IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CONFIGURATION MANAGEMENT USING ANSIBLE

[1]Upmanyu Sharma, [2]Bhavik Sharma, [3]Simmi Dutta, [4]Akhil Sharma

[1,2]Student, [3]Head Department, [4]Assistant Professor

[1,2,3,4]Computer Engineering

[1,2,3,4] Government College of Engineering and Technology, Jammu, India

Abstract: Configuration Management in Industry was a purely manual task that is to be done by System Administrator. Automation helps in replacing repetitive tasks and helps in saving time, money, and increasing productivity. But the Industry is now changing a lot with the popularity of DevOps, Cloud Computing, and new Automation Tools. DevOps is in demand nowadays as it shortens the life cycle of Software Development. With today's demand for automation, consistency, and the move towards cloud and DevOps, companies from different sectors are adopting easy-to-use tools that help them to achieve their goal by reducing complexities. Therefore, it is of the utmost importance for a company to have its services installed, configured, and running as quickly as possible and as consistent as possible to help reduce costs. This paper focuses on Configuration Management of Apache Web Server, Kubernetes Cluster, Docker, Hadoop Cluster on the AWS Cloud Using Ansible.

Keywords:- Configuration Management, Automation, Cloud Computing, Devops, Ansible

I. INTRODUCTION

A. Devops

In Today's world the organizations want to shift from manual work to automation to decrease the cost of releasing software. To compete in the market, organizations want to release their software earlier and with better frequency.

DevOps is a set of practices that combines software development (Dev) and IT operations (Ops). Main Goal of DevOps is to reduce the time between development and operation of software without affecting the quality. IBM has coined the term Collaborative DevOps as "designing processes for coordinating software development teams with IT operations teams".

There are various examples of organization practicing Devops including Flickr, Netflix and Etsy. Allspawn and Hammond.

By adopting DevOps and variants thereof, many organizations have improved the software delivery which increases the productivity. It consists of various stages such as continuous development, continuous integration, continuous testing, continuous deployment, and continuous monitoring.

Teams that adopt DevOps culture, practices, and tools become high-performing, building better products faster for greater customer satisfaction.

Cloud Computing often referred to as "the cloud", in simple terms means storing or accessing your data and programs over the internet rather than your computer's hard drive.

The services it offers has been divided into three different models:

- SaaS (Software as a service) - It provides you with end to end product that is run and completely managed by the service provider.
- PaaS (Platform as a Service) - It removes the need for you to manage underlying infrastructure (usually hardware and operating systems), and allows you to focus on the deployment and management of your applications.
- IaaS (Infrastructure as a Service) - IaaS contains the basic building blocks for cloud IT. It typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space.

Types of cloud computing

There are three different ways to deploy cloud service:

- Public Cloud - Owned and Operated by third-party cloud service providers, which deliver their computing resources like servers and storage over the internet.
- Private Cloud – Used exclusively by a single business or organization.
- Hybrid Cloud – Combines public and private cloud, bound together by technology that allows data and applications to be share between them.

B. 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. Automation tools are software applications that help users to test various desktop, web, and mobile applications. These tools provide automation solutions in order to automate the testing process.

I. Automation Tool: Ansible

Ansible is an open source IT Configuration Management, Deployment & Orchestration tool. Main aim of Ansible is to provide large productivity gains to a wide variety of automation challenges.

It is a helpful tool that allows you to create groups of machines, describe how these machines should be configured or what actions should be taken on them. It issues all commands from a central location to perform these tasks.

