional host-centric paradigms to the access decisions evaluated base on target resources, units that scattered over large networks like the internet. A threat to security, introduced due to ICT, can be defined as any circumstances in which any defence method that administers access to the computing system may be under risk of harm. Security in computing may have essential elements like authentication with identity establishment, resource base access control, data integrity, non-repudiation and denial of services attacks, etc.[8]. It is very significant to incorporate different security provisions for preventing computing devices, network infrastructures, and communication channels from cyber attacks by well-defined protocols and achieving a precise regulation of information security [9]. The extension from a single domain system to the many domain systems regarding the security system with the policy-based security framework for an electric grid with the independently specified authorization from PDP and implemented with consideration of privacy using digital credentials for improving trust and assigning a role for all domains[10]. An authentication and authorization protocol with the integration of anonymous certificates using public keys along with standard authentication using XACML servers is proposed in [11], where the proposed work ensured the total secrecy by securing any identity from the illegal use by providing anonymous identities. ABAC is a sound access control authorization tactic for preventing requests from executing a group of action based on the decisions taken through evaluating policy, rules, or associations against the attributes related to the subject, object, demanded action, and, in a few cases, environment circumstances [12]. To improve the reliability and efficiency and for also achieving the equivalent security level as in conventional SCADA architectures, the Integration of Virtual SCADA in the Cloud is proposed with cryptography to achieve effective incorporation of Smart Grid and cloud in[13]. Open research issues for cloud computing in Smart Grid are Security Framework for SG Applications, Increasing Robustness, Defining Communication Protocols and a Model for Network Utilization, Economic Data Centres and Large Scale Cloud Platforms, Timely Demand Response, and Efficient Streaming with Clouds [14]. We prefer to work in the domain of security framework for smart grid applications. Incorporating technology like cloud computing in the Smart Grid can help a lot for the continuously evolving architecture envisioned to improve the performance in terms of cost, computing power, and energy efficiency[15]. Study of different use cases proposing to look for a solution that is capable of providing soft collusion amongst the numerous organizations beyond their personal constraints of ICT, grid, and consumers[16]. The model for storing a massive amount of data with capabilities to do the processing using Hadoop can help to achieve reliability having an efficient processing power using parallel processing in Hadoop [17]. Smart grid and cloud computing both are distributed types of which makes it an easy victim of DDoS kind attack and to counter that it requires to be benefited from the inherent attributes of Cloud Computing so that computation load distributes across extensive provisions of computational resources to achieve the balance for a swift increase in computational requirements by utilizing the capabilities of cloud computing [18] and suggested to utilizing the capabilities of cloud computing to distribute this computational load across a huge pool of computational resources to balance for a swift increase in computational requirements. The Smart Grid Data Cloud [16] is appropriate for open energy business with an idea of data clearinghouse having large upright incorporated utilities and relations between transmission system operators with prototyping of a synchronized smart meter data management system. Various cloud technologies participate aggressively in the smart grid that has a significant role in recovering from the disaster attentiveness level, in developing toughness of energy systems against the disasters, in improving the standard of power system optimization, providing the correctness of power system simulation and cloud technologies can endorse the right to use of renewable, green and sustainable power to the Smart Grid (SG) [19]. The need for various stakeholders to share data and to assist with each other's is emergent day by day. This circumstance requires the description of approaches for simply defining and successfully enforcing the discriminatory sharing needs of data stored at different stakeholders, probably also crossing administrative boundaries and various domains irrespective of the enterprise[20]. The smart grid has smart devices that are going to perform an essential role in the realization of this concept. These smart devices are IoT based devices which need to communicate to the control systems. There are many tasks which demand IoT based solution and Air Quality monitoring is one of the such task. Air Quality monitoring is an IoT-based solution proposed in[21] that has the favoured approach of communication topology where the accuracy of the

design is measured against Quality of service while considering throughput and power consumption. An IoT-based intelligent solution utilizes interconnected smart devices to monitor various parameters, collecting data to communicate it to the control server through the MQTT broker, and making analysis of data to predicting air quality[22]. Any cloud-based Smart Grid model must envision for improving the performance parameter concerning the service request of Smart grid users and the response of the service provider. To study performance impact, different load balancing algorithms like round-robin, throttled, artificial bee colony (ABC), ant colony optimization (ACO), and particle swarm optimization are implements, and analysis is presented in[23]. The services of Cloud are opted to develop a user dashboard, which provides dynamic data regarding operating status, details of generation, and consumption of power, tariff schedule, which is referred by the name Utility-Consumer Interactive Information System[24]. Developing 100 Smart cities is a visionary mission that is in the development phase in India, demanding specialized involvement from various domains for agreement and improvement of standard processes and products that require sound access control to prevent shared data. Based on this, a solution based on the combination of Ethereal smart contracts, eIDAS-based attribute and identity management, and the distributed file system IPFS is discussed in [25]. Globally many standard defining institutions like IEC, IEEE, and NIST, etc. are actively involved in standardization actions of Smart Grids. An Indian organization like the Bureau of Indian Standards (BIS) is making progress in various standards considering various technologies. Therefore, BIS must specify and regulate the standards which give priority and importance to the security challenges in cyberspace also[26]. To satisfy the requirements like flexibility, privacy, and integrity, the extended version of the ABAC model that can incorporate the concept of privacy is proposed in [27]. The implementation of the policy specifically crafted using XACML by carefully identified entities for the same with the scope and purpose of the SealedGRID - Smart Grid project proposed in [28]. A cautiously evaluated and implemented plan of solutions for providing security against cyber threat will ultimately escort to a secure end-to-end structure which can be adopted by grid for its operation. But an effort to implement a full proof structure as the initial step of securing the Smart Grid against cyber attacks can delay the implementation of considerably needed security provisions. Security provisions are available in ready to use for today's need that suggests talking about the thought of constant enhancement, whereby the security stance of the Grid should be in a steady state of development[29].

