

ACCELERATING CLOUD-NATIVE APPLICATIONS: AUTOMATED KUBERNETES CLUSTER DEPLOYMENT

Dr. Smita Chavan¹, Dr. Rupali Mahajan², Dr. Prajakta Ajay Khadkikar³

¹Government College of Engineering, Information Technology Department, Aurangabad, India

²Vishwakarma Institute of Information Technology, Computer Engineering Department, Pune, India

³Pune Institute of Computer Technology, Computer Engineering Department Pune
Email: rupali.mahajan@viit.ac.in

Abstract: Nowadays, cloud computing provides a lot of services, as per user application services are used. Paper is based on DevOps. It can be mostly specified like a set of practices expected to decrease the time among committing a change to a system also the change being sited into normal production, although guaranteeing high quality. In the present scenario, IAAS clouds significance has been understood ahead of measures. Main objective of the proposed system is continuous delivery (CD) and continuous integration (CI), is called the core of devops. In the proposed system, from understanding DevOps and why DevOps is necessary in today's world is clearly mentioned. As we know, in a production environment, deploying Kubernetes clusters manually is difficult and time consuming. So, we are creating a Kubernetes cluster on a public cloud, where instances and the Kubernetes cluster will be created using ansible. One of the additional functionalities of this project is, even if the node count is increased or decreased, change will be triggered by Jenkins and a new Kubernetes cluster will be launched on the public cloud seamlessly. So the idea of the system by developing a project using some advanced tools, to overcome the drawbacks of the existing work. System also looked at the working mechanism of each module.

Keywords: Kubernetes, Ansible, Git, Jenkins, AWS, DevOps

1 Introduction

In a production environment, deploying Kubernetes clusters manually is difficult and time consuming. So, we are creating a Kubernetes cluster on a public cloud, where instances and the Kubernetes cluster will be created using ansible. One of the additional functionalities of this project is, even if the node count is increased or decreased, change will be triggered by Jenkins and a new Kubernetes cluster will be launched on the public cloud seamlessly. Using ansible systems also automate the deployment of pods, services, and provide high availability of pods using HPA (Horizontal Pod Autoscaler) in Kubernetes cluster.

The IT industries are adopting consequently with the growing need meanwhile the business requirement in addition to market competition are all indicating in the direction of cloud. Why all these scaling up the reason is being seen in Information Technology that later customers are challenging swift service deployment above the cloud [1].

To create a kubernetes cluster in AWS, we need an IAM (Identity Access Management) role of EKS (Elastic Kubernetes Service) which each organization does not provide to the employee. Even if we have an IAM role of EKS we need to create a cloud formation to create cluster on AWS and it's a tedious task to perform the same steps for many clusters. Proposed system does not need all these services to create a cluster on AWS. System just need access to Amazon EC2 (Elastic Compute Cloud) service that usually each employee/student has, so if an employee/student needs to create a kubernetes cluster on AWS quickly, he can do that using this proposed system. For creating multiple clusters system uses Jenkins which provides continuous delivery, where any change in the git repository is found, it will create a new kubernetes cluster ready for production environment [2].

2 Literature Survey

Research study of the proposed systems is about all the papers studied throughout implementation. Through this study, developers will be able to gain more knowledge and understanding to develop new software. As a result, the developer would be able to improve the weakness and integrate the existing strength with new features to improve functionality of existing software. As a part of literature in this stage, findings, summary, analysis, synthesis of the system will be done. This is to ensure the full understanding of the system and that the suitable software and tools are used [3].

In the existing scenario the complete process of creation of cluster nodes, deployment of kubernetes and connecting nodes to master node/s was a whole manual process. Based on some research papers which are found out some drawbacks in the existing system. Few papers have cloud infrastructure as automated, using Docker with Robust Container Security are used for Continuous integration and continuous delivery [4].

Other papers showed provisioning of automated application using Ansible configuration management in IAAS cloud.

Drawbacks according to research study are

- No container orchestration, no kubernetes cluster is deployed in this project, which is a container orchestration tool for handling high workload and performing complex tasks [5].
- Manual configuration requires more time and needs to be done repeatedly. Unlike manual configuration, Automation tools make the work easier for the DevOps team and completes the task without any interruption [6]
- Less secure as compared to the Kubernetes, docker provides less security to the application as the traffic directly hits the container. [7].
- High resources Consumption As compared to the containerized environment application deployed directly on the EC2 instances require more resources.
- No version control helps to keep a track of all the changes made in the git repository or code.
- Provide IAAS clients are responsible for managing all the aspects such as OS, data, networking, storage, etc.

3 Methodology

On the basis of study performed about all the functions that deal proposed system, following requirements are specified.

3.1 Hardware Requirement:

As the whole project is deployed on public cloud, there is no particular hardware requirement for the base machine. For the instances that we are going to launch on AWS, following are the requirements.

Master Node:-

Table 1. Hardware requirement for master node.

RAM	2 GB
CPU	2 CPU
STORAGE	10 GB

Worker Node:-

Table 2. Hardware requirement for worker node.

