iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** II **Month of publication:** February 2021

DOI: https://doi.org/10.22214/ijraset.2021.32950

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue II Feb 2021- Available at www.ijraset.com

Secure Attribute-based User Access Control over AWS Cloud

Sucharita Khuntia[1], Dipti Krishna[2], Shirish Sahay[3]
[1, 2, 3]*Computer Science and Engineering, CV Raman Global University*

Abstract: ***Every day billions and trillions of data are generated from different sources. To store such a huge amount of data need large storage, for that cloud becomes the most effective solution due to the properties like flexibility, cost-effectiveness, and scalability. It has become a suitable and efficient option for the storage of data. But with the storage of data on a cloud platform, it is very essential to avail all possible securities to the data stored in the cloud. That aim can be achieved by providing access control to the data and storage. Storage can be made secure by implementing user-level access methods. User-level Access Control prohibits the storage at the user level itself. All the users do not have the same permissions in storage access. Some users are allowed to only read as well as write data/files in the storage, some are allowed to only read data and write data in the storage. All this access control is achieved by writing certain rules concerning the storage of the cloud platform. For more efficiency and security purposes along with access control on storage, access control can also be implemented over the stored data through Attribute Based Encryption. So the data stored in the cloud can be accessed by permitted users. The storage requires an application to address the reading, writing of data and imposing access control over the data stored in it. In this paper, we propose an Attribute-based User Access Control method. We create an application over Amazon Web Services (AWS) that facilitate uploading data on the cloud platform. We keep the cloud storage of the application security and access to the stored data is controlled by implementing our idea. The security of the cloud storage and data stored is controlled at the user level. This application also prevents spamming from unauthorized users to keep data more secure. Thus, our proposed Attribute-based User Access Control method proves a suitable and effective solution for providing security to the AWS cloud as well as data stored over its platform.***

Keywords: Cloud Computing, Amazon Web Services, Attribute-based User Access Control, User-level Access Control, Cipher-text Policy Attribute Based Encryption

I. INTRODUCTION

The typical business application architecture has evolved manifold ranging from a desktop-centric installation to desktop free web services and service-oriented architectures (SOA) over the past few years. Nowadays an outsized amount of knowledge is continuously generated from different sources like social networking sites, android smartphones, and sensors, etc. Our traditional computer systems are not sufficient enough to store and process this huge amount of data. Cloud proves to be an appropriate option for this problem due to its elasticity and scalable nature. Users can store their data in the cloud and can share their data with different other users too. Cloud Computing is omnipresent and the speed of the transformation of business models in all the industries because of the cloud is more and more accelerated. Cloud Computing is often used as an umbrella term under which several technologies, components, approaches, and various sorts of applications come. To compete and establish its position during this changing digitalized market, a company must evaluate its current business standard and leverage technologies such as the Internet of things (IoT), big data, Augmented Reality, Machine Learning, etc. These emerging technologies need heavy computing power, large storage capacity, and huge IT infrastructure that can be provided by Cloud computing services according to companies' requirements. There are many benefits of using the cloud such as flexibility, scalability, elasticity, reliability, etc., through which we will leverage the expertise of others in a cost-efficient way. This technology mainly uses remote servers over a network. Due to all these additional benefits, enterprise organizations see an enormous scope in cloud computing for flourishing their IT infrastructure. But the major problem with cloud storage is once the data is outsourced to the cloud, the user loses control over the data. It is mandatory to style an efficient access system to share users' stored data only to authorized users as per their requirements. Unlike traditional access control, the info owner wants to reserve rights to define the access control policy for the data manually before its release on the cloud. Moreover, the data owner wants to encrypt the data consistent with the access control policy before putting the info on the cloud because the safety on the cloud could be compromised. Thus, the access control mechanism becomes a strenuous issue.

CP-ABE (Cipher-text Policy Attribute Based Encryption) with a hidden access control policy is an efficient solution. In CP-ABE, the access to the data is controlled by the user. It allows users to share their data stored in the cloud with other authorized users while keeping the access control policy encrypted and also enables them to define access control in terms of access formula over the attributes within the system. In the most effective system the size of cipher-text, time of encryption, and time of decryption scale linearly with the complexity of the access formula. In CP-ABE, the user can manually set some access policy and therefore the data is encrypted consistent with the described access policy. At the receiving end, the user whose attributes satisfy the specified access policy can only able to decrypt the data. It allows users to encrypt and decrypt their data consistent with their attributes.

Although CP-ABE may be a notable access control scheme it is suffering from privacy preservation of access policy due to the possibility of leaking user's private information because the access policy attached in plain text format to the cipher-text. So, the adversary may not get the data/file but can easily collect users' sensitive information. To deal with this problem many hidden policy schemes have been proposed. From partially hiding the access policy to fully hide the access policy through attribute bloom filter with attribute localization algorithm. The paper [1] discussed a new hidden policy CP-ABE to provide big data access control with a privacy-preserving policy. The contribution of this paper includes encryption of the user data with CP-ABE with threshold MSSS with a tree-based access structure, then apply mask technique to hide attributes of access policy which associated with cipher-text. The masking technique applied to every attribute of the access policy. If any user wants to access the stored data from the cloud, the de-mask technique is applied on masked access policy to get original attributes of access policy, then the authorized users can reconstruct the encrypted document by combining their share.

In our paper by taking inspiration from the paper [1], we have implemented the Attribute-based User Access control method over the AWS cloud platform for better security of the data stored in the Cloud. We tried to use the user access policy on an internet site that is hosted on the cloud. With the assistance of an access policy, the data uploaded to the website is monitored and can be uploaded only by the authorized user, thus preventing the backend storage from spamming. To illustrate the implementation of our idea, we have created an internet site for uploading documents and hosted this website on the cloud using an on-demand cloud computing platforms AWS (Amazon Web Service) and used its EC2 (Elastic Cloud Compute) service to make a virtual instance where the web site server are often hosted and used a real-time database and back-end as a service provider, Firebase Storage for securely storing the documents. The restriction is applied within the firebase to restrict unwanted users from accessing our website for uploading documents. This paper also showed the application of the Attribute-Based Encryption method on the data uploaded on the website over the AWS platform to keep control over the data. So that only the authorized users are permitted can be accessed those data.

II. RELATED WORK

Cloud storage as we know become a necessary part of every enterprise in today's scenario. But still, it faces major security issues regarding security and privacy-preserving. The following literature review based on various research papers related to the security aspects of Cloud computing.

As mentioned in [2] Sun et al. described Cloud Computing as many computing components such as computing, storage, and software resources from where consumers can purchase required services, are providing efficient resources to end-users. Researchers have been developed many privacy policies in the cloud such as access control methods, encryption, and trust but still, these policies lack overall protection because of Structural characteristics of the cloud computing environment such as security problems. Nodes involved in computing are diverse, distributed, unable to compute effectively and also the property of on-demand service of the cloud makes the risk of disclosing privacy in the process of transmission, process, and storage. Cloud computing technology and other existing technologies' vulnerabilities can be transferred to cloud computing technology may become a major security threat.

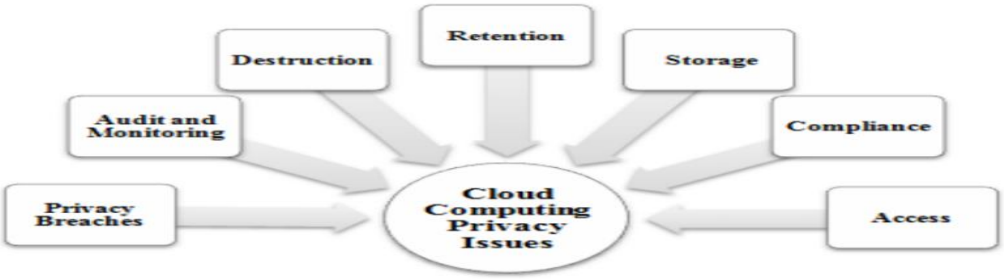


Fig 1. Privacy security risk in Cloud Computing

When corporation businesses have been moving towards cloud platforms, they must have trust in cloud infrastructure, maintenance, and security control. As mentioned by Coles-Kemp et al. [3], 4 deployment models available in cloud system those are private cloud, public cloud, community cloud, and hybrid clouds. In terms of cloud computing security issues shown in Fig.1, there are very big differences between these deployment models due to different controls offered to the enterprises by using any infrastructure of cloud '. The deployment decides the level of control is processed by users as some of the models need great dependence on any cloud vendor as compared to others for data security. Data ownership becomes more complex due to its responsibility and data control. That causes questions not only for ownership but also regarding the safety of data. Another concern is the dependency between the on-going services available whenever required and the connectivity of the internet. In [4] Hui et al. have been proposed a method that is the integration of the Role-based access control model and Attribute-based access control model that is known as RABAC. These models used to establish a relation between the users, their roles, and the permissions and to make the access control model easy and flexible attribute-based access control rules are dynamically applied to the mappings of user-role, role-permission, and user permission. However, still these models suffering from drawbacks in terms of granularity of access control. ACPC is proposed by Heng et al.[5] which is an efficient, secure, and fine-grained Access control mechanism of P2P storage Cloud, that enables the owner of the data to delegate many laborious to a server of the cloud.

In [6] Praveen et al. discussed the Attribute-Based Encryption technique. This is a public key cryptography technique, which provides the facilities to securely sharing data among multiple users. In this scheme data is encrypted by using attributes and decryption takes place by the secret key of the authorized user that is related to an access policy of the data. In [7] Ali et al. discussed by creating a fully hierarchical-based ABE scheme it is made flexible for cloud servers to perform many vigorous tasks. It is also mentioned user side storage cost is less than the cost of similar schemes. Koo et al. described in [8], to generate a function of information retrieval while the privacy and security issues addresses, an encryption scheme is available that is searchable and exploits attribute-based encryption scheme with mixed up attributes that handle different problems such as redundant data present for the same file, a poor response regarding any access policy and the computational overhead occurs due to searching an entity. Challagidad et al. [9] mentioned Role hierarchy algorithm and Hierarchy access structure can also be used to achieve data privacy and multi-authority access for users ' stored data in the cloud. To overcome the issues of data security Lai et al. [10] told cipher-text policy attribute-based encryption can be used to keep data confidential even when the cloud server is not highly secured. In [11] Xu et al. described CPABE that as Cipher-text Policy Attribute-Based Encryption is mostly regarded as the most suitable and secure data protection mechanism in Cloud server due to its flexibility and scalability features for access control. However, Lui et al. [12] discussed sensitive information is contained by CPABE may cause privacy leaks of the data providers and the data receivers.

In [13] Khan et al. have proposed a hidden policy CPABE method based on And-gate access structure but Helil et.al [14] described the tree-based access structures can express more strongly and has more flexible access control capabilities. CPABE with the hidden policy that is CP-ABE-HP based on tree-based access structure protects the policy efficiently and having flexible access control capabilities. Due to the hidden access control nature of CP-ABE-HP, the data owners can share their encrypted data with authorized users. This Hidden policy scheme can be deployed in a mobile cloud environment also. In [15] Odelu et al. discussed by using the Hidden policy CPABE scheme the data outsourced to the cloud environment can be remain protected from unauthorized users. The CPABE designed to deploy in the mobile cloud generally contains the constant size of secret keys and ciphertexts as mobile devices are generally resource-restricted.

III.PROBLEM STATEMENT

In this section, we discuss the Preliminaries and Technologies Used in our proposed method.

A. Preliminaries

1) *CP-ABE:* This is a Public Key Encryption technique. In this method, users can encrypt and decrypt their data according to the data attributes. Users can define an access policy for the data and that access policy is represented as an access tree over the attributes. They can also perform encryption of the data based on a defined access policy. The authorized users who will be satisfying the access policy can only be eligible to decrypt the data.

CPABE consist of 4 algorithms: Set up, Keygen, Encryption, and Decryption

a) Set up: Input is Security Parameter that generates Public Key and Master Key as output.

b) Keygen: This algorithm takes Public Key, Master Key and Attributes set as input and generates Secret Key

c) Encryption: The inputs of this algorithm are Public Key, Plain text message, and Access Policy defined by the user. It generates Cipher-text with masked Access Policy as output.

d) Decryption: This algorithm takes input as Cipher-text policy along with masked Access policy associated with it then generates Plain text message as Output.

2) *Access Structure:* A tree-based Access Structure is used in which the user-defined Access Policy represents as an Access Tree. The interior nodes of an access tree is a threshold gate such as AND gate OR gate and the left nodes contain attributes. Whose attributes of secret key satisfy the access tree can only be able to decrypt Cipher-text.

B. Technologies Used

1) *AWS Cloud*: AWS is a daughter company of Amazon. It provides on-demand cloud computing services and APIs to its customers such as an individual, corporation, or government on the basis of pay as you use policy. A various set of technical infrastructure, computing resources, and features have been provided by these cloud computing web services. By using AWS, We can request such as compute power, storage, and other AWS services within few minutes and have the flexibility to choose the appropriate platform for development or any programming model that is most suitable to solve our problem. It can be easily identified from other on-demand cloud services providers in the IT computing sector as it is flexible, cost-effective, and elastic. The top 5 services are provided by AWS: Amazon Elastic Cloud Compute(EC2), Amazon S3(Simple Storage Service), Amazon Virtual Private Cloud(VPC), Amazon CloudFront, and Amazon Relational Database Services (RDS).

2) *PuTTY:* It is a free and open-source terminal-based emulator application. It acts as a client for the SSH, Telnet, rlogin, and raw TCP computing protocols. The name of PuTTY is no definite meaning but TTY is a terminal name in Unix tradition. Originally it was written for Microsoft windows but now it is ported to many other operating systems. There also some official ports available for Unix like platforms but there are some work in progress port available for Classic Mac OS, MAC OS, and also unofficial ports.

3) *FileZilla:* FileZilla client also called FileZilla. It is an open-source, free, and FTP client of cross-platform. It contains an FTP server, a pair of programs, and an FTP client. FTP protocols are used in a network for transferring files. This software is absolutely legal and the file transfers may or may not be dependent on whether the files are protected through copyright or not or if the permission is granted by the owner of the copyright or not.

4) *Firebase:* It is a platform for mobile and web application development provided by Google that helps to create, improve and grow high-quality applications. There are a total of 19 products that come under firebase which are used by more than a million applications. It provides various services such as Google Analytics, Real-time Database, Cloud Messaging, Storage, Hosting, and many more. It also provides services to monitor the performance of the application and maintains a detailed report on the errors of the application.

IV. OUR PROPOSED IMPLEMENTATION

We show the detailed implementation process of creation and deployment of our website in AWS Cloud in a step-wise manner with proper snap shorts.