III. Open Stack & ABAC authorization

A. Access control of OpenStack

The Identity service of OpenStack extends the idea of groups and roles. Users belong to a specific group, and domain-specific roles assigned to various groups as per the requirements. Any of the services from the available pool of services of OpenStack must indicate the roles of the user trying to access the service. The OpenStack's policy enforcer middleware constrained the request and referred the rules specified into policy and then the user's group or roles and their relations to evaluate the permission. The policies are associated with each service to constrain the access request by the security component for evaluating the applicable permission to the demanded resource. The policy enforcement middleware provides domain-specific access control to OpenStack resources with the provision of the roles only. Each OpenStack service defines the access policies for its resources in an associated policy file configured and managed by specializing security service referred to by the name Keystones. The policy file contains different policy rules specified in the policy.json having JSON format that keeps the policy, and the instance is shown in figure 2[30]. Figure 2 specifies the policies which are defined for a service to control access for an operation like create, update, and delete resources to only legitimate users having the role of cloud_admin, which has been specified by a combination of role = admin and domain_id = admin_domain_id. The same way the operations like getting and list resources are made accessible to users having the role of cloud_admin or admin.

```
{
    "admin_required": "role:admin",
    "cloud_admin": "rule:admin_required and domain_id:admin_domain_id",
    "service_role": "role:service",
    "service_or_admin": "rule:admin_required or rule:service_role",
    "owner" : "user_id:%(user_id)s or user_id:%(target.token.user_id)s",
    "admin_or_owner": "(rule:admin_required and domain_id:%(target.token.user.domain.id)s) or rule:owner",
    "admin_or_cloud_admin": "rule:admin_required or rule:cloud_admin",
    "admin_and_matching_domain_id": "rule:admin_required and domain_id:%(domain_id)s",
    "service_admin_or_owner": "rule:service_or_admin or rule:owner",

    "default": "rule:admin_required",

    "identity:get_service": "rule:admin_or_cloud_admin",
    "identity:list_services": "rule:admin_or_cloud_admin",
    "identity:create_service": "rule:cloud_admin",
    "identity:update_service": "rule:cloud_admin",
    "identity:delete_service": "rule:cloud_admin",

    "identity:get_endpoint": "rule:admin_or_cloud_admin",
    "identity:list_endpoints": "rule:admin_or_cloud_admin",
    "identity:create_endpoint": "rule:cloud_admin",
    "identity:update_endpoint": "rule:cloud_admin",
    "identity:delete_endpoint": "rule:cloud_admin",
}
```


Figure 2: Policy File instance

By default, at present, the policy engine specifies the terminal rules of type Role-based, Field-based, or Generic rules. For example, at a particular instance, the Generic rule is successful if the project identifier for the resource is equal to the project identifier of the user who is submitting the request (tenant_id:% (tenant_id) s). An OpenStack has a setup of predefined scripts or functions, which are by default following role-based access control policies throughout all the services provided by the cloud platform of OpenStack. The policy engine of specific service triggers the policies for every time the request's operation, specific attributes being used matches an API request. RBAC uses Roles for assigning the kind of permission users can have by owning the specific role. Permission or access rights for a particular concept of the tenant/domain provided via Roles. User or a group can endure a role as per the specification of requirements of a particular participating domain, following the constraint of a different name for each role adopted. There are pre-defined bonding between a Role, a Resource, and an Identity, and also the assignment relation between these three participating entities or tuples. Keystone is the identity service used by OpenStack for authentication (authN) and high-level authorization (authZ) with the adoptions of token-based solutions. The mechanism or the solution adopted in the form of the token can confirm and supervise the tokens of access request handed over with the desire of validating the authentication of a user whose user id and passwords have previously been confirmed.

In OpenStack, the user means a sole user who consumes any of the available services through provided APIs, and for that, he must be a part of a definite domain through pre-registered for that domain. The identifiers of such users have not been constrained throughout all the domains but are bound to be exclusive to their own domain only. A bunch of users is collected under one shelf and referred to as groups. Comparable to the individual user, a group also associates with some exclusive domain holding the property of ownership of that domain. The groups' identifiers are not restricted to intra domains of the whole organization but require to be exclusive for its own domain. OpenStack provides services that are responsible and authorized to a particular domain or tenants to access the available resources and data. The Projects or Tenants are the foundational elements of ownership to associate the ownership of any resource with the specified project. Domains specify a compositor or box which holds tenants with its users and groups while everyone is registered to very precisely to a single domain only. The assignment service provides data about roles and role assignments. A powerful concept of the namespace is incorporated to make an API searchable by its names or identity throughout.

B. Attributes ammendement

OpenStack is flexible enough to be deployed by different clients with different requirements though a generic yet flexible approach is needed. The solution with which the clients may define, apply and manage their own customized authorization policy and with this objective, we reconfigured the PEP (Policy Enforcement Point) to disable the default way of authorization and delegates' authorization to an external authorization policy engine. The Identity service can directly provide end-user authentication, which should be modified to use external authorization methods to adhere to a particular organization's security policies and requirements. For external policy checks, we have to configure and modify the OpenStack default and establish the communication to the external policy engine, and for that modification we made is highlighted in figure 3. We added attributes with the default token mechanism that is assigned and verified in decision making by the policy engine, which refers to the policies specified with the crafted rules adopting the requirements of the considered attributes. After making the recommended changes, we have to intercept the POST request (Figure 4) for the sake of authorization by the policy engine residing at the given URL, and the policy engine returns the response based on the rules specified through policies over there. By default, POST request within the OpenStack framework comes with payload data that requires converting into the appropriate formats so that it becomes capable of holding hold attribute values for the sake of our purpose.

```
"os_compute_api":os-keypars:index":"http://127.0.0.1:777",

"os_compute_api":os-keypars:create":"http://127.0.0.1:777",

"os_compute_api":os-keypars:delete":"http://127.0.0.1:777",

"os_compute_api":os-keypars:show":"rule:admin_api or user_id:%(user_id)s",
```

Figure 3: Redirection to Policy Engine

```
from http.server import BaseHTTPRequestHandler, HTTPServer
import payload
class MyHandler(BaseHTTPRequestHandler):

        def do_POST(self):
                content_length = int(self.headers['Content-Length'])
                body = self.rfile.read(content_length)
                body=str(body,'utf-8')
                self.send_response(200)
                self.end_headers()
                if payload._check_(body,"dept")=='computer':
                        self.wfile.write(b'True')

def main():
        print('starting server on port 7777...')
        server_address = ('127.0.0.1', 7777)
        httpd = HTTPServer(server_address, MyHandler)
        httpd.serve_forever()

main()
```

Figure: 4 Instance of Server Script of policy engine

We must make changes in default coding scripts so that the composition of requiring domain-specific attributes becomes possible for the asked ABAC approach. Scripts shown through figure 5 (Views.py, Tables.py, Forms.py, Test.py) are some of the python scripts that need to be modified within existing OpenStack to support the ABAC approach. The required changes must also be incorporated in relevant HTML/JAVA scripts to reflect the changes of python scripts through User Interface.

Figure 5: Scripts to accommodate attributes

We have modified the existing OpenStack framework to accommodate the required changes for proof of concept implementation for demonstrating the proposed work. We have done experiments with services of OpenStack that have command compute_api for various operations like create, delete, index, and show on keypairs of each tenant that are permissible to a user base on his valid authentication. The newly designed component of authorization collects the additional details of attribute data from the MySQL database that are needed to merge with the authentication information in the payload that traverses through the requested operation referred to as token to verify the appropriateness of changes. We have created two users with the role of admin and having the value of the name attribute as Yagnik and Divyesh. In addition to the role, we have introduced the attribute department attached to users having values like a computer or cyber for the department for the desired shifting towards ABAC from RBAC. Figure 6 (a) demonstrates the default OpenStack users with the role. The modified and updated OpenStack having users with the role and attributes both is demonstrated by figure 6 (b).