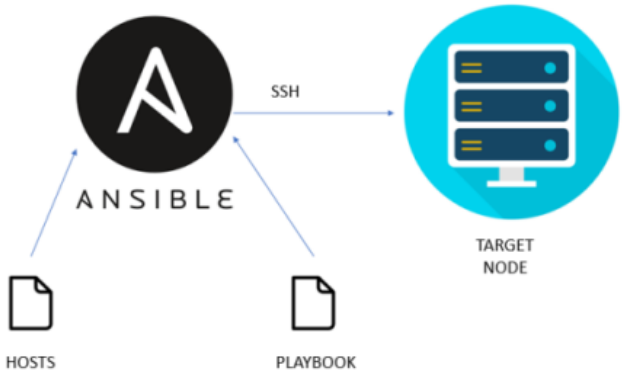


Fig. 1: Architecture of Ansible

Ansible works by connecting to nodes and pushing out small programs called as ansible modules. It then executes these modules over SSH by default and then remove them when finished. Ansible management node is the controlling node, which controls the entire execution of the Playbook. It's the node from which you are running the installation, and the inventory file provides the list of the host where the modules need to be run. The management node makes ssh connection, and then it executes the modules on the host machines and installs the product. It removes the modules once they are installed. So that's how ansible works.

Advantages

- Simple
- Agentless
- Powerful & Flexible
- Efficient

Disadvantages

- Complex to set up the entire stuff because it requires good understanding of Ruby.
- Huge amount of documentation.
- Requires an agent to be installed and pulled configuration

The ansible package provides a base configuration file located at /etc/ansible/ansible.cfg. This file is used if no other configuration file is found.

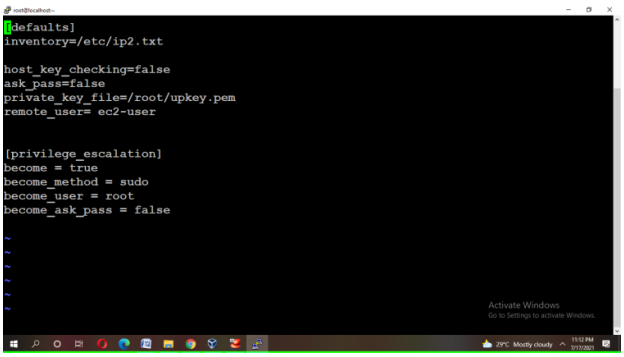


Fig. 2: Ansible Configuration file

II. Implementation

A. Hadoop

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.

A.1 HDFS:

The Hadoop Distributed File System (HDFS) is the primary data storage system used by Hadoop applications. HDFS employs a NameNode and DataNode architecture to implement a distributed file system that provides high-performance access to data across highly scalable Hadoop clusters.
The main components of **HDFS** are the **NameNode** and the **DataNode**.

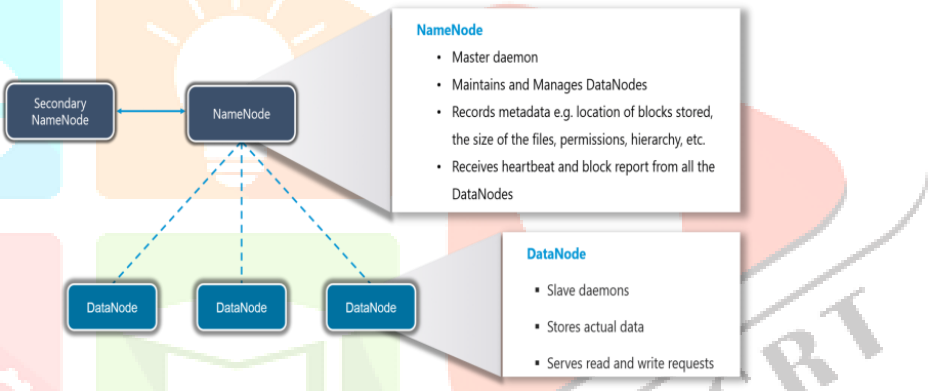


Fig. 3: Hadoop – HDFS

NameNode
- It maintains and manages the DataNodes (slave nodes).
- It is also known as Master.
- It stores the **metadata** of all the blocks stored in the cluster.
- It does not store the actual data.
- It knows the list of the blocks and its location for any given file in HDFS.
- It is a single point of failure in Hadoop cluster.

DataNode
- It is responsible for storing the actual data in HDFS.
- It is also known as the Slave.
- NameNode and DataNode are in constant communication.
- DataNode is usually configured with a lot of hard disk space.