RAM	1 GB
CPU	2 CPU
STORAGE	10 GB

• Worker Node:

Hardware requirement for worker node RAM 1 GB CPU 2 CPU STORAGE 10 GB

3.2 Software Requirement:

Table 3. Software requirement

Operating System	Linux
Software	Ansible <= 2.7 Jenkins Git

3.3 Tools

□ Kubernetes :

Orchestration engine for scaling, automating deployment and management of containerized applications is called Kubernetes which is an open source container. Cloud native computing foundation (CNCF) by Kubernetes hosts the open source project, as well called K8s, which is an open-source system for same functionalities which are described by CNCF. Later operations of Kubernetes are done for container level instead of hardware level. It offers few common applicable features general to PaaS offerings, like load balancing, scaling, deployment, users will work on integration of their monitoring, logging, and alerting solutions. Kubernetes can be installed on top of many CRI i.e. container runtime interface such as containerd, docker and cri-o. Kubernetes is available as a service in AWS as EKS, GCP as GKE and Azure as AKS [8] Uses of Kubernetes

- Load balancing and service discovery: using their own IP address or using the DNS name, a container is revealed by Kubernetes. Distribution of network traffic and load balance can be done by Kubernetes, hence the deployment is stable, just in case high traffic to a container.
- Orchestration storage: - with the help of Kubernetes it's easy to automatically mount your choice storage system like local storages and public cloud.
- Rollbacks and rollouts automated: - to create new containers for users deployment, new container adopts all their resources and remove existing containers users can automate Kubernetes. Here Kubernetes uses different types of deployment such as rolling, canary and blue-green deployment.
- Bin packing automatic: - Kubernetes with a cluster of nodes you can use to run containerized tasks. How much CPU and memory (RAM) each container needs, you can tell to Kubernetes. To make the best use of your resources, it can fit containers onto your nodes. This is called a request and limit where a container can request more resources on node if they are available and though a container is not allowed to use additional than its required limit.
- Self-healing:- by restarting Kubernetes containers kills containers that fail, replaces containers, that don't reply to your health check which is user-defined, and doesn't promote them to clients till they are ready to attend. These health checks are called Probes in Kubernetes.
- Configuration management and Secret: - it is used to store and manage sensitive information, like OAuth tokens, passwords, and keys like SSH [9] [10].

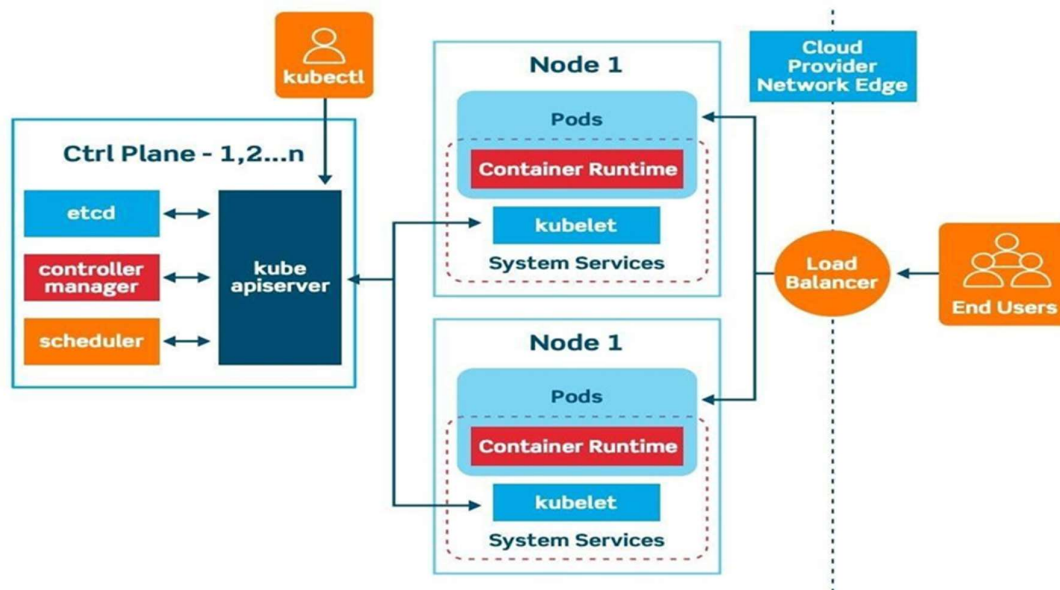


Fig.1. Architecture of Kubernetes.

As mentioned in fig.1, after deploying Kubernetes, cluster we will get. A Kubernetes cluster consists of nodes from worker machines set, which run containerized applications.

Control Plane: as per above diagram control plane accomplishes all pods and nodes in the cluster. It also uses control loop to monitor the state of cluster [11].

Node are treated as worker machine uses system services also responsible for container runtime

□ **Ansible**

Ansible is configuration management tool also called is an open-source automation tool, as it can configure multi- ple systems parallely. Ansible is agentless as it does not require any agent at the client side like other configuration management tools such as CEPH, PUPPET. Ansible uses Python as a backend for processing yaml files. Ansible controller and managed nodes which are the clients talk to each other through password less sash. Ensile is a push based tool where the controller sends the configuration data to the managed node without the nodes asking for it. Ansible requires the controller node to be a Linux machine, but can configure different OS easily. Ansible uses yaml file format to write playbooks. Playbooks consist of different tasks to perform on the nodes. These nodes are specified inside a file named as inventory.

Why Ansible ?