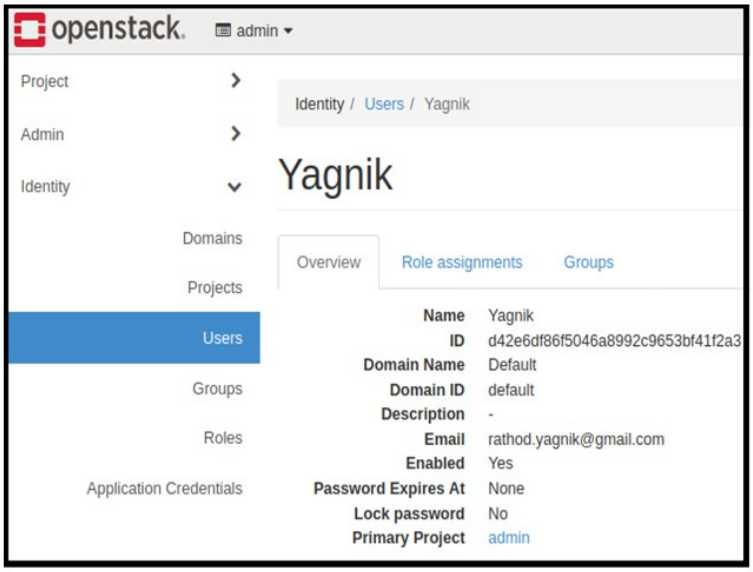

Figure 6 a: Users with RBAC

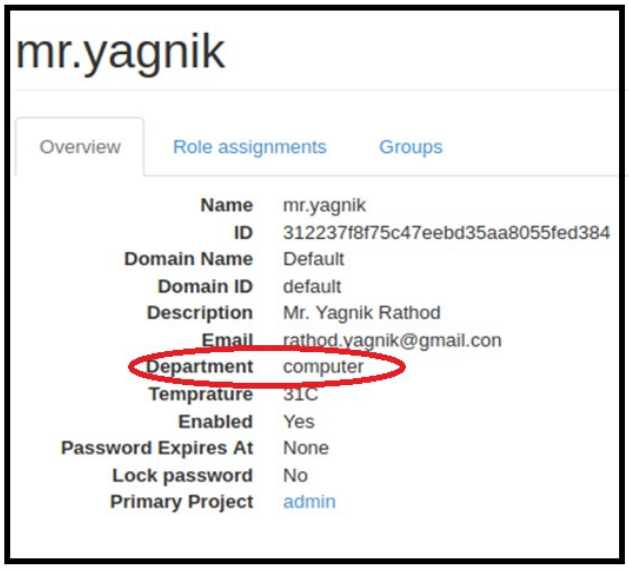

Figure 6 b: Users with ABAC

We have modified the OpenStack's default rule to constraint access in such a way that the users having the role of admin must be assigned to the computer department only to become authorized to create and delete operations. One instance of rules is specified below for better understanding.

- Rule 1: User with the role of an admin belongs to the computer department is authorized to use all commands of any services.
  - Figure 7 shows the execution of requests and responses for the above rule 1.
- Rule 2: User with role admin who belongs to any other department is authorized to perform only index and show operation of the services.
  - Figure 8 shows the execution of requests and responses for the above rule 2.

