Full Project Report:

Edge Computing as a Service

Team 6: Dhanraj Vedhanth, Ananya Nunna, Meenalatha Bhathula, Ashwin Comandur Sounderarajan

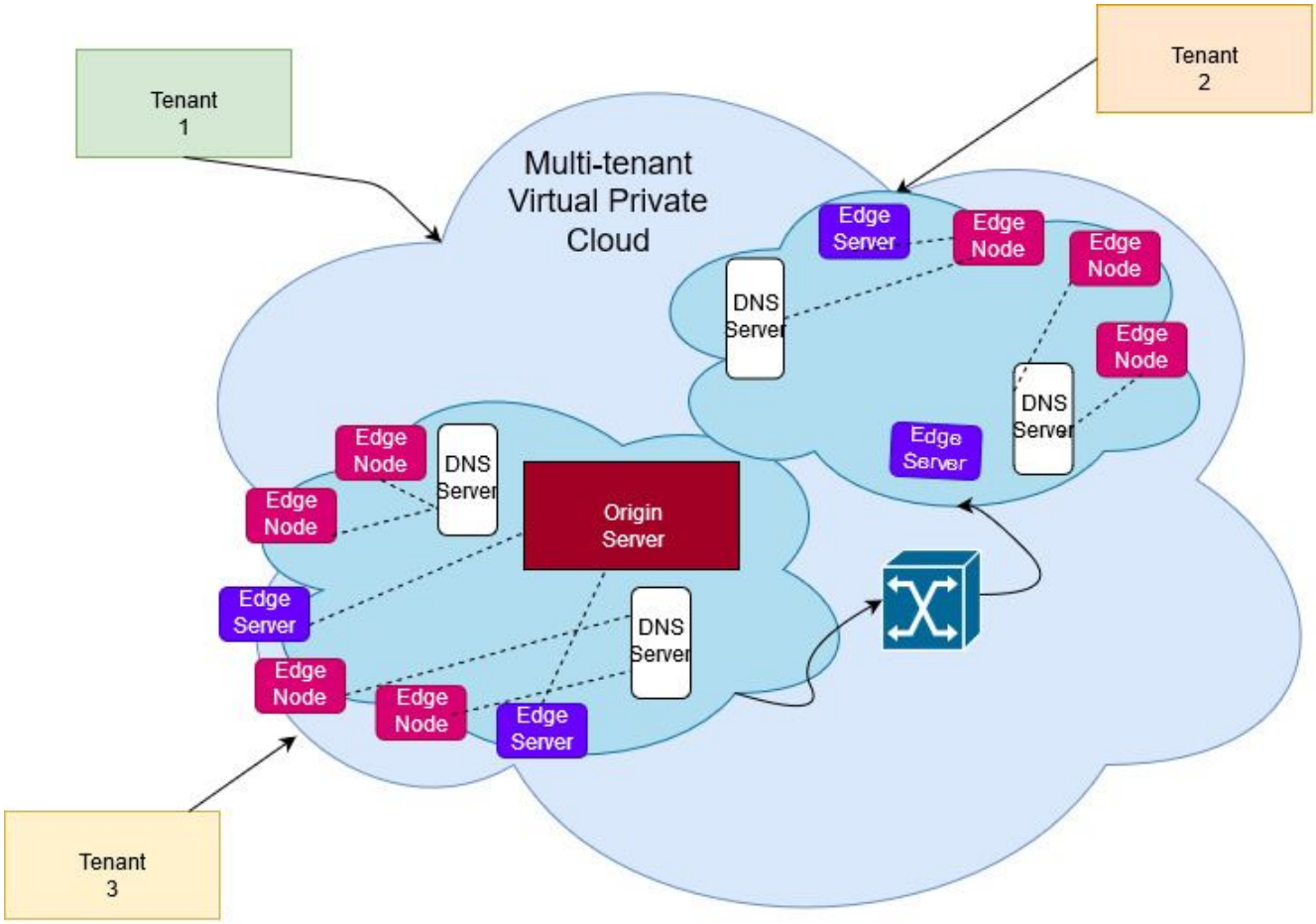
Contents:

Project Objective

The goal of our project is to implement edge services through content delivery networks with DNS resolution for a multi-tenant architecture. These terms today are widely used and often have very thin distinctions between them. In general, edge computing brings cloud resources such as compute speeds, storage and networking, closer to applications, devices or users. This architecture allows complex event processing to happen in a system closer to the client(at edge servers), by eliminating round-trip issues and enabling actions to happen quicker (by analyzing through metrics like Round trip delay time or RTT). While CDNs are a type of node servers (broadcast points of presence -POPs) that helps or mitigates the function of the root/main/origin server in a client-server model, by providing multi-layer cache services.

In our model, we have edge devices or nodes which are connected to edge servers, making up the CDN network that is offered to our customer the client. Hence the goal of our project is to create a multitenant architecture that is hosted by a number of VPCs. In order to find the closest CDN node (or edge server) to an edge client, we plan to use content based DNS routing to find the closest edge server with the relevant cached content. For example, Netflix is configured in such a way that, 3 http links are provided for every play of the video, to start storing video data stream into a buffer, which may vary in video resolution and data bit rate.

Model Draft of our Project output:



Related Work

#1 Akamai Edge Computing

Akamai Edge Computing

Akamai has application delivery networks that can accelerate entire web or IP-based applications, media delivery networks that provide HD-quality delivery of live and on-demand media, and EdgeComputing networks that deploy and execute entire Java J2EE applications in a distributed fashion. Akamai comprises of multiple delivery networks, each tailored to a different type of content such as static web content, streaming media, or dynamic applications. At a high level, these delivery networks share a similar architecture, but the underlying technology and implementation of each system component differs in order to best suit the specific type of content, streaming media, or application being delivered.

The following are the main components of the Akamai delivery system:

Mapping System: When the user types a URL into his/her browser, the domain name of the URL is translated by the mapping system into the IP address of an edge server to serve the content.

Edge Server Platform

A large global deployment of servers are located in thousands of sites around the world. These servers are responsible for processing requests from nearby users and serving the requested content.

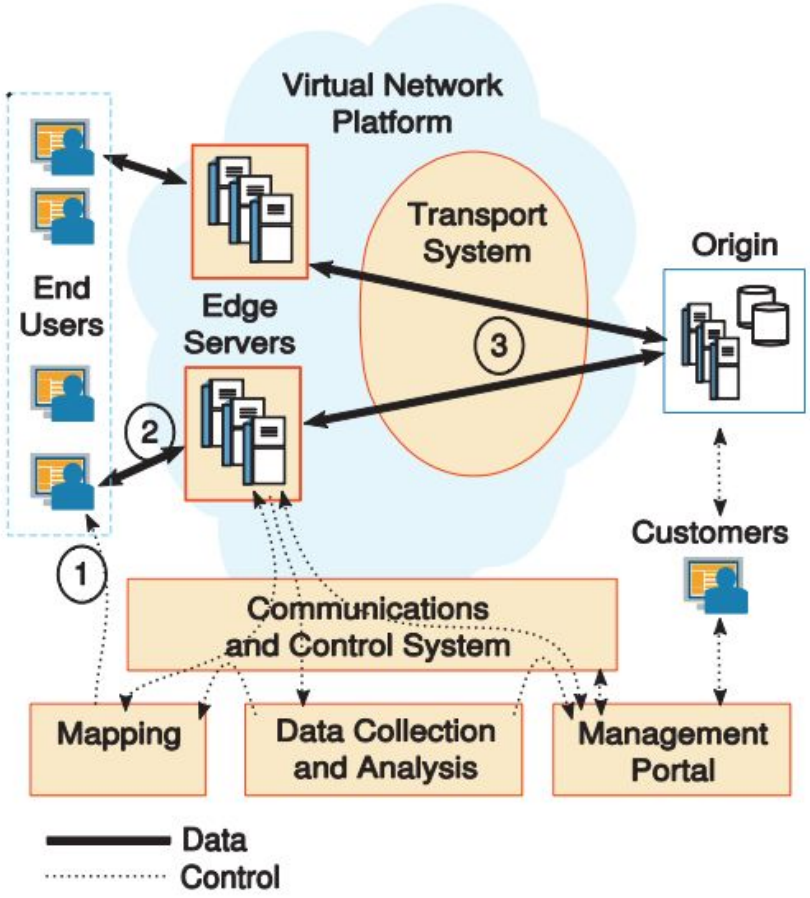
Origin Server: The origin server is used in the case of dynamic content. Caching mechanism cannot entirely work when content is customized for each user. Dynamic content needs to be fetched from the Origin server.

Transport System: The transport system is used to download the required data in a reliable and efficient manner. More generally, the transport system is responsible for moving data and content over the long-haul Internet with high reliability and performance.

Communications and Control System: Communications and Control System is used for disseminating status information, control messages, and configuration updates in a fault-tolerant and timely fashion

Data Collection and Analysis System: The data collection and analysis system is responsible for collecting and processing data from various sources such as server logs, client logs, and network and server information. The collected data can be used for monitoring, alerting, analytics, reporting, and billing.

Management Portal: A configuration management platform allows an enterprise customer to retain fine-grained control how their content and applications are served to the end user. Configurations are updated across the edge platform from the management portal via the communications and control system. In addition, the management portal provides the enterprise with visibility on how their users are interacting with their applications and content, including reports on audience demographics and traffic metrics.