A. Creation and Deployment of Website to AWS cloud

1) New EC2 instance creation process: Login to AWS account

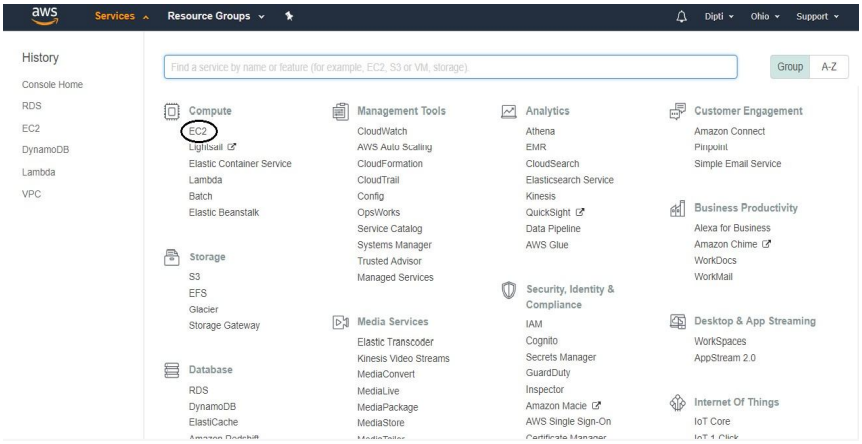


Fig.2 AWS Home page

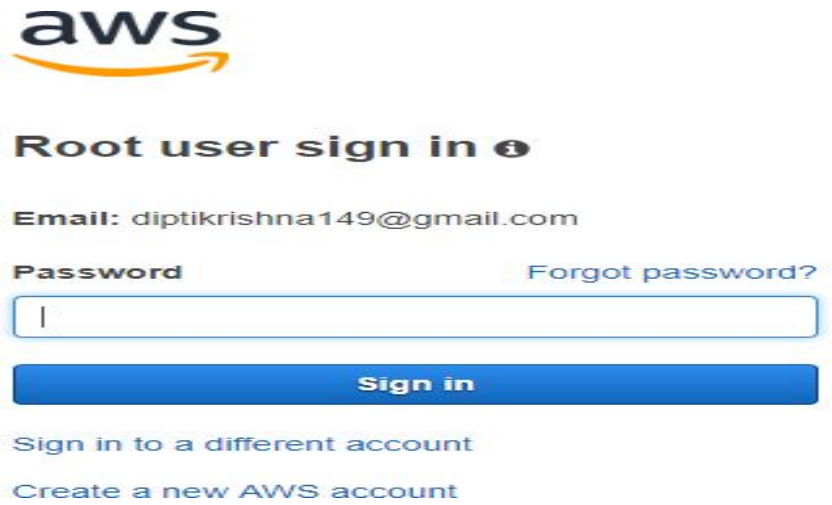


Fig.3 Login page of AWS Console

2) For launching the EC2 dashboard, Login to the AWS Console then selects the EC2 Service. In the dashboard click on "Launch Instance" that will launch the EC2 instance.

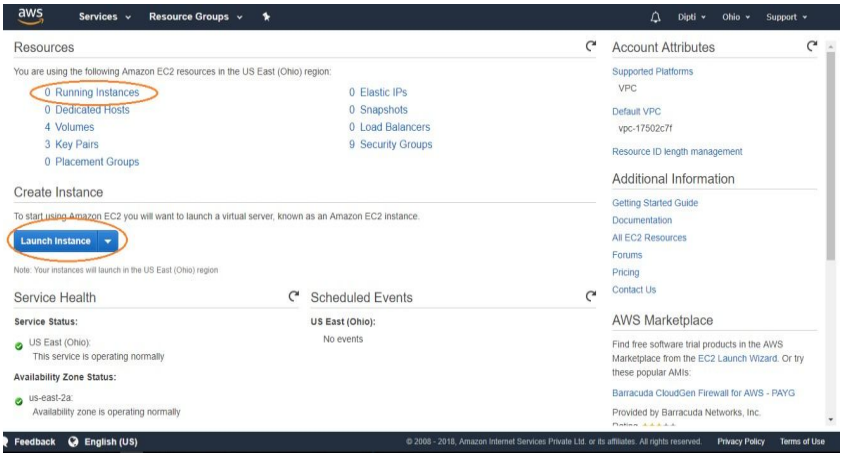


Fig.4 Lunching EC2 Instance

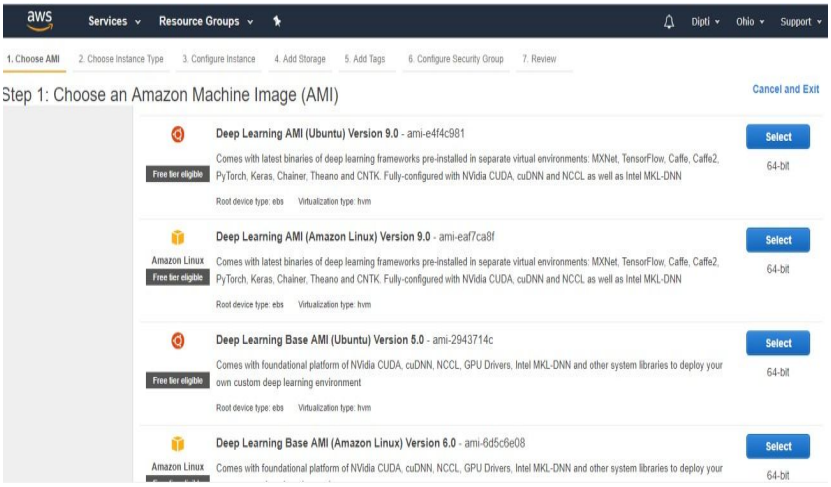


Fig.5 Selecting AMI

3) In the below figure there are various instance types are shown. Select the T1 Micro then press "Next configuration setting" button.

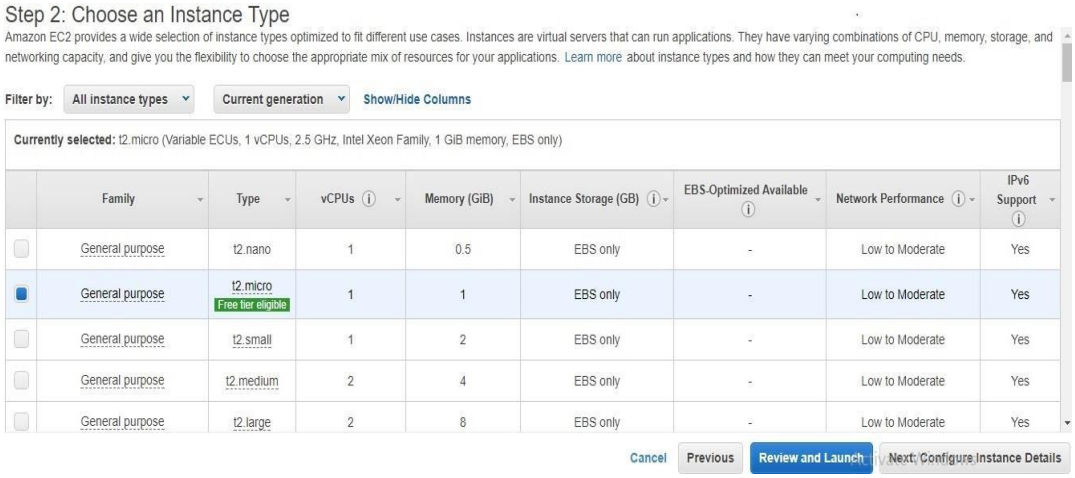


Fig. 6 Selecting Instance type

4) Provide IAM role as None and click on Next: Add Storage

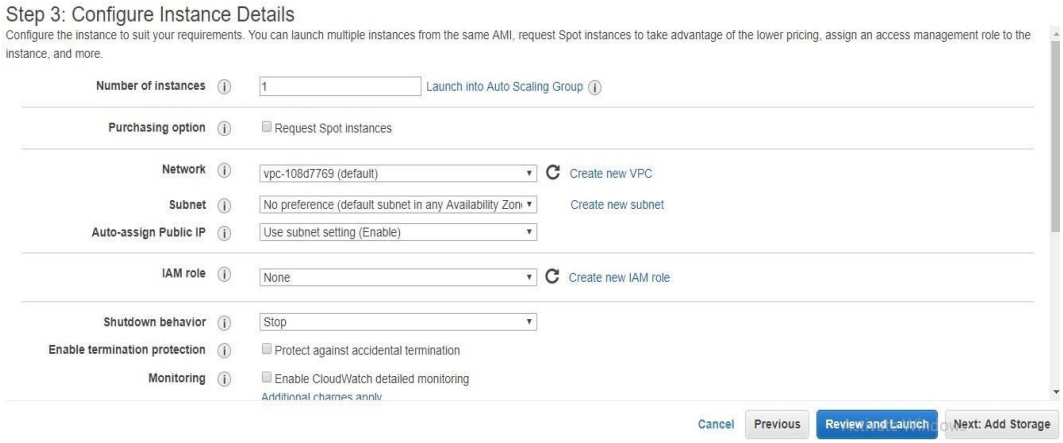


Fig 7. Instance Configuration

5) Provide the storage related information

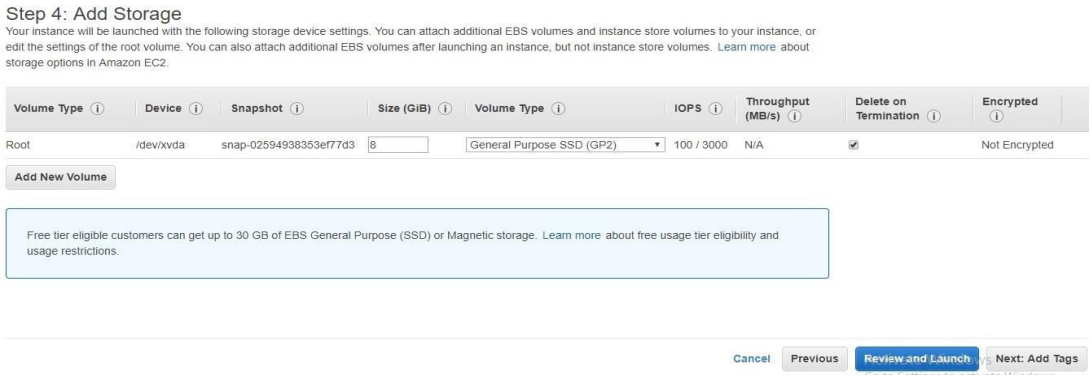


Fig 8. Storage information in Amazon EC2 account

6) Provide the tags for the AWS instance

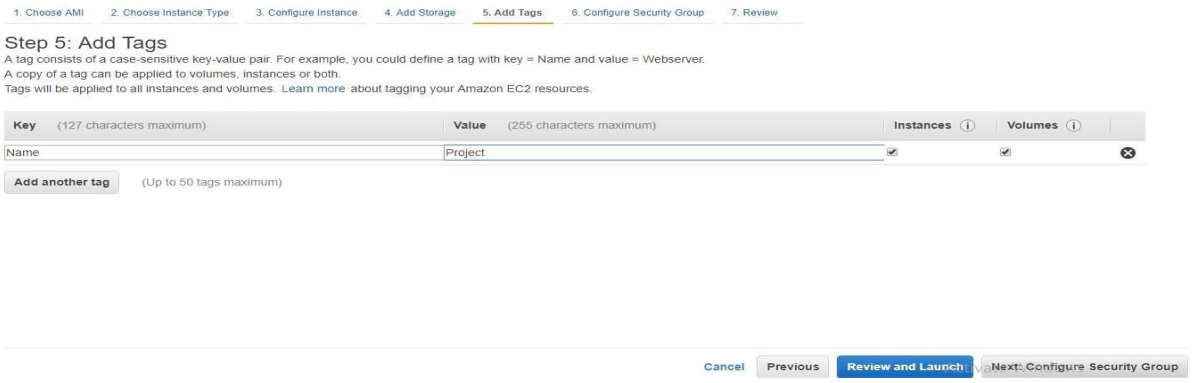


Fig.9 Tag information in Amazon EC2

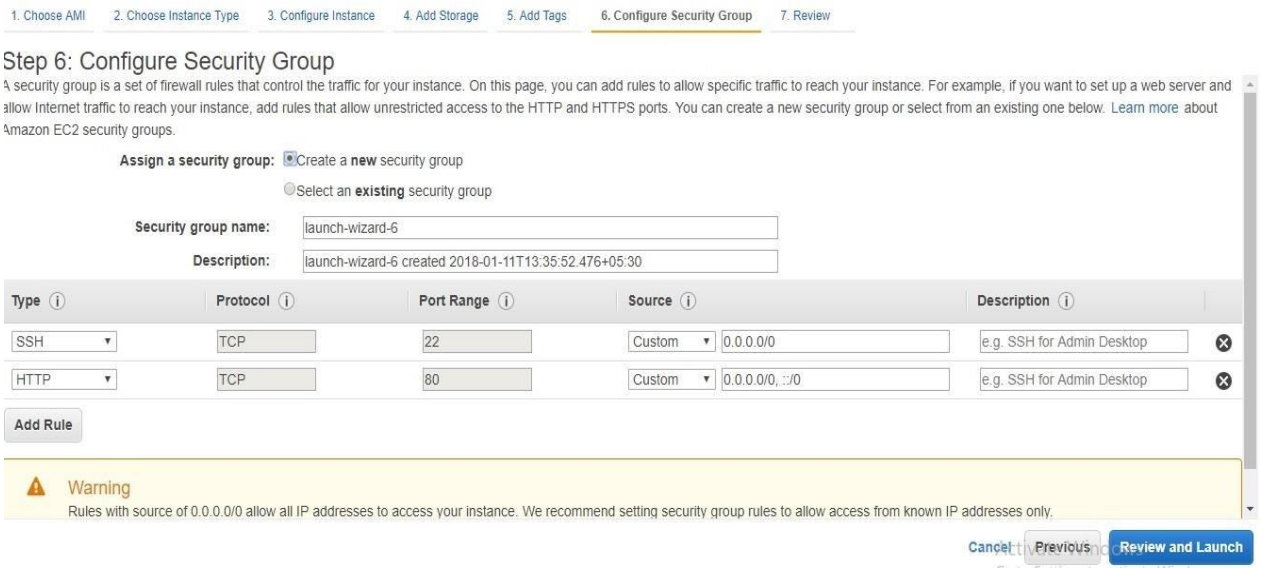


Fig 10. Security group Configuration

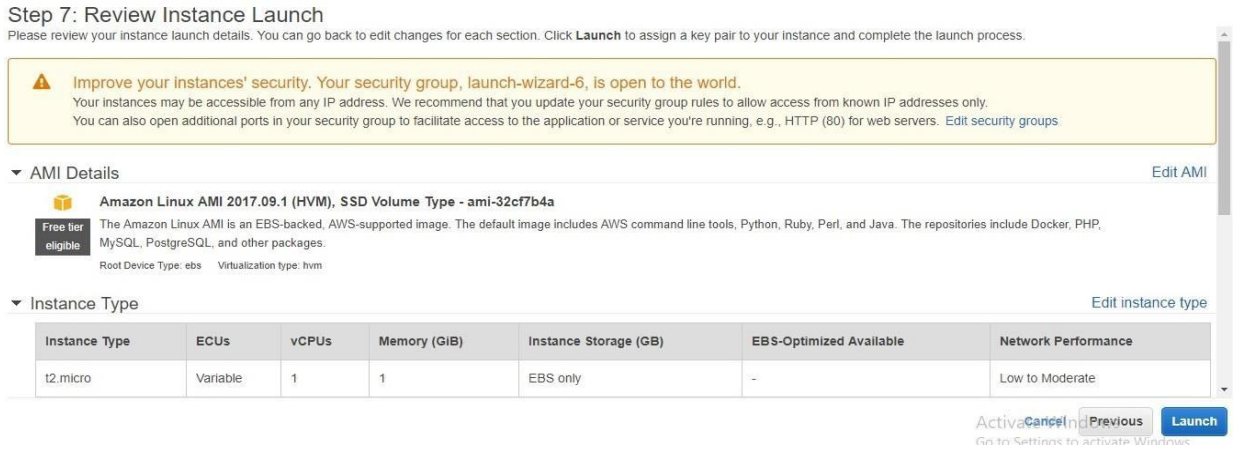


Fig. 11 Review of Instance Type

7) For the security of the instance, select the existing key-pair or create a new key-pair. Provide the name of the key pair and click on "Create & Download our key pair" to create a new key-pair.

