

CONFIGURING WEBSERVER AND ADDING SERVER

AUTHENTICATION USING ANSIBLE

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and
Engineering

By

YALLA RITHWIK REDDY

(Reg. No. 37110865)

YECHURI SANDEEP

(Reg. No. 37110871)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI
SALAI, CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA



INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bona fide work of **YALLA RITHWIK REDDY (Reg. No. 37110865)** and **YECHURI (Reg.No.37110871)** who carried out the project entitled **“CONFIGURING WEBSERVER AND ADDING SERVER AUTHENTICATION USING ANSIBLE”** under my supervision from **August 2020** to **March 2021**.

Internal Guide

Dr. J. Refonna, M.E., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I **YECHURI SANDEEP** here by declare that the Project Report entitled **“CONFIGURING WEBSERVER AND ADDING SERVER AUTHENTICATION USING ANSIBLE”** is done by me under the guidance of **Dr. J. Refonna, M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D.**, and **Dr. L. Lakshmanan, M.E., Ph.D.**, **Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide. **Dr. J. Refonna, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

In this project, our task is to develop an webserver to access in a network it is added with additional security with the help of ansible. It is provided with security to protect with outsiders to access it. It is built in Linux os with the help of ansible. playbooks holds the actions to be performed. And we are able to manage passwords. By this we got a secured webserver in a fast and easy way. In today's world any company in existence needs to have hundreds to servers to provide seamless services to its users, Here the challenge arises where the employees of company has to configure all the servers which is very long process as configuring each server sometimes takes days. It is very inefficient process as there may also be human errors which are very difficult to find. By using Ansible we can easily configure thousands of servers only with one click without any errors as the process automated.

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE No.
	ABSTRACT	V
1	INTRODUCTION	1
	1.1 Architecture System	1
2	LITERATURE SURVEY	5
3	AIM AND SCOPE OF PRESENT INVESTIGATION	6
4	EXPERIMENTAL OR MATERIALS AND METHODS	7
	4.1 REQUIREMENT SPECIFICATION	7
	4.2 SOURCE CODE	9
5	RESULT AND DISCUSSION	12
	5.1 SCREENSHOTS	12
6	SUMMARY AND CONCLUSION	15
	6.1 REFERENCES	15

CHAPTER 1

INTRODUCTION

Imagine that you are a web developer and you have created web pages which needs to be deployed so that public world may access it, But the challenge every developer face is that they do not know how web servers works and how to configure them.

We can use ansible to write a playbook in yaml language with tasks in them such that it can configure webserver dynamically for web developers, web developers just need to write the code for web pages and upload them In github, this github link needs to be placed in playbook, That is it ,anyone can deploy their website code to public world even without knowing how webserver works or how to configure it.

We can also add server authentication so that only verified users can access the website after deploying it.

After configuring any server the first thing to ensure is that the server is secure, Hackers may try to take down our server using DOS(denial of service) attack where he sends huge traffic to our server.

It keeps our server busy and the companies clients couldn't connect to server as the server is busy due to the traffic sent by hacker. In order to avoid DOS attack we can implement server authentication so that only registered users can access the server.It supports any type of devices like mobile phone or laptops.

Architecture System

Architecture diagram given in figure 1.1 gives Ansible is a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs.

Being designed for multi-tier deployments since day one, Ansible models your IT infrastructure by describing how all of your systems inter-relate, rather than just managing one system at a time.

It uses no agents and no additional custom security infrastructure, so it's easy to deploy - and most importantly, it uses a very simple language (YAML, in the form of Ansible Playbooks) that allow you to describe your automation jobs in a way that approaches plain English.

In this section, we'll give you a really quick overview of how Ansible works so you can see how the pieces fit together.

Ansible works by connecting to your nodes and pushing out scripts called "Ansible modules" to them. Most modules accept parameters that describe the desired state of the system. Ansible then executes these modules (over SSH by default), and removes them when finished. Your library of modules can reside on any machine, and there are no servers, daemons, or databases required.