- Configuration management: - Ansible configurations are simple data descriptions of infrastructure and are both parsable by machines and readable by humans. We can configure OS, switches and routers using ansible.
- Application deployment: - using ansible we can install different packages/software on many managed nodes parallely in no time. All we need to do is add the nodes ip address in inventory and run the playbook consisting of tasks.
- Cloud Provisioning: - We can also automate different public, private and hybrid clouds using ansible. For eg. We can create VM's, VPC, storage instances and automate different services on cloud using ansible.
- Ad-hoc task execution: - It is used to run a single command on all managed nodes specified in an inventory file.
- Network automation: - As specified before Ansible can be used for automating different types of routers and switches. For eg. We can create a DHCP server in router or create VLANs in switch using ansible.
- Multi-node orchestration: - to define your infrastructure Ansible's orchestration is used as per need [12].

□ **Git**

These are available for software development according to wide range of operating system and for IDEs .It is used as distributed model, branching and merging is easy, workflow is flexible, use the process that best fits you [13].

□ **Jenkins**

An open source automation tool called Jenkins. It provides simple methods for continuous delivery environment and continuous integration for language combinations and source code repositories

What is continuous integration?

Afterward a code commit, the software is built and tested instantly is known as Continuous Integration. In huge projects every commit code is built and tested. After test pass it goes for

deployment process. Finally after successful deployment the code is ready to production. This pipeline consists servers (Jenkins), open source tools, source control tool (SVN, GIT CVS) frameworks for automation testing (UFT, Selenium etc.)

It is portable to all the major platforms because it is built with Java [14].

3. System Development

3.1 Project Scope

The scope of this project is to launch the kubernetes cluster using automation tool like Ansible on AWS. As it is launched on a public cloud anyone with the authenticity can access the kubernetes cluster from anywhere without any difficulty. If anyone wants to launch many clusters with variable resources, it can be done using this system as we are using Jenkins for the CI/CD purpose, which will check for any change in the git repository (contains the YAML file for creating a kubernetes cluster with specified resources). Any change in the parameters of the YAML file will trigger the Jenkins and a new kubernetes cluster will be launched with the specified hardware resources on AWS instantly.

3.2 Proposed System

Implemented modules are following

3.2.1 Module 1

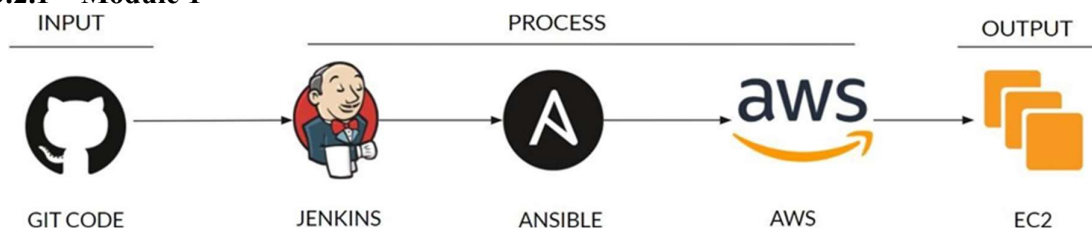


Fig.2. Module 1.

The above figure shows the flow of module 1. This module uses 2 files as input i.e. .boto file and task.yml file which are stored on git repository as shown in figure:

- .boto: This file contains secret key, the access key and the session token required for authentication with AWS when used with ansible. (AWS provides sessions for using this keys, every time the session is refreshed we need to change the .boto file) 13 39
- task.yml this is an ansible playbook file that runs on the local machine. File contains the ec2 module that creates instances on AWS using .boto file required for authenticity.

In the processing, first we create a Jenkins job that will continuously hit the repository every minute to check any new commit or any change to the files in the git repository. i.e. .boto or task.yml. If any of the files in the git repository is updated with a new commit, it will notify Jenkins job. Then the Jenkins job will then be triggered and will clone the updated repository in the specific location (home directory of the Jenkins user i.e. /var/lib/jenkins). After cloning the repository, ansible will run the task.yml file using ansible-playbook command on the same system and create EC2 instances with specified parameters. The execution of ansible playbook will result in creation of EC2 instance on AWS which is the output for this module. The EC2 instance will be created with a public ip, which can be used to access or make any changes to the instance using windows powershell or Linux terminal. Here the instance will be created with a specified name like master or node to identify their roles in the kubernetes cluster.

3.2.2 Module 2

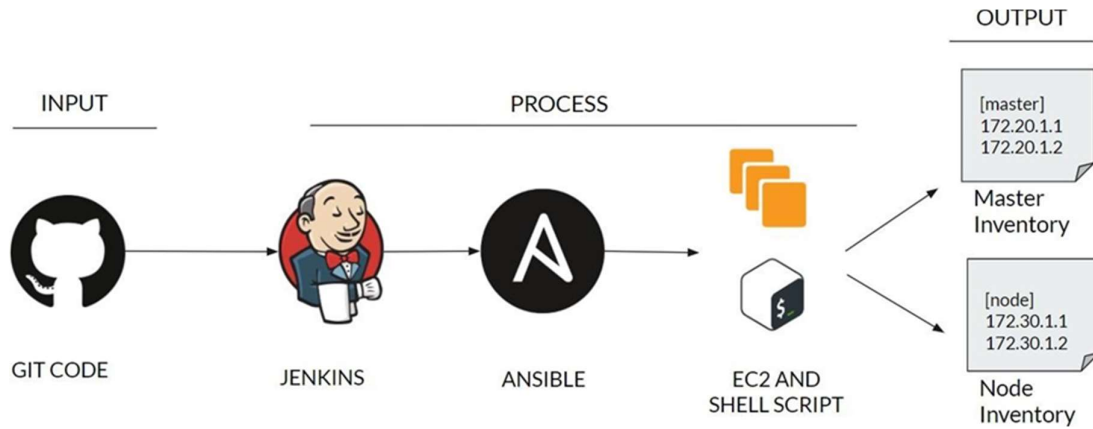


Fig. 3. Module 2.

The above figure shows the flow of module 2. As shown, it will use a git code as its input. The git code will have the same .boto file. Here, the task.yml file will contain the code to fetch the private IP addresses of the launched EC2 instances. The process will be the same as mentioned in the earlier module. The fetched IP addresses will be stored in the file and the ip addresses will be segregated in master and node groups. This file will be further used as an inventory for running ansible-playbook on specific managed nodes to configure master and worker nodes.

3.2.3 Module 3

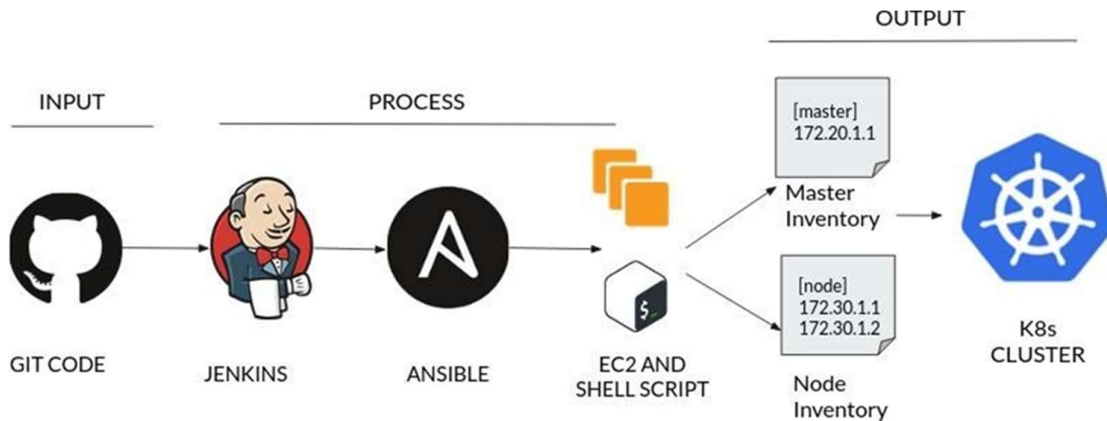


Fig. 4. Module 3.

The above figure shows the flow of module 3. In this module we created a dynamic inventory which consists of private IP addresses of node and master that we have deployed on the AWS platform. The created dynamic inventory will be used by ansible playbooks placed in git. This ansible playbook consists of many plays which will be used to deploy the kubernetes cluster. Any changes in this play will be reflected on AWS by deploying a new kubernetes cluster with specified nodes.

4. Results

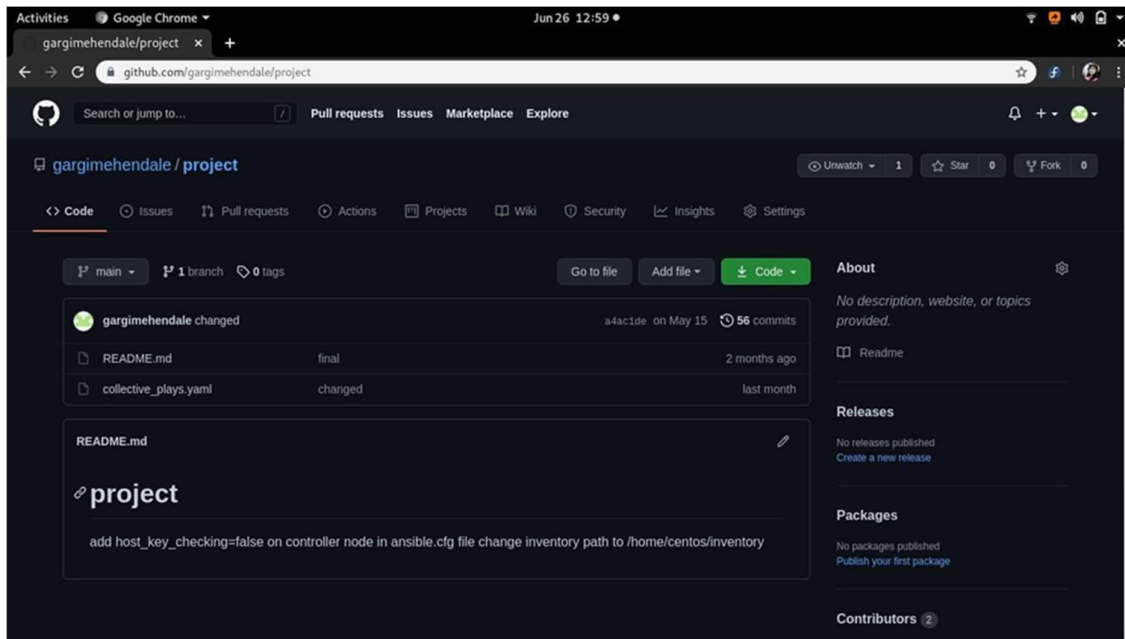


Fig. 5. Git Repository.

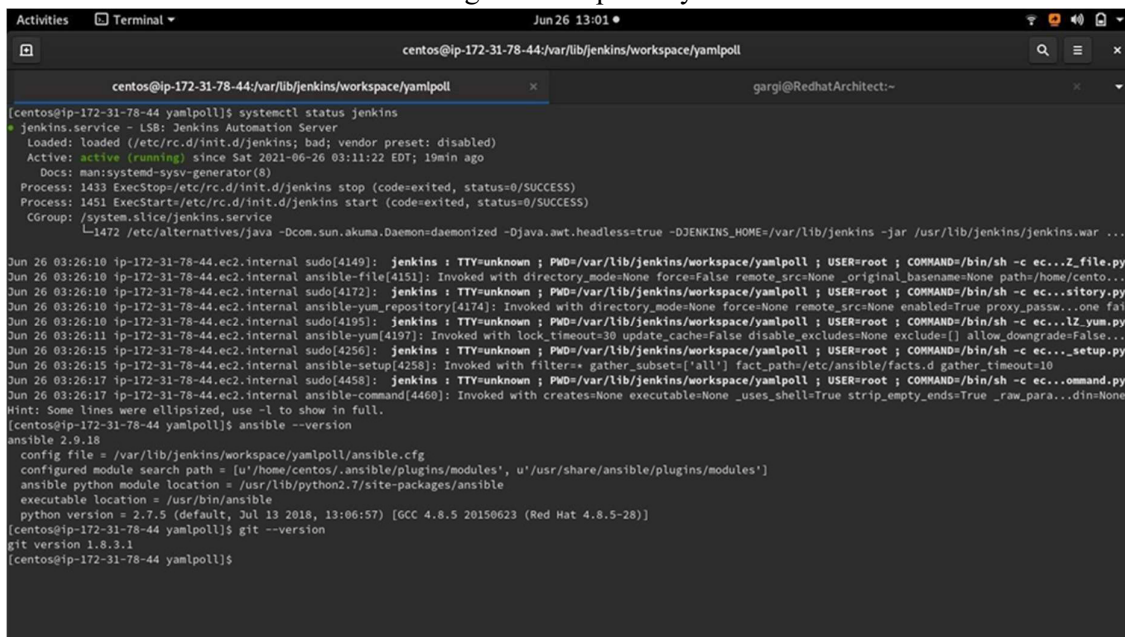


Fig. 6. Installed software.

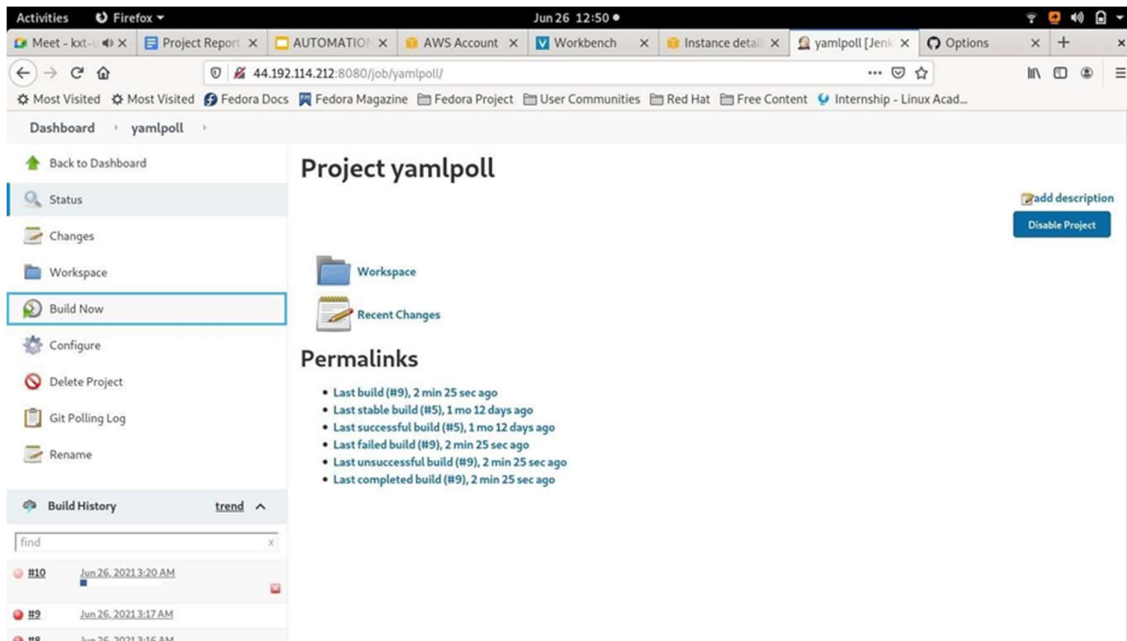


Fig. 7. Jenkins SCM poll trigger.

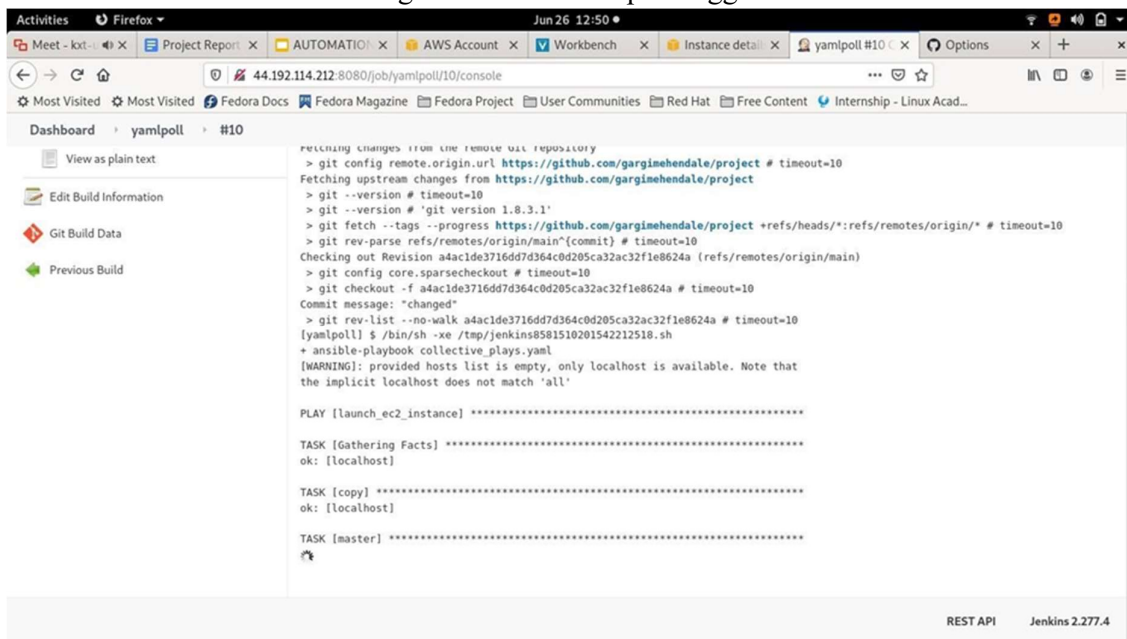


Fig. 8. Console output 1.



Fig. 9. Console output 2.

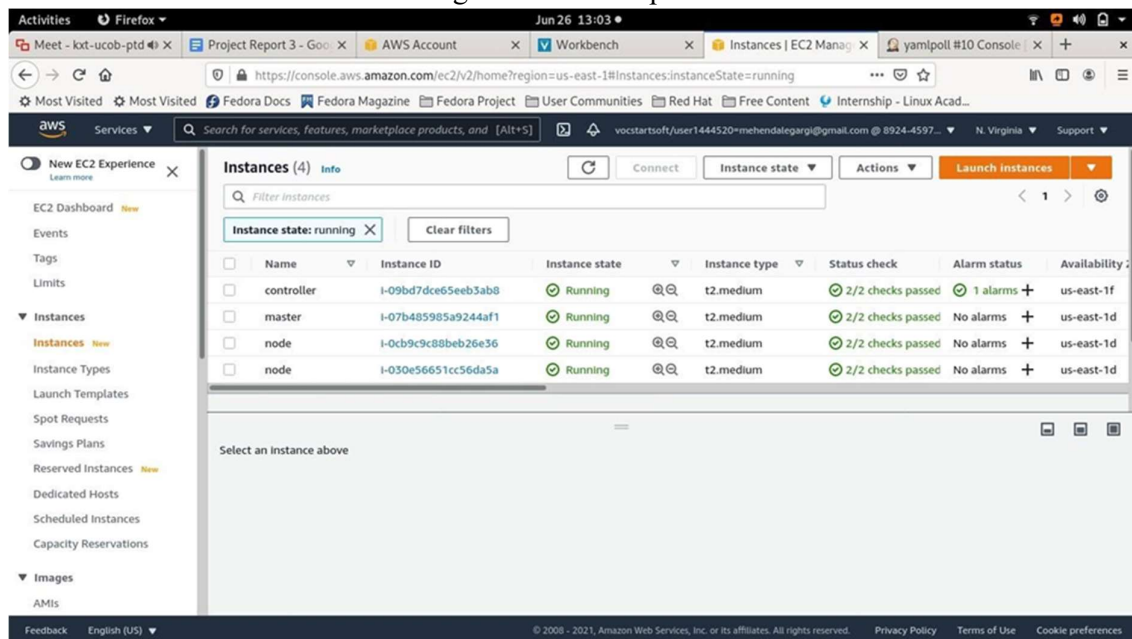
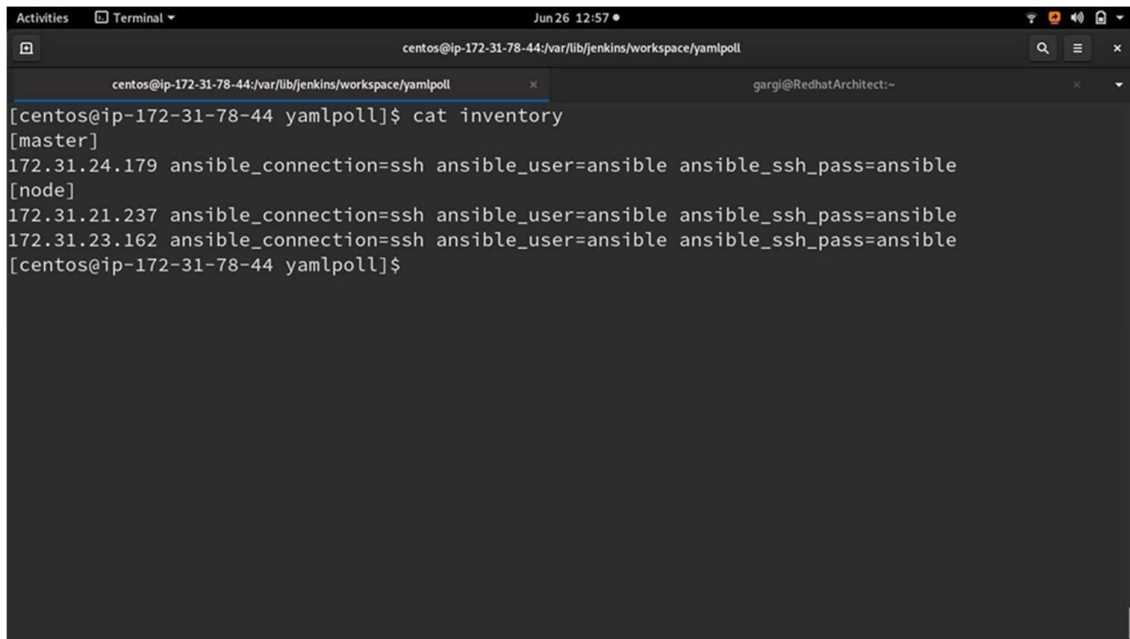


Fig. 10. EC2 dashboard with created instances.

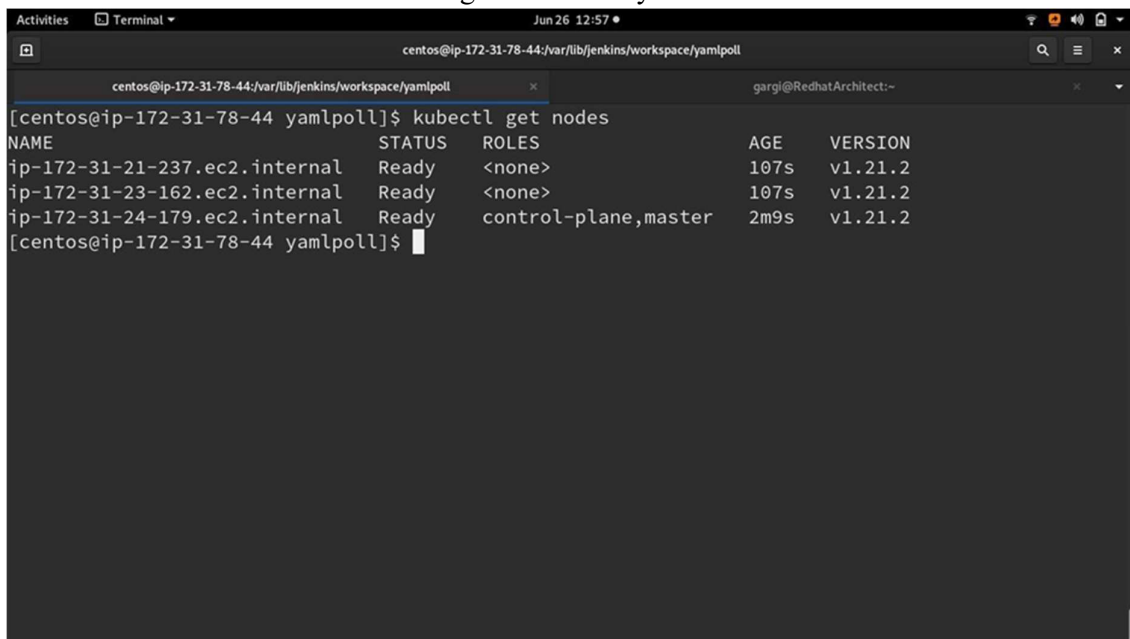


```

[centos@ip-172-31-78-44 yamllpoll]$ cat inventory
[master]
172.31.24.179 ansible_connection=ssh ansible_user=ansible ansible_ssh_pass=ansible
[node]
172.31.21.237 ansible_connection=ssh ansible_user=ansible ansible_ssh_pass=ansible
172.31.23.162 ansible_connection=ssh ansible_user=ansible ansible_ssh_pass=ansible
[centos@ip-172-31-78-44 yamllpoll]$

```

Fig. 11. Inventory file.



```

[centos@ip-172-31-78-44 yamllpoll]$ kubectl get nodes
NAME                                STATUS    ROLES                                AGE      VERSION
ip-172-31-21-237.ec2.internal        Ready     <none>                               107s     v1.21.2
ip-172-31-23-162.ec2.internal        Ready     <none>                               107s     v1.21.2
ip-172-31-24-179.ec2.internal        Ready     control-plane,master                 2m9s     v1.21.2
[centos@ip-172-31-78-44 yamllpoll]$

```

Fig. 12. Kubernetes Cluster.