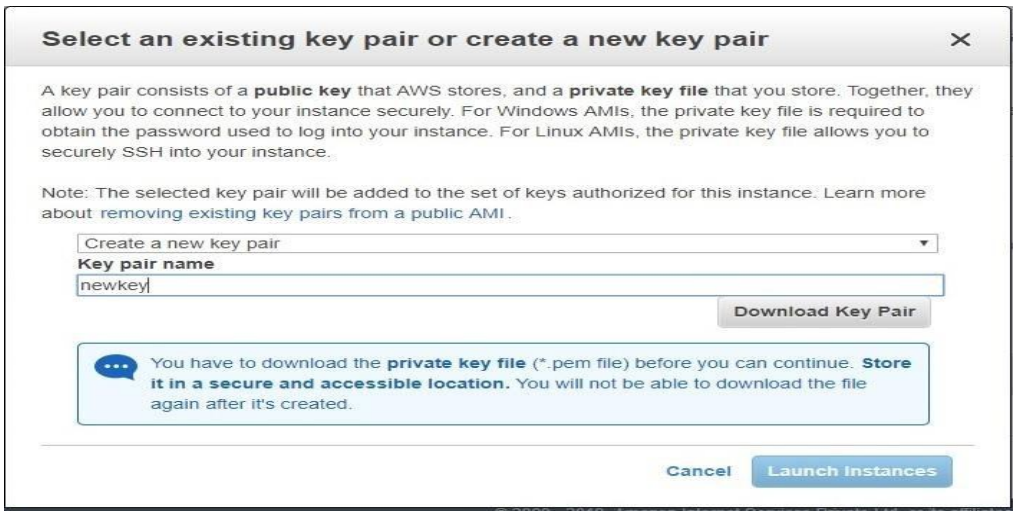


Fig 12. New key pair creation

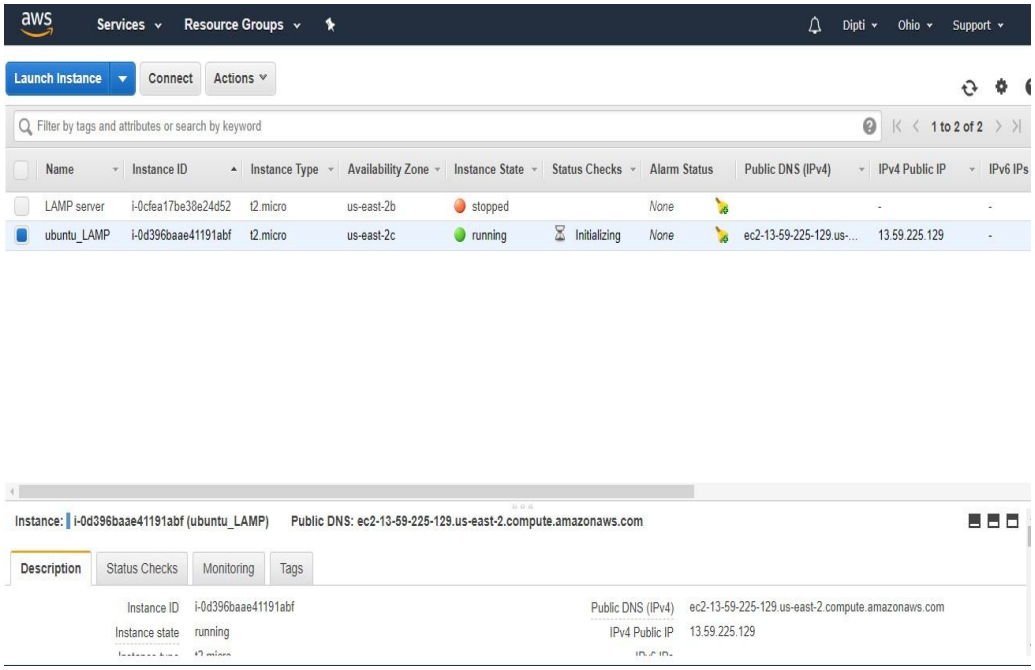


Fig 13. Launch EC2 instance

Steps to Launch EC2 instance

a) Connecting to the EC2 Instance Remotely

b) Once the instance is launched, we should connect it. We use PuTTY and PuTTYgen tools to connect to the UBUNTU instance by using the public DNS of the instance.

c) For connecting from a Windows machine, for that the key-pair file myfirstkey.pem needs to be converted to .ppk file. To perform this task of conversion, PuTTYgen tool is used.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue II Feb 2021- Available at www.ijraset.com

8) Open the PuTTYgen tool.

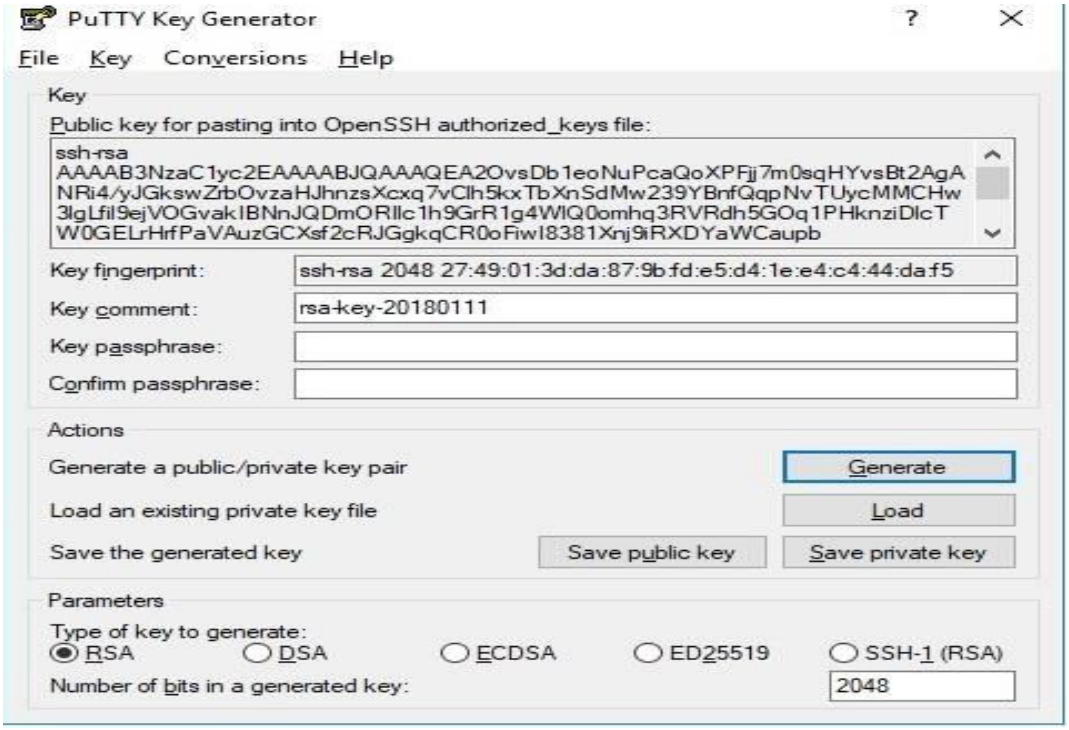


Fig. 14 Dashboard of PuTTYgen tool

9) Click the Load button and select the myfirstkey.pem and load it in PuTTygen.

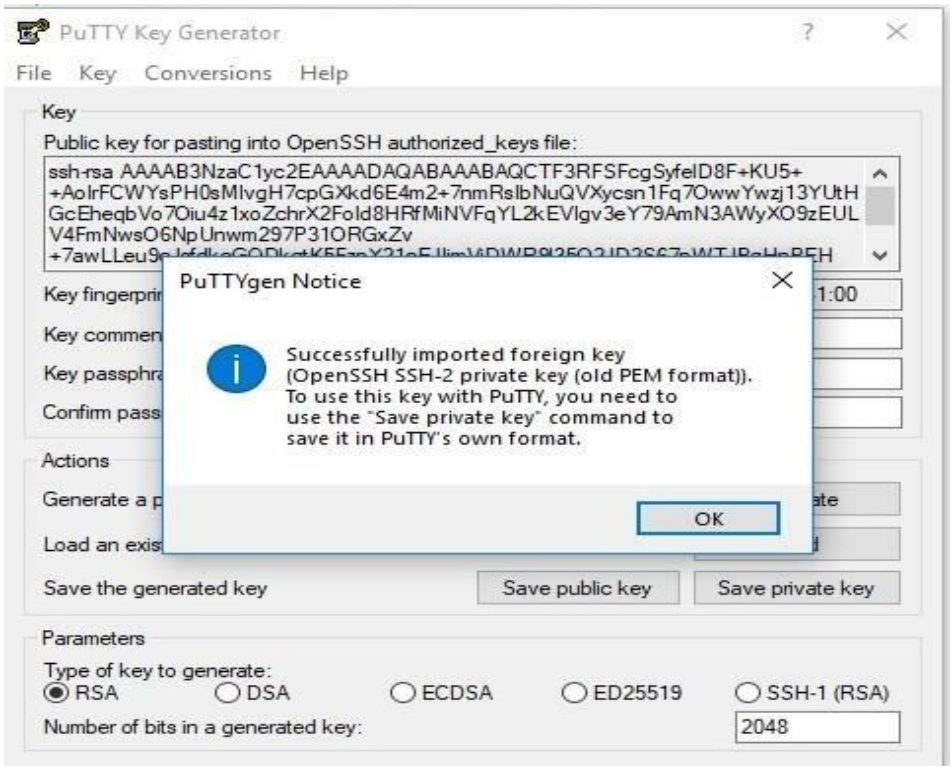


Fig. 15 successfully .pem file loaded in PuTTygen

10) Click on Save then Private Key and save the file as myfirstkey.ppk.

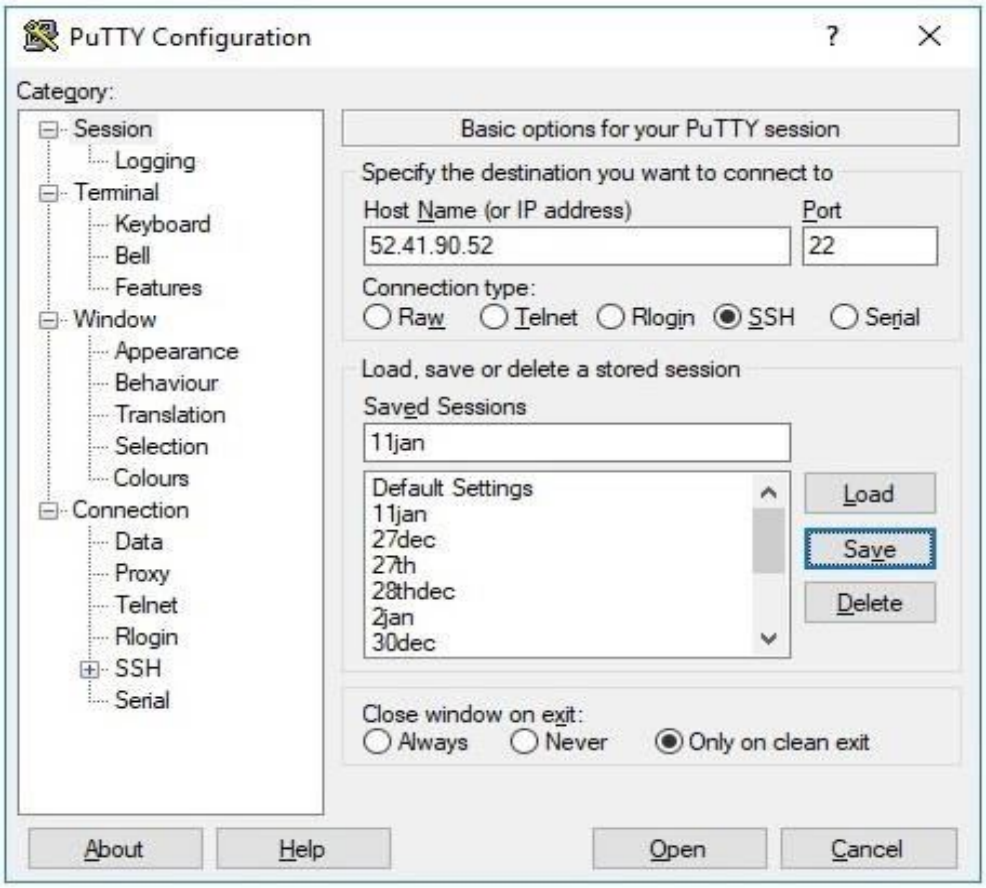


Fig. 16 Private key Saved with .ppk extension

11) Start PuTTy by entering the public DNS that can be found in the Host name/IP address field.

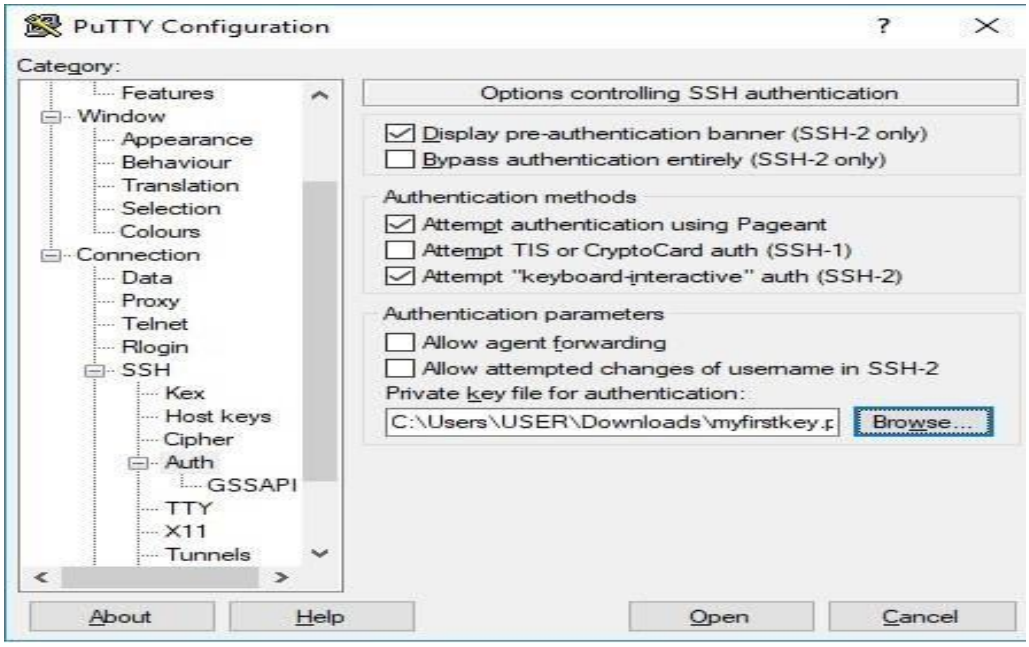


Fig. 17 Starting of PuTTy

12) In the Category tree, select SSH >Auth provide the key-pair file to launch of the instance that connect to the instance.

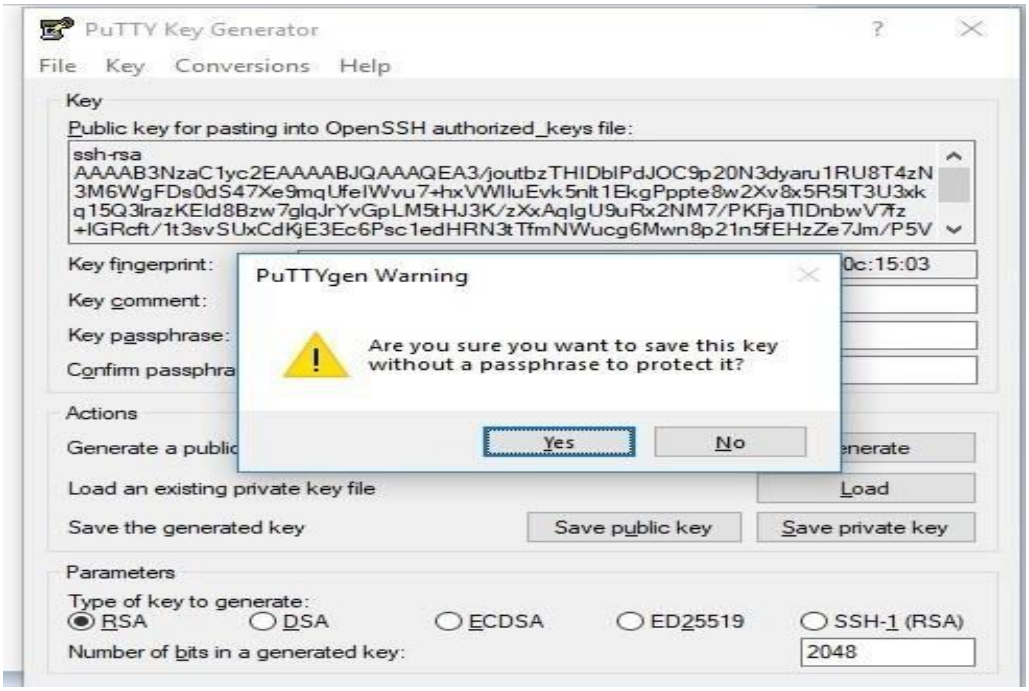


Fig.18 Connecting to Instance

13) Open the command window (telnet) that launches to connect to the AWS instance.