A.2. Implementation of HDFS Cluster:

1. Master Node (Name Node)

```
- hosts: master
  tasks:
  - name: Copy jdk package
    copy:
       src: /root/jdk-8u171-linux-x64.rpm
       dest: /root/jdk-8u171-linux-x64.rpm

  - name: Copy hadoop package
    copy:
       src: /root/hadoop-1.2.1-1.x86_64.rpm
       dest: /root/hadoop-1.2.1-1.x86_64.rpm

  - name: Installing jdk package
    command: rpm -iv /root/jdk-8u171-linux-x64.rpm --force
    ignore_errors: yes


  - name: Installing hadoop
    command: rpm -iv /root/hadoop-1.2.1-1.x86_64.rpm --force
    ignore_errors: yes
```

Fig 4(a): Playbook for Master Node

```
    ignore_errors: yes

  - file:
      state: directory
      path: /nn

  - name: Copy hdfs-site.xml file
    copy:
      src: /root/master_files/hdfs-site.xml
      dest: /etc/hadoop/

  - name: Copy core-site.xml file
    copy:
      src: /root/master_files/core-site.xml
      dest: /etc/hadoop/

  - name: Format the Namenode
    command: hadoop namenode -format -force

  - name: Starting Hadoop Service
    command: hadoop-daemon.sh start namenode
```

Fig. 4(b) Playbook for Master Node

Above Playbook as show in Fig. 4(a) and 4(b) will configure the setup for master. The Playbook will perform the following tasks: copy the jdk package, copy the hadoop package, install the jdk package, install the hadoop package, create a file directory, copy the hdfs-site.xml file to destination folder, copy the core-site.xml file, format the namenode and then finally start the master node.

2. Slave Node (DataNode)

```
hosts: slave
tasks:

- name: Copy jdk package
  copy:
     src: /root/jdk-8u171-linux-x64.rpm
     dest: /root/jdk-8u171-linux-x64.rpm

- name: Copy hadoop package
  copy:
     src: /root/hadoop-1.2.1-1.x86_64.rpm
     dest: /root/hadoop-1.2.1-1.x86_64.rpm


- name: Installing jdk package
  command: rpm -ivh /root/jdk-8u171-linux-x64.rpm --force


- name: Installing hadoop

  command: rpm -iv /root/hadoop-1.2.1-1.x86_64.rpm --force
"slave.yml" 38L, 829C
```

Fig. 5(a) Playbook for Data Node

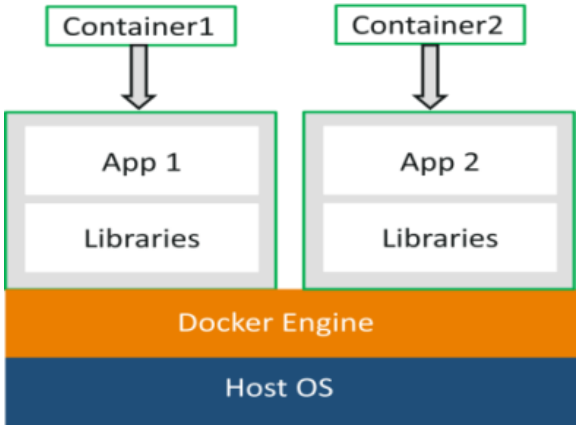
```
- name: Installing hadoop

  command: rpm -iv /root/hadoop-1.2.1-1.x86_64.rpm --force

- file:
    state: directory
    path: /dn

- name: Copy hdfs-site.xml file
  copy:
    src: /root/slave_files/hdfs-site.xml
    dest: /etc/hadoop/

- name: Copy core-site.xml file
  copy:
    src: /root/slave_files/core-site.xml
    dest: /etc/hadoop/

- name: Startting Slave Node
  command: hadoop-daemon.sh start datanode
```

Fig. 5(b) Playbook for Data Node

Above playbook **as shown in Fig 5(a) and Fig 5(b)** will configure the setup for Slave Node**.** The Playbook will perform the following tasks: copy the jdk package, copy the hadoop package, install the jdk package, install the hadoop package, create a file directory, copy the hdfs-site.xml file to destination folde**r, copy the core-site.xml file** and then finally start the **slave**r node.