4.1 Sample Codes

collective_plays.yaml

```

- name: launch_ec2_instance
  hosts: localhost
  vars:
    count: 2
    var: ec2.instances
  tasks:
    - copy:

```

```

/

content: |
    [master]
    [node]
dest: ./inventory
name: master
ec2:
    key_name: newkey
    instance_type: t2.medium im-
age: ami-0015b9ef68c77328d re-
gion: us-east-1
group: securegroup
count: 1
wait: yes
instance_tags:
Name: master user_data: |
    #!/bin/bash
    sudo useradd ansible
    PASSWD="ansible"
    sudo echo ${PASSWD} | passwd --stdin ansibl
    sudo echo "ansible ALL=(ALL) NOPASSWD: ALL" >>
/etc/sudrs

    sudo sed -i 'PasswordAuthentication yes/s/^#//'
```

CONCLUSION

In the proposed system, from understanding DevOps and why DevOps is necessary in today's world. We also analyzed the existing system and came across some drawbacks. So this system with the idea of developing a project using some advanced tools, to overcome the drawbacks of the existing project and also add some new features to our project. System also looked at the working mechanism of each module. Then further studied in detail about devops tools and integrated these tools and deployed the kubernetes cluster on AWS cloud which is highly reliable and secure.

FUTURE SCOPE

Developments in the DevOps industry are necessary as new tools are coming into picture. So here are some further enhancements that can be added to proposed system

- Containerisedceph cluster.
- Customizing running cluster.
- Backend with Ceph Storage cluster.
- Integrate with a graphical dashboard.
- Kubernetes on CRI-O.

References

1. John T. J. Mathieson , Thomas Mazzuchi , and Shahram Sarkani” The Systems Engineering DevOps Lemniscate and Model-Based System Operations” IEEE SYSTEMS JOURNAL, VOL. 15, NO. 3, SEPTEMBER 2021 3890-3991.

2. Nishant Kumar Singh* , Sanjeev Thakur† Himanshu Chaurasiya‡ and Himanshu Nagdev “Automated Provisioning of Application in IAAS Cloud using Ansible Configuration Management” International Conference on Next Generation Computing Technologies Dehradun, India, 978-1-4673-6809-4/81-85NGCT-2015.
3. Sriniketan Mysari Vaibhav Bejgam ,” Continuous Integration And Continuous Deployment PipelineAutomationUsing Jenkins Ansible”,International Conference on Emerging Trends in Information Technology and Engineering 1-4,ic-ETITE-2020.
4. Francesco Minna and Balakrishnan Chandrasekaran “Understanding the Security Implications of Kubernetes Networking |”University of Trento and Vrije Universiteit Amsterdam ,46-56, September/October 2021.
5. Jiang Wenhao¹ , Li Zheng,”Vulnerability Analysis and Security Research of Docker Container”, IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE) 978-1-7281-8304-6/20/ IEEE 354-357, 2020.
6. “Cloud Based Integration of CRM, BPM & DM for SME’s”, March 2015 volume 3 Issue 3, ISSN: 2001-5569
7. PAVOL BARAN “Kubernetes Operator for managing OAuth2 tokens” Bachelor’s Thesis Brno, Spring 1-49, 2022.
8. Bandung - Padang “ High Availability Storage Server with Kubernetes” International Conference on Information Technology Systems and Innovation -ICITSI2020 ISBN: 978-1-7281-8196- IEEE,74-78, 2020.
9. Chia-Chen Chang and Shun-Ren Yang En-Hau Yeh and Phone Lin Jeu-Yih Jeng “A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning” 978-1-5090-5019-2, IEEE 1-6, 2017.
10. Steve Buchanan Janaka Rangama Ned Bellavance “Introducing Azure Kubernetes Service A Practical Guide to Container Orchestration” Distinguished Engineer, Microsoft 1-14.
11. Khairunnisa Abdullah² , Iman Hazwam Abd Halim³ , Rafiza Ruslan⁴,Khairunnisa Abdullah , Iman Hazwam Abd Halim , Rafiza Ruslan”Network Automation using Ansible for EIGRP Network " Journal of Computing Research and Innovation (JCRINN) Vol. 6 No. 4 Journal of Computing Research and Innovation (JCRINN) Vol. 6 No. 4 (pp59-69) <https://jcrinn.com> : eISSN: 2600-8793,2021.
12. David Badalyan * , Oleg Borisenko Ivannikov ,“Original software publication Ansible execution control in Python and Golang for cloud orchestration”, Institute for System Programming of the RAS, Moscow, Russia journal homepage: www.elsevier.com/locate/softx <https://doi.org/10.1016/j.softx.2022.101126> 2352- 7110/© 2022 SoftwareX 19 101126 1-7,2022.
13. Rf-16-Git.pdf Diomidis Spinellis 100 IEEE SOFTWARE | PUBLISHED BY THE IEEE COMPUTER SOCIETY 0740-7459/12IEEE 101-102, 2012.
14. Sikender Mohsienuddin Mohammad ,“Continuous Integration and automation” IJCRT | Volume 4, Issue 3 July 2016 | ISSN: 2320-2882 IJCRT1133440 International Journal of Creative Research Thoughts (IJCRT) www.ijcrt.org 938-944, © 2016