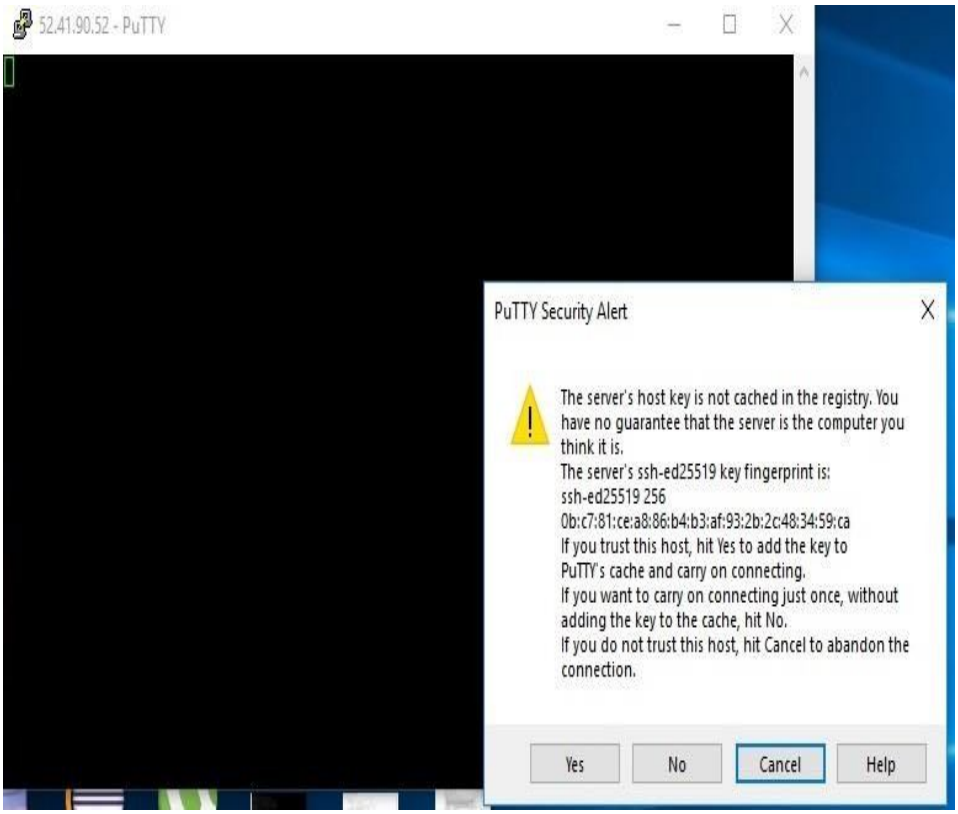


Fig. 19 Lunching of Command Window

14) Give the correct IP address to display the below Linux prompt

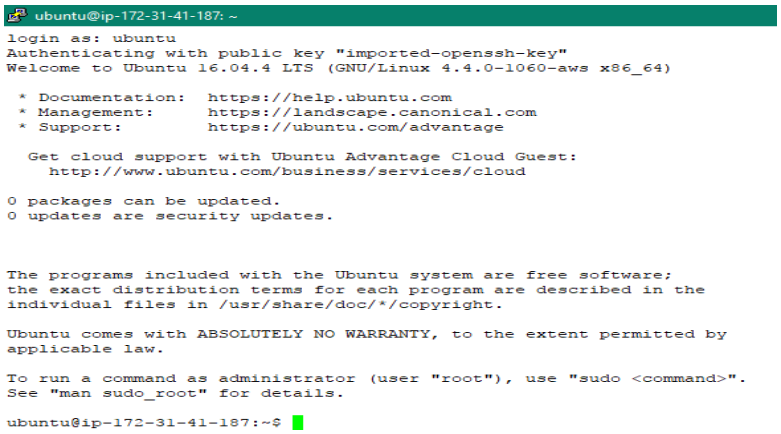


Fig. 20 Linux Prompt

15) To Process of Installing LAMP server in UBUNTU, we need LAMP stack, Linux Operating System acronym, Apache HTTP server, MySQL relational database management system and general-purpose scripting language PHP, is a web service stack consisting of a group of open source software which is installed together on a server to enable it to host dynamic websites and web apps.

16) To assure that all software packages are up to date, perform a quick software updation on the instance by typing command : $ sudo apt-get update

Fig 21. Updating on AWS Instance

17) To install Apache and Allow in Firewall type command: $ sudo apt-get install apache2

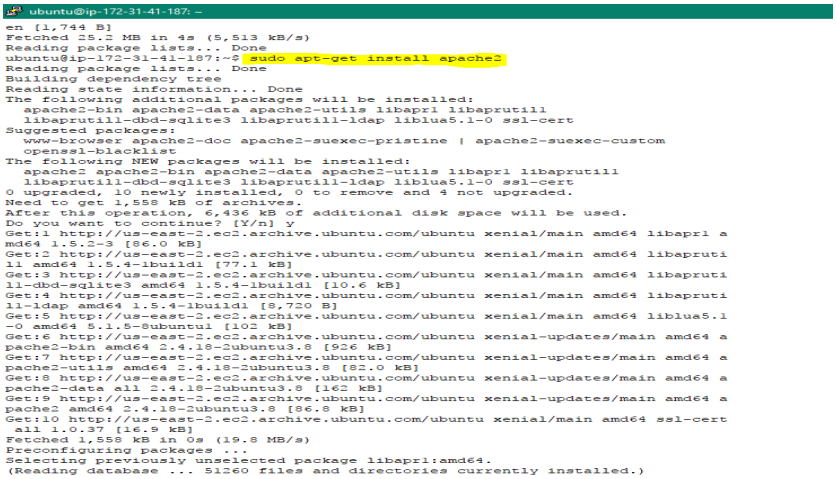


Fig 22. Installing Apache

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue II Feb 2021- Available at www.ijraset.com

18) Set Global ServerName to Suppress Syntax Warnings through command : $ sudo apache2ctl configtest

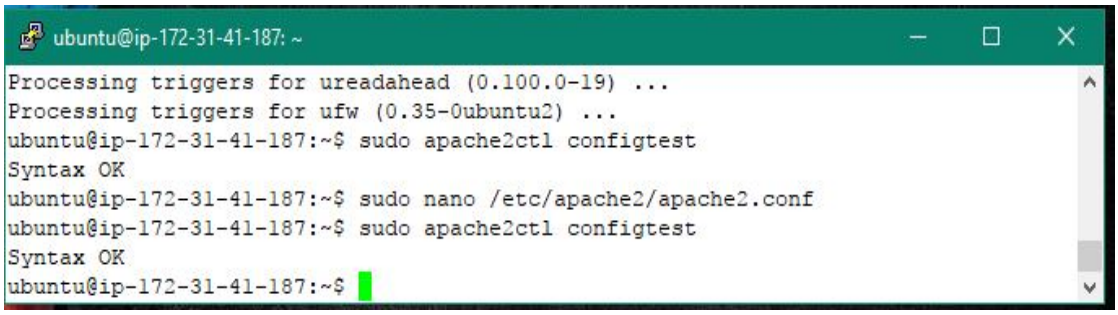


Fig 23. Global Server Name setting

19) Open the main configuration file with text edit by the command : $ sudo nano /etc/apache2/apache2.conf

20) The file inside adds a ServerName directive to that file, pointing to the primary domain name, or we can use the server's public IP address.

21) Then Save and close the file when you are finished

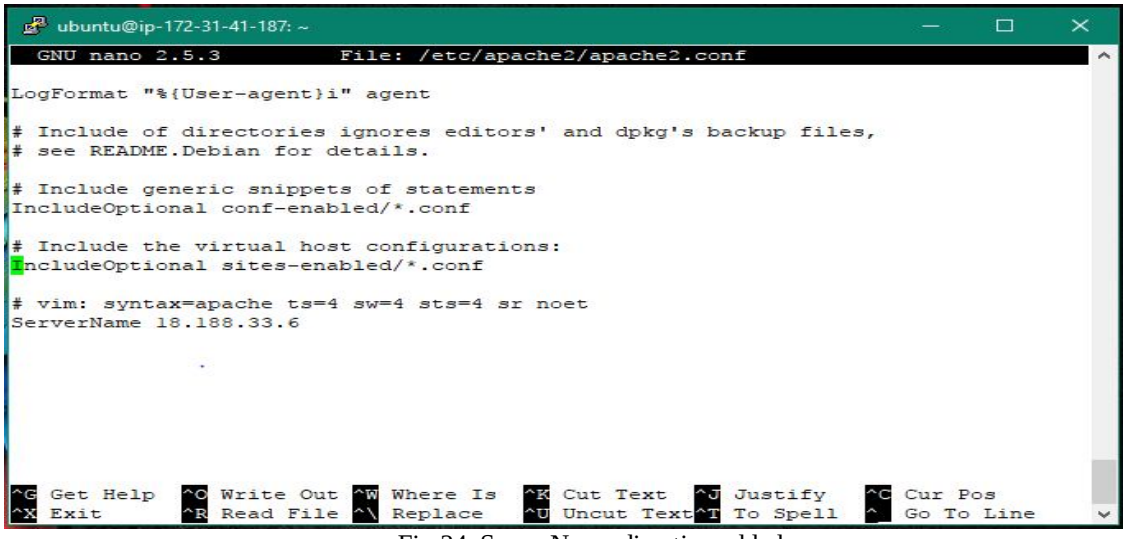


Fig 24. ServerName directive added

22) Check the syntax errors through command : $ sudo apache2ctl configtest

23) Restart Apache to implement the changes by: $ sudo systemctl restart apache2

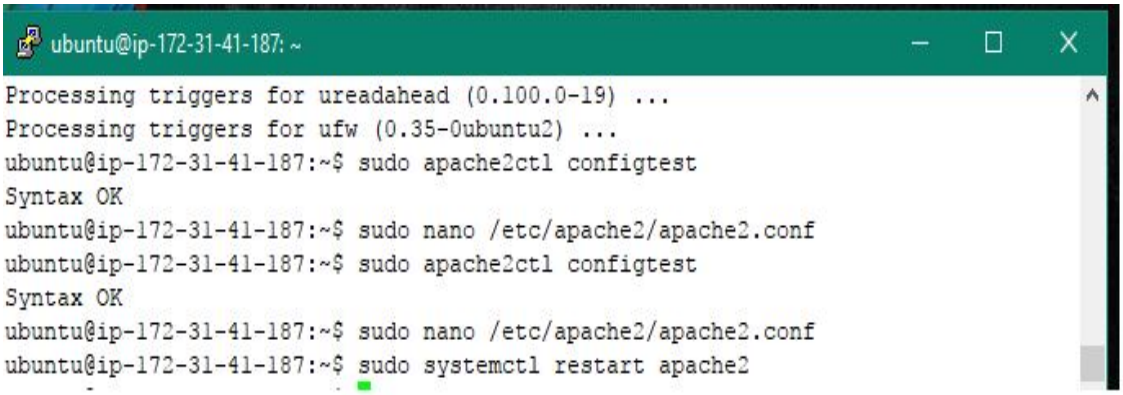


Fig 25. Apache Server restarted

24) Adjust the Firewall to Allow Web Traffic through: $ sudo ufw app list

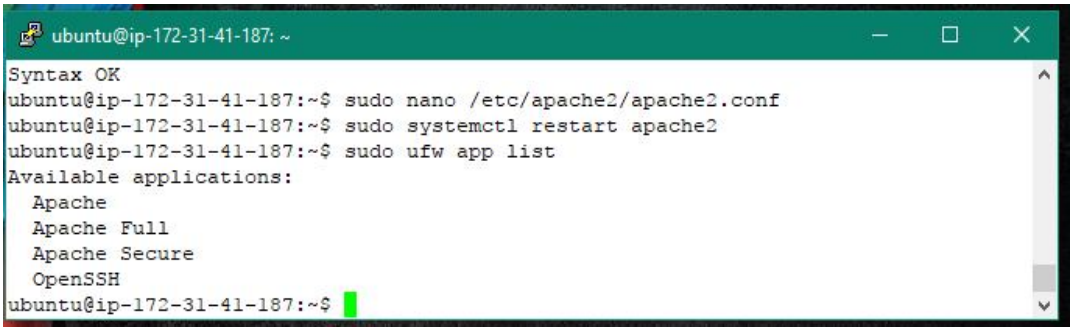


Fig 26. Firewall Adjusted

25) The Apache Full profile should show that it enables traffic to ports 80 and 443 through $ sudo ufw app info "Apache Full

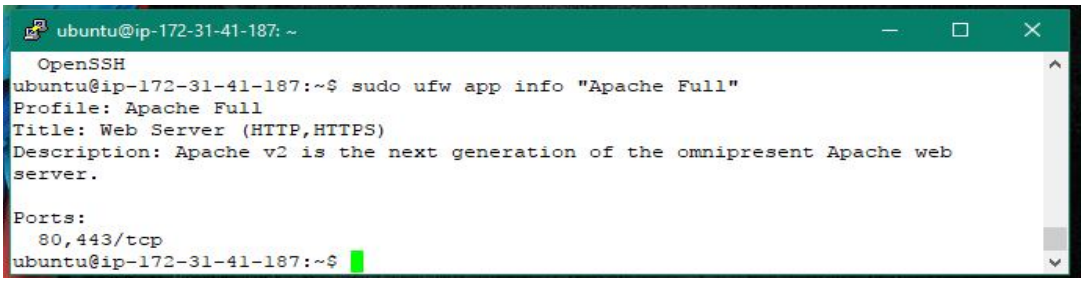


Fig 27. Traffic to ports 80 and 443 is enabled

26) Allow incoming traffic to the profile through : $ sudo ufw allow in "Apache Full"

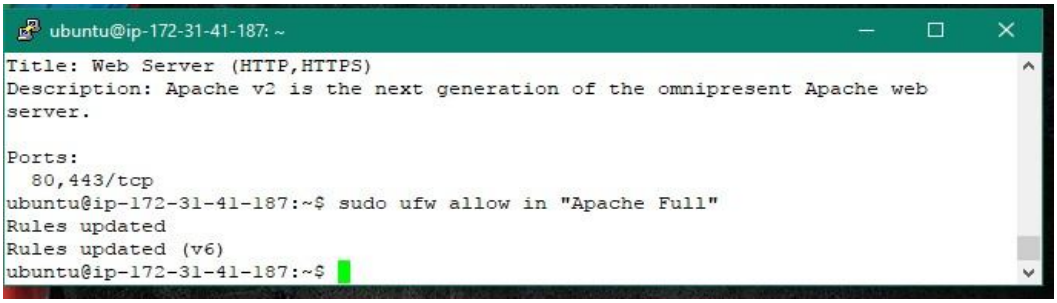


Fig 28. Traffic is allowed to the profile

27) Testing Apache through: http://your_server_IP_address

Fig 29. Apache Server

28) Steps to install Apache Server in Ubuntu

a) Install PHP through command: $ sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql

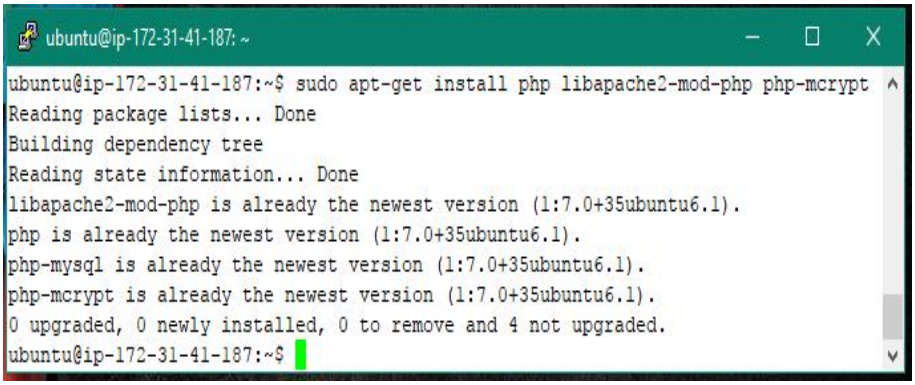


Fig 30. PHP installed

b) Open the dir.conf file through a text editor with root privileges then for moving the PHP index file given above to the first position after the DirectoryIndexspecification, for that restart the Apache service through command: $ sudo systemctl restart apache2

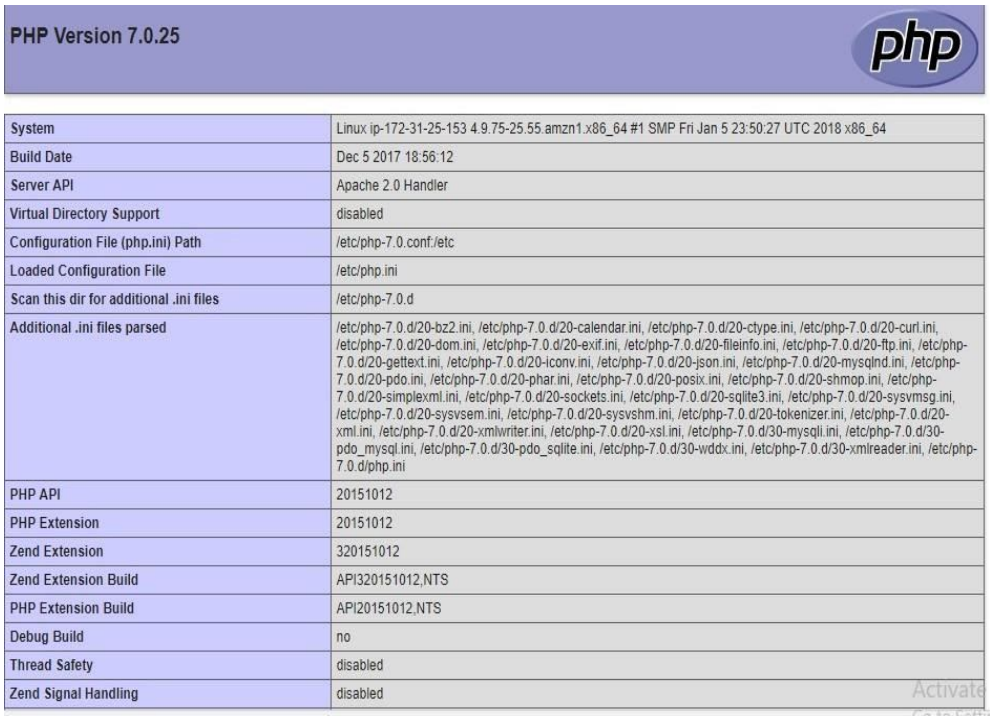
29) To test Lamp Server

30) Install the server and set permissions correctly, then create a PHP file through the user account in the /var/www/html directory.

31) Also PHP file should be created in the Apache document root through the command: $ sudo nano /var/www/html/info.php

32) Then a blank file will open. Write valid PHP code inside the file through " http://your_server_IP_address/info.php" command

33) Type the just created URL in the web browser. The URL act as public DNS address of the instance followed by file name and a forward slash.



System	Linux ip-172-31-25-153 4.9.75-25.55.amzn1.x86_64 #1 SMP Fri Jan 5 23:50:27 UTC 2018 x86_64
Build Date	Dec 5 2017 18:56:12
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php-7.0.conf./etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php-7.0.d
Additional .ini files parsed	/etc/php-7.0.d/20-bz2.ini, /etc/php-7.0.d/20-calendar.ini, /etc/php-7.0.d/20-ctype.ini, /etc/php-7.0.d/20-curl.ini, /etc/php-7.0.d/20-dom.ini, /etc/php-7.0.d/20-exif.ini, /etc/php-7.0.d/20-fileinfo.ini, /etc/php-7.0.d/20-ftp.ini, /etc/php-7.0.d/20-gettext.ini, /etc/php-7.0.d/20-iconv.ini, /etc/php-7.0.d/20-json.ini, /etc/php-7.0.d/20-mysqlnd.ini, /etc/php-7.0.d/20-pdo.ini, /etc/php-7.0.d/20-phar.ini, /etc/php-7.0.d/20-posix.ini, /etc/php-7.0.d/20-shmop.ini, /etc/php-7.0.d/20-simplexml.ini, /etc/php-7.0.d/20-sockets.ini, /etc/php-7.0.d/20-sqlite3.ini, /etc/php-7.0.d/20-sysvmsg.ini, /etc/php-7.0.d/20-sysvsem.ini, /etc/php-7.0.d/20-sysvshm.ini, /etc/php-7.0.d/20-tokenizer.ini, /etc/php-7.0.d/20-xml.ini, /etc/php-7.0.d/20-xmlwriter.ini, /etc/php-7.0.d/20-xsl.ini, /etc/php-7.0.d/30-mysqli.ini, /etc/php-7.0.d/30-pdo_mysql.ini, /etc/php-7.0.d/30-pdo_sqlite.ini, /etc/php-7.0.d/30-wddx.ini, /etc/php-7.0.d/30-xmlreader.ini, /etc/php-7.0.d/php.ini
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012
Zend Extension Build	API320151012,NTS
PHP Extension Build	API20151012,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled

Fig. 31 PHP installed and tested successfully for the deployment of the LAMP server

B. Transferring the file from windows to UBUNTU using FileZilla

1) Open FileZilla

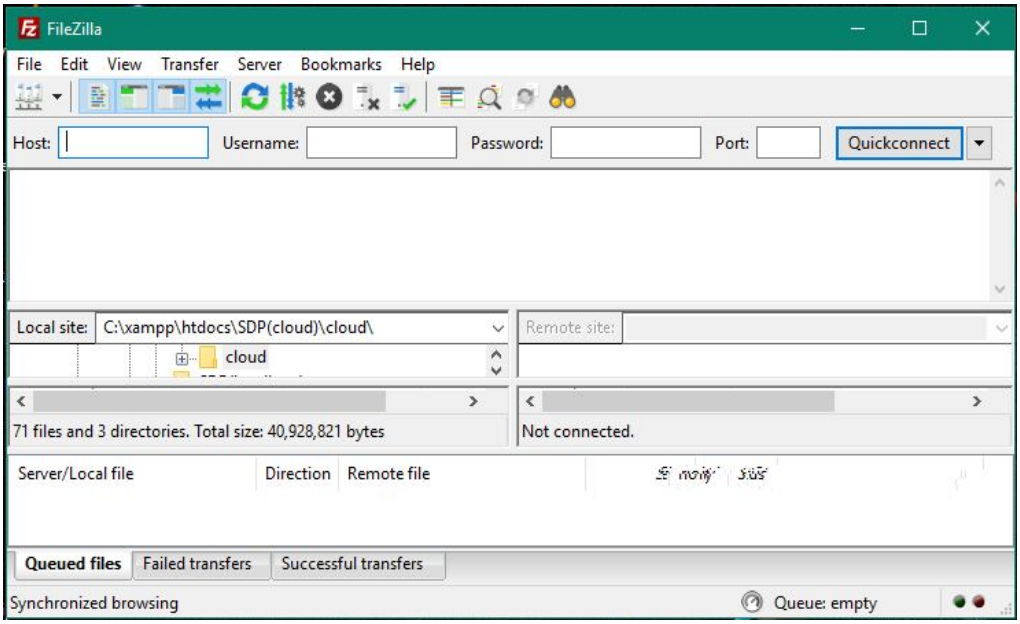


Fig. 32 FileZilla dashboard

2) Go to Settings in Edit tab, SFTP attach the private key pair of the created UBUNTU instance

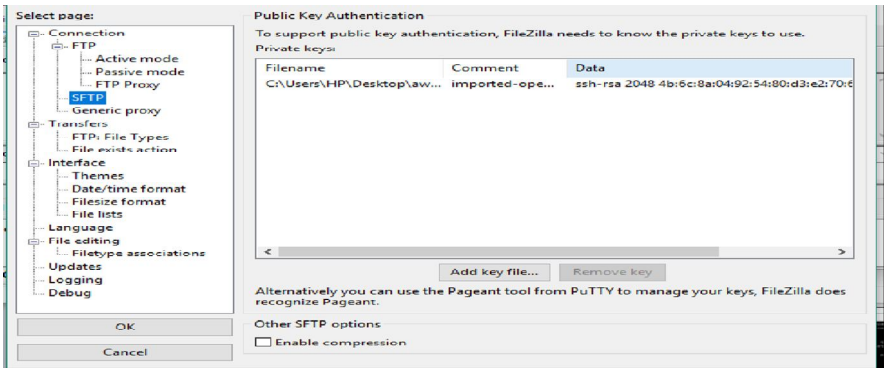


Fig. 33 Private Key is attached

3) Start FileZilla with our public IP address and user name.

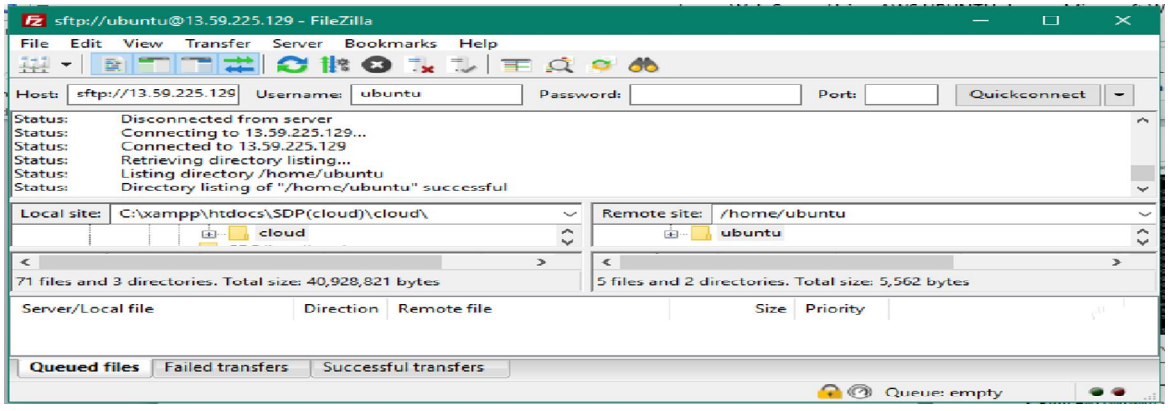


Fig. 34 Starting of FileZilla

4) Now transfer our desired dynamic PHP file from Windows to Ubuntu instance by simply uploading the file here

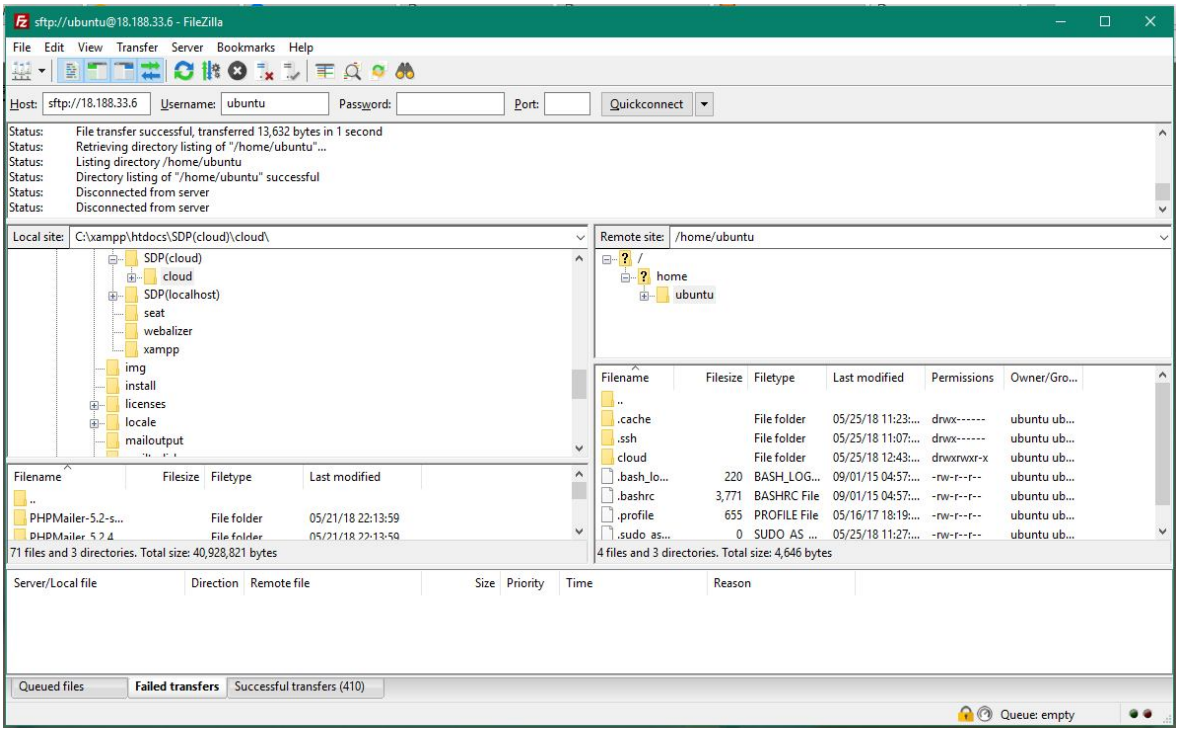


Fig. 35 Transferring of PHP file

5) Make sure the transferred file are in /var/www/html directory of Ubuntu instance

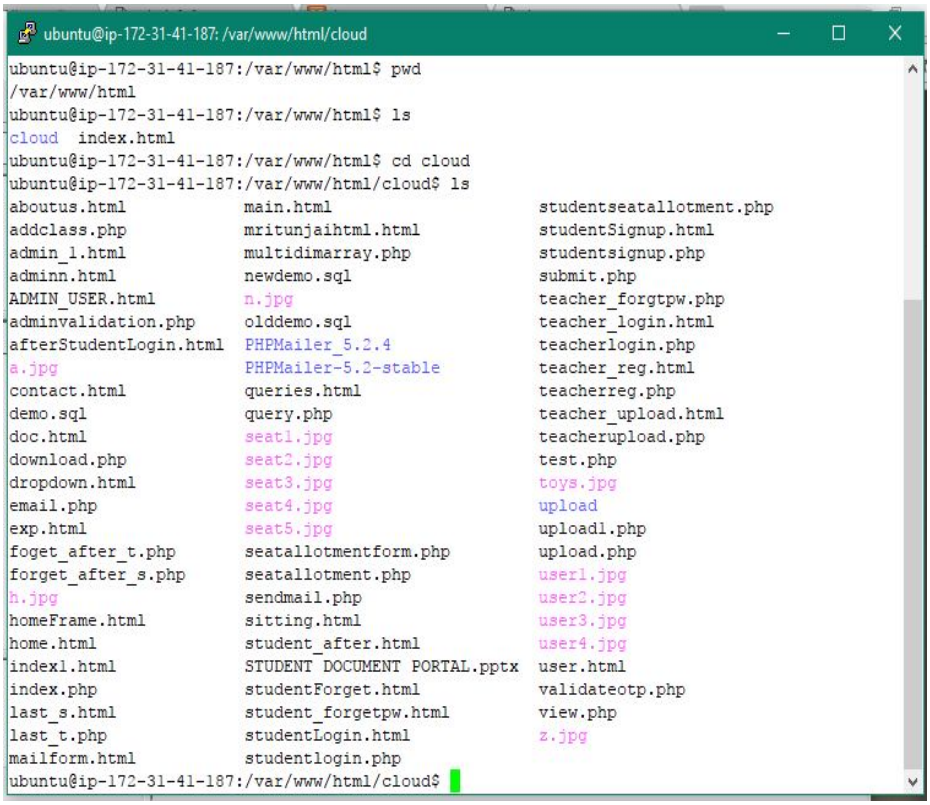


Fig. 36 Php file is transfered

C. Accessing Website Globally

Give the public ip address of the UBUNTU instance followed by the directory in which the whole php file is stored in /var/www/html in any browser to access the website globally.