#2 Amazon Cloudfront Edge Computing

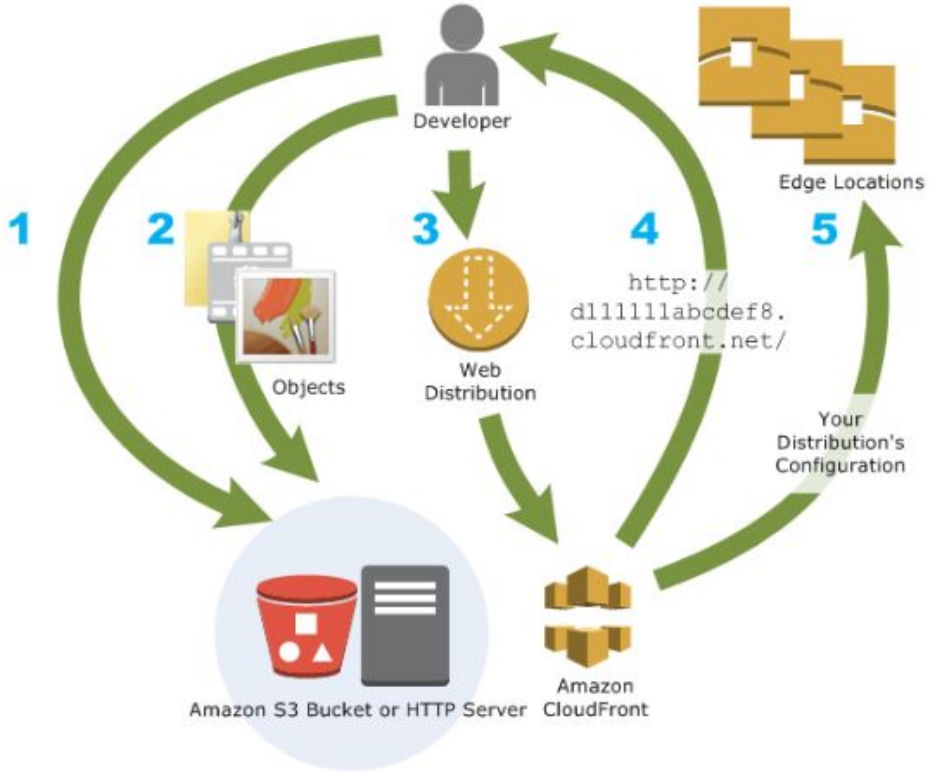
Amazon Cloudfront: Amazon CloudFront is a web service that speeds up distribution of static and dynamic web content, such as .html, .css, .js, and image files. CloudFront distribution tells CloudFront where content is to be delivered from, and the details about how to track and manage content delivery. Then CloudFront uses computer edge servers that are close to the viewers to deliver the content quickly when someone wants to see it or use it.

Origin Servers: An origin server stores the original, definitive version of the objects. If content is being served over HTTP, your origin server is either an Amazon S3 bucket or an HTTP server, such as a web server. The HTTP server can run on an Amazon Elastic Compute Cloud (Amazon EC2) instance or on a server that the user can manage; these servers are also known as custom origins.

Cloudfront Distribution: CloudFront distribution tells CloudFront which origin servers to get files from when users request the files through the user's web site or application. At the same time, the user can specify details such as whether CloudFront should log all requests and whether the distribution is to be enabled as soon as it's created.

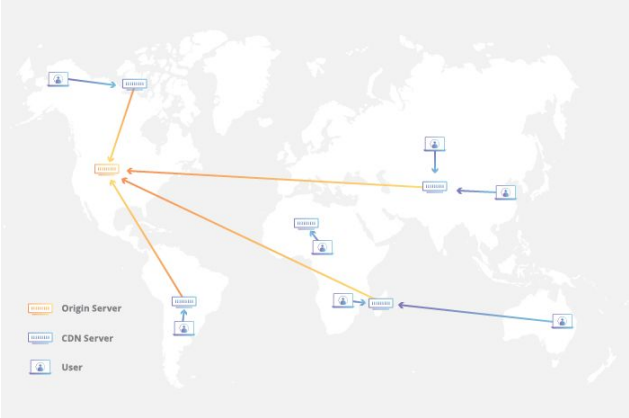
Management: CloudFront assigns a domain name to the new distribution that can be seen in the CloudFront console, or that is returned in the response to a programmatic request, for example, an API request. An alternate domain name can be used instead.

Points of Presence: CloudFront sends distribution's configuration to all of its edge locations or points of presence (POPs); collections of servers in geographically-dispersed data centers where CloudFront caches copies of your files.



#3 Cloudflare Edge Computing

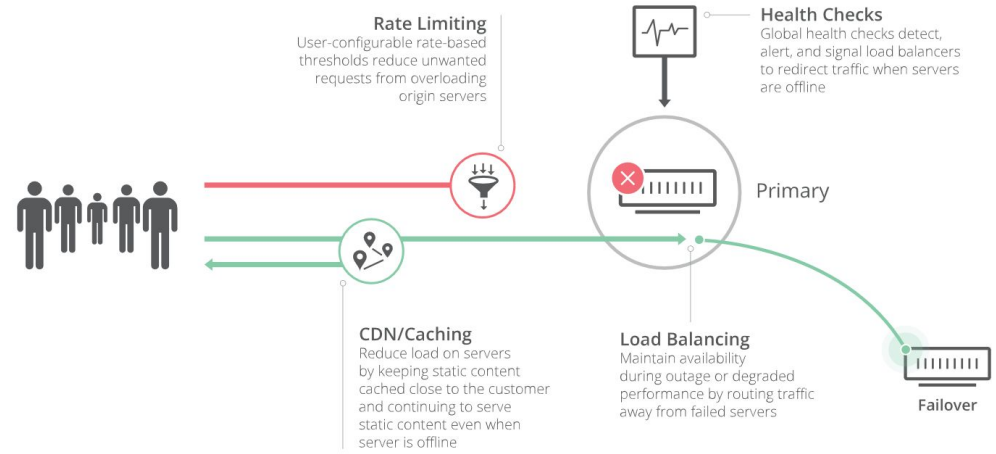
Cloudflare is an American web infrastructure and security company, mainly providing content delivery network services along with DDoS mitigation, Internet security, caching, routing, load balancing, DDoS mitigation, and distributed DNS services. They have many CDN- datacenters across the globe, that host their clients with the motto of making all things hardware as online.



Some of their services that are of interest to us:

(1) Availability as a service while hosting web-pages

They have fail-safe operations when the origin server faces unexpected failures by redirecting traffic to healthy origins.



(2) CDN as a service

Their CDNs sits on the network between end-user web browsers and website origin servers.
When traffic goes from the web browser to Cloudflare, it fulfills the request from cache whenever or otherwise, it goes back to the origin web server in a second connection. They allow the customers to customise the CDNs by allowing them to choose how each individual URLs would get cached and for how long. In order to enhance performances of shorter video start up times, they provide the following

packaged functionalities: SSL/TLS encryption, cache configuration, tiered caching and DDoS Protection. CDNs use reverse proxy rather than a forward proxy to communicate with the origin server.
Hence the key components involved in their system:

(a) Anycast: Anycast routing is used to route the incoming requests to the nearest datacenter or a variety of different location or nodes. On the same datacenters they also provide DNS as a service with enhanced security features.

(b) Internet Exchange points: An Internet exchange point(IXP) is a physical location through which internet companies such as ISPs and CDNs connect with each other. They exist on the edge of different networks to allow transit between other networks for network providers.

(c) Datacenters: 115 PoPs and 6 million domains

(d) Argo Smart Routing and Tiered Cache: Argo analyzes and optimizes routing decisions to manage web traffic along the most reliable network paths. Argo Smart Routing uses latency and packet loss data collected from each request to pick optimal paths across the Internet. If in case some data requested is not in cache, the CDN must connect to the origin server. Argo Tiered Cache lowers this origin load, increases cache hit ratios, and improves end user experience by first asking other Cloudflare PoPs if they have the requested content when a cache miss occurs. This results improve performance for visitors, as links traversed between Cloudflare PoPs are generally shorter and faster than links between PoPs and origins.

Milestone 2 Progress Log

We have completed creating our automated Virtual Private Cloud environment. Given below in Section 1, is the description of our VPC model and Section 2 describes the Management feature and Section 3 is the Functional Feature that we implemented.

Section 1 VPC

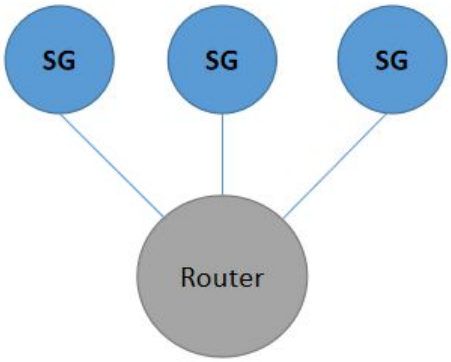
Flow of VPC Environment:

First we take as **inputs from each customer**, ie. one single tenant, the following details :

1) The number of subnets they require:
    a) The Subnet Id (asked on loop depending on the number of subnets they require)
    b) The number of VMs (asked on loop for every subnet Id)
    c) How many VCPUs do you want?
2) Do you want an Internet connection?

Ex: 3 subnets were asked.

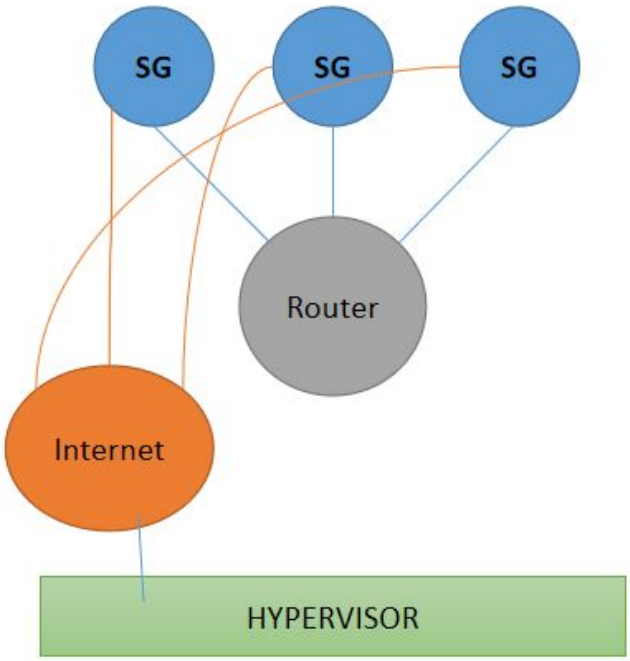
The Python code is automated to generate the following architecture first:



Here the SG and Routers are namespaces, the SG namespace acts as the default gateway for every subnet ID specified by the tenant. The router allows inter-subnet communication within a single tenant. It has routes configured to every subnet from one another.