You can write your own modules, though you should first consider whether you should. Typically you'll work with your favorite terminal program, a text editor, and probably a version control system to keep track of changes to your content. You may write specialized modules in any language that can return JSON (Ruby, Python, bash, and so on)

Module Utility

When multiple modules use the same code, Ansible stores those functions as module utilities to minimize duplication and maintenance. For example, the code that parses URLs is `lib/ansible/module_utils/url.py`.

Inventory

By default, Ansible represents the machines it manages in a file (INI, YAML, and so on) that puts all of your managed machines in groups of your own choosing.

To add new machines, there is no additional SSL signing server involved, so there's never any hassle deciding why a particular machine didn't get linked up due to obscure NTP or DNS issues.

If there's another source of truth in your infrastructure, Ansible can also connect to that. Ansible can draw inventory, group, and variable information from sources like EC2, Rackspace, OpenStack, and more.

Here's what a plain text inventory file looks like:

```
---  
[webserver]  
www1.example.com  
www2.example.com  
  
[dbserver]  
db0.example.com  
db1.example.com
```

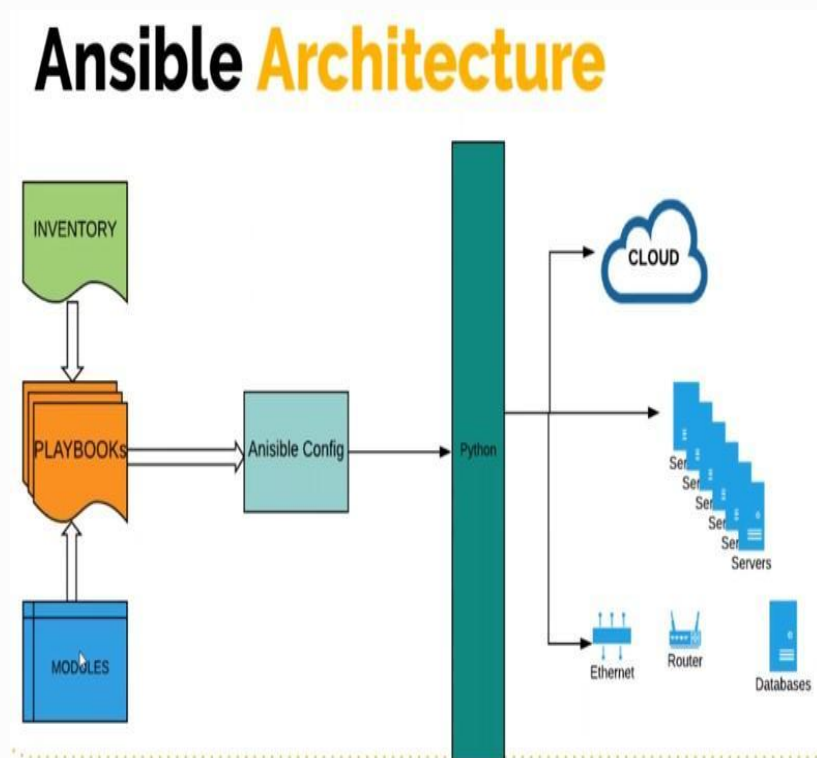
Once inventory hosts are listed, variables can be assigned to them in simple text files (in a subdirectory called 'group_vars/' or 'host_vars/' or directly in the inventory file).

Or, as already mentioned, use a dynamic inventory to pull your inventory from data sources like EC2, Rackspace, or OpenStack.

PLAYBOOKS

Playbooks can finely orchestrate multiple slices of your infrastructure topology, with very detailed control over how many machines to tackle at a time. This is where Ansible starts to get most interesting.

Ansible's approach to orchestration is one of finely-tuned simplicity, as we believe your automation code should make perfect sense to you years down the road and there should be very little to remember about special syntax or features.