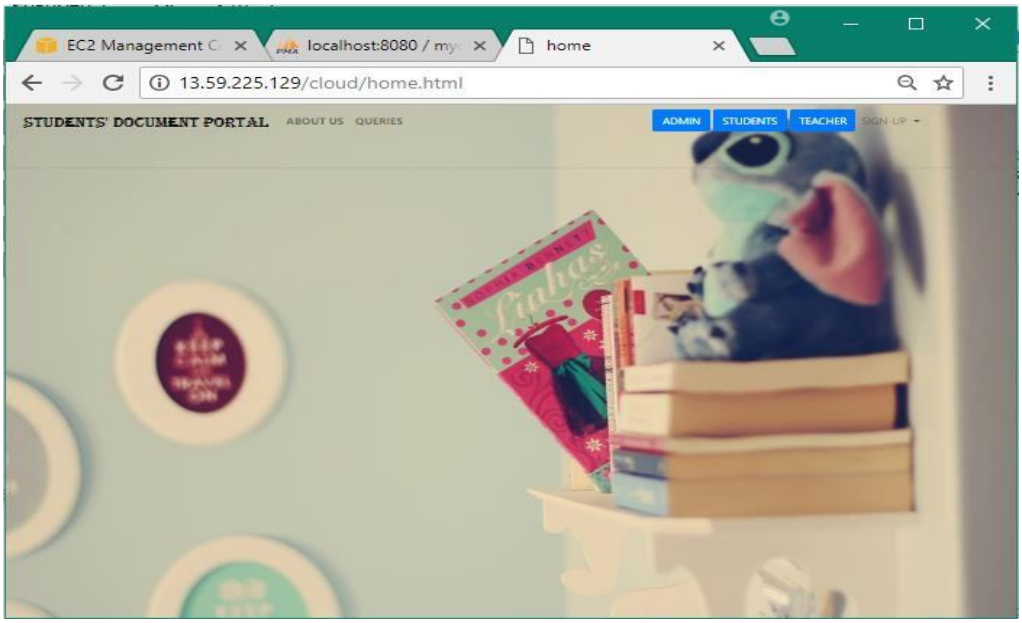


Fig. 37 Home page of the Website

D. Development Of Web Application And Adding User-Level Security At Backend Storage

The front end part of the web application was developed using HTML that is HyperText Markup Language and CSS(Cascading Style Sheet). HTML along with CSS enables a user to make simple web pages. The back end of the website was configured using Firebase. Firebase is a platform that provides web and mobile application development. Google acquired Firebase in 2014.

1) Development of web Application

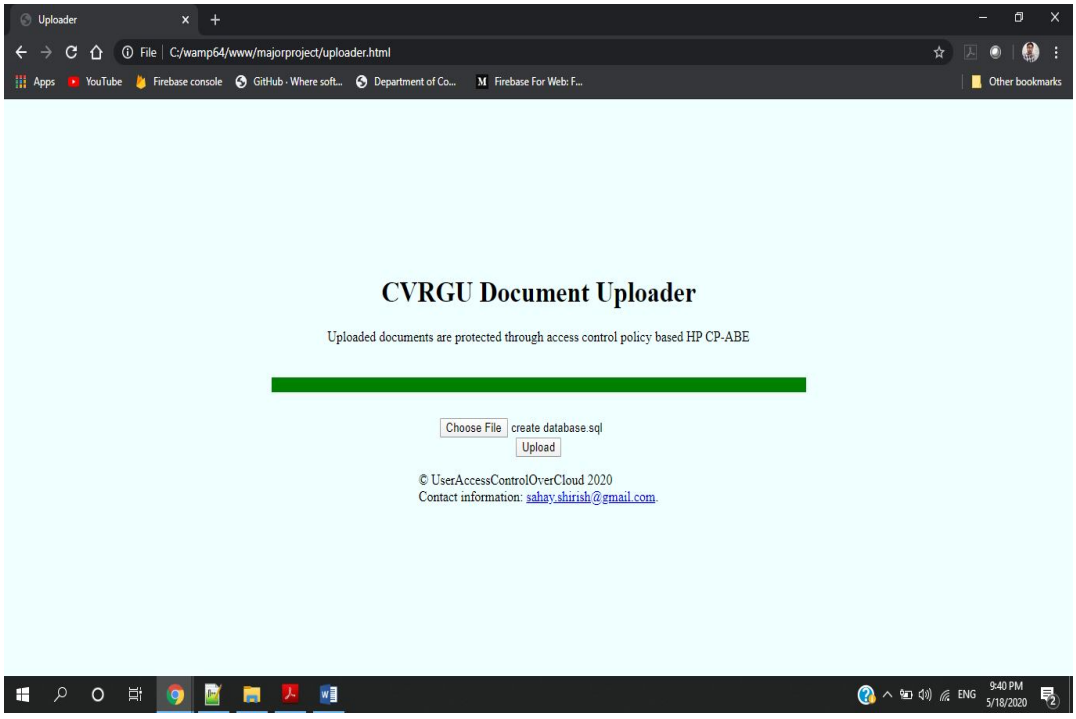


Fig. 38 Home page of the created account in Firebase

2) Click on add project from firebase console

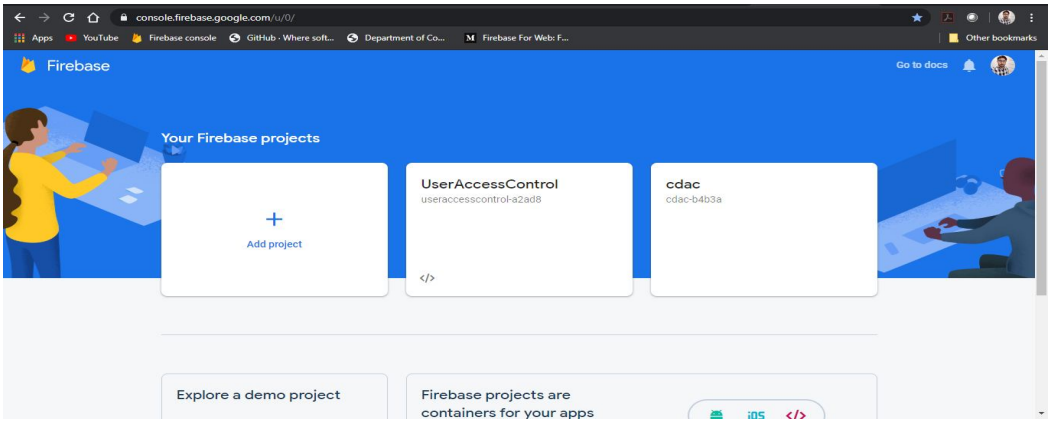


Fig. 39 Dashboard of Firebase

3) Provide the project name

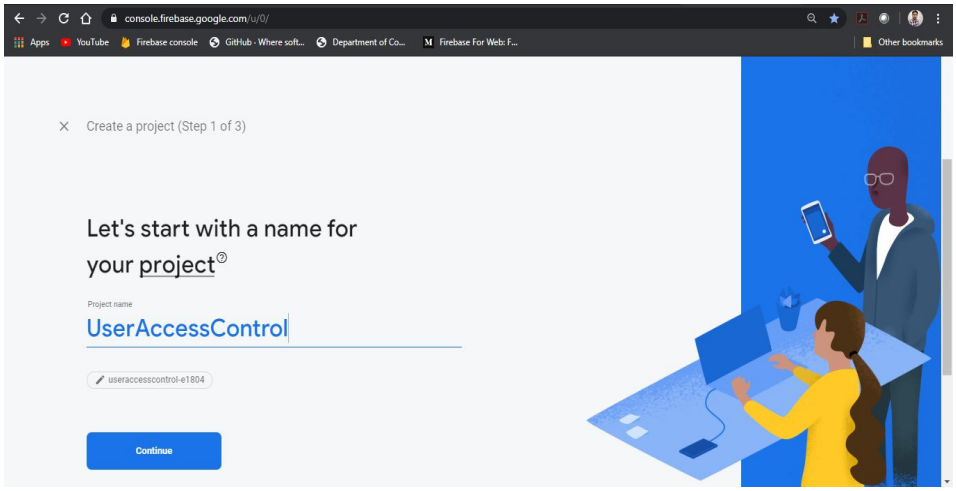


Fig. 40 A Project name added

4) Select Google analytics if required and click on continue to create the project.

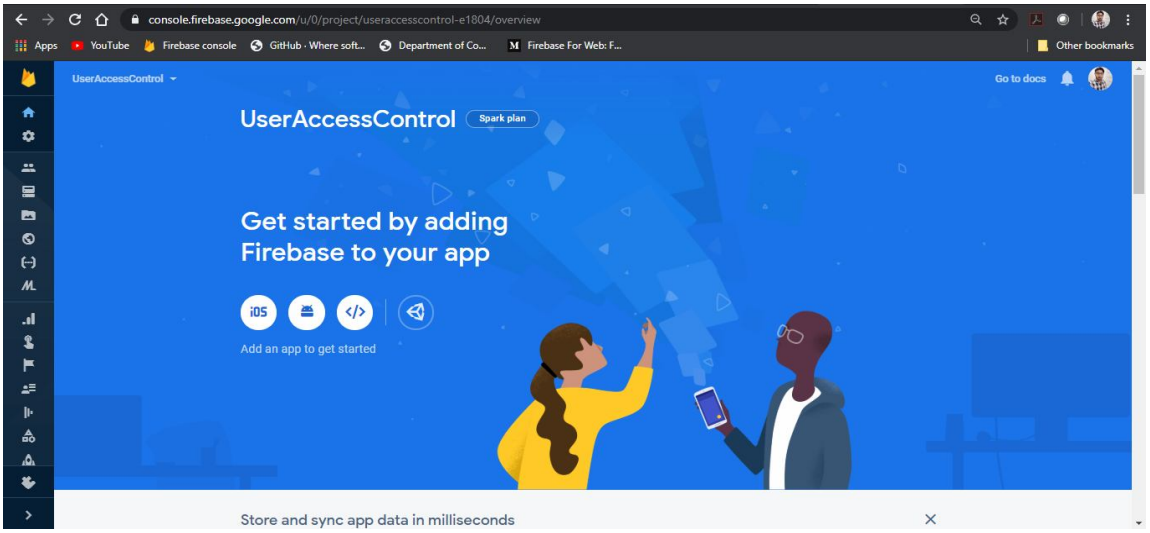


Fig. 41 Project is created

5) Add firebase to the web Application

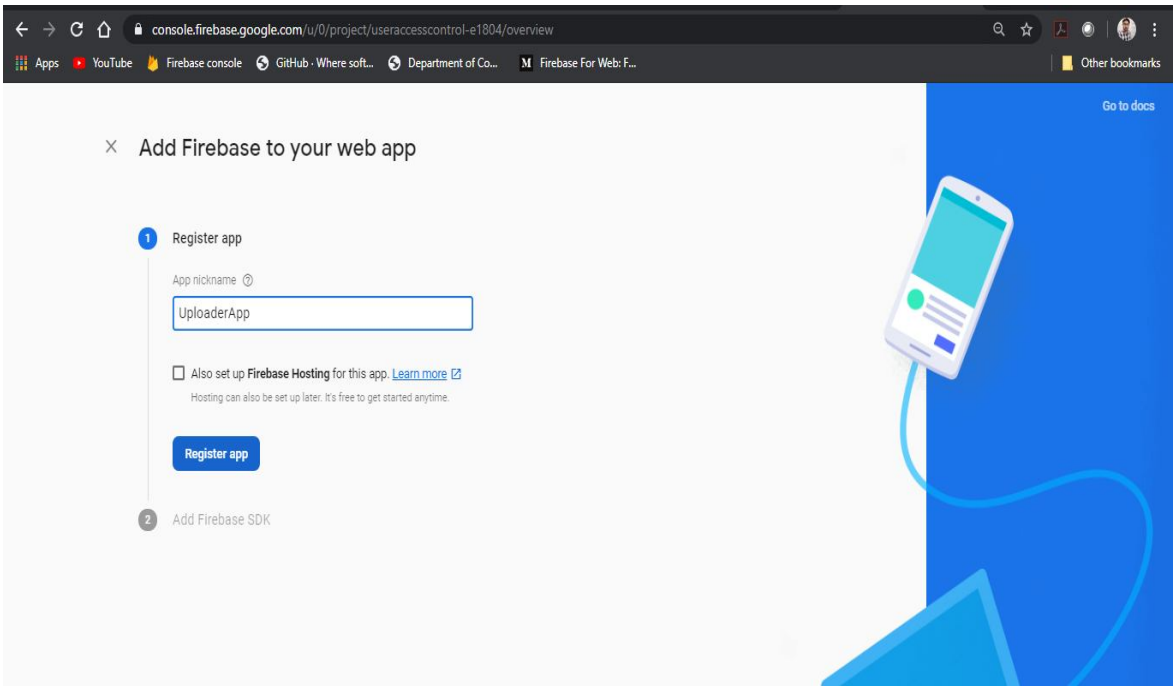


Fig. 42 Firebase is added

6) Firebase provides javascript code to add Firebase SDK to the created web application. There are multiple ways to add firebase SDK like through Firebase Hosting URL, through Node.js applications or CDN. Click on continue to console to proceed.

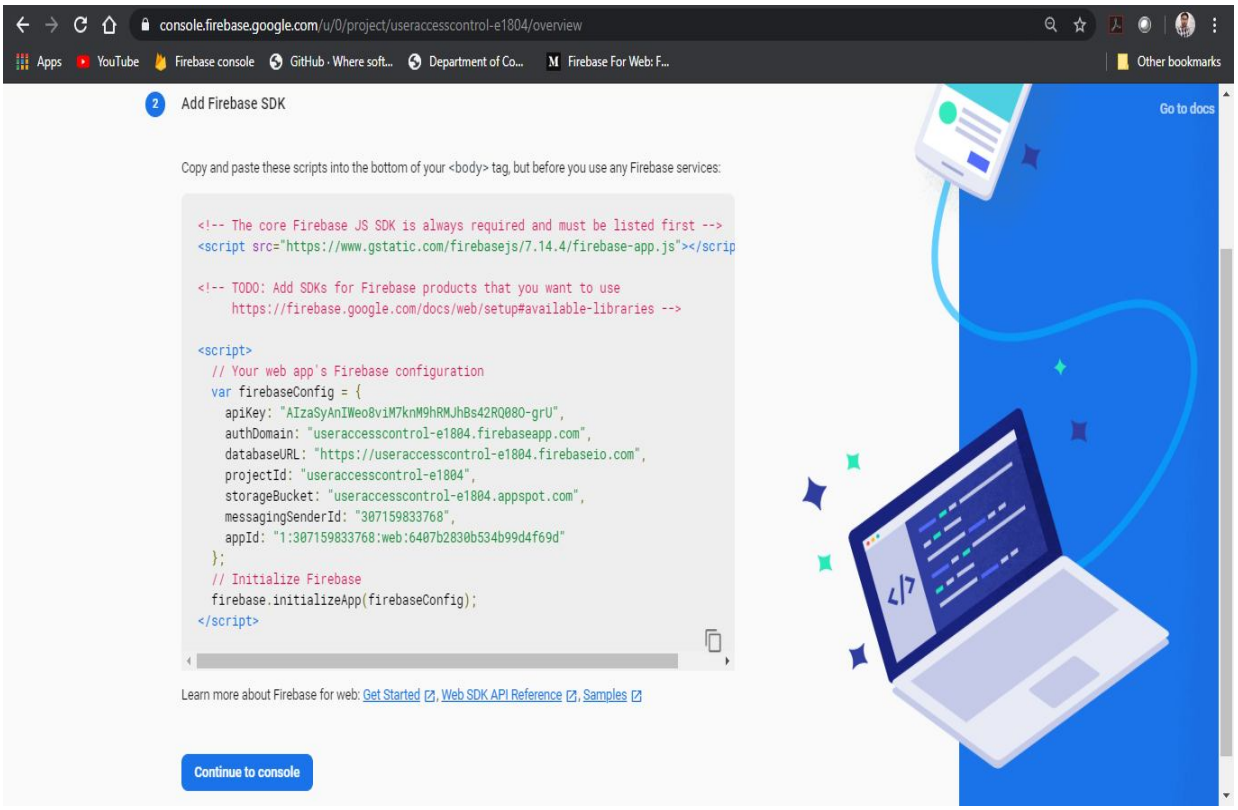


Fig. 43 Adding Firebase SDK

7) Having pasted this script in our web application source code, the firebase is integrated into our application and is ready for use.
8) To provide login in the firebase project through emails, the sign-in through email property in enabled in the firebase project
9) Here other methods of signing in are also available. One can choose any way among the given ways to add to the project as per the requirements and convenience.