Naming Convention:
Tenant<tenantno>_SG_<subnetno>
Router<tenantno>

Following this, the tenant is also asked if they require Internet connection, upon which a namespace called Internet is created, which enables internet connectivity for all the subnet gateways of the tenant.



This Internet Namespace is connected to the Hypervisor, hosting the VPC.

Naming Convention:
NS_int<>

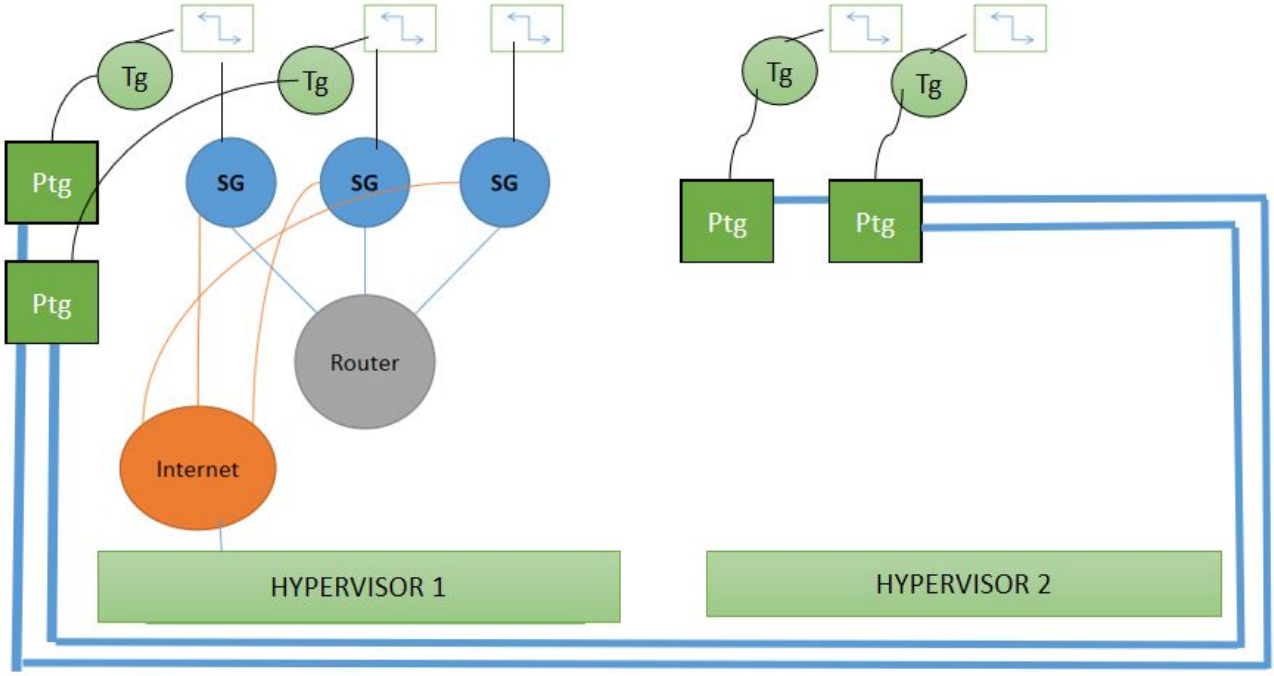
We have an If condition, where, if the number of VMs per subnet is greater than one, then it creates the other VM on another Hypervisor VM which balances the load.

After this step, the Tunnel Gateway namespaces, and OVS bridges for each subnet are created.

For the case of #subnet 1- 3Vms, subnet 2- 2Vms, subnet 3- 1 Vm we see the following two figures depicting the steps toward creating the complete architecture of the VPC.

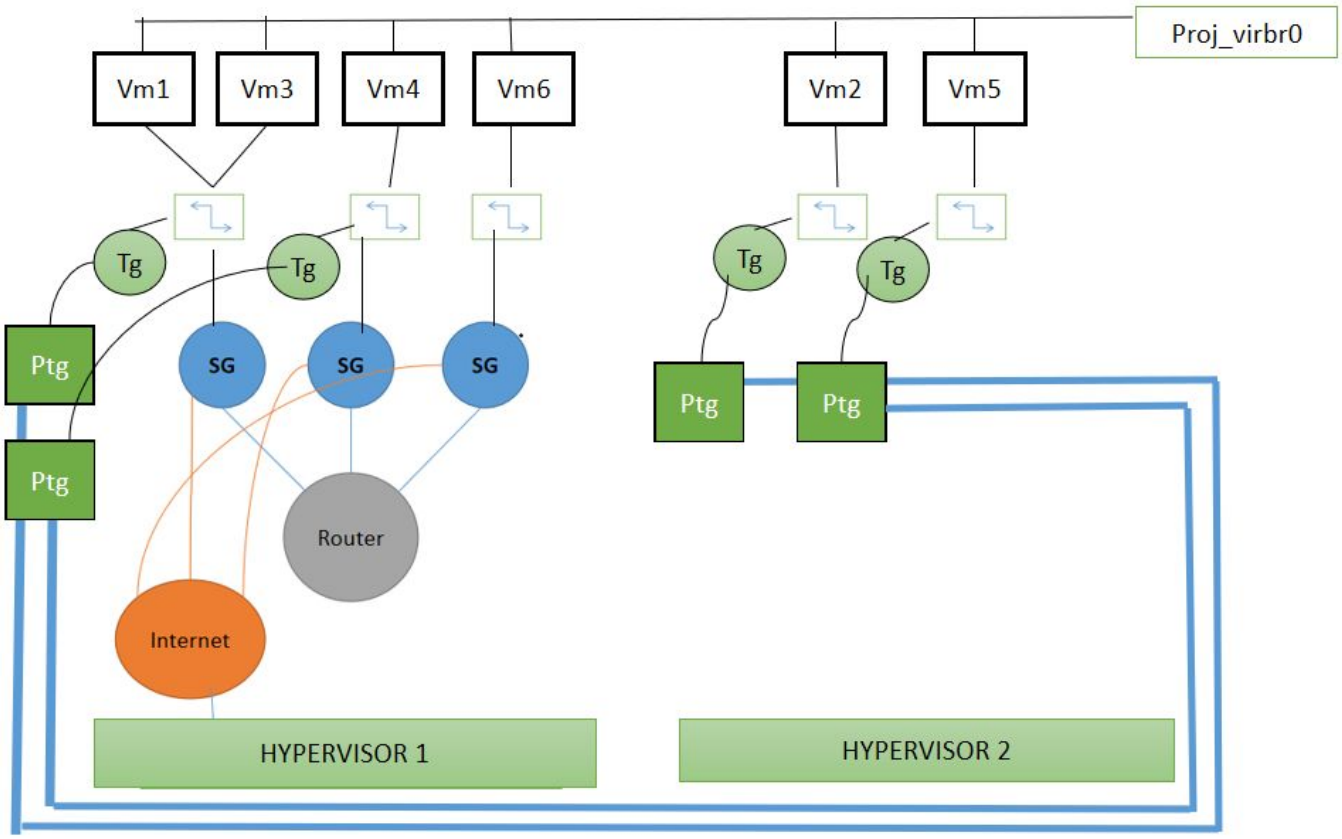
Naming Convention:
Tg: NS_tenant<tenantno>_<subnetno>
Ptg: NS_proj<tenantno>

The below figure depicts the architecture generated by us when you run the script for a single tenant



with the mentioned specifications. As you can see, the Vms generated are highlighted by the black blocks, and all VMs connected to the same subnet under the same hypervisor are connected to the same OVS bridge. Following which a default tunnel gateway namespace is created only to those switches carrying subnets that extend to another hypervisor. The blue dashed line in the image indicates the VXLAN tunnel.

In the back end, these VMs are created using virsh define commands with a default virtual machine CentOS images that already have pre installed collectd and python packages. It is connected to a new default network called the proj_default which acts as the management interface through which we ssh into.



Datapath:

1. Inter subnet communication on the same hypervisor:
    a. Source: L3 forwarding(VM 1)- L2 forwarding(OVS switch)- Destination: L3 forwarding(VM3) (for example in the above case)
    b. The L2 switch is used to pass traffic between the two VMs.
2. Inter Subnet communication on different hypervisor:
    a. Source: L3 forwarding(VM1)- L2 forwarding (OVS switch)- VXLAN TUNNEL(through Tunnel gateway(Tg)) - L2 forwarding- Destination: L3 forwarding(VM 2)
    b. So the entire topology is seen as an L2 Lan network communication
3. One subnet to another Subnet (on the same hypervisor):
    a. Source: L3 forwarding(VM1)-L2 forwarding (OVS bridge)-L3 forwarding (Subnet gateway 1)- L3 forwarding(Router)- L3 forwarding(Another subnet gateway 2)- L2 forwarding (OVS switch)-Destination: L3 forwarding (VM4)
    b. Router allows each tenant to have their subnets communicate with each other.
4. One subnet to another subnet (on different hypervisor):
    a. Source: L3 forwarding (VM5) - Tunnel Gateway(Tg) VXLAN Tunnel - L2 forwarding (OVS switch)- L3 forwarding (Router)- L2 forwarding (OVS switch)- L3 forwarding (VM 3)
    b. So every VM on the hypervisor 2 uses VXLAN tunnel to communicate with any other Vm irrespective of its own subnet or a different subnet
5. One subnet to another subnet (on Hypervisor 2):
    a. Similar to the previous case:
    b. Source: L3 forwarding (VM5) - Tunnel Gateway(Tg) VXLAN Tunnel - L2 forwarding (OVS switch)- L3 forwarding (Router)- L2 forwarding (OVS switch)- L2 Tunnel gateway VXLAN TUNNEL- Destination: L3 forwarding (VM 2)
6. Any Vm to internet:

a. Source: L3 forwarding(VM1)- L2 forwarding (OVS Switch) - L3 forwarding (SG) -L3 forwarding (Internet namespace)- L3 forwarding (proj_virbro)-management interface- Public network
b. If the VM is on Hypervisor 2, then the VM goes till the OVS switch of its subnet on hypervisor 1 and follows the same path as mentioned above.