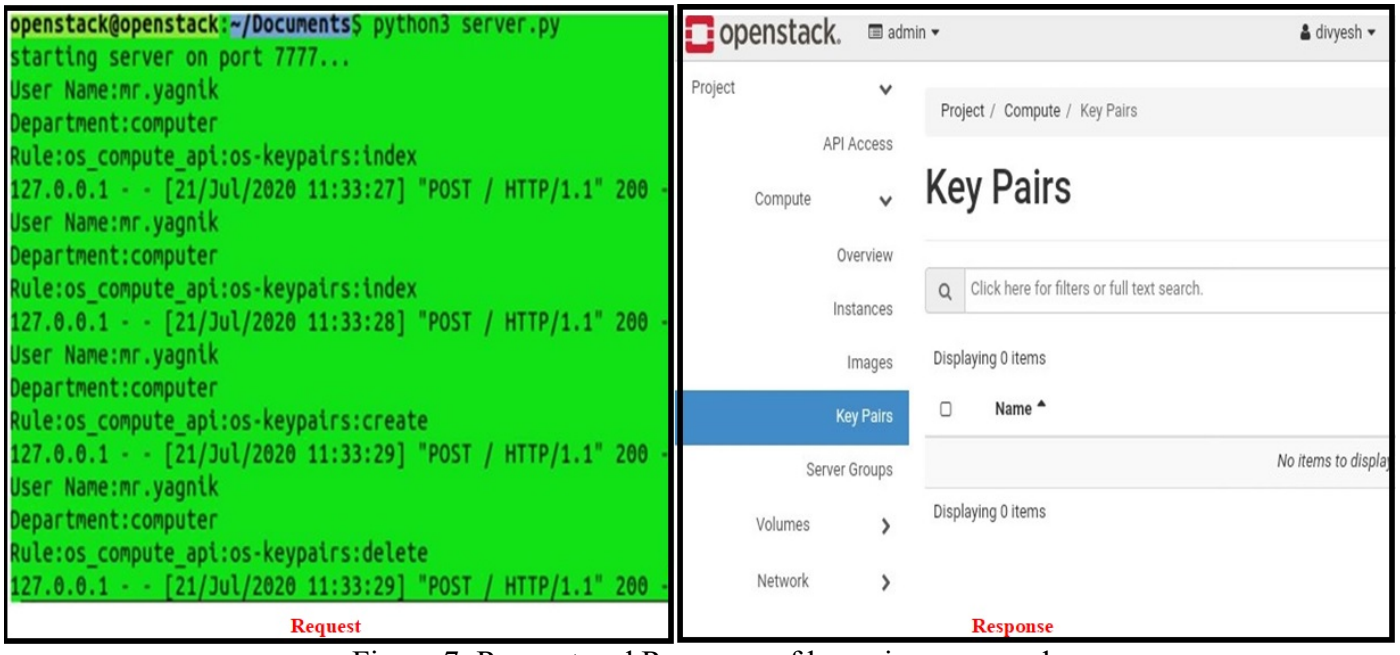

Figure 7: Request and Response of keypairs command

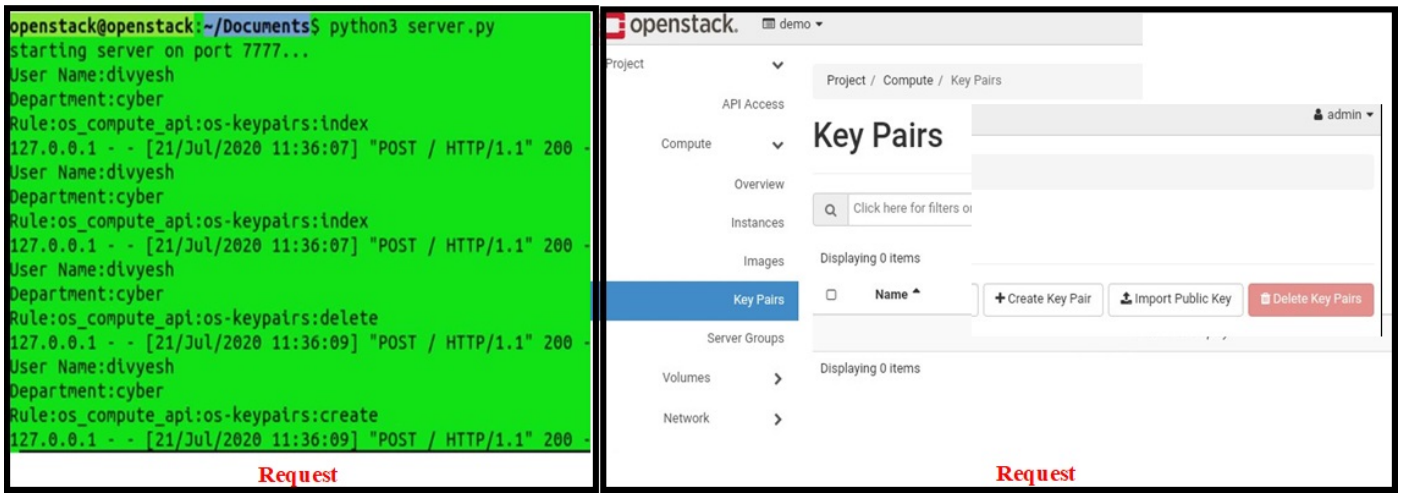

Figure 8: Request and Response of keypairs command

We have also integrated Smart devices with OpenStack as it is quite essential for the actual realization of the Smart grid. This requirement addresses by incorporating data of remotely placed temperature sensor that communicates data through a communication line for displaying it on a legitimate authorized user's OpenStack dashboard is shown in figure 9.

Figure 9: Dashboard of openstack user Yagnik with Smart device-IOT Data

C. Policy crafting for Smart Grid

The authorization decisions reflect the rules defined in the underneath enterprise, and for providing accurate authorization appropriateness of the design for the policies is extremely crucial. The Composition of policies is required for many policies though they are following different concepts of access control, especially for the Smart Grid. Here, we have prepared policies and rules for the energy domain that is base on the Smart Grid with the consultation from the engineers of the generation and distribution unit of the government of Gujarat. Though there are many stakeholders for any state's department of energy, we are only involved in the crucial two of that referred to as Generation and Distribution. The hierarchical structure of the roles is identified for the associate organizations of the stakeholders to define the appropriate access rights by carrying the much-needed role as presented in figure 10.

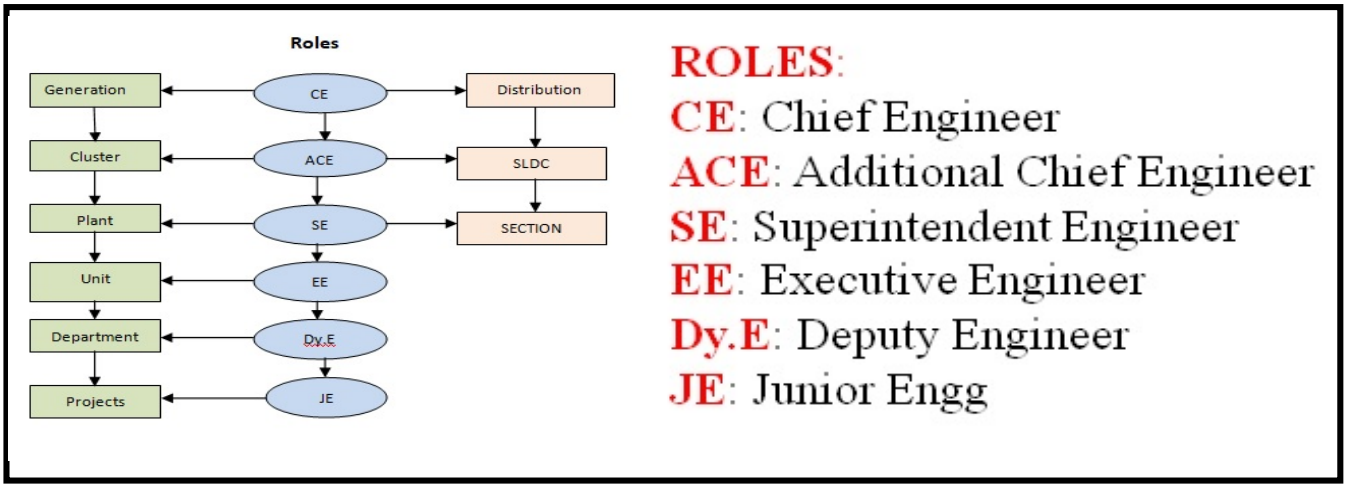

Figure 10: Role Hierarchy

The specifications of the various components mentioned in the Role Hierarchy are defined below.

Distribution: The distribution center has many SLDCs, where SLDC is a State load Distribution Center and responsible for distributing electricity for a particular geographical area. Each SLDC has several SECTIONS which are responsible for pre-assigned duties. Each SLDC and SECTION has its own Data Objects and Users with specified ROLE.

Generation: Generation center has several CLUSTERS under it. Each cluster is responsible for managing Electricity generation as per the demand of one of the geographical areas. Each of these clusters has several PLANTS that are responsible for generating electricity as per its own declared capacity (for example, 800 Megawatt). Each PLANT has several UNITS that are actually generating electricity as per its dynamic situation (for example, a UNIT announce the declare capacity as 300 MW for the next 24 hrs compared to the actual capacity of 500 MW). Each UNIT has several Departments that have different Projects running under it.

The organization structure of the State's DOE is shown in figure 11 as per the specification of generation and distribution defined.

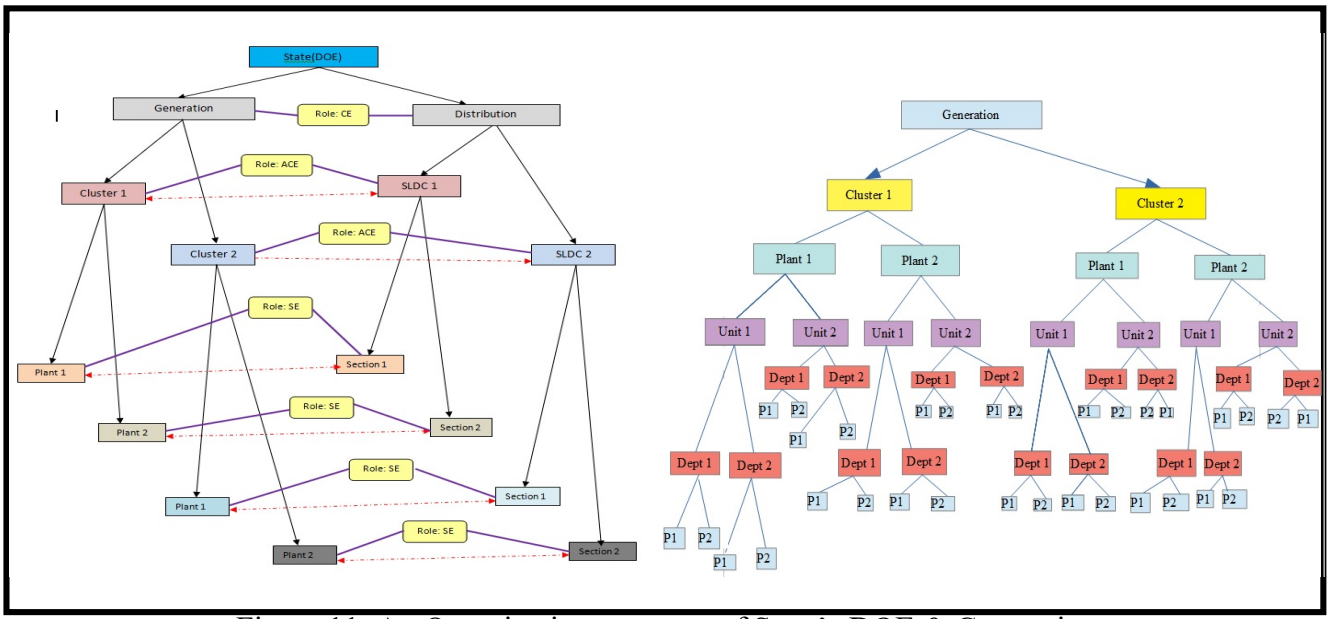

Figure 11: An Organization structure of State's DOE & Generation

The policy crafted incorporating defined rules for this organizational structure (inter-division) is shown in figure 12 (a). An instance of that rule incorporating constraints of association within the policy schema is shown in figure 12 (b).

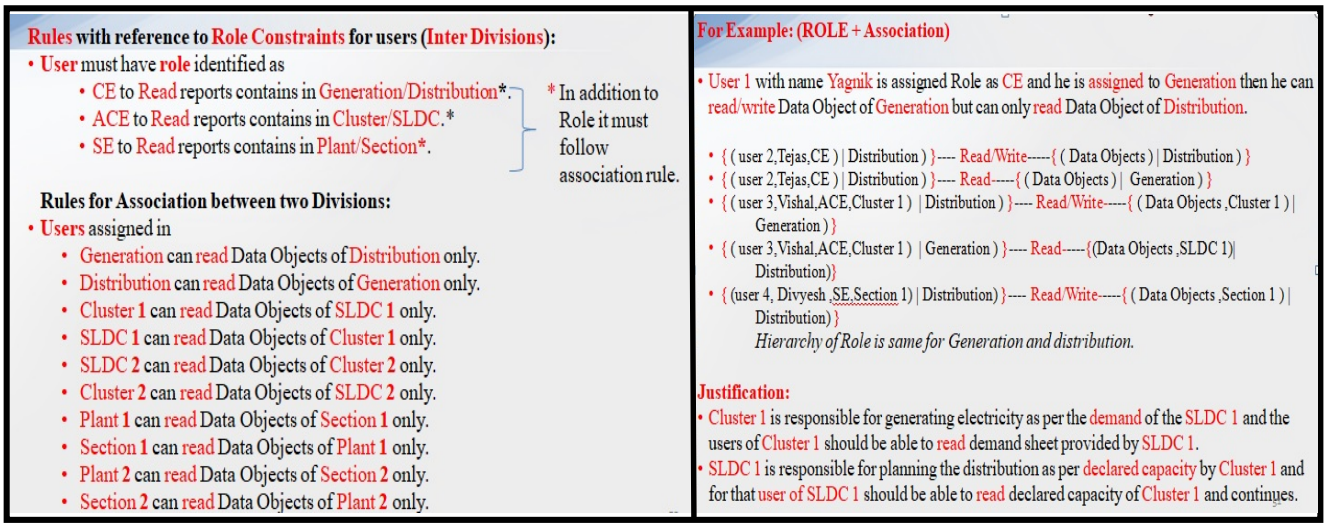

Figure 12 (a): Policy Rules with Role for users (Inter Divisions) Figure 12 (b): instance of Policy Rules

Similarly, figure 13 shows sample rules for intra division, distribution, and inheritance.

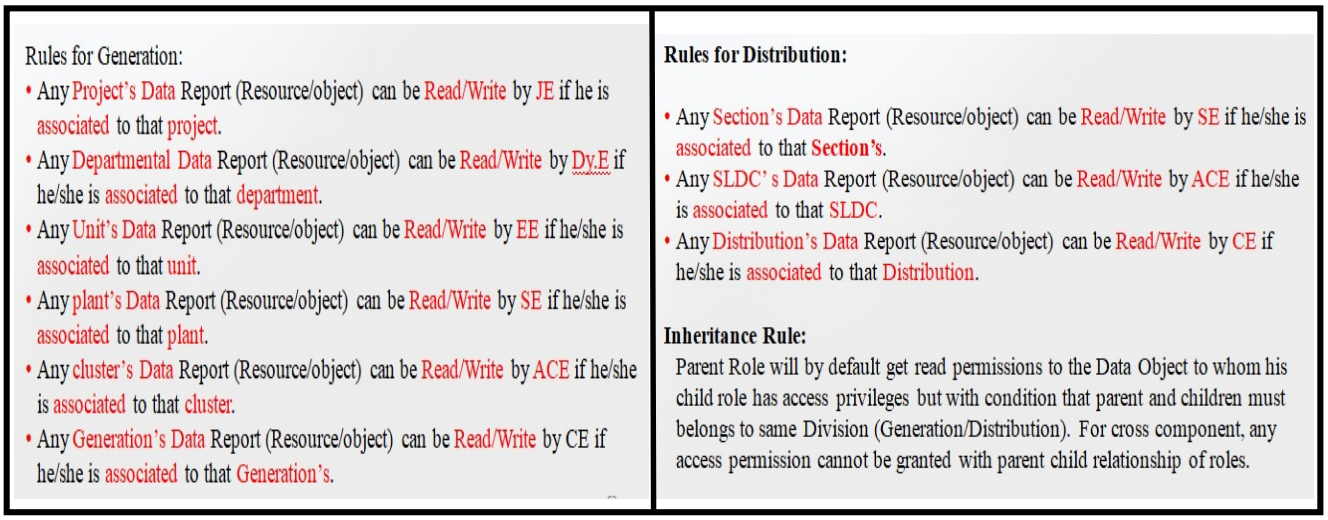

Figure 13: Policy Rules intra division, distribution and inheritance

The users holding the role of PRO (Public Relations Officer) can read/write consumer reports following the constraint that both of them are associated with the same section. Users with the role of the consumer can read Data Objects that are assigned to him only. (For example, the Role of PRO, consumer, and sensor designated in Section 1 have no access to Data objects of Section 2). The graphical representation specifies the permitted operation of customized rules applicable for Sections under Distribution is as shown in figure 16.

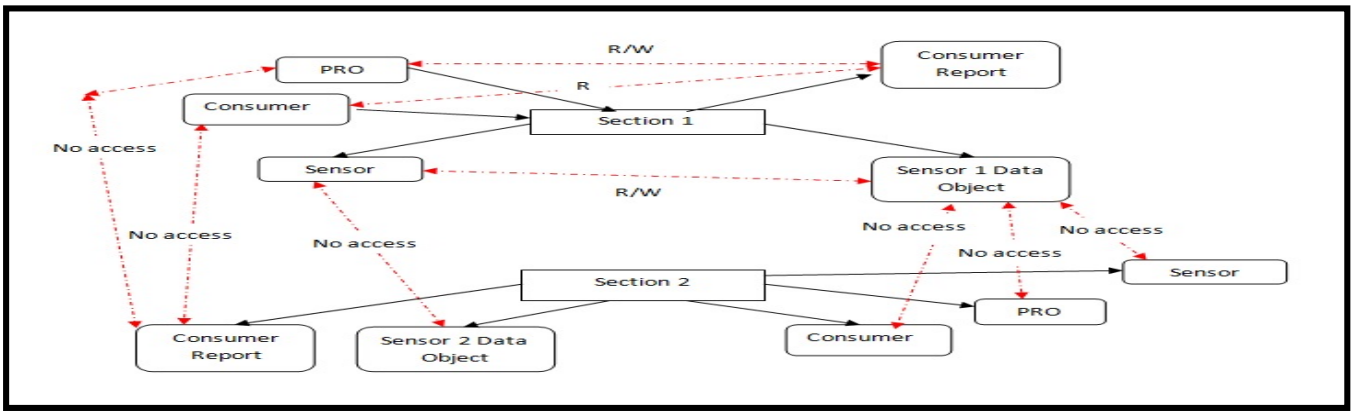

Figure 14: Customized rules for integrating Smart devices

Rules defined for inter and intra divisions as specified above are difficult to achieve with a simple rule-based approach, which demands additional support to role-based policies to satisfy the rule effectively. The conventional security approaches holding the current policy classes based on RBAC needs the improvement in existing models to achieve the desired objective by designing supplementary policy classes that operate in synchronization with the default security models. In the RBAC based policy class, the role of a user authorizes access to the resource using the role, role-assignments, and that kind of policy class does not satisfy the requirements identified for our Smart Grid case study single-handedly. We can achieve a greater level of flexibility by utilizing the various approaches of access-control to draft the different policy classes as per the need of stakeholders. Therefore, We have designed three policy classes referred to by RBAC, Generation (ABAC-based), and Distribution (ABAC-based) as specified in figure 16 to meet our desired objective of association and inter/intra divisional access. The permissions for any access request concludes by evaluating the combination of the rules defined in multiple policies specified in different policy classes during policy designing with the proposed approach. We have examined the rules drafted by us for validation purposes with the Access Control Policy Tool (ACPT) provided by the Computer Security Division of NIST because the correctness of work strongly depends on the correctly defined policies. The ACPT tool can help to identify any conflicts of the rules and give us the chance to rectify it.

For the implementation of these policies classes, while taking benefit of the power of the OpenStack cloud, we need the policy engine that will decide authorization instead of OpenStack's default RBAC policy engine. The policy engine used in the proposed framework follows an attribute-based access control mechanism by default that has core features like the ability to configure, enforce several access-control policies, and the ability to protect resources under multiple instances of different policy classes. It is also suitable for applications where information is stored locally or in a grid or cloud that asserted different policies in each context. The implementation of the authorization engine follows a three-tier architecture that has a presentation layer, a business layer, and a data layer. The users can access the resources based on his is evaluated permission through the presentation layer, a business layer that is a core policy engine server containing the PDP, and a data layer contains all the relevant information in the database. Policy administrators can configure all policies, data objects, and users through the policy files or can use the UI tool for the same. Users can have access to the resources that are accessible via the user-specific customized dashboard based on their authorization that displays the resources for which he is entitled to operations like read, write, execute, etc. The policy engine determines authorized users, subjects, operations, and objects. An authorized user obtains access to the policy engine system by presenting his/her credentials to establishing a session. System entities that must be protected are referred to here as objects having global meaning shared under one or more policies.

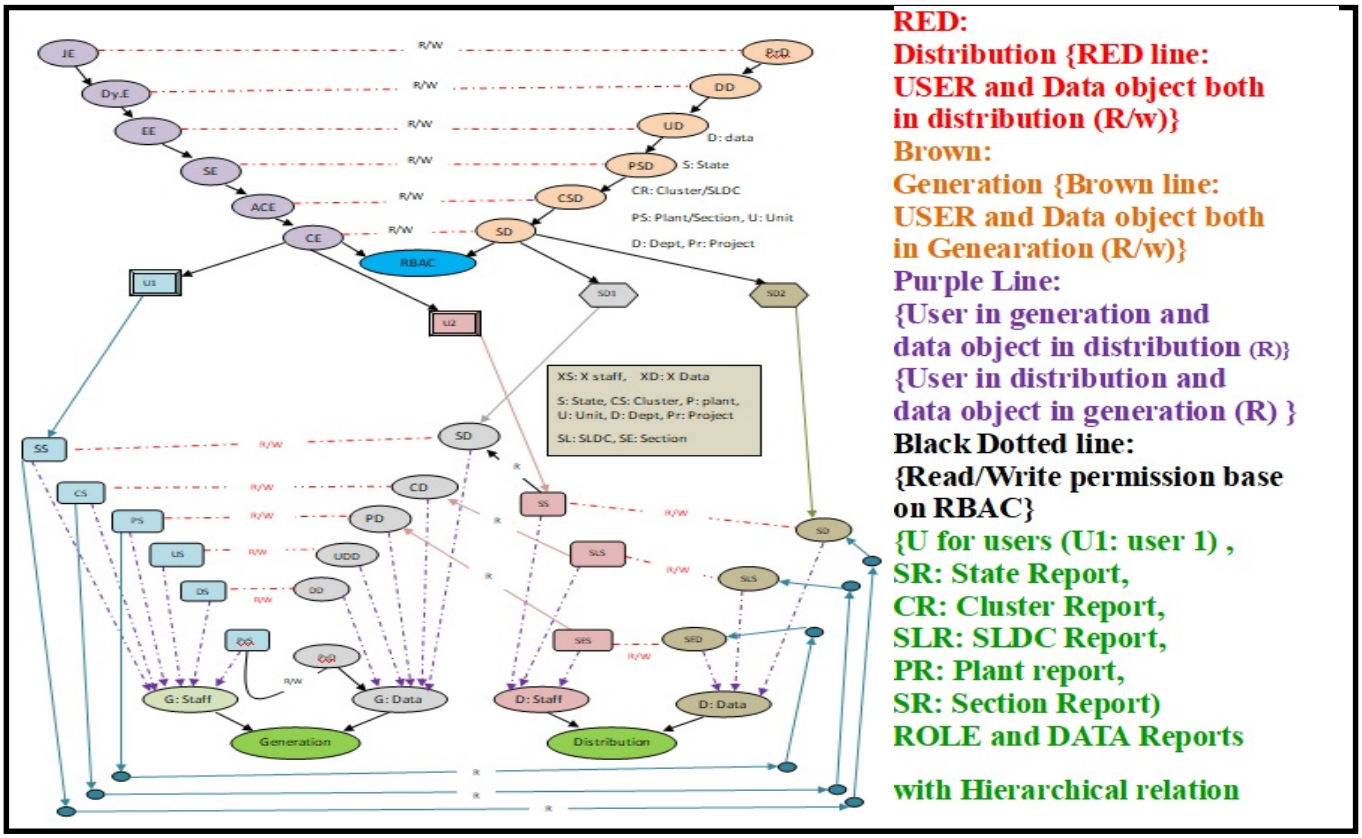

Figure 15: Policy Diagram for smart grid case study

The authorization engine considers the permission as triplets of the user, operation, and object. The request made by the subject (operation, object) is granted by the reference mediation function if and only if permission triplets exist. The permissions are not individually managed but instead derive from a set of policies specific to user and object attributes. The authorization engine tool specifies the association between the user, operation, and resource that means it actually assigns the permissions through triplets. The concept of graph theory is applied to specify and store the association triplets between the requester and the resources. Graph searching methods like DFS, BFS is used to determine the permission at the time to user's access request base on the active session and active attribute of users only. The design and development of Policy classes, users, and data objects are as per figure 15, which is then implemented and configured using policy tools as shown in figure 16.

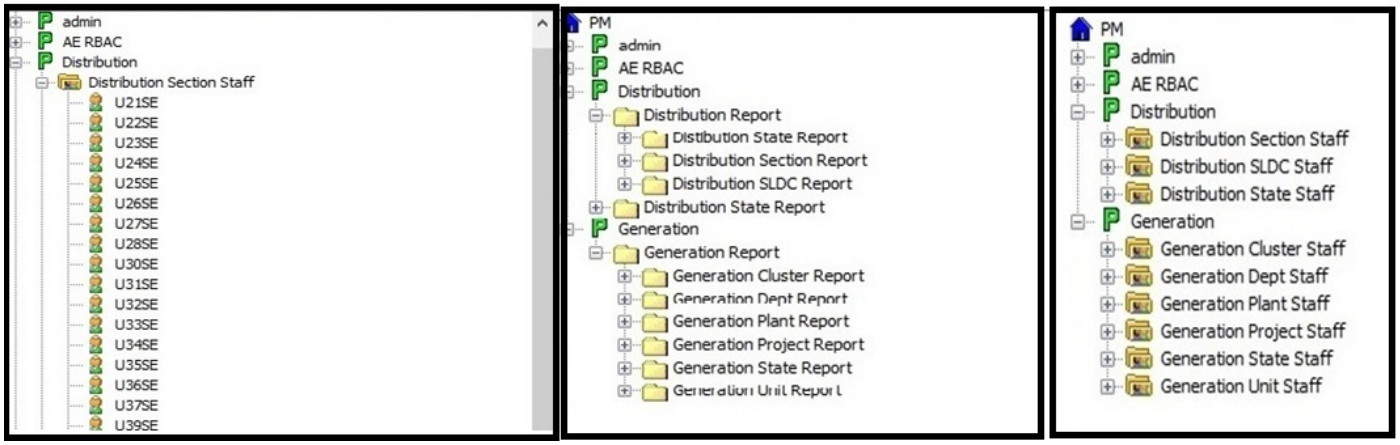

Figure 16: Policy Tool

D. Results & Perfomance analysis

We have created codes to evaluate and compute the time consumed for a group having several access requests. Here, our prime purpose is to assess the time consumed in the access control framework of OpenStack by the default policy engine, also for the proposed approach that integrates the selected attributes into the access request redirected to the external policy engine. Figure 18 (a) shows the analysis, which informs about the total time consumed in fulfilling the access requests of an authenticated user, whereas the interpretation of Figure 18 (b) notifies the time taken during the checking of policies only in OpenStack. Three different variants symbolizing distinct methods for access requests that users can make if he has valid credentials with a predefined role are compared in the graph. The default role base policy in OpenStack highlights by the blue line. The default role base policy of OpenStack with redirection to the external engine configured with RBAC, highlights by the red line. The approach with the integration of attributes to extend the role-based decision making policy with redirection to the external authorization engine bypassing the default engine of OpenStack, highlights by the green line. Figure 17 (a) shows the total time consumed in making a request and getting the response for these three variants is nearly similar. Figure 17 (b) shows the time taken in evaluating policy for each of these variants varies compare to its counterpart. A limitation of the executive power of the external policy engine and network base communication methods becomes a bottleneck and increases the time taken in the evaluation of policies.

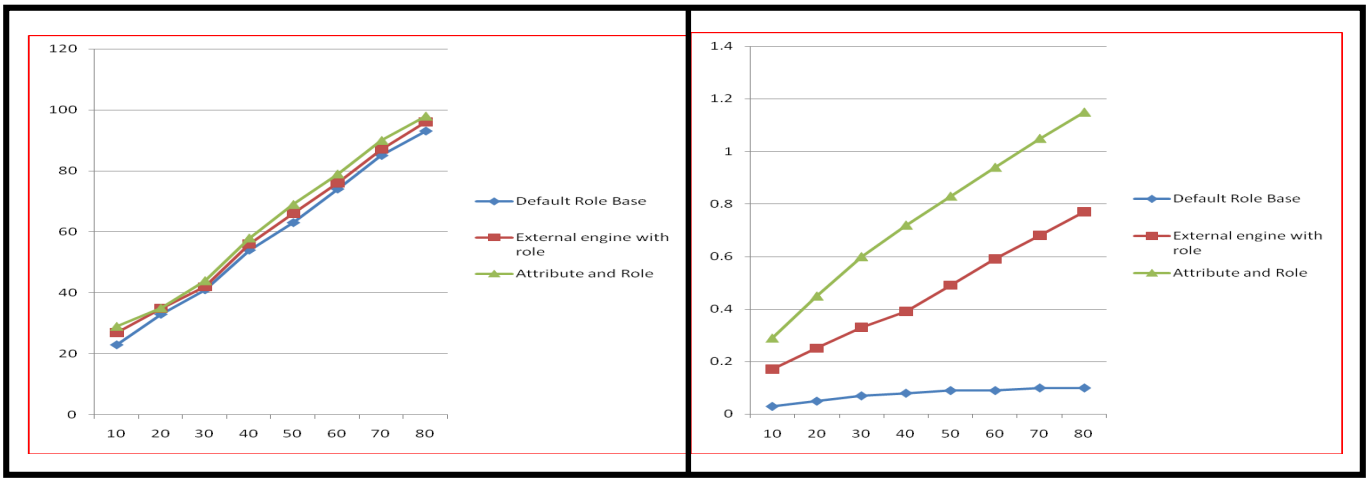

Figure 17 a: Overall Time Taken in Requests-Response

Figure 17 b: Policy Check Time for Requests by User

Here our primary objective is to insist on our anticipated ABAC approach and its integration with the OpenStack cloud avoiding its default. When we compare the performance based on the evaluations of our access control and authorization approach against the default one, then only we can understand the cost of applying our idea to real systems. It becomes apparent that there is always a challenge of balancing execution speed with added constraints of security added due to the functionality for user benefits. There are different findings associated with the performance of the total task of accomplishment. Stared with the first thing, as implementation was not targeting the constraints of the performance, that's why optimization is not done in an appropriate way to improve the performance. Our work was more focused on the impact and appropriateness of the provided solution in a real-world scenario with OpenStack. We understand the limitations and accept that the enforcement architecture of this model needs extended improvement customized for better performance that can only make him widely acceptable in the real world infrastructure domains. Also, there is a role to play by the network that is also very important in considering the time consumed for each access request from OpenStack to an external server, which is currently residing on some remote node.