CHAPTER 2

LITERATURE SURVEY

Ansible is a tool used by our head DevOps engineer and others who elect to do so. It is mainly used for automating server setup/tear down and ensuring concurrency exists across all our application platforms. It's an incredible tool for setting up any environment without having to install the program on the server you wish to target.

Ansible allows us to manage our systems with over SSH with an easy to read YAML format. It is easy to check playbooks into source control to share with others in the organization. Ansible playbooks are easy and quick to develop.

CHAPTER 3

AIM AND SCOPE OF PRESENT INVESTIGATION

The project aim is to create a web server and host the server with a website. This is added with additional authentication. First we need to create a website and write an ansible playbook so we can launch remotely and dynamically any where in linux OS the password are saved in a password file in hash algorithm form where the passwords are encrypted. So anyone cannot understand passwords that we assigned. After this we need to host server using ansible where the required connections established using playbook. We need to give the address of website which we need to host. Every company needs to have multiple servers in order to provide the customers uninterrupted web servers, managing all these servers may be difficult to manage, by using ansible we can manage all of them with just one click.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS

REQUIREMENT SPECIFICATION

The hardware and software requirements for this project is as below

HARDWARE REQUIREMENTS

System- any system which supports virtualization(because we are going to perform this project on virtual machine)

Hard disk- 40GB

Ram -1GB

SOFTWARE REQUIREMENTS

RHEL8 iso image

Any virtualization tool (I have used virtual box)

Ansible software

Python version 3

AUTOMATION

Automation is the creation and application of technologies to produce and deliver goods and services with minimal human intervention. The implementation of automation technologies, techniques and processes improve the efficiency, reliability, and/or speed of many tasks that were previously performed by humans.

Automation is being used in a number of areas such as manufacturing, transport, utilities, defense, facilities, operations and lately, information technology.

ANSIBLE

Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling infrastructure as code. It runs on many Unix-like systems, and can configure both Unix-like systems as well as Microsoft Windows. It includes its own declarative language to describe system configuration. Ansible was written by Michael DeHaan and acquired by Red Hat in 2015. Ansible is agentless, temporarily connecting remotely via SSH or Windows Remote Management (allowing remote PowerShell execution) to do its tasks.

Ansible is a Python-dependent configuration-management software, where both the controlling node and the target machine must have Python and its dependent packages installed. Ansible does not require a single controlling machine where orchestration begins. Ansible works against multiple systems in your infrastructure by selecting portions of Ansible's inventory, stored as edit-able, version-able ASCII text files. Not only is this inventory configurable, but you can also use multiple inventory files at the same time and pull inventory from dynamic or cloud sources or different formats (YAML, INI, etc.). Any machine with Ansible utilities installed can leverage a set of files/directories to orchestrate other nodes. The absence of a central-server requirement greatly simplifies disaster-recovery planning. Nodes are managed by this controlling machine – typically over SSH. The controlling machine describes the location of nodes through its inventory. Sensitive data can be stored in encrypted files using Ansible Vault since 2014. In contrast with other popular configuration-management software — such as Chef, Puppet, and CFEngine — Ansible uses an agentless architecture, with Ansible software not normally running or even installed on the controlled node. Instead, Ansible orchestrates a node by installing and running modules on the node temporarily via SSH. For the duration of an orchestration task, a process running the module communicates with the controlling machine with a JSON-based protocol via its standard input and output.

When Ansible is not managing a node, it does not consume resources on the node because no daemons are executing or software installed.

ANSIBLE PLAYBOOK

An Ansible playbook is a blueprint of automation tasks—which are complex IT actions executed with limited or no human involvement. Ansible playbooks are executed on a set, group, or classification of hosts, which together make up an Ansible inventory.

Ansible playbooks are essentially frameworks, which are prewritten code developers can use ad-hoc or as starting template. Ansible playbooks are regularly used to automate IT infrastructure (such as operating systems and Kubernetes platforms), networks, security systems, and developer personas (such as Git and Red Hat CodeReady Studio).

Ansible playbooks help IT staff program applications, services, server nodes, or other devices without the manual overhead of creating everything from scratch. And Ansible playbooks—as well as the conditions, variables, and tasks within them—can be saved, shared, or reused indefinitely.

ANSIBLE INSTALLATION

As ansible is built using python we need to have python I our controller node and after that we can install ansible with just one command “ pip3 install ansible ”

IMPLEMENTING ANSIBLE PLAYBOOK

After writing the code,specifyingip of managed node in inventory file and updating user database,we can execute ansible playbook by using “ansible-playbook code.yml”

COMPANIES USING ANSIBLE

Following is non exhaustive list of companies using ansible

Trivago

Razorpay

Walmart

Source code

```
- name: "webserver"
  hosts: final
  tasks:
    - name: "creating firewall rule for port 80"
  firewall:
    port: 80/tcp
    state: enabled
  - name: "creating directory for mounting cdrom"
  file:
    path: "/data"
    state: directory
    register: folder
  - name: "mounting cdrom to directory"
  mount:
    path: "/data"
  src: "/dev/sr0"
  fstype: iso9660
    opts: ro,noauto
    state: mounted
  when: folder.failed==false
```

```
    register: mount
  - name: "yum repository for appstream"
yum_repository:
  name: "appstream"
  description: "contains appstreamsoftwares"
baseurl: "file:///data/AppStream"
  enabled: yes
gpgcheck: no
  when: mount.failed==false
  register: appstream
  - name: "yum repository for baseos"
yum_repository:
  name: "baseos"
  description: "contains baseossoftwares"
baseurl: "file:///data/BaseOS"
  enabled: yes
gpgcheck: no
  when: mount.failed==false
  register: baseos
  - name: "install http software"
  package:
    name: "httpd"
    state: present
  when: [appstream.failed==false,baseos.failed==false]
  register: http
  - name: "copy web pages to webserver root directory"
get_url:
  url:
    "https://raw.githubusercontent.com/rithwik12345/HTML_TEST_CODE/master/index.html"
  dest: "/var/www/html"
  when: http.failed==false
```

```
    register: webpage
  - name: "enabling security in config file"
    replace:
      path: "/etc/httpd/conf/httpd.conf"
  regexp: "AllowOverride None"
    replace: "AllowOverrideAuthConfig"
    register: security
  - name: "copying details of server security file"
    copy:
      dest: "/var/www/html/.htaccess"
      src: ".htaccess"
      when: security.failed==false
    register: htaccess
  - name: "copying user credentials to server"
    copy:
      dest: "/etc/www.passwd"
      src: "/etc/www.passwd"
      when: htaccess.failed==false
  - name: "restart httpd service "
    service: name: "httpd" state: restarted
  when:[webpage.failed==false,htaccess.failed==false]
```

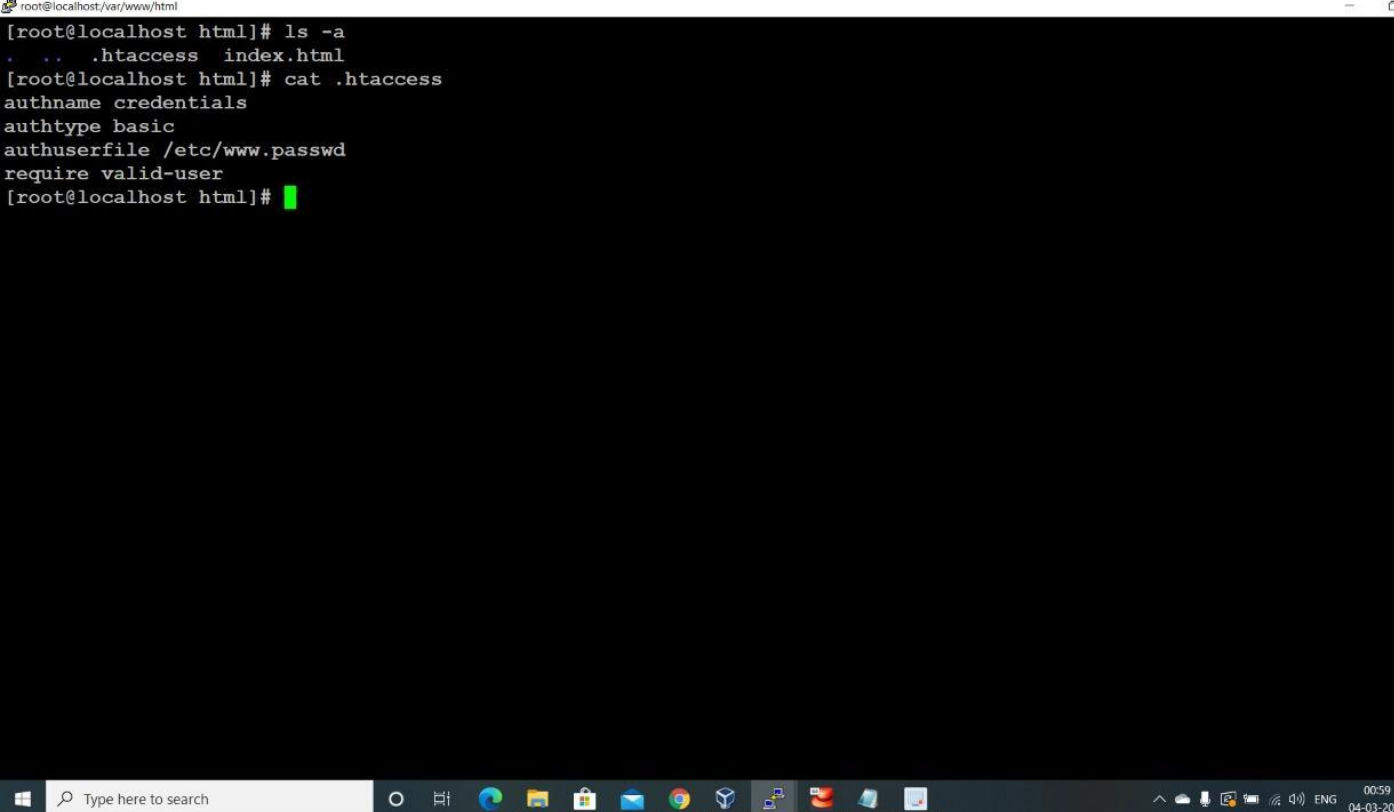
CHAPTER 5

RESULT AND DISCUSSION

Thus we have successfully executed ansible playbook to configure webserver with server authentication

As ansible is declarative language we only tell it what to do , ansible it self finds a way to perform the task. By using ansible we can eliminate errors caused by humans because whole process is automated and we can use a single playbook indefinitely so writing a playbook is a one time task.

SCREENSHOTS



```
root@localhost:/var/www/html
[root@localhost html]# ls -a
.  ..  .htaccess  index.html
[root@localhost html]# cat .htaccess
authname credentials
authtype basic
authuserfile /etc/www.passwd
require valid-user
[root@localhost html]#
```