IP Address assignment Mechanism:

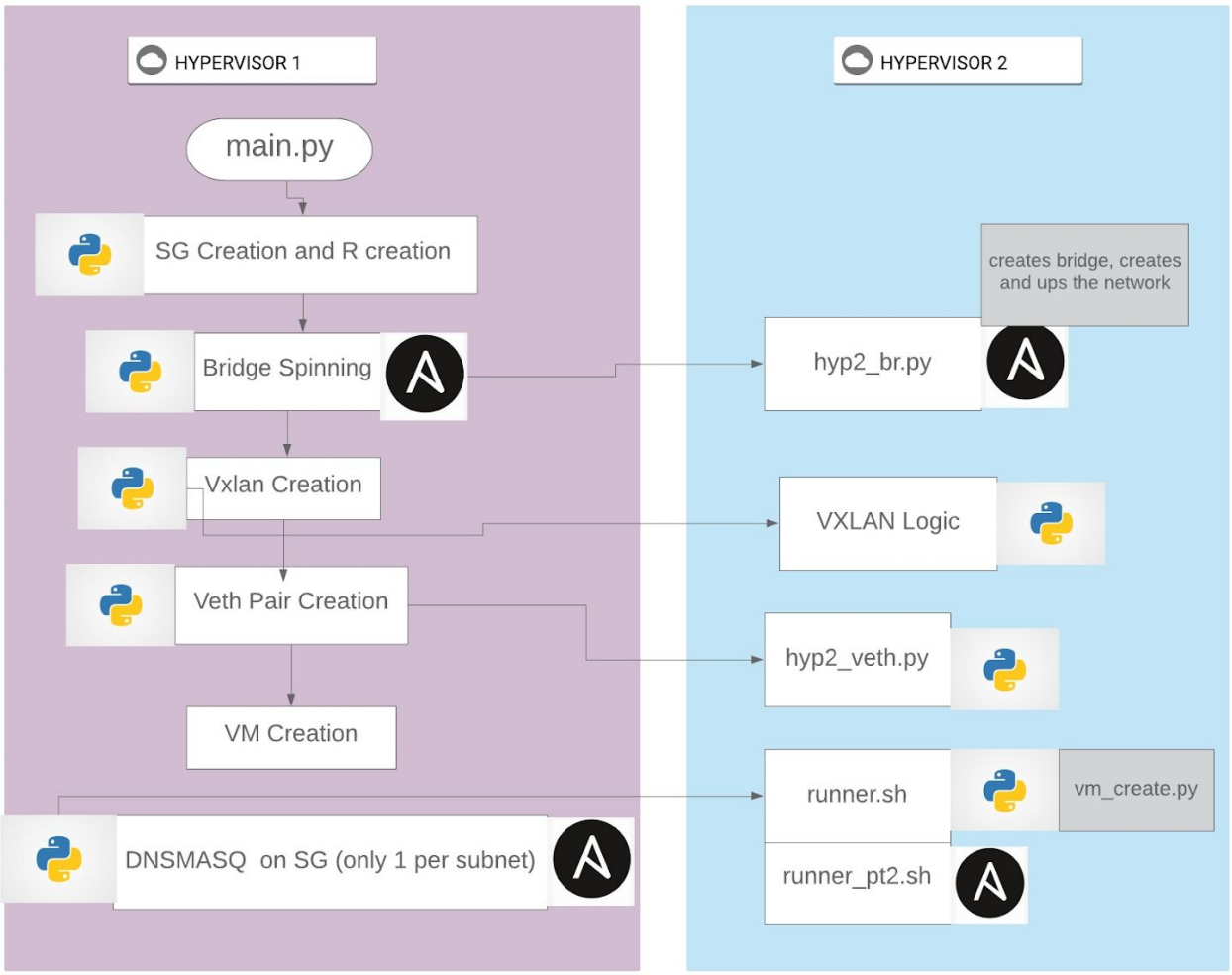
1. The subnets given by the customer are assigned to all interfaces from the VMs till the subnet gateway. However we do place a hard limitation where the customer cannot give away the 185.0.0.0 subnet as it is used by the designer, to make the connections for the VXLAN tunnel and the internet data path.
2. For every LAN, by splitting the load across two hypervisors, we assign every consecutive IP address alternatively to VMs on both the hypervisors consequently giving Even or Odd addresses one each hypervisor.
3. For our functional features, our origin server which is the hypervisor 1 is given the interface IP of 24.16.10.8/24 (randomly chosen to be used).

Southbound and Northbound Interfaces:

The Northbound interface for the customer in our case is the interface on the subnet gateway that connects to the OVS bridge. This is assigned with the first address of the usable range in which the tenant as asked for (Example: 10.0.0.0- northbound interface : 10.0.0.1). The southbound interface would be the interface of every subnet gateway that connects to the router. This is assigned by the designer with /30 subnet mask in the reserved subnet ID that is 185.0.0.0/8.

How did we make it?

Our ENTIRE VPC is Automated!



We use Python and Ansible at different parts of the VPC automation to create the model.
The **main.py** in our VPC directory is the python script used to make our model. We use Classes and global file pointers to create a structure for the code.
Its flow of automating can be divided into 3 parts.

Part 1:
Tool: Python script
After inputs are taken in at the main.yml, the **TenantL3comm.py** is called by the main.py, which creates the subnet gateway and router namespaces. This is created on only one hypervisor. This is because no load balancing is required as namespaces take up insignificant amount of CPU Usage.

Part 2a:
Tool: Python script
The script **vxlan_Create.py** is used to attach every switch to a tunnel gateway(Tg) and a path tenant gateway(Ptg) namespace and corresponding veth pairs for every subnet. Once the tunnel starts setting up in the hypervisor 1, where the main.py function runs, paramiko is used to do the same using the called function on the other hypervisor for all corresponding VMs in the other hypervisor.

Part 2b:
This part runs two parts of the architecture, which includes OVS(L2) bridge creation (one for each subnet) and VXLAN tunneling.
Tool: Ansible - roles
The logic behind creating bridges:

- If number of VMs is lesser than 1 for each subnet ?:
    - The bridge is made only in the hypervisor 1

Functional block of Ansible:
****
The **/vpc/ansible** has the following yml files used to run our roles script. The bridge_main.yml uses the 2 roles: **/test_role** and **/net_define**.

Task1 - Creates the bridge
[The /test_role runs tasks with **/tasks/main.yml** by using the template stored in **ovs_l2.xml.j2** and it edits this file each time it has to create the bridge and the network by using the main.yml in vars file (which has all the variables that must be modified in the xml file each time a new bridge is made for a particular subnet)]

Task 2- To define and start the network file
[The /vpc/ansible has the following yml files that runs roles- /net_define which runs the **/task/mail.yml** and this uses the **/vars/mail.yml** in vars to edit the standard configuration of xml file each time a network has to be made for a particular subnet]

Task 3- Adding Veth Pairs
The main.py is the main function that runs on the hypervisor 1 and thus takes care of the job of connecting each switch to the Subnet Gateway and the Tunnel Gateway.
****

- If number of VMs are more than one in each subnet ?:
    - We use netmiko to call an ansible function saved in the other hypervisor to create the bridge

Functional block of Ansible:
****
Even in the hypervisor 2: **/vpc/ansible** has the following yml files used to run our roles script. The bridge_main.yml uses the 2 roles: **/test_role** and **/net_define**.

Task1 - Creates the bridge
[The hyp2_br.py calls the bridge_main.yml which creates the bridges, creates the network and brings up the network]

Task 2- To add Veth Paris
[The hyp2_veth.py runs the function of adding veth pairs]
****
Part 3:
Tool: Ansible , Python, Bash script

The final part sets up VM installation configuration files and runs the dnsmasq function. The inputs to this are the number of VMs, subnet number, the default gateway, dhcp1 dhcp2 and cidr. The img_to_cpy.img is used as a standard image of the VM that we replicate to our customers. It consists of collectd and python packages installed.

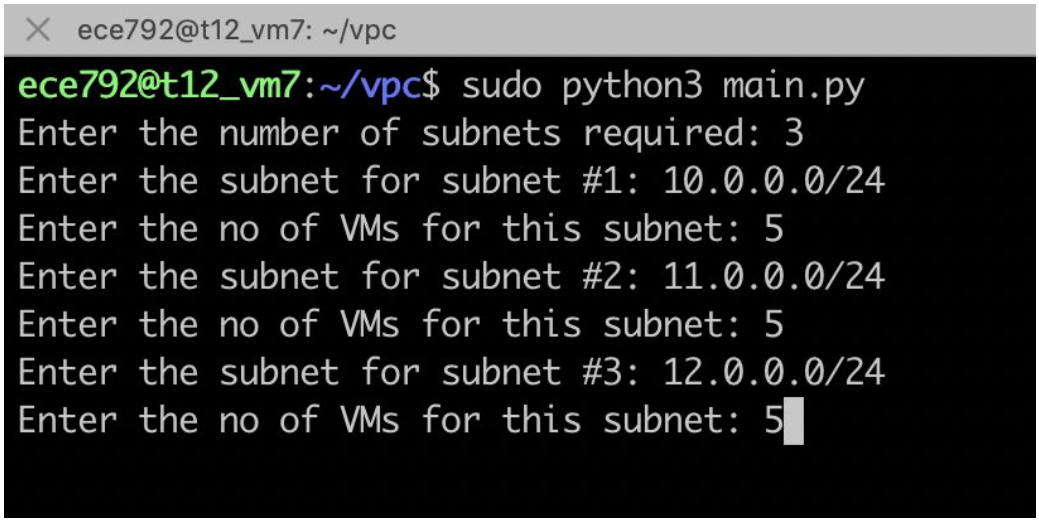
Functional block of Ansible:

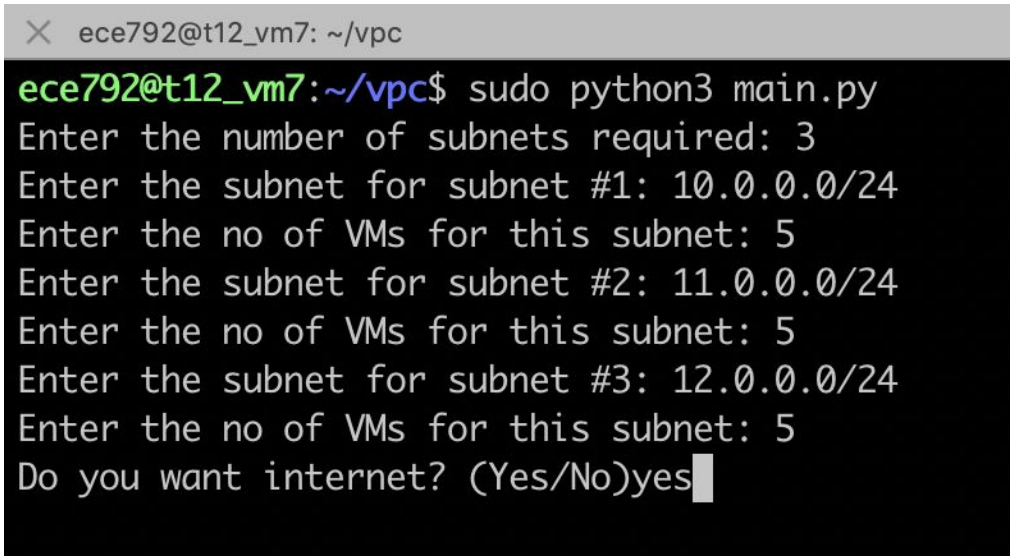
****

The runner.sh calls vm_create.py which updates the var files of the corresponding ansible roles which is gen_vm. Next the runner_pt2.sh runs the ansible playbook vm_create.yml.

****

SCREENSHOTS OF THE OUTPUT:





```
ece792@t12_vm7: ~/vpc
Enter the number of subnets required: 3
Enter the subnet for subnet #1: 10.0.0.0/24
Enter the no of VMs for this subnet: 5
Enter the subnet for subnet #2: 11.0.0.0/24
Enter the no of VMs for this subnet: 5
Enter the subnet for subnet #3: 12.0.0.0/24
Enter the no of VMs for this subnet: 5
Do you want internet? (Yes/No)yes
{'10.0.0.0/24': '5', '11.0.0.0/24': '5', '12.0.0.0/24': '5'}
ip netns add Router3
ip netns add Tenant3_SG_1
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.148/30
Broadcast ID -> 185.0.2.151
185.0.2.149
185.0.2.150


[['185.0.2.149', '185.0.2.150']]
[['185.0.2.149', '185.0.2.150']]
ip link add ns_proj type veth peer name Tenant3_SG_1
ip link set ns_proj netns Tenant3_SG_1
ip link set Tenant3_SG_1 netns proj_internet
ip netns exec Tenant3_SG_1 ip link set ns_proj up
ip netns exec proj_internet ip link set Tenant3_SG_1 up
ip netns exec Tenant3_SG_1 ip addr add 185.0.2.150/30 dev ns_proj
ip netns exec proj_internet ip addr add 185.0.2.149/30 dev Tenant3_SG_1
ip netns exec Tenant3_SG_1 ip route add default via 185.0.2.149
ip netns exec Tenant3_SG_1 iptables -t nat -A POSTROUTING -s 0.0.0.0/0 -j MASQUERADE -o ns_proj
Tenant3_SG_1
yes
Router3
10.0.0.0/24
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.152/30
Broadcast ID -> 185.0.2.155
185.0.2.153
185.0.2.154


[['185.0.2.153', '185.0.2.154']]
[['185.0.2.153', '185.0.2.154']]
ip link add ns_proj1 type veth peer name Tenant3_SG_1
ip link set ns_proj1 netns Tenant3_SG_1
ip link set Tenant3_SG_1 netns Router3
ip netns exec Tenant3_SG_1 ip link set ns_proj1 up
ip netns exec Router3 ip link set Tenant3_SG_1 up
ip netns exec Tenant3_SG_1 ip addr add 185.0.2.154/30 dev ns_proj1
```

```
ece792@t12_vm7: ~/vpc
[['185.0.2.157', '185.0.2.158']]
[['185.0.2.157', '185.0.2.158']]
ip link add ns_proj type veth peer name Tenant3_SG_2
ip link set ns_proj netns Tenant3_SG_2
ip link set Tenant3_SG_2 netns proj_internet
ip netns exec Tenant3_SG_2 ip link set ns_proj up
ip netns exec proj_internet ip link set Tenant3_SG_2 up
ip netns exec Tenant3_SG_2 ip addr add 185.0.2.158/30 dev ns_proj
ip netns exec proj_internet ip addr add 185.0.2.157/30 dev Tenant3_SG_2
ip netns exec Tenant3_SG_2 ip route add default via 185.0.2.157
ip netns exec Tenant3_SG_2 iptables -t nat -A POSTROUTING -s 0.0.0.0/0 -j MASQUERADE -o ns_proj
Tenant3_SG_2
yes
Router3
11.0.0.0/24
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.160/30
Broadcast ID -> 185.0.2.163
185.0.2.161
185.0.2.162


[['185.0.2.161', '185.0.2.162']]
[['185.0.2.161', '185.0.2.162']]
ip link add ns_proj1 type veth peer name Tenant3_SG_2
ip link set ns_proj1 netns Tenant3_SG_2
ip link set Tenant3_SG_2 netns Router3
ip netns exec Tenant3_SG_2 ip link set ns_proj1 up
ip netns exec Router3 ip link set Tenant3_SG_2 up
ip netns exec Tenant3_SG_2 ip addr add 185.0.2.162/30 dev ns_proj1
ip netns exec Router3 ip addr add 185.0.2.161/30 dev Tenant3_SG_2
ip netns exec Router3 ip route add 11.0.0.0/24 via 185.0.2.162
ip netns exec Tenant3_SG_2 ip route add 10.0.0.0/24 via 185.0.2.161
ip netns exec Tenant3_SG_2 ip route add 12.0.0.0/24 via 185.0.2.161
ip netns add Tenant3_SG_3
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.164/30
Broadcast ID -> 185.0.2.167
185.0.2.165
185.0.2.166


[['185.0.2.165', '185.0.2.166']]
[['185.0.2.165', '185.0.2.166']]
ip link add ns_proj type veth peer name Tenant3_SG_3
ip link set ns_proj netns Tenant3_SG_3
ip link set Tenant3_SG_3 netns proj_internet
ip netns exec Tenant3_SG_3 ip link set ns_proj up
```

```
ece792@tt12_vm7: ~/vpc
185.0.2.166


[['185.0.2.165', '185.0.2.166']]
[['185.0.2.165', '185.0.2.166']]
ip link add ns_proj type veth peer name Tenant3_SG_3
ip link set ns_proj netns Tenant3_SG_3
ip link set Tenant3_SG_3 netns proj_internet
ip netns exec Tenant3_SG_3 ip link set ns_proj up
ip netns exec proj_internet ip link set Tenant3_SG_3 up
ip netns exec Tenant3_SG_3 ip addr add 185.0.2.166/30 dev ns_proj
ip netns exec proj_internet ip addr add 185.0.2.165/30 dev Tenant3_SG_3
ip netns exec Tenant3_SG_3 ip route add default via 185.0.2.165
ip netns exec Tenant3_SG_3 iptables -t nat -A POSTROUTING -s 0.0.0.0/0 -j MASQUERADE -o ns_proj
Tenant3_SG_3
yes
Router3
12.0.0.0/24
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.168/30
Broadcast ID -> 185.0.2.171
185.0.2.169
185.0.2.170


[['185.0.2.169', '185.0.2.170']]
[['185.0.2.169', '185.0.2.170']]
ip link add ns_proj1 type veth peer name Tenant3_SG_3
ip link set ns_proj1 netns Tenant3_SG_3
ip link set Tenant3_SG_3 netns Router3
ip netns exec Tenant3_SG_3 ip link set ns_proj1 up
ip netns exec Router3 ip link set Tenant3_SG_3 up
ip netns exec Tenant3_SG_3 ip addr add 185.0.2.170/30 dev ns_proj1
ip netns exec Router3 ip addr add 185.0.2.169/30 dev Tenant3_SG_3
ip netns exec Router3 ip route add 12.0.0.0/24 via 185.0.2.170
ip netns exec Tenant3_SG_3 ip route add 10.0.0.0/24 via 185.0.2.169
ip netns exec Tenant3_SG_3 ip route add 11.0.0.0/24 via 185.0.2.169
10.0.0.0/24 5
10.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  10.0.0.1
DHCP range: 10.0.0.2 10.0.0.253
10.0.0.1 10.0.0.2 10.0.0.253 10.0.0.254 24
ip netns add NS_tenant3_1
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_1

Running the playbook for bridge creation
```

```
ece792@tt12_vm7: ~/vpc
ip netns exec Tenant3_SG_3 iptables -t nat -A POSTROUTING -s 0.0.0.0/0 -j MASQUERADE -o ns_proj
Tenant3_SG_3
yes
Router3
12.0.0.0/24
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.2.168/30
Broadcast ID -> 185.0.2.171
185.0.2.169
185.0.2.170


[['185.0.2.169', '185.0.2.170']]
[['185.0.2.169', '185.0.2.170']]
ip link add ns_proj1 type veth peer name Tenant3_SG_3
ip link set ns_proj1 netns Tenant3_SG_3
ip link set Tenant3_SG_3 netns Router3
ip netns exec Tenant3_SG_3 ip link set ns_proj1 up
ip netns exec Router3 ip link set Tenant3_SG_3 up
ip netns exec Tenant3_SG_3 ip addr add 185.0.2.170/30 dev ns_proj1
ip netns exec Router3 ip addr add 185.0.2.169/30 dev Tenant3_SG_3
ip netns exec Router3 ip route add 12.0.0.0/24 via 185.0.2.170
ip netns exec Tenant3_SG_3 ip route add 10.0.0.0/24 via 185.0.2.169
ip netns exec Tenant3_SG_3 ip route add 11.0.0.0/24 via 185.0.2.169
10.0.0.0/24 5
10.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  10.0.0.1
DHCP range: 10.0.0.2 10.0.0.253
10.0.0.1 10.0.0.2 10.0.0.253 10.0.0.254 24
ip netns add NS_tenant3_1
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_1

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ***********************************************************************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] ****************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo


TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] *******************************************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)
```

```
ece792@tt2_vm7: ~/vpc
ip netns exec Tenant3_SG_3 ip addr add 185.0.2.170/30 dev ns_proj1
ip netns exec Router3 ip addr add 185.0.2.169/30 dev Tenant3_SG_3
ip netns exec Router3 ip route add 12.0.0.0/24 via 185.0.2.170
ip netns exec Tenant3_SG_3 ip route add 10.0.0.0/24 via 185.0.2.169
ip netns exec Tenant3_SG_3 ip route add 11.0.0.0/24 via 185.0.2.169
10.0.0.0/24 5
10.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  10.0.0.1
DHCP range: 10.0.0.2 10.0.0.253
10.0.0.1 10.0.0.2 10.0.0.253 10.0.0.254 24
ip netns add NS_tenant3_1
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_1

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] *****************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo


TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ************************************************************************************************************************************
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_1 t3_1
```


```
ece792@tt2_vm7: ~/vpc
ip netns exec Tenant3_SG_3 ip addr add 185.0.2.170/30 dev ns_proj1
ip netns exec Router3 ip addr add 185.0.2.169/30 dev Tenant3_SG_3
ip netns exec Router3 ip route add 12.0.0.0/24 via 185.0.2.170
ip netns exec Tenant3_SG_3 ip route add 10.0.0.0/24 via 185.0.2.169
ip netns exec Tenant3_SG_3 ip route add 11.0.0.0/24 via 185.0.2.169
10.0.0.0/24 5
10.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  10.0.0.1
DHCP range: 10.0.0.2 10.0.0.253
10.0.0.1 10.0.0.2 10.0.0.253 10.0.0.254 24
ip netns add NS_tenant3_1
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_1

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] *****************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo


TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ************************************************************************************************************************************
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_1 t3_1
```


```
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_1)

PLAY RECAP ********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_1 t3_1
/home/ece792
(3, 1)
3
ovs_l2_net: t3br
Running the playbook

PLAY [Patience is virtue - Bridge and Network creation] ***********************

TASK [Gathering Facts] ********************************************************

TO PASSSSSSSS NS_tenant3_1
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.1.12/30
Broadcast ID -> 185.0.1.15
185.0.1.13
185.0.1.14


[['185.0.1.13', '185.0.1.14']]
[['185.0.1.13', '185.0.1.14'], ['185.0.129.13', '185.0.129.14']]
ip link add ns_proj type veth peer name NS_tenant3_1
ip link set ns_proj netns NS_tenant3_1
ip link set NS_tenant3_1 netns proj1
ip netns exec NS_tenant3_1 brctl addbr BR_NS
ip netns exec NS_tenant3_1 ip link set ns_proj up
ip netns exec proj1 ip link set NS_tenant3_1 up
ip netns exec NS_tenant3_1 ip addr add 185.0.1.14/30 dev ns_proj
ip netns exec proj1 ip addr add 185.0.1.13/30 dev NS_tenant3_1
sh vxlan-create.sh NS_tenant3_1 42 ns_proj vxlan0 185.0.129.14 185.0.1.13 BR_NS
File Deleted
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c251d0 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
```