B. Docker

Docker is a platform which packages an application and all its dependencies together in the form of containers. This containerization aspect ensures that the application works in any environment.



From the above diagram, we can every application has its own set of libraries and dependencies and run on separate container. Each application is independent of other applications which gives surety that different application will not interfere with one another.

B.1 Implementation:

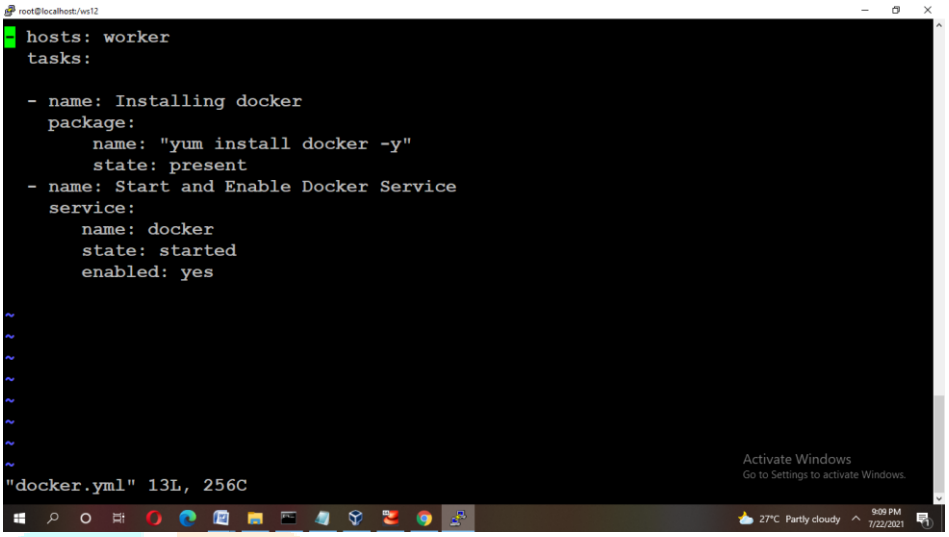


Fig 6 Playbook for Docker Setup

Above Playbooks will configure the setup for the docker on the target node. The above playbook will perform the following tasks: install the docker, start and enable the service.

C. Web Server

Apache HTTP Server is a free and open-source web server that delivers web content through the internet. It is commonly referred to as Apache and after development, it quickly became the most popular HTTP client on the web. It's widely thought that Apache gets its name from its development history and process of improvement through applied patches and modules but that was corrected back in 2000. It was revealed that the name originated from the respect of the Native American tribe for its resiliency and durability.

C.1 Implementation:

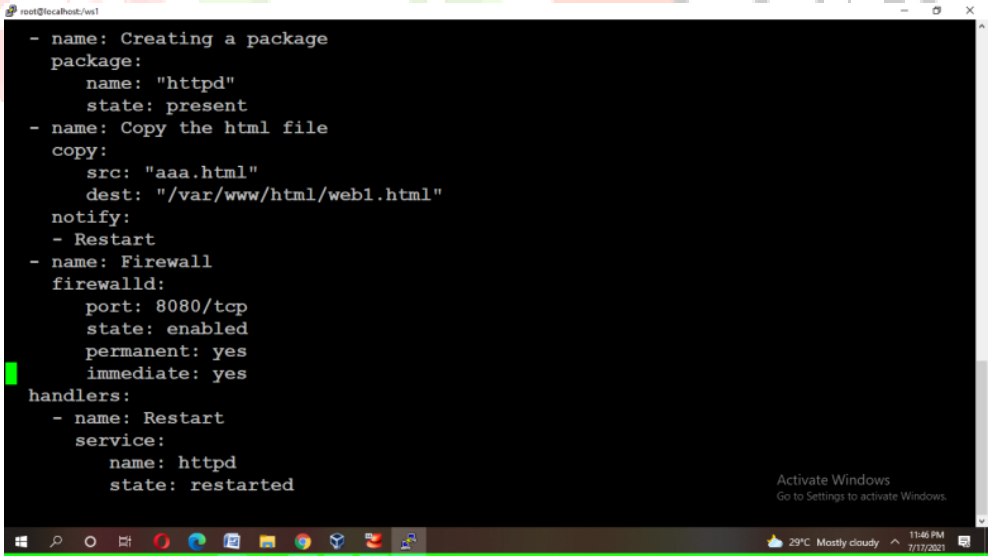


Fig. 7(a) Playbook for Web Server

Above Playbook will configure the setup for web server. The playbook will perform the following tasks: install the package httpd apache, copy the html website to root document folder (/var/www/html), start the web service. We can make changes in the website and again run the same playbook which will automatically restarts the Apache Service and changes will be deployed.

D. Kubernetes

Kubernetes is an open-source container management (orchestration) tool. It helps in the automation of manual processes involved in deploying, managing, and scaling containerized applications.

Case Study by Adidas – How they got benefitted from Kubernetes

In recent years, the adidas team was happy with its software choices but accessing all of the tools was a problem. They found the solution with containerization, agile development, continuous delivery, and a cloud native platform that includes Kubernetes and Prometheus.

Just six months after the project began, 100% of the adidas e-commerce site was running on Kubernetes. Load time for the e-commerce site was reduced by half. Releases went from every 4-6 weeks to 3-4 times a day. With 4,000 pods, 200 nodes, and 80,000 builds per month, adidas is now running 40% of its most critical, impactful systems on its cloud native platform.

Kubernetes Architecture/Kubernetes Components

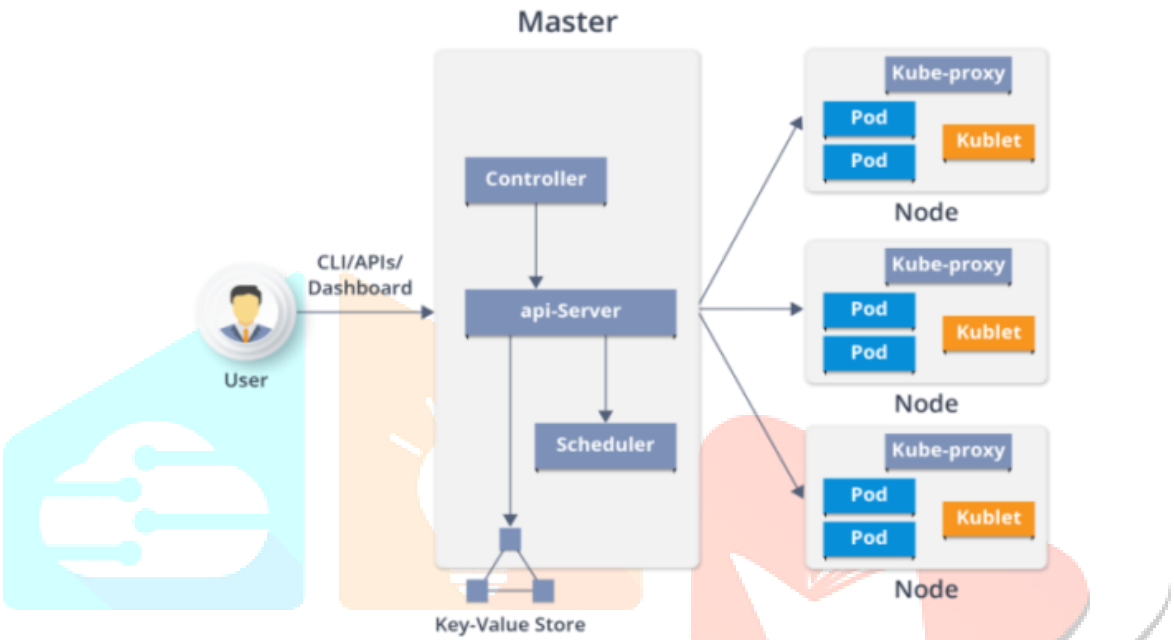


Fig 8: Architecture of Kubernetes

Kubernetes Architecture has the following main components:

- Master nodes
- Worker/Slave nodes
- Distributed key-value store(etcd.)

Master Node

The master node is the origin for all task assignments. It coordinates processes such as:

- **Scheduling** and scaling applications
- **Maintaining** a cluster's state
- **Implementing** updates

Worker Node

The **worker nodes** are the components that run these applications. Worker nodes perform tasks that is assigned by the master node. They can either be virtual machines or physical computers, all operating as part of one system.

D.1 IMPLEMENTATION:

Configuration of Master Node

```
- name: Installing docker
  package:
    name: docker
    state: present

- name: Start and Enable Docker Service
  service:
     name: docker
     state: started
     enabled: yes
- name: Adding repository for kubernetes
  yum_repository:
    name: kubernetes
    description: RPMforge YUM repo
    baseurl: https://packages.cloud.google.com/yum/repos/kubernetes-el7-$basearch
    enabled: yes
    gpgcheck: yes
    gpgkey: https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages
.cloud.google.com/yum/doc/rpm-package-key.gpg
    exclude: kubelet kubeadm kubectl
```

Fig. 9(a) Playbook for Master Node

```
- name: Installing kubectl kubeadm and kubelet
  yum:
     name:
       - kubeadm
       - kubectl
       - kubelet
     state: present
     disable_excludes: kubernetes
- name: Enable kubelet
  shell: "systemctl enable kubelet --now"


- name: Setting the driver for docker by coping daemon.json file
  copy:
      src: "daemon.json"
      dest: "/etc/docker/daemon.json"
- name: Restarting the docker services
  service:
      name: "docker"
      state: restarted
- name: Installing the iproute-tc
```

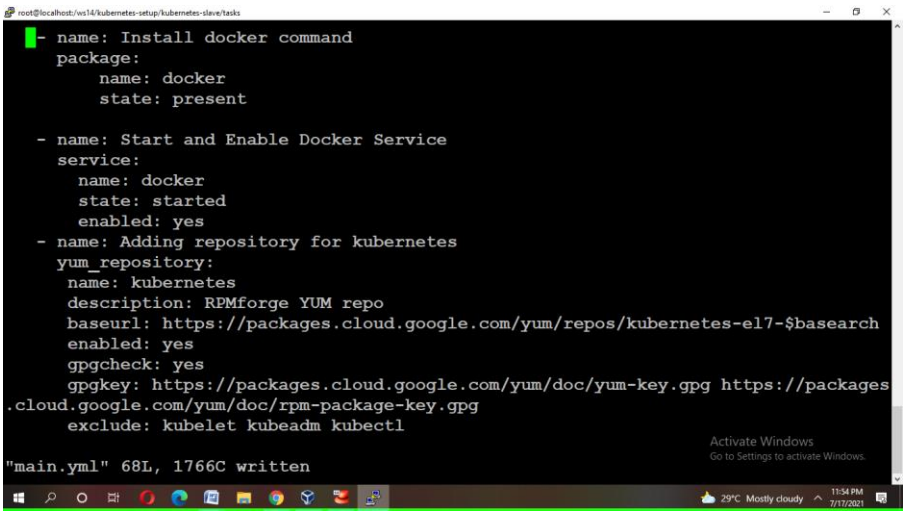
Fig. 9(b) Playbook for Master Node

```
- name: Installing the iproute-tc
  package:
     name: "iproute-tc"
     state: present


- name: Pull the images
  shell: "kubeadm config images pull"

- name: Initiating the kubeadm
  command: "kubeadm init --pod-network-cidr=10.240.0.0/16 --ignore-preflight-error
s=NumCPU --ignore-preflight-errors=Mem"

- name:
  shell: |
      mkdir -p $HOME/.kube
      cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
      chown $(id -u):$(id -g) $HOME/.kube/config
- name: Install Network Add-on
  command: kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/maste
r/Documentation/kube-flannel.yml
```

Fig. 9(c) Playbook for Master Node

The above playbook will configure the setup for master node in the target node.
Above Playbook will perform the following tasks: install the docker, enable the docker service, add yum repo for kubernetes, install kubectl, kubeadm and kubelet, enable kubelet service, copy the daemon.json file for setting the docker driver, restart the docker service, install the iproute-tc, pull the image, initiate the kubeadm and install the network add-on.

Configuration of Slave Node

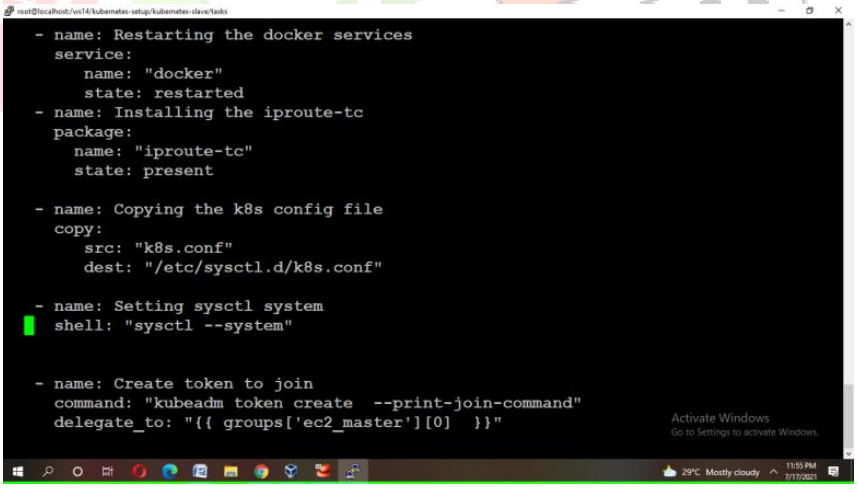


```
- name: Install docker command
  package:
      name: docker
      state: present

- name: Start and Enable Docker Service
  service:
    name: docker
    state: started
    enabled: yes
- name: Adding repository for kubernetes
  yum_repository:
   name: kubernetes
   description: RPMforge YUM repo
   baseurl: https://packages.cloud.google.com/yum/repos/kubernetes-el7-$basearch
   enabled: yes
   gpgcheck: yes
   gpgkey: https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages
.cloud.google.com/yum/doc/rpm-package-key.gpg
    exclude: kubelet kubeadm kubectl

"main.yml" 68L, 1766C written
```

Fig. 10(a) Playbook for Slave Node

```
- name: Installing kubectl kubeadm and kubelet
  yum:
    name:
      - kubeadm
      - kubectl
      - kubelet
    state: present
    disable_excludes: kubernetes

- name: Enable kubelet
  service:
      name: kubelet
      state: started
      enabled: yes

- name: Setting the driver for docker by coping daemon.json file
  copy:
     src: "daemon.json"
     dest: "/etc/docker/daemon.json"

- name: Restarting the docker services
```

Fig. 10(b) Playbook for Slave Node



```
- name: Restarting the docker services
  service:
     name: "docker"
     state: restarted
- name: Installing the iproute-tc
  package:
    name: "iproute-tc"
    state: present

- name: Copying the k8s config file
  copy:
     src: "k8s.conf"
     dest: "/etc/sysctl.d/k8s.conf"

- name: Setting sysctl system
  shell: "sysctl --system"


- name: Create token to join
  command: "kubeadm token create  --print-join-command"
  delegate_to: "{{ groups['ec2_master'][0]  }}"
```