root@localhost:~#

```
<head>
<title>WEBSERVER</title>

</head>
<body style="color:white;background-color: black">
  <div><marquee><mark><h1 style="color:white">WELCOME TO MY JOURNAL</h1></mark></marquee></div>
  <div><center><h1><i><u>PONDICHERRY</u></i></h1></center>
</div>

  <div style="padding-top: 30px;padding-bottom: 100px">Pondicherry (or Puducherry), a French colonial settle
in India until 1954, is now a Union Territory town bounded by the southeastern Tamil Nadu state. Its French legac
preserved in its French Quarter, with tree-lined streets, mustard-colored colonial villas and chic boutiques. A s
de promenade runs along the Bay of Bengal and passes several statues, including a 4m-high Gandhi Memorial.

  <br>The earliest recorded history of the municipality of Puducherry can be traced to the second centur
. The Periplus of the Erythraean Sea mentions a marketplace named Poduke (ch 60). G. W. B. Huntingford suggested t
might be a site about 2 miles from the modern Puducherry, which was possibly the location of Arikamedu (now part o
iyankuppam). Huntingford noted that Roman pottery was found at Arikamedu in 1937. In addition, archaeological exca
ons between 1944 and 1949 showed that it was "a trading station to which goods of Roman manufacture were imported
ng the first half of the 1st century" Subsequent investigation by Vimala Begley from 1989 to 1992 modified this as
ment, and now place the period of occupation from the third or second century BC to the eighth century AD.

In 1674, the municipality of Pondicherry (Pondichéry) became a French colony of the French colonial empire. Togeth
ith Chandernagor (already French since 1673), Mahé (since 1721), Yanam (Yanaon) (since 1731), Karaikal (Karikal) (
e 1739) and Masulipatam (1760), it formed the French colony of French India, under a single French governor in Pon
erry, although French rule over one or more of these enclaves was repeatedly interrupted by British occupations. T
erritories of French India were completely transferred to the Republic of India de facto on 1 November 1954, and d
re on 16 August 1962, when French India ceased to exist, becoming the present Indian constituent union territory o
ndicherry, combining four coastal enclaves (with the exception of Chandernagor, which merged with the state of Wes
ngal in 1954).
  </div>
</body>
</html>
[root@localhost html]#
```

root@localhost:~#

```
[root@localhost ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: active (running) since Thu 2021-03-04 00:55:39 IST; 3min 16s ago
     Docs: man:httpd.service(8)
  Main PID: 4980 (httpd)
    Status: "Total requests: 3; Idle/Busy workers 100/0;Requests/sec: 0.0159; Bytes served/sec: 28 B/sec"
    Tasks: 213 (limit: 9084)
   Memory: 25.1M
    CGroup: /system.slice/httpd.service
            └─4980 /usr/sbin/httpd -DFOREGROUND
              └─4990 /usr/sbin/httpd -DFOREGROUND
                └─4991 /usr/sbin/httpd -DFOREGROUND
                  └─4992 /usr/sbin/httpd -DFOREGROUND
                    └─4993 /usr/sbin/httpd -DFOREGROUND

Mar 04 00:55:39 localhost.localdomain systemd[1]: Starting The Apache HTTP Server...
Mar 04 00:55:39 localhost.localdomain httpd[4980]: AH00558: httpd: Could not reliably determine the server's fully
Mar 04 00:55:39 localhost.localdomain httpd[4980]: Server configured, listening on: port 80
Mar 04 00:55:39 localhost.localdomain systemd[1]: Started The Apache HTTP Server.
lines 1-19/19 (END)
```

root@rithwik:~/finalproject

```
TASK [yum repository for baseos] *****
*****
changed: [192.168.1.104]
```

```
TASK [install http software] *****
*****
changed: [192.168.1.104]
```

```
TASK [copy web pages to webserver root directory] *****
*****
changed: [192.168.1.104]
```

```
TASK [enabling security in config file] *****
*****
changed: [192.168.1.104]
```

```
TASK [copying details of server security file] *****
*****
changed: [192.168.1.104]
```

```
TASK [copying user credentials to server] *****
*****
changed: [192.168.1.104]
```

```
TASK [restart httpd service] *****
*****
changed: [192.168.1.104]
```

```
PLAY RECAP *****
*****
192.168.1.104 : ok=12 changed=11 unreacha
ble=0 failed=0 skipped=0 rescued=0 ignored=0
```

root@rithwik:~/finalproject

```
[root@rithwik finalproject]# ansible-playbook code.yml
```

```
PLAY [webserver] *****
*****
```

```
TASK [Gathering Facts] *****
*****
ok: [192.168.1.104]
```

```
TASK [creating firewall rule for port 80] *****
*****
changed: [192.168.1.104]
```