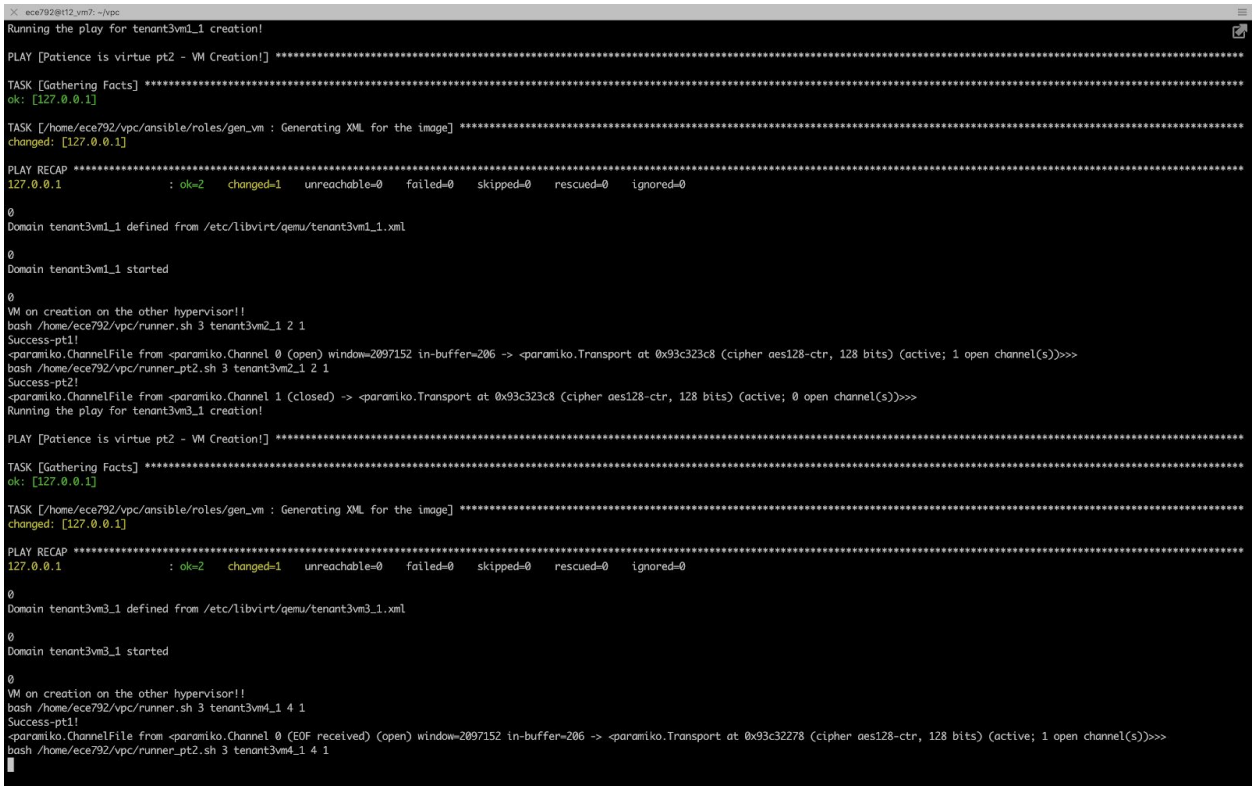
```
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.1.12/30
Broadcast ID -> 185.0.1.15
185.0.1.13
185.0.1.14


[['185.0.1.13', '185.0.1.14']]
[['185.0.1.13', '185.0.1.14'], ['185.0.129.13', '185.0.129.14']]
ip link add ns_proj type veth peer name NS_tenant3_1
ip link set ns_proj netns NS_tenant3_1
ip link set NS_tenant3_1 netns proj1
ip netns exec NS_tenant3_1 brctl addbr BR_NS
ip netns exec NS_tenant3_1 ip link set ns_proj up
ip netns exec proj1 ip link set NS_tenant3_1 up
ip netns exec NS_tenant3_1 ip addr add 185.0.1.14/30 dev ns_proj
ip netns exec proj1 ip addr add 185.0.1.13/30 dev NS_tenant3_1
sh vxlan-create.sh NS_tenant3_1 42 ns_proj vxlan0 185.0.129.14 185.0.1.13 BR_NS
File Deleted
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c251d0 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
ok: [localhost]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] ***
changed: [localhost] => (item=tenant3br_1)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather
than running sudo

TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********
changed: [localhost] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ***********************
ok: [localhost] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_1)

PLAY RECAP ********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
```

```
ece792@t12_vm7: ~/vpc
File Deleted
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c251d0 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
ok: [localhost]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] ***
changed: [localhost] => (item=tenant3br_1)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather
than running sudo

TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] *******
changed: [localhost] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ***********************
ok: [localhost] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_1)

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_1 ip a add 10.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_1 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=10.0.0.2,10.0.0.253,12h --dhcp-option=3,10.0.0.1
0
0
Running the play for tenant3vm1_1 creation!
```