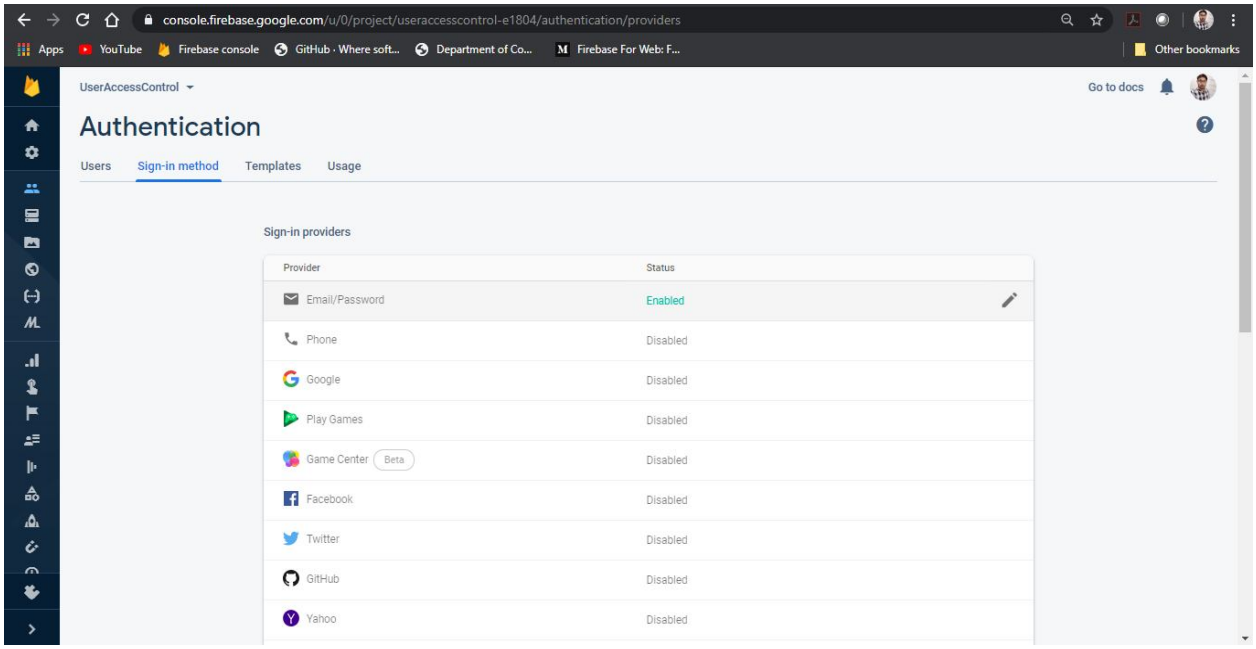


Fig. 44 Various way to signing to Firebase account

10) All the users added to this project through the web application will be shown

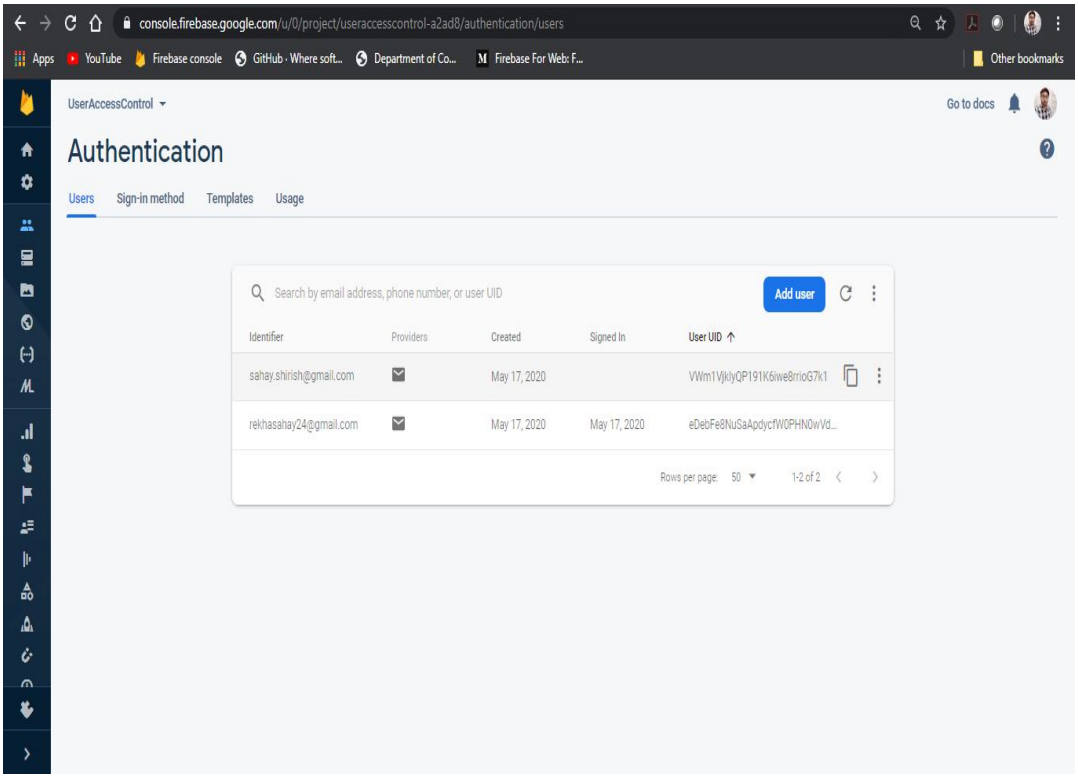


Fig. 45 List of users added to Firebase account through the web application

11) It required performing the setup of the storage in the project where the files will be uploaded to and downloaded from.

12) The storage section in the project is accessed for configuration. It shows a general rule depicting the default security provided by firebase. Any required number of security instructions can be added here like who can access storage, restrictions on the size of files etc.

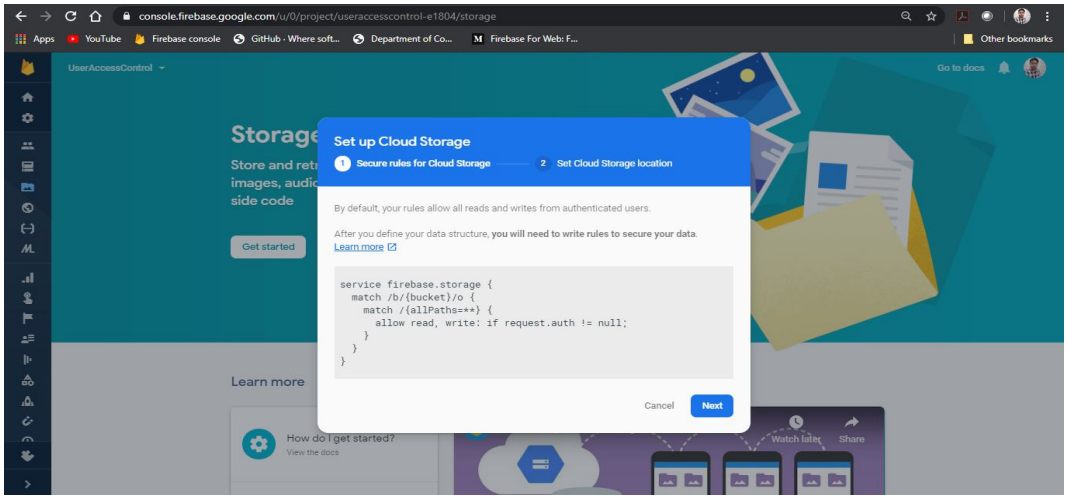


Fig. 46 Set up of Cloud storage

13) After clicking on next, it asks to choose the cloud storage location and then creates a storage bucket. This storage bucket is like a memory location provided by the firebase where the uploaded documents will be stored.

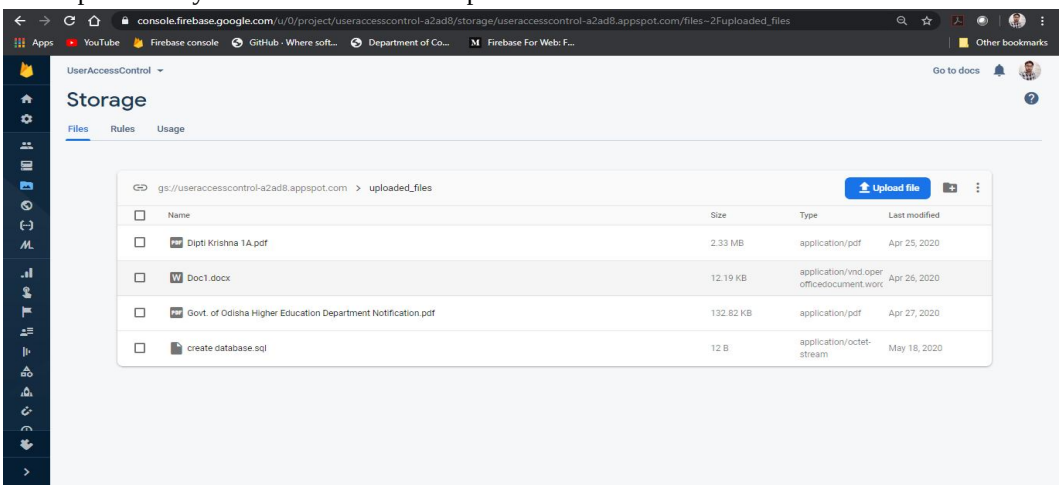


Fig. 44 Web Page of our implemented Cloud storage

V. ATTRIBUTE BASED ENCRYPTION TECHNIQUE IN AWS

After creating a website in AWS Cloud to keep secure the data stored on the website. We apply a hidden policy Cipher-text Policy Attribute-Based Encryption scheme over data have shared in our deployed website for efficient access control over the data stored in our deployed website in the AWS cloud.

Our proposed idea implemented attribute based encryption technique by following 3 algorithms.

A. Set Up

Security parameter is taken by AWS that generates Public Key (PK) along with Master key (MK).

1) *Input:* ƛ as security parameter and Ą as attribute set

2) *Output:* Public key (PK) along with Master key (MK)

a) G_1 is generated that is a bilinear group of p as prime order and g as generator

b) Choose β *and* k randomly

c) Compute $y = e(g,g)^{\beta}$
d) For every attribute $a_i \in A_t$
e) Choose secret share $s \in z_p$ randomly
f) Choose h_{t1} *and* h_{t2} randomly
g) Compute $T_i = g^{h_{i1}h_{i2}(h_{i1}+h_{i2})}$

Return $PK = (G_1, g, y, \forall a_i \in A_t : (T_i, s), k)$, $MK = (\beta, k, \forall a_i \in A_t : (h_{i1}, h_{i2}))$

B. Encryption

The Data uploaded in the website hosted in the AWS cloud encrypts under τ access policy and public key generated through AWS EC2 instance. After encryption generates cipher-text of the data/file and encrypted masked access policy that is embedded along with cipher-text. The generated cipher-text stored in the AWS cloud.

1) *Input:* Public key $PK = (G_1, g, y, \forall a_i \in A_t : (T_i, s), k)$, Master key $MK = \beta, k, \forall a_i \in A_t : (h_{i1}, h_{i2})$created by AWS, data F and access policy τ
2) *Output:* Cipher-text (CT) of the data/file under τ
a) Choose secrete ω *and* $r \in z_p$ randomly
b) Calculate $C_1 = g^{\omega}$ and $C_2 = Fe(g,g)^{\beta\omega}$
c) In τ from top to down manner for each node a
d) If a root node and $a_i = att(a)$, choose p_a randomly from z_p and form $t-1$ polynomial degree, let $p_a = w$ then shift value is calculated $v_i = p_a - s_i$ and a secret share y is assigned to each child node i.e $s_y = v_i(index(y))$
e) Else if a is non-leaf choose p_a randomly and form $t-1$ polynomial degree, $p_a = w = p_{parent\,(a)}(index(a))$ and assign $s_y = v_i(index(y))$ *to* each child node
f) If set of leaf nodes is S
g) In τ for a leaf node $a \epsilon S$, $a_i = att(a)$
h) Compute $C_i = T_i^{(v_i+s)} = g^{(v_i+s)h_{i1}h_{i2}(h_{i1}+h_{i2})}$
i) If $\tau = N$, $[N =$ real number
j) Compute $m = \tau * k$
k) Else $m = ASCII(\tau) * k$
l) Compute $D_i = g^{r/h_{i1}}$ and $D'_i = g^{r/h_{i2}}$

Return Cipher-text $CT = (C_1, C_2, \forall a \in S, C_i, D_i, D'_i, m)$ of data/file

C. Decryption

1) *Input:* Cipher-text $CT = (C_1, C_2, \forall a \in S, c_i, D_i, D'_i, m)$ of data/file and Master key created in AWS
2) *Output:* Data/File F in plain text format
a) If $\tau = N$, $[N =$ real number
b) Compute $\tau = m/k$
c) Else ASCII value of $\tau = m/k$
d) For every leaf node y in τ , where $i = att(y), i \in S$
e) Compute $D_y = e(C_i, D_i D'_i)$

$$= e(g^{(v_i+s)t_{i1}t_{i2}(t_{i1}+t_{i2})}, g^{r/t_{i1}}, g^{r/t_{i1}})$$
$$= e((g,g)^{r(v_i+s)})$$
$$= e(g,g)^{rp_a}$$
$$= e(g,g)^{rw}$$

f) Else $D_y = \Gamma$

g) For every non-leaf node y in τ

h) k'_y Is denoted for S_y that is size of child nodes of a set such that $D_y \neq \Gamma$, If no size child node of a set exist then the node will not satisfy by attribute set S return Γ

i) Else Compute $D_y = \Pi_{z \in S_y} F^{\Delta_{l,S'_y}}$

$$= \Pi_{z \in S_y} e((g,g)^{rp_a})^{\Delta_{l,S'_y}}$$

$$= \Pi_{z \in S_y} e((g,g)^{rp_{parent(z)}})^{(index(z))^{\Delta_{l,S'_y}}}$$

$$= \Pi_{z \in S_y} e((g,g)^{rw\Delta_{l,S'_y}}$$

$$= e(g,g)^{w}$$

Where Δ_{l,S'_y} is called as Lagrange Coefficient, $l = (index(z)), S'_y = \forall_z \in S_y$ that is $index(z)$

VI. EXPERIMENTAL RESULTS

We have experimented on Windows 10, AMD RYZEN 5(RADEON Graphics). To implement our proposed algorithm, we use Python 3.8 and PyCharm 2020.3.2 x64 and configured AWSTool kit for PyCharm plug in it. After testing the proposed algorithm we use AWS SDK to implement it through Amazon S3 Client side encryption. This technique encrypts our data and its access policy before storing it in our deployed website [discussed in section 4] in AWS. Amazon S3 encryption client randomly generates separate data encryption key that encrypt data of each object of Amazon S3. Through data encryption keys the data and its access policy encrypts and uploads in Amazon S3. The data encryption key is encrypted through master key and save the encrypted data encryption key as metadata of object in Amazon S3. The client used client side master key to decrypt the saved encrypted data encryption key. The client downloads data from Amazon S3 and by using master key it decrypts data encryption key then uses the data encryption key that decrypt the data along with its corresponding access policy. Therefore the data stored in our deployed web application is secure, controlled by the user and according to the access policy defined for corresponding data only authorized users can able to access those data.