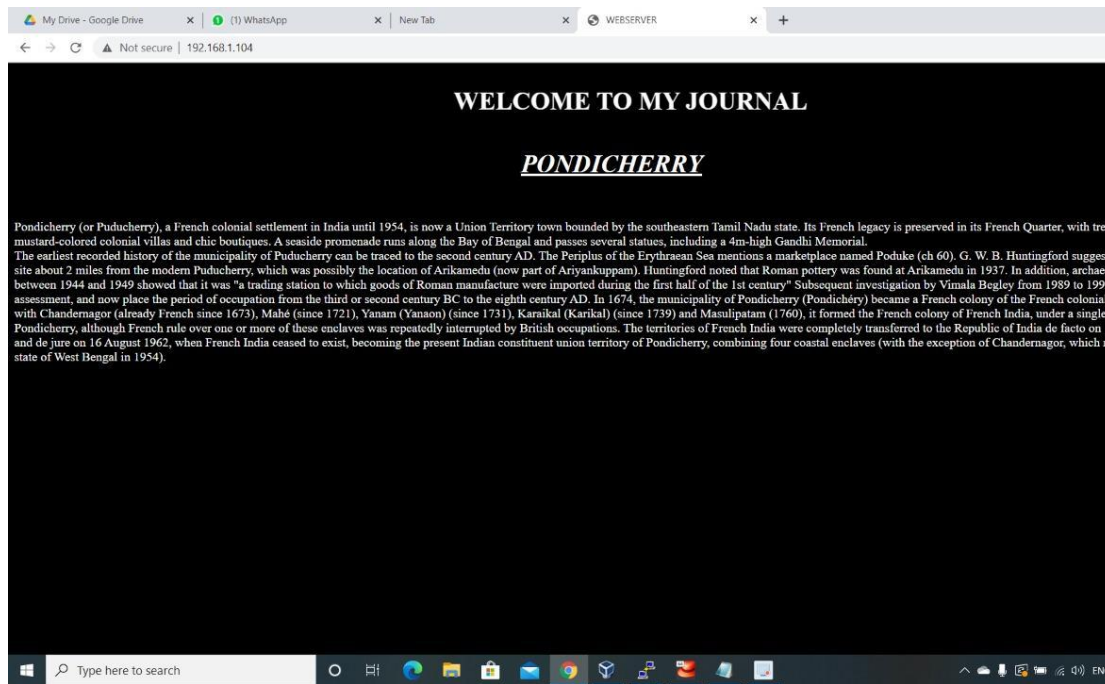
```
TASK [creating direcorry for mounting cdrom] *****
*****
changed: [192.168.1.104]
```

```
TASK [mounting cdrom to directory] *****
*****
changed: [192.168.1.104]
```

```
TASK [yum repository for appstream] *****
*****
changed: [192.168.1.104]
```

```
TASK [yum repository for baseos] *****
*****
changed: [192.168.1.104]
```

```
TASK [install http software] *****
*****
changed: [192.168.1.104]
```



CHAPTER 6

SUMMARY AND CONCLUSION

Implementing this project involved many steps such as writing playbook in yaml language, creating user database, configuring ansible in controller node, managing inventory file etc. The approach used in this project is a top down one concentrating on what first, then how and moving to successive levels of details.

By using ansible we have successfully configured another server as webserver with server authentication

References

[1] L. Hochstein and R. Moser, Ansible: Up and Running: Automating Configuration Management and Deployment the Easy Way. " O'Reilly Media, Inc.", 2017.

[2] S. Thakur, S. C. Gupta, N. Singh, and S. Geddam, "Mitigating and patching system vulnerabilities using ansible: A comparative study of various configuration management tools for iaas cloud," in Information Systems Design and Intelligent Applications, pp. 21–29, Springer, 2016.