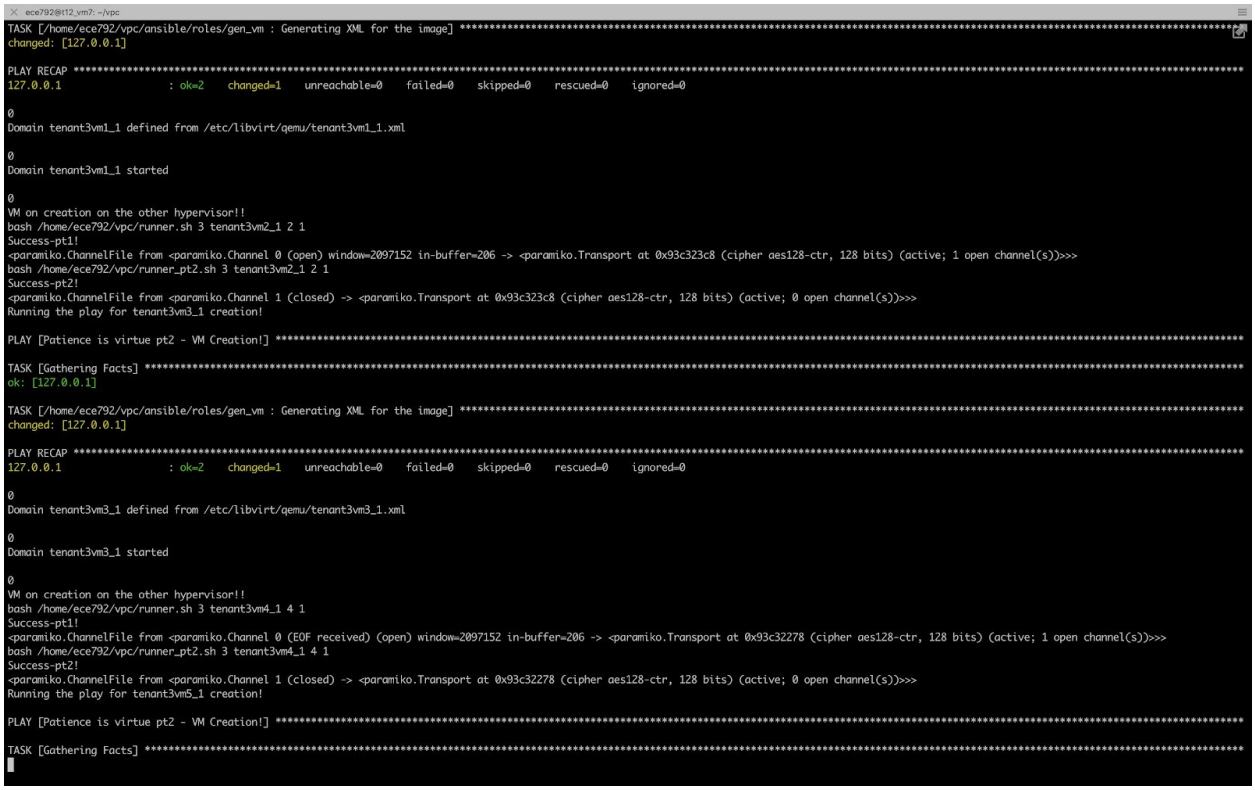
```
ece792@t12_vm7: ~/vpc
TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_1', u'network': u'tenant3br_1'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_1)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_1)

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_1 ip a add 10.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_1 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=10.0.0.2,10.0.0.253,12h --dhcp-option=3,10.0.0.1
0
0
Running the play for tenant3vm1_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_1 defined from /etc/libvirt/qemu/tenant3vm1_1.xml

0
```

```
ece792@t12_vm7: ~/vpc
changed: [localhost] => (item=tenant3br_1)

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_1 ip a add 10.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_1 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=10.0.0.2,10.0.0.253,12h --dhcp-option=3,10.0.0.1
0
0
Running the play for tenant3vm1_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_1 defined from /etc/libvirt/qemu/tenant3vm1_1.xml

0
Domain tenant3vm1_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_1 2 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_1 2 1
```

```
ece792@t12_vm7: ~/vpc
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_1 ip a add 10.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_1 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=10.0.0.2,10.0.0.253,12h --dhcp-option=3,10.0.0.1
0
0
Running the play for tenant3vm1_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_1 defined from /etc/libvirt/qemu/tenant3vm1_1.xml

0
Domain tenant3vm1_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_1 2 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_1 2 1
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
```



```
ece792@tt12_vm7: ~/vpc
Running the play for tenant3vm1_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_1 defined from /etc/libvirt/qemu/tenant3vm1_1.xml

0
Domain tenant3vm1_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_1 2 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_1 2 1
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm3_1 defined from /etc/libvirt/qemu/tenant3vm3_1.xml

0
Domain tenant3vm3_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_1 4 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c32278 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_1 4 1
```



```
ece792@tt12_vm7: ~/vpc
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_1 defined from /etc/libvirt/qemu/tenant3vm1_1.xml

0
Domain tenant3vm1_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_1 2 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_1 2 1
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c323c8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm3_1 defined from /etc/libvirt/qemu/tenant3vm3_1.xml

0
Domain tenant3vm3_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_1 4 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c32278 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_1 4 1
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c32278 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm5_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
```

```
ece792@tt2_vm7: ~/vpc

0
Domain tenant3vm3_1 defined from /etc/libvirt/qemu/tenant3vm3_1.xml

0
Domain tenant3vm3_1 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_1 4 1
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c32278 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_1 4 1
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c32278 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm5_1 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm5_1 defined from /etc/libvirt/qemu/tenant3vm5_1.xml

0
Domain tenant3vm5_1 started

0
11.0.0.0/24 5
11.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  11.0.0.1
DHCP range: 11.0.0.2 11.0.0.253
11.0.0.1 11.0.0.2 11.0.0.253 11.0.0.254 24
ip netns add NS_tenant3_2
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_2

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
```

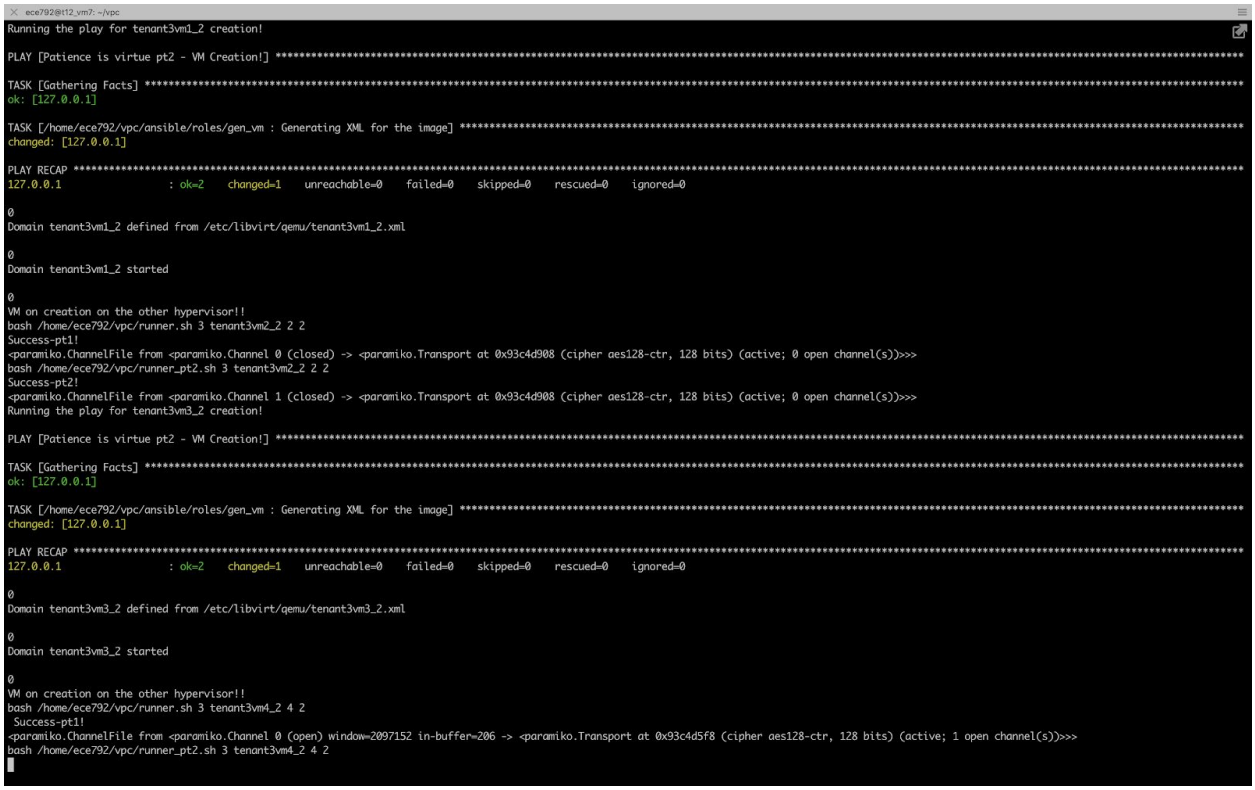
```
ece792@tt2_vm7: ~/vpc

0
Domain tenant3vm5_1 started

0
11.0.0.0/24 5
11.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  11.0.0.1
DHCP range: 11.0.0.2 11.0.0.253
11.0.0.1 11.0.0.2 11.0.0.253 11.0.0.254 24
ip netns add NS_tenant3_2
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_2

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] *************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo


TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ************************************************************************************************************************************
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_2', u'network': u'tenant3br_2'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_2 t3_2
```