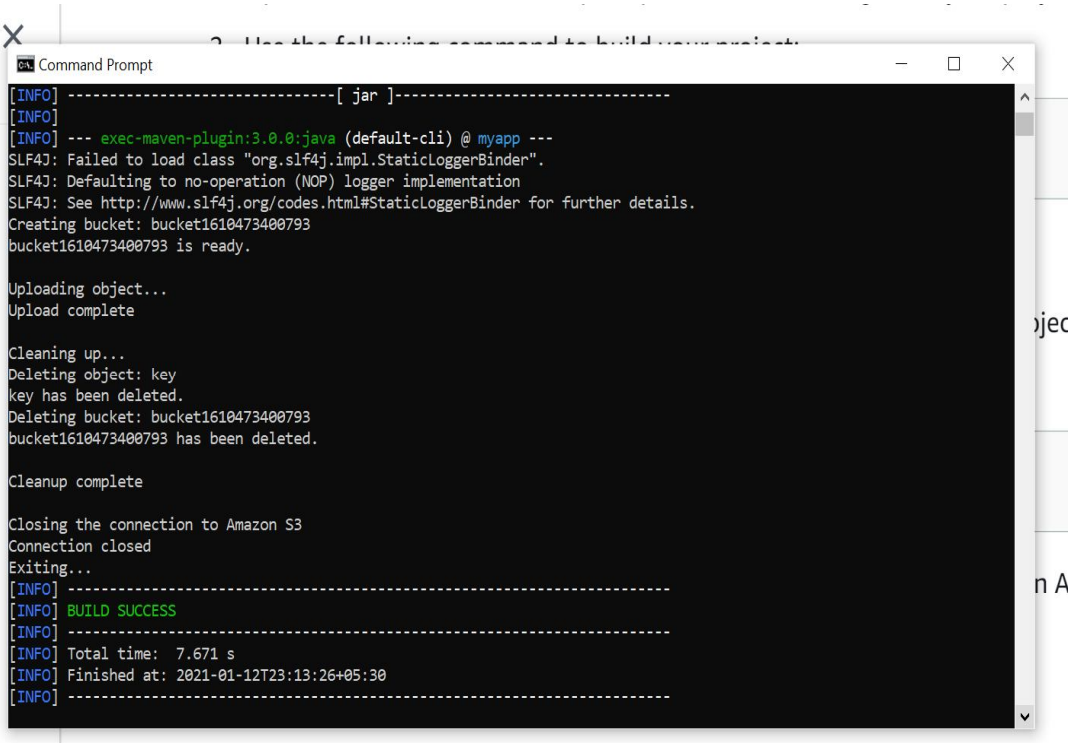


Fig. 45 After data uploaded in AWS through AWS SDK

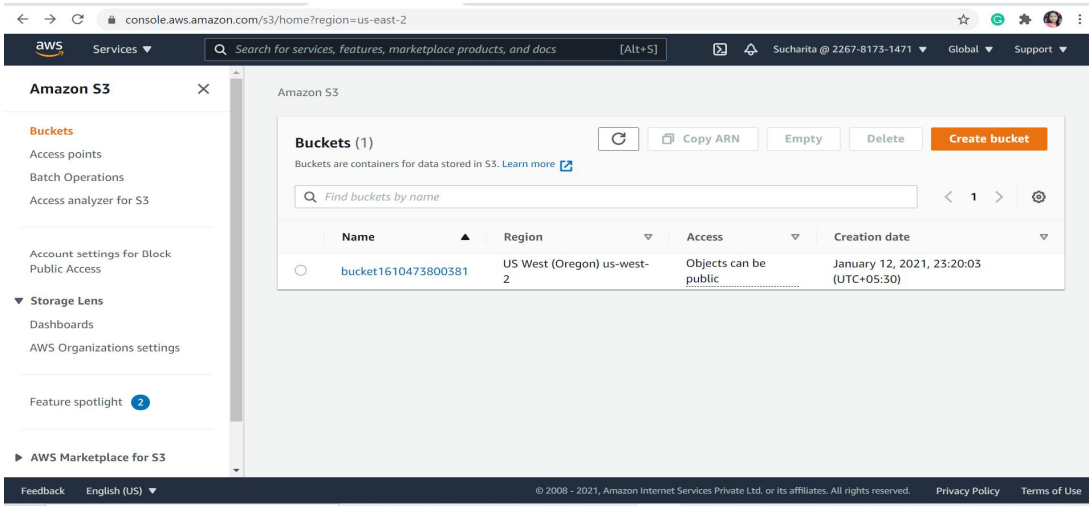


Fig.46 Uploaded Data successfully stored in AWS Server

Data stored in Amazon S3 can be copied to Amazon EC2 instance through AWS Command Line Interface by following the below command ec2-user ~$ aws s3 cp s3://my_bucket/my_folder/my_file.ext my_copied_file.ext

VII. RESULT ANALYSIS

Applying access control rules on the users in the storage, this application ensures secure and authorized access to the firebase cloud storage. The hosting of application on the Amazon EC2 instance provides secure remote hosting of the application and it can be accessed globally. Unauthorized users do not have access to cloud storage.

Various methods to add attribute-based encryption over the files/documents shared over the cloud platform. All the encryption properties were attached to the documents. Taking this idea we decided to implement user-level access control in our paper because that is also a suitable way to achieve secured access. Under that implementation, we tried to create controlled access among registered users. Our application achieves this to a great extent. All users do not have read and write permission. The application only allows authorized users to access the application. The authorized users are those who have '*@cvrgu.com' extension in their email id. Otherwise, the users cannot register into the application. This application is meant to be used within an organization; therefore there is no threat of any non-organizational member trying to access the application. The storage is kept private so unnecessary users cannot get into the storage. By applying Attribute based Encryption scheme to our application, we made secure to the data stored in our application that deployed in AWS Cloud. Therefore, only authorized users can able to access the data stored in AWS server.

Amazon EC2 instance provides a secure server for hosting the web application. Using EC2 instance ensures that the application can be accessed from anywhere globally. That means that a user with the EC2 generated website URL can access this application from anywhere. Firebase platform's storage is well secured with access control rules. The entire application performs on the cloud and no data is being kept in any secondary storage. The application is thus made secure for its users.

VIII. CONCLUSION

In this paper, we have showed step by step process of creating a web application that facilitates the uploading of documents in the cloud platform. The cloud storage is kept private and access to stored data is controlled by using a user-level access control method. This method ensures only authorized users have access to cloud storage. The use of the Firebase platform has helped to achieve the above-mentioned access control method. The website is hosted on the Amazon EC2 instance so one who is authorized to use this application can globally access this application. This website is secured through a user-level access policy. This property ensures the least spamming on the website by unauthorized users. We used Attribute Based Encryption technique to encrypt and decrypt our data stored in our web application in AWS through Amazon S3 Client side encryption. So that, our data security can be controlled and data can be accessible to the only authorized persons. The authorized person is permitted to access the stored data through the access policy. The access policy is defined by the user for their respective data and stored in the cloud server along with encrypted data. This paper helps in understanding the deployment of any website on the cloud platform and how to manage the secure access of data on the cloud platform.

IX. ACKNOWLEDGMENT

None

REFERENCES

[1] S. Khuntia and P.S. Kumar, "New Hidden Policy CP-ABE for Big Data Access Control with Privacy-preserving Policy in Cloud Computing," 2018 Int. Conf. Compute. Commun. Netw. Technol. ICCCNT 2018,pp. 1-7, 2018, doi: 10.1109/ICCCNT.2018.8493698.

[2] P. J.Sun, "Security and privacy protection in cloud computing: Discussions and challenges," J. Netw. Comput. Appl., vol.160, p.102642, 2020, doi: 10.1016/j.jnca.2020.102642.

[3] L.Coles-Kemp, J. Reddington, and P. A. H. Williams, "Looking at clouds from both sides: The advantages and disadvantages of placing personal narratives in the cloud," Inf. Secure. Tech. Rep.,vol. 16, no. 3-4,pp. 115-122, 2011, doi: 10.1016/j.istr.2011.09.001.

[4] H. Qi, X. Di, and J.Li, "Formal definition and analysis of access control model based on role and attribute",J. Inf. Secure. Appl., vol. 43, pp. 53-60, 2018, doi: 10.1016/j.jisa.2018.09.001.

[5] H.He, R. Li, X. Dong, and Z.Zhang, "Secure, efficient and fine-grained data access control mechanism for P2P storage cloud," IEEE Trans. Cloud Comput., vol.2,no. 4, pp.471-484, 2014, doi: 10.1109/TTC.2014.2378788.

[6] P.K. P,S.K.P, and A.P.J.A, "Attribute based encryption in cloud computing: A survey, gap analysis, and future directions", J. Netw. Comput. Appl., vol. 108, pp. 37-52, 2018, doi: 10.1016/j.jnca.2018.02.009.

[7] M. Ali, J.Mohajeri, M. R. Sadeghi, and X. Liu, " A fully distributed hierarchical attribute- based encryption scheme," Theor. Comput. Sci., vol.815,pp. 25-46, 2020, doi: 10.1016/j.tcs.2020.02.030

[8] D.Koo, J.Hur, and H. Yoon, "Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage," Comput. Electr. Eng. , vol. 167, no.2019, pp. 34-46, 2013, doi: 10.1016/j.compeleceng.2012.11.002.

[9] P.S. Challagidad and M. N. Birje, "Efficient Multi-authority Access Control using Attribute based Encryption in Cloud Storage," Procedia Comput Sci., vol. 167, no. 2019, pp. 840-849, 2020, doi: 10.1016/j.procs.2020.03.423

[10] J. Lai, R. H. Deng, Y. Yang, and J. Weng,"Adaptable ciphertext-policy attribute-based encryption," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Bioinformatics), vol. 8365 LNCS, pp. 199-214, 2014, doi: 10.1007/978-3-319-04873-4_12.

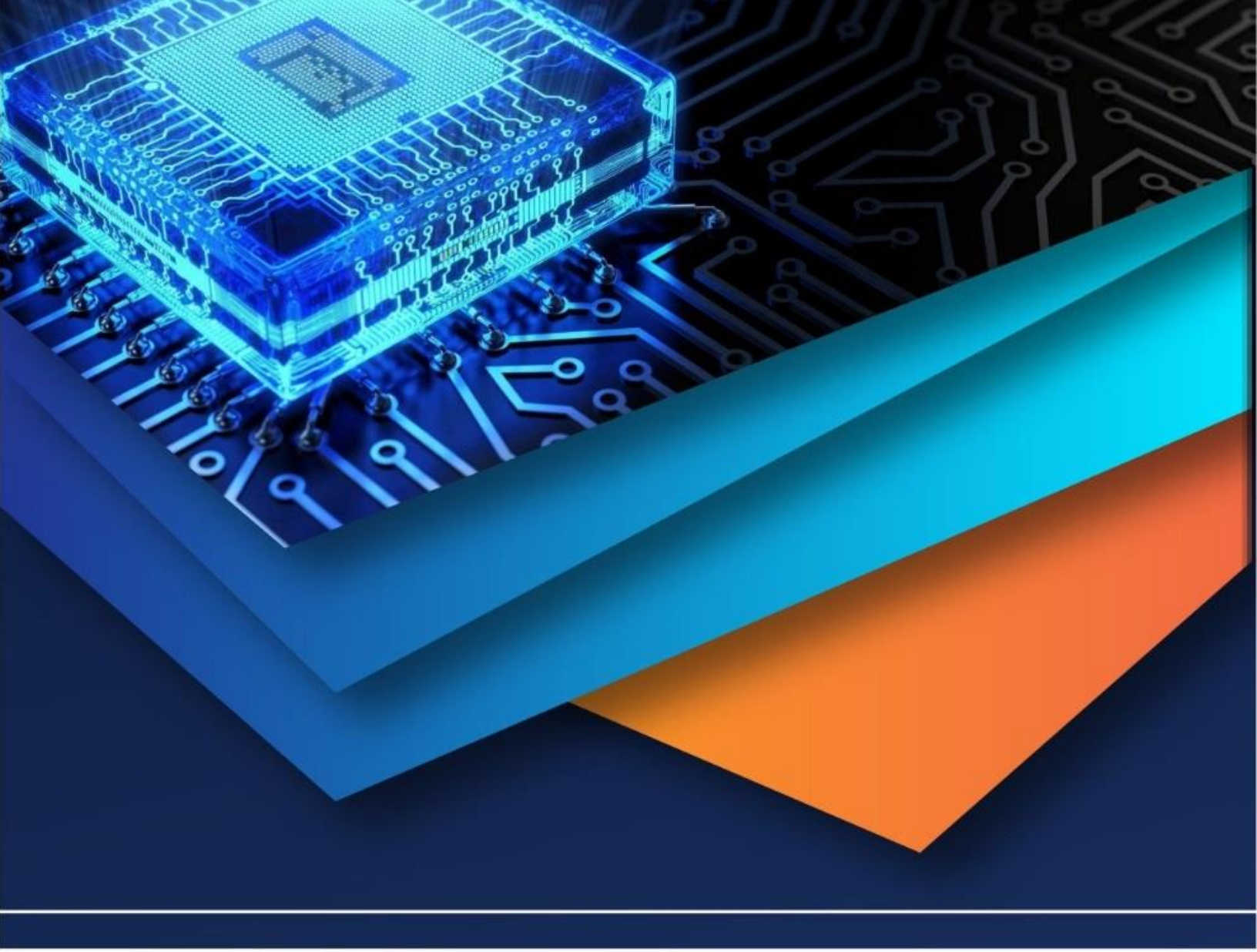
[11] R. Xu, Y.Wang, and B. Lang,"A tree-based CP-ABE scheme with hidden policy supporting secure data sharing in cloud computing," Proc.-2013 Int. Conf. Adv. Cloud Big Data, CBD 2013, pp.51-57, 2013, doi: 10.1109/CBD.2013.9.

[12] J.Liu, X.Huang, and J. K.Liu,"Secure sharing of Personal Health Records in cloud computing: Cipheretext-Policy Attribute-Based Signcryption," Futur. Gener. Comput. Syst., vol. 52, pp. 67-76, 2015, doi: 10.1016/j.future.2014.10.014.

[13] F.Khan, H. Li, L. Zhang, and J. Shen,"An Expressive Hidden Access Policy CP-ABE," Proc. -2017 IEEE 2nd Int. Conf. Data Sci. Cyberspace, DSC 2017, pp. 178-186, 2017, doi: 10.1109/DSC.2017.29.

[14] N.Helil and K.Rahman,"CP-ABE access control scheme for sensitive data set constraint with hidden access policy and constraint policy," Secur. Commun. Networks, vol.2017, 2017, doi: 10.1155/2017/2713595.

[15] V.Odelu, A.K. Das, Y. S. Rao, S.Kumari, M.K. Khan , and K. K. R . Choo,"Pairing-based CP-ABE with constant-size ciphertexts and secret keys for cloud environment," Comput. Stand. Interfaces. vol.54, pp. 3-9,2017,doi: 10.1016/j.csi.2016.05.002.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY