National College of Ireland

Configuration Manual

Docker Conatiner Deployment in Diverse Network regions
Programme Name

Jayesh Chitnawis
Student ID: 22244328

School of Computing
National College of Ireland

Supervisor: Rashid Mijumbi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Jayesh Chitnawis
Student ID:	22244328
Programme:	Programme Name
Year:	2024
Module:	Docker Conatiner Deployment in Diverse Network regions
Supervisor:	Rashid Mijumbi
Submission Due Date:	16/09/2024
Project Title:	Configuration Manual
Word Count:	1500
Page Count:	10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Jayesh Chitnawis
Date:	16th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Jayesh Chitnawis
22244328

Docker Conatiner Deployment in Diverse Network regions

1 Introduction

This configuration manual will describe the steps of implementing and deploying the database or docker application in the network infrastructure proposed in the research paper.

Here is the detailed information for developing such kind of network. This scalable environment for docker containerised applications and relational database service instances. This environment will spread over multiple regions which gives us special features called high availability and fault tolerance with also multiple availability zones (AZs). This procedure includes Virtual Private cloud hosting for containers and also for RDS.

This environment requires ideal up time and data consistency across multiple nodes with heterogeneous network locations and diverse networks. This setup was proven to manage complex workloads, maintaining high standards.

1.1 Pre-requisites:

We confirm the following prerequisites are met:

1. AWS account: AWS Account with valid and appropriate super-user permissions to create and manage services. AWS CLI command line interface makes easier management of resources and IAM users.

2. Networking knowledge: understanding of networking and basic concepts. Routing and security groups. And familiar with AWS VPC controls for the creation of VPC with subnets and route tables.

3. Docker: docker installed on each Virtual machine (VM) and docker understanding to pull images and manage containers.

4. Text editor: to ensure and save YAML files on the local device.

5. Linux VM: we can create EC2 AWS Linux as well or else we can have our own Linux system.

2 Execution step to launch AWS EC2 Instance

2.1 AWS Management Console

Step1: open the AWS Management Console (Sign in using your AWS account credentials)
Step2: navigate to EC2 Dashboard
Step3: In the dashboard click "launch instance" It will redirect you "instance " wizard.
Step4: Name your instance > select Amazon Machine Image > for current instance we go with "no key pair required"
Step5: for network settings we click on edit > select Subnet > for current instance we can go for subnet US-East-1a > in security group same as it.
Step6: now click on "Review and Launch"
Step7: After launching it should redirect to EC2 Dashboard and it goes for view instance on Dashboard, As it will come running after some time, so here instance will be ready to use.
Step8: once an instance is open, click on connect, and it will redirect to AWS CLI (Command line Interface).

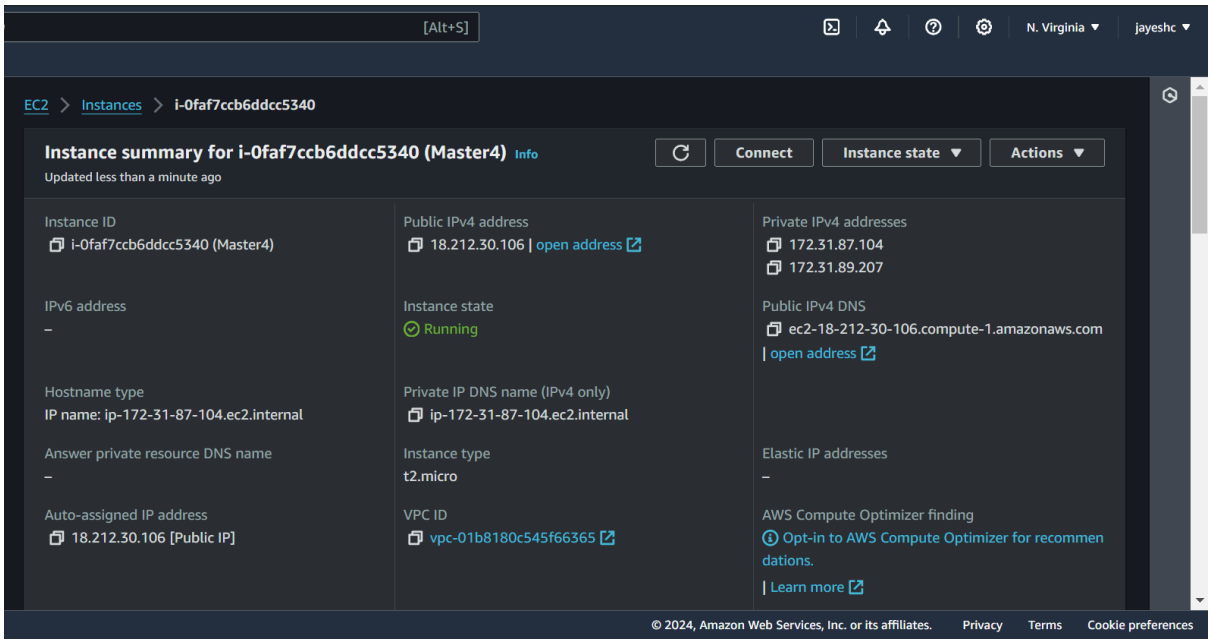

Figure 1: Master Node EC2

3 Docker installation on EC2 instances

Step1: Before installing Docker, you should update the package index.
"**sudo yum update -y**"
"**sudo yum upgrade**"

Step2: Installing docker To install Docker on Amazon Linux, you can use the following command:
"**sudo yum install docker -y**"

Step3: Install Docker Compose
"**sudo yum install docker-compose**"

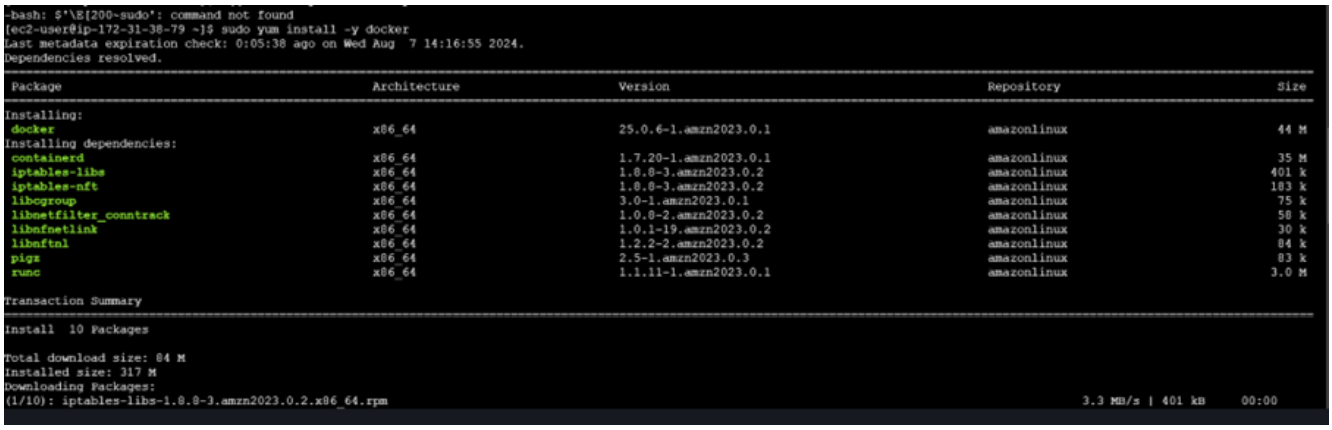

Figure 2: Docker installation on Master node

here show case for master node, As we are using push model, we need to perform same steps on Slave nodes as well.

As we perform Same operations on Slave, for we need to deploy 1 EC2 as Slave, this slave can created any other region as well and use of different subnet is also allowed as well.

3.1 Pulling source code From GitHub

On master node and on Slave node we pull code from GitHub,

Docker file for Master and Slave is stored in GitHub repository. it also consist of one YAML file of Docker Compose, reason to use Docker Compose, as it will define and share multi-container applications. It have YAML file as it servers the purpose of defining the services with single command. If we do not use Docker Compose, then we would need to write many arguments in Docker.

3.2 VxLAN and OVS Connection

As Master is on in another region or in another subnet and Slave is another as well so, to connect them we use VxLAN as tunnel for connection. OVS supports VxLAN and provides managing and routing of traffic packets between VM's

Perform Both the Steps in Master as well as Slave Node.

Add Bridge

For Master Node

Step1: ovs-vsctl add-br master-br

For Slave Node

Step2: ovs-vsctl add-br slave-br

Internal Port Bridge

For Master node

step3: "$sudo - ovs - vsctl - add - portmaster - br$
$master - internal - -set - interface$
$master - internal - type = internal$"

For Slave node

Step4: "$sudoovs - vsctladd - portslave - br$
$slave - internal - -set - interfaceslave - internal - type = internal$"

Creating Vxlan tunnel Through Private IP

For Master node

step5: $sudoovs - vsctladd - portmaster - brmaster - vxlan - - setinterfacemaster - vxlantype = vxlanoptions : remote_ip = 172.31.20.180 options : key = 2000$

for Slave node

step6: "$sudoovs - vsctladd - portslave - brslave - vxlan - -setinterfaceslave - vxlantype = vxlanoptions : remote_ip = 172.31.89.207 options : key = 2000$"

Now we Assigned them Static IP.
Adding overlay IP

For Master node

step7: "sudo ifconfig master-internal 10.20.31.100/24 mtu 1450 up"

for Slave node

step8:"sudo ifconfig slave-internal 10.20.32.200/24 mtu 1450 up"

To Prove OVS architecture

step9:" sudo ovs-vsctl show"

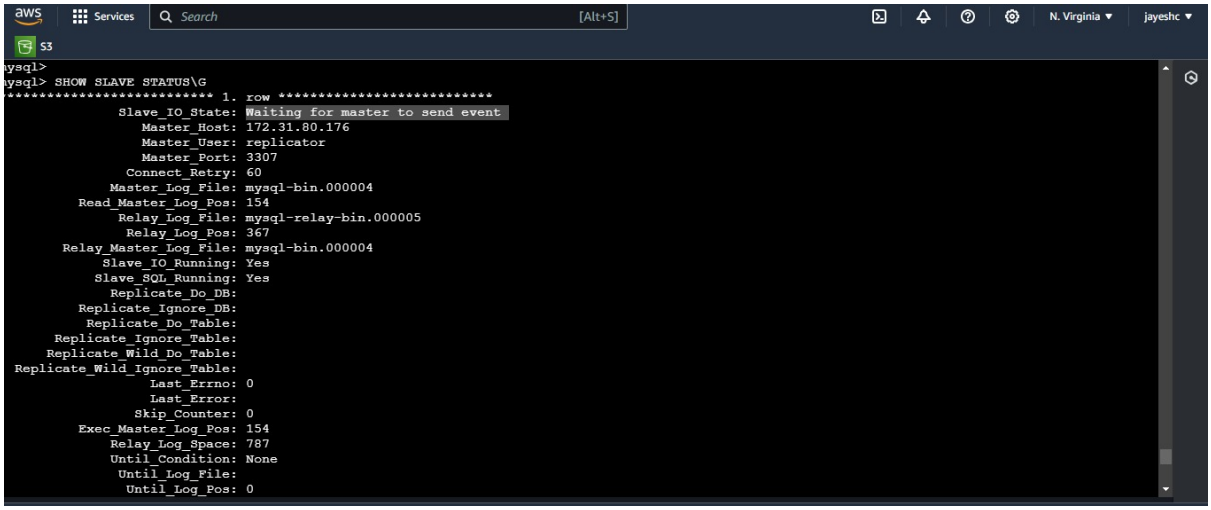

Figure 3: Master and Slave valid connection

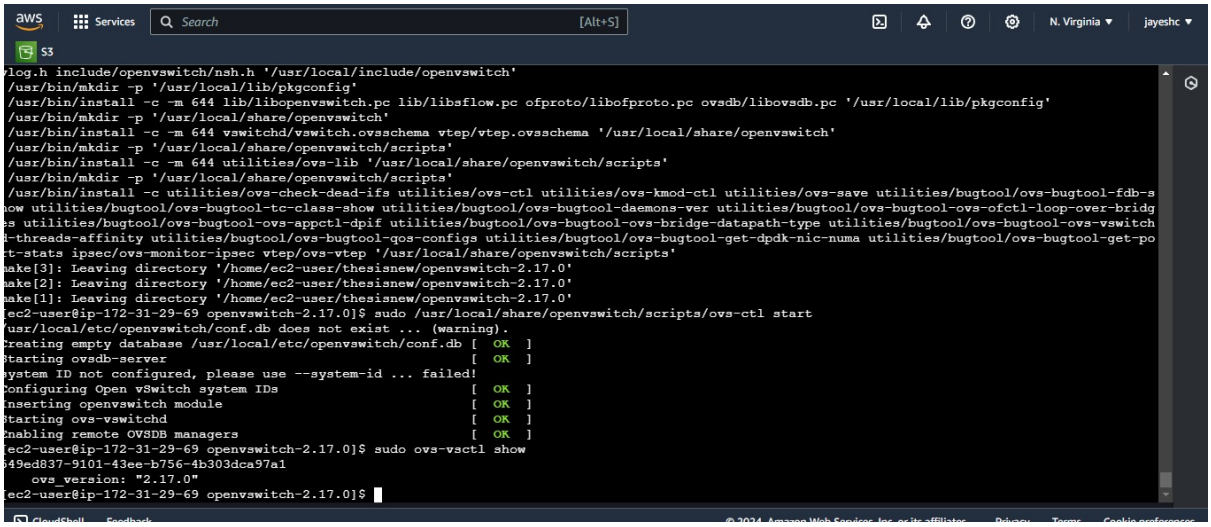

Figure 4: OVS installation

4 For Creation of NFV

For creation of NFV, virtualized network.

We create an EC2 instance for virtualized functions. And Navigate to Route tables. Here we use NFV instance IP on VxLAN master, so all packets route through NFV.

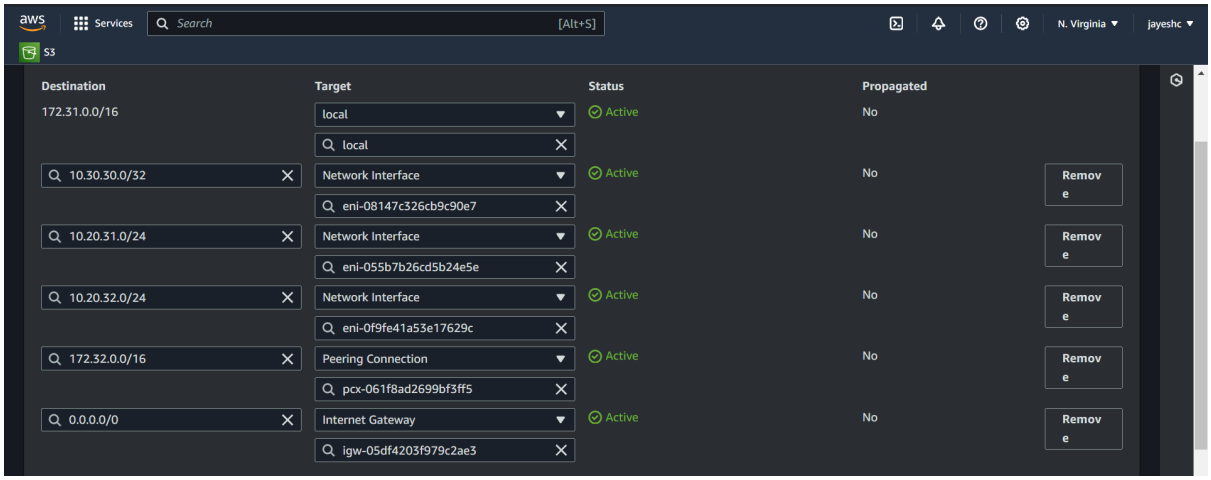

Figure 5: Routing Traffic to NFV

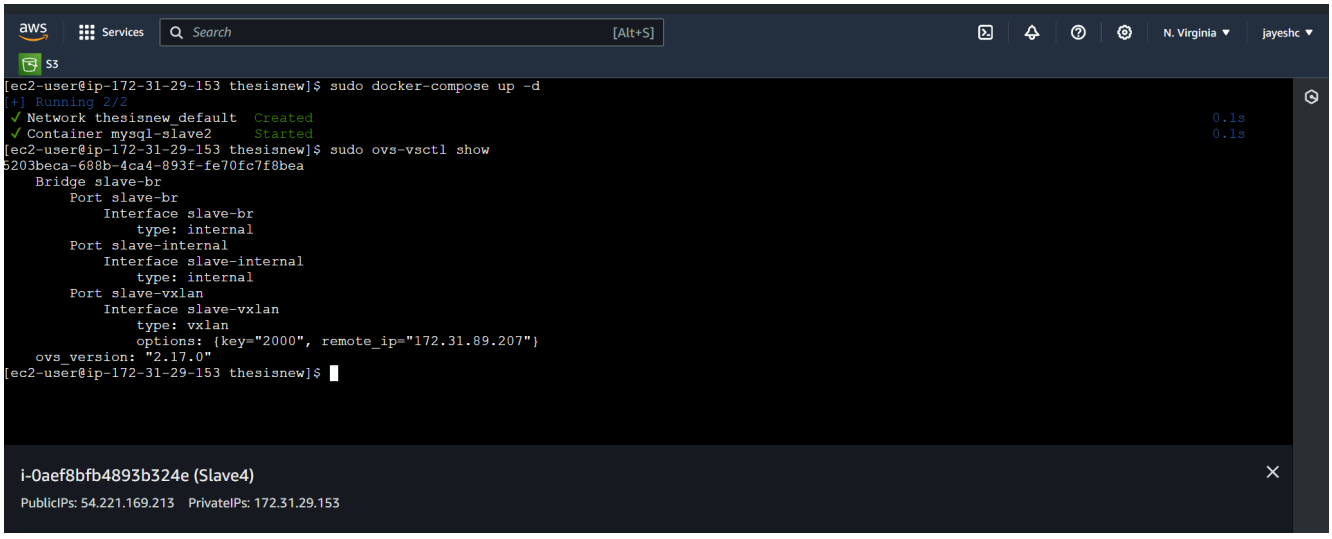

Figure 6: Connection at slave bridges

5 Deploying RDS Multi region and Availability Zone

```
docker exec -it mysql-slave2 mysql -uroot -prootpassword -e "
STOP SLAVE IO_THREAD FOR CHANNEL '';
CHANGE MASTER TO
    MASTER_HOST='10.20.31.100',
    MASTER_USER='replicator',
    MASTER_PASSWORD='passpass23',
    MASTER_PORT=3306,
    MASTER_LOG_FILE='mysql-bin.000001',
    MASTER_LOG_POS=154;
START SLAVE;"
```

Figure 7: Connection of RDS at Master Node

6 To check status of RDS database at Master node

```
-- Check Master status
docker exec -it mysql-master2 mysql -uroot -prootpassword -e "SHOW MASTER STATUS\G"
```

Figure 8: Connection of RDS at Master Node

If we want to "*Ping*" and check connection, then we need to get Master node Internal IP. As here we ping Slave from Master Node. To check Master-Internal IP we use *ifconfig* command. And note down Master IP.

7 To check status of RDS database at Master node

We use *tcpdump* command for analyzing packet traffic.

8 Creating AWS VPC's with Multi region and Multi-AZ

This infrastructure aims to design robust and scalable RDS setup here will achieve Multi region and Multi-AZ.

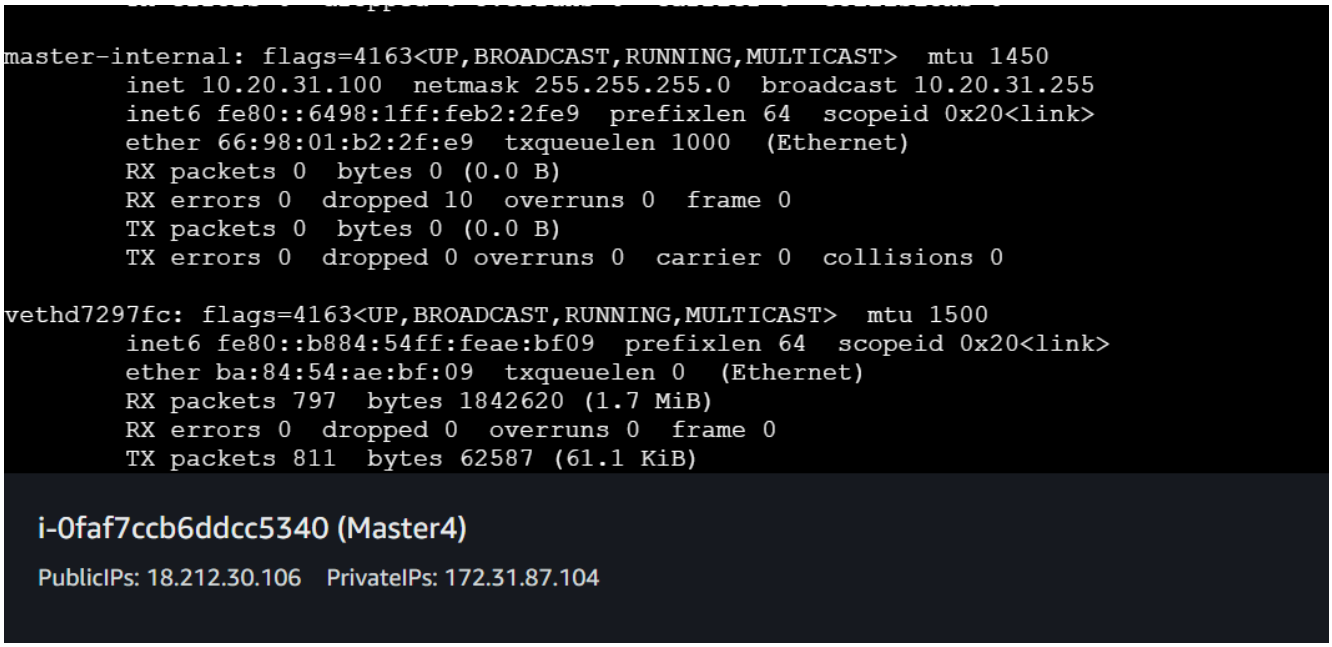

Figure 9: Note Master-Internal IP

```
-- Check Master status
docker exec -it mysql-master2 mysql -uroot -prootpassword -e "SHOW MASTER STATUS\G"
```

Figure 10: Connection of RDS at Master Node

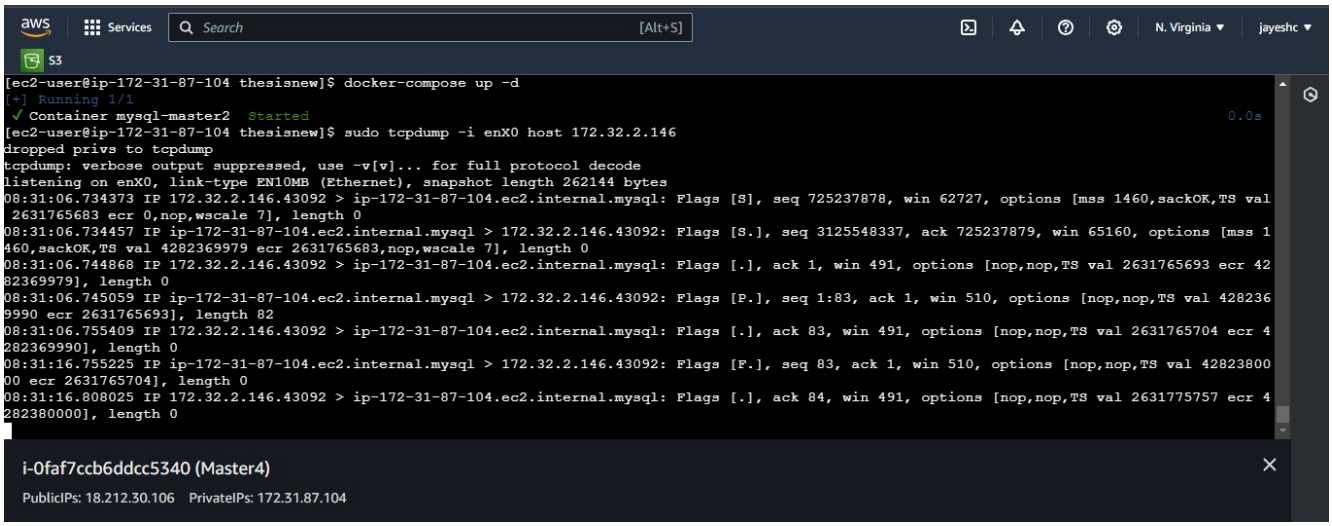

Figure 11: Packet Traffic in Master node

Step 1: Creating VPC with AWS VPC dashboard Here we assume IPv4 CIDR and Tenancy set as basic/default Launching VPC with 3 subnets

Step2: Subnet 1: MasterDB this needs to be a private subnet

Step3: Current availability zone can be US-East-1a

Step 4: And small range of IPv4 CIDR

Step 5 Subnet 2: slaveDB

step 6: Availability zone be US-EAST-1b

Step 7: And small range of IPv4 CIDR

Step 8: Public IP: (auto assigned)

Step 9: Subnet 3: (NFV Load Balancer): NFV-sLB

Step 10: Availability zone be US-EAST-1a And small range of IPv4 CIDR

Step 11: Public IP: (auto assigned)

Step 12: Configuring routing table to confirm MasterDB subnet

Step 13: Private routing table For masterDB subnet / no internet gateway association

Step 14: Public routing table For slave and NFV connected to internet gateway.

Step 15: SlaveDB will use LB's ip to connected masterDB

Step 16: Configure listeners to handle incoming packets and request from MasterDB

Step 17: Creating secondary VPC with one subnet Single subnet

Step 18: Availability zone be US-EAST-1a And small range of IPv4 CIDR

Step 19: Public IP: (auto assigned) connected to internet gateway VPC Peering

Step 20 Connection 2 VPC is established by Transit gateway which is also known as AWS Transit Gateway.

Step 21: So here we prove cross region connectivity. Setting up RDS instance

Step 22: MasterDB from first VPC on instance and hence subnet enabling Multi-AZ

Step 23: So slaveDB configure for read and mimic Setting cross region Cloning.

Step 24: As cross connectivity is ensured, MasterDB from first VPC and SingleDB from another VPC prove to achieve RDS Multi-region