IV. Conclusion

We proposed an access control and authorization model that is integrated with the cloud and applicable for the Smart Grid domain. We also integrated the IoT devices that can be part of the smart grid component in the cloud and non-cloud environment and analyze the feasibility with the OpenStack cloud. Most importantly, the design and development of policy rules and frameworks prove applicable for our case study for the Smart Grid domain that is incorporable with the cloud platform of OpenStack. These multiple policies designed for the real world enterprises can permit minimum access rights to a user and make it possible to specify advanced fine-grained authorization policies for avoiding existing problems. We believe this work will facilitate the transition towards ABAC based access control and authorization models and will open prospective avenues to apply ABAC in real-world applications using the open-source cloud platform. To attain the performance enhancement, We believe that solutions like to configure the customize superior server to host the policy engine with a vision towards the improvement in policy assessment time by caching results locally that obtains through the assessment of policies. We keep this work as open for future work.

V. References

[1] U.S. National Institute of Standards and Technology, "Guidelines for Smart Grid Cybersecurity NISTIR 7628 Revision 1," *U.S. Dep. Commer. NISTIR*, vol. 1, no. September, p. 668, 2014, doi: 10.6028/NIST.IR.7628r1.

[2] C. Wei, "A conceptual framework for smart grid," *Asia-Pacific Power Energy Eng. Conf. APPEEC*, 2010, doi: 10.1109/APPEEC.2010.5448786.

[3] H. Melvin, "The role of ICT in evolving SmartGrids," in *The 10th International Conference on Digital Technologies 2014*, Jul. 2014, pp. 235–237, doi: 10.1109/DT.2014.6868720.

[4] A. R. Metke and R. L. Ekl, "Smart grid security technology," *Innov. Smart Grid Technol. Conf. ISGT 2010*, pp. 1–7, 2010, doi: 10.1109/isgt.2010.5434760.

[5] M. B. Line, I. A. Tøndel, and M. G. Jaatun, "Cyber security challenges in Smart Grids," *IEEE PES Innov. Smart Grid Technol. Conf. Eur.*, pp. 1–8, 2011, doi: 10.1109/ISGTEurope.2011.6162695.

[6] A. Bereş, B. Genge, and I. Kiss, "A Brief Survey on Smart Grid Data Analysis in the Cloud," *Procedia Technol.*, vol. 19, pp. 858–865, 2015, doi: 10.1016/j.protcy.2015.02.123.

[7] "OpenStack Docs: Keystone, the OpenStack Identity Service." .

[8] A. R. Anggraini and J. Oliver, *Access control systems*, vol. 53, no. 9. 2019.

[9] R. S. Sandhu and P. Samarati, "1994 Access Control 1.pdf," *IEEE Communications Magazine*, pp. 40–48, Sep. 1994.

[10] H. Cheung, C. Yang, and H. Cheung, "New Smart-Grid Operation-Based Network Access Control," in *2015 IEEE Energy Conversion Congress and Exposition (ECCE)*, 2015, pp. 1203–1207.

[11] U. Khalid, A. Ghafoor, M. Irum, and M. A. Shibli, "Cloud based secure and privacy enhanced authentication & authorization protocol," *Procedia Comput. Sci.*, vol. 22, pp. 680–688, 2013, doi: 10.1016/j.procs.2013.09.149.

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

[13] C. Alcaraz, I. Agudo, D. Nu, and J. Lopez, "Managing Incidents in Smart Grids a` la Cloud," 2011, doi:

10.1109/CloudCom.2011.79.
[14] M. Yigit, V. C. Gungor, and S. Baktir, “Cloud Computing for Smart Grid applications,” *Comput. Networks*, vol. 70, pp. 312–329, 2014, doi: 10.1016/j.comnet.2014.06.007.
[15] P. Naveen, W. K. Ing, M. K. Danquah, A. S. Sidhu, and A. Abu-Siada, “Cloud computing for energy management in smart grid - An application survey,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 121, no. 1, 2016, doi: 10.1088/1757-899X/121/1/012010.
[16] S. Rusitschka, K. Eger, and C. Gerdes, “Smart Grid Data Cloud: A Model for Utilizing Cloud Computing in the Smart Grid Domain,” pp. 483–488, 2010, doi: 10.1109/smartgrid.2010.5622089.
[17] H. Bai, Z. Ma, and Y. Zhu, “The application of cloud computing in smart grid status monitoring,” *Commun. Comput. Inf. Sci.*, vol. 312 CCIS, pp. 460–465, 2012, doi: 10.1007/978-3-642-32427-7_64.
[18] A. Califano, E. Dincelli, and S. Goel, “Using Features of Cloud Computing to Defend Smart Grid against DDoS Attacks,” *10th Annu. Symp. Inf. Assur.*, no. June, p. 44, 2015.
[19] B. Fang *et al.*, “The contributions of cloud technologies to smart grid,” *Renew. Sustain. Energy Rev.*, vol. 59, no. June, pp. 1326–1331, 2016, doi: 10.1016/j.rser.2016.01.032.
[20] John vacca, *cloud computing security foundation and challenges*, vol. 53, no. 9. CRC press, 2013.
[21] V. Barot, V. Kapadia, and S. Pandya, “QoS Enabled IoT Based Low Cost Air Quality Monitoring System with Power Consumption Optimization,” vol. 20, no. 2, pp. 122–140, 2020, doi: 10.2478/cait-2020-0021.
[22] V. Barot and V. Kapadia, “Towards building a scalable IoT based system for carbon monoxide monitoring and forecasting,” *Int. J. Adv. Sci. Technol.*, vol. 29, no. 3, pp. 5583–5590, 2020.
[23] S. Zahoor, S. Javaid, N. Javaid, M. Ashraf, F. Ishmanov, and M. K. Afzal, “Cloud-fog-based smart grid model for efficient resource management,” *Sustain.*, vol. 10, no. 6, pp. 1–21, 2018, doi: 10.3390/su10062079.
[24] P. Naveen, W. Kiing, I. Michael, K. Danquah, A. S. Sidhu, and A. Abu-Siada, “A Cloud Associated Smart Grid Admin Dashboard,” *Technol. Appl. Sci. Res.*, vol. 8, no. 1, pp. 2499–2507, 2018.
[25] F. Buccafurri, C. Labrini, and L. Musarella, “Smart-contract based access control on distributed information in a smart-city scenario,” *CEUR Workshop Proc.*, vol. 2580, 2020.
[26] V. R. Elonnai Hickok, “Cyber Security of Smart Grids in India,” *Centre for Internet and Society (CIS)*, 2016.
[27] M. Ed-Daibouni, A. Lebbat, S. Tallal, and H. Medromi, “Toward a New Extension of the Access Control Model ABAC for Cloud Computing,” in *Advances in Ubiquitous Networking*, 2016, pp. 79–89.
[28] G. Suciu, C. Istrate, A. Vulpe, M.-A. Sachian, and M. Vochin, “Attribute-based Access Control for Secure and Resilient Smart Grids,” 2019, doi: 10.14236/ewic/icscsr19.9.
[29] E. D. Knapp and R. Samani, *Applied Cyber Security and the Smart Grid: Implementing Security Controls into the Modern Power Infrastructure*. 2013.
[30] “Openstack documnetation Policies,” 2020. https://docs.openstack.org/security-guide/identity/policies.html.