Fig 10(c) Playbook for Slave Node

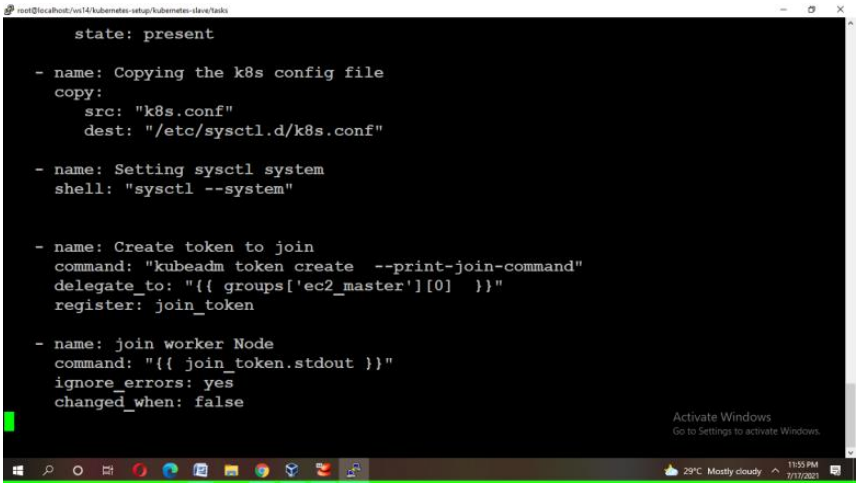


Fig. 10(d) Playbook for Slave Node

The above playbook will configure the setup for slave node in target node.
Above Playbook will perform the following tasks: install the docker, enable the docker service, add yum repo for kubernetes, install kubectl, kubeadm and kubelet, enable kubelet service, copy the daemon.json file for setting the docker driver, restart the docker service, install the iproute-tc,copy the kubernetes config file, change the system sysctl settings, create the token by running the command on master node and finally join the worker node with the master.

III. CONCLUSION

After experimentation and analysis, it can be observed that the manual installation, updation and configuration of different software needs a lot of time and man power. With the help of automation tool like Ansible it becomes easy to configure the technology inside the Instance on Cloud. Ansible's simplicity and ability to decrease the complexity of other tools has made it a reliable applicant for your environment.
In this paper we were able to implement setup for various technologies like Hadoop, Kubernetes, Docker and Webserver using Ansible Playbook which will take few minutes to complete rather than doing manually which may take upto hours.

IV.FUTURE WORK

As we have seen that configuration using Automation tools reduces the time as well as need of manpower to deploy an application. Ansible is one such DevOps tool which can be used to configure applications on remote machines. There are multiple configuration management tools available in the market that we can use to deploy and for configuration but it is not easy to choose which one is best. It depends on the developer's choice which tool is best according to their comfortability. In future, we can try to configure the setup on cloud through other configuration management tools.

V. REFERENCES

[1] Pranav T P, Charan S, Darshan M R. (2021). Devops Methods for Automation of Server Management using Ansible International Journal of Advanced Scientific Innovation, 1(2), 7-13.
[2] Ramandeep Singh , Dr. Ravindra Kumar Purwar, 2019, Cloud Automation with Configuration Managementusing CHEF Tool, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 08, Issue 04 (April – 2019),
[3] Masek, Pavel & Štůsek, Martin & Krejčí, Jan & Zeman, Krystof & Pokorny, Jiri & Kudlacek, Marek. (2018). Unleashing Full Potential of Ansible Framework: University Labs Administration. Proceedings of the XXth Conference of Open Innovations Association FRUCT. 426. 10.23919/FRUCT.2018.8468270.
[4] https://www.edureka.co/blog/devops-tutorial
[5] https://www.redhat.com/en
[6] https://www.edureka.co/blog/aws-cli/
[7] https://www.edureka.co/blog/what-is-hadoop/
[8] https://www.edureka.co/blog/docker-explained/
[8] https://www.edureka.co/blog/kubernetes-tutorial/
[9] https://kubernetes.io/case-studies/adidas/
[10]Elastic Compute Cloud(EC2), Amazon Web Services, Amazon.com, http://aws.amazon.com/ec2.
[11] Ansible, http://docs.ansible.com.