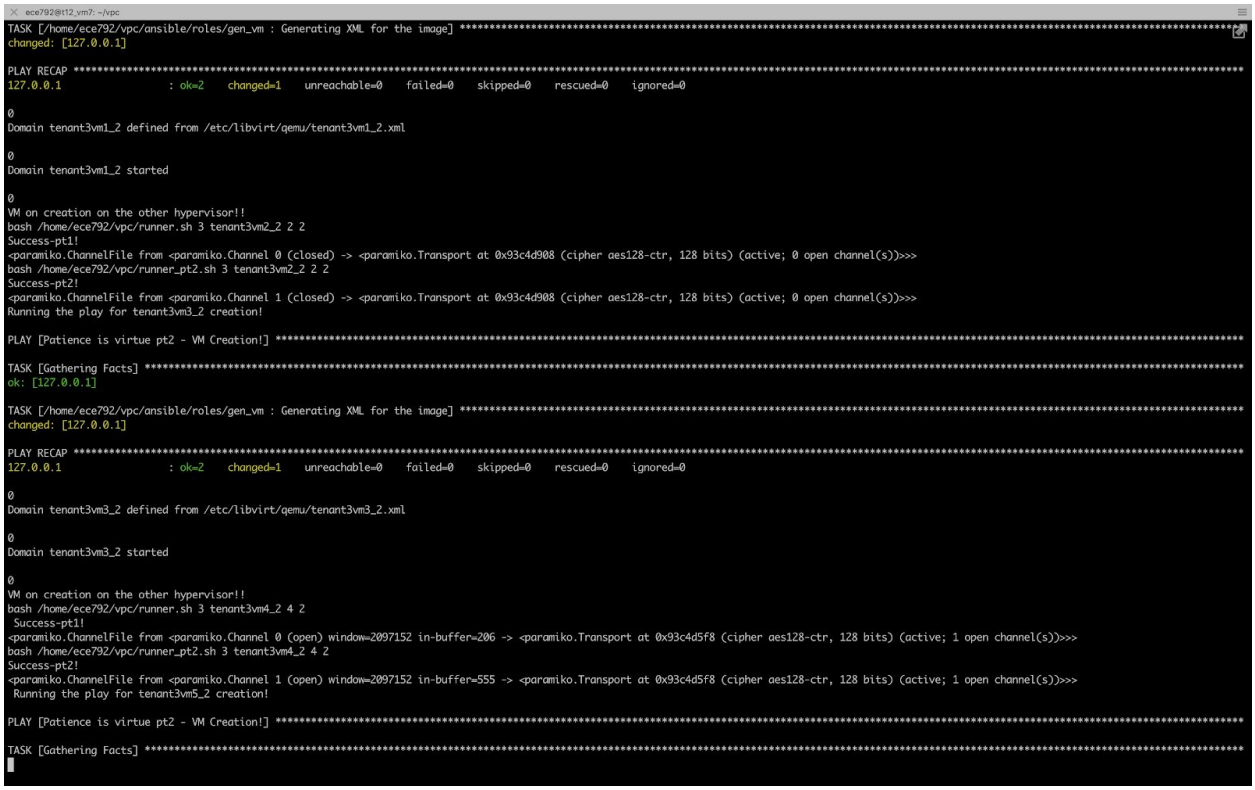
```
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_2', u'network': u'tenant3br_2'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_2)

PLAY RECAP ********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_2 t3_2
/home/ece792
(3, 2)
3
ovs_l2_net: t3br
Running the playbook

PLAY [Patience is virtue - Bridge and Network creation] ***********************

TASK [Gathering Facts] ********************************************************

TO PASSSSSSSS NS_tenant3_2
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.1.16/30
Broadcast ID -> 185.0.1.19
185.0.1.17
185.0.1.18


[['185.0.1.17', '185.0.1.18']]
[['185.0.1.17', '185.0.1.18'], ['185.0.129.17', '185.0.129.18']]
ip link add ns_proj type veth peer name NS_tenant3_2
ip link set ns_proj netns NS_tenant3_2
ip link set NS_tenant3_2 netns proj1
ip netns exec NS_tenant3_2 brctl addbr BR_NS
ip netns exec NS_tenant3_2 ip link set ns_proj up
ip netns exec proj1 ip link set NS_tenant3_2 up
ip netns exec NS_tenant3_2 ip addr add 185.0.1.18/30 dev ns_proj
ip netns exec proj1 ip addr add 185.0.1.17/30 dev NS_tenant3_2
sh vxlan-create.sh NS_tenant3_2 42 ns_proj vxlan0 185.0.129.18 185.0.1.17 BR_NS
File Deleted
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=575 -> <paramiko.Transport at 0x93c487b8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
```

```
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=575 -> <paramiko.Transport at 0x93c487b8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>

ok: [localhost]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] ***
changed: [localhost] => (item=tenant3br_2)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather
than running sudo

TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********
changed: [localhost] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ***********************
ok: [localhost] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_2', u'network': u'tenant3br_2'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_2)

PLAY RECAP ********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 2
VETH FUNCTION ON THEIR END?????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_2 ip a add 11.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_2 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=11.0.0.2,11.0.0.253,12h --dhcp-option=3,11.0.0.1
0
0
```

```
ece792@t12_vm7: ~/vpc
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_2', u'network': u'tenant3br_2'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_2)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_2)

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 2
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_2 ip a add 11.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_2 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=11.0.0.2,11.0.0.253,12h --dhcp-option=3,11.0.0.1
0
0
Running the play for tenant3vm1_2 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
```

```
ece792@t12_vm7: ~/vpc

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 2
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_2 ip a add 11.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_2 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=11.0.0.2,11.0.0.253,12h --dhcp-option=3,11.0.0.1
0
0
Running the play for tenant3vm1_2 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_2 defined from /etc/libvirt/qemu/tenant3vm1_2.xml

0
Domain tenant3vm1_2 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_2 2 2
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c4d908 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_2 2 2
```


ece792@tt2_vm7: ~/vpc
Running the play for tenant3vm1_2 creation!
PLAY [Patience is virtue pt2 - VM Creation!] ***********
TASK [Gathering Facts] ***********
ok: [127.0.0.1]
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ***********
changed: [127.0.0.1]
PLAY RECAP ***********
127.0.0.1 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
0
Domain tenant3vm1_2 defined from /etc/libvirt/qemu/tenant3vm1_2.xml
0
Domain tenant3vm1_2 started
0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_2 2 2
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c4d908 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_2 2 2
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c4d908 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_2 creation!
PLAY [Patience is virtue pt2 - VM Creation!] ***********
TASK [Gathering Facts] ***********
ok: [127.0.0.1]
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ***********
changed: [127.0.0.1]
PLAY RECAP ***********
127.0.0.1 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
0
Domain tenant3vm3_2 defined from /etc/libvirt/qemu/tenant3vm3_2.xml
0
Domain tenant3vm3_2 started
0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_2 4 2
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c4d5f8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_2 4 2


ece792@tt2_vm7: ~/vpc
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ***********
changed: [127.0.0.1]
PLAY RECAP ***********
127.0.0.1 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
0
Domain tenant3vm1_2 defined from /etc/libvirt/qemu/tenant3vm1_2.xml
0
Domain tenant3vm1_2 started
0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_2 2 2
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x93c4d908 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_2 2 2
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x93c4d908 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_2 creation!
PLAY [Patience is virtue pt2 - VM Creation!] ***********
TASK [Gathering Facts] ***********
ok: [127.0.0.1]
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ***********
changed: [127.0.0.1]
PLAY RECAP ***********
127.0.0.1 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
0
Domain tenant3vm3_2 defined from /etc/libvirt/qemu/tenant3vm3_2.xml
0
Domain tenant3vm3_2 started
0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_2 4 2
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c4d5f8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_2 4 2
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (open) window=2097152 in-buffer=555 -> <paramiko.Transport at 0x93c4d5f8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
Running the play for tenant3vm5_2 creation!
PLAY [Patience is virtue pt2 - VM Creation!] ***********
TASK [Gathering Facts] ***********

```
ece792@tt2_vm7: ~/vpc

0
Domain tenant3vm3_2 defined from /etc/libvirt/qemu/tenant3vm3_2.xml

0
Domain tenant3vm3_2 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_2 4 2
 Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x93c4d5f8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_2 4 2
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (open) window=2097152 in-buffer=555 -> <paramiko.Transport at 0x93c4d5f8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
 Running the play for tenant3vm5_2 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm5_2 defined from /etc/libvirt/qemu/tenant3vm5_2.xml

0
Domain tenant3vm5_2 started

0
12.0.0.0/24 5
12.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  12.0.0.1
DHCP range: 12.0.0.2 12.0.0.253
12.0.0.1 12.0.0.2 12.0.0.253 12.0.0.254 24
ip netns add NS_tenant3_3
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_3

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
```

```
ece792@tt2_vm7: ~/vpc

0
Domain tenant3vm5_2 started

0
12.0.0.0/24 5
12.0.0.0
24
The subnet mask is: 255.255.255.0

Default gateway is:  12.0.0.1
DHCP range: 12.0.0.2 12.0.0.253
12.0.0.1 12.0.0.2 12.0.0.253 12.0.0.254 24
ip netns add NS_tenant3_3
ovs_l2_net: tenant3br
ovs_l2_net: tenant3br_3

Running the playbook for bridge creation

PLAY [Patience is virtue - Bridge and Network creation] ************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/test_role : Creating the damn bridge, using roles and other shizzzzzz.] *************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo


TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ************************************************************************************************************************************
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_3', u'network': u'tenant3br_3'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)

PLAY RECAP *********************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_3 t3_3
```

```
ok: [127.0.0.1] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ********************************************************************************************************************
changed: [127.0.0.1] => (item={u'name': u'tenant3br_3', u'network': u'tenant3br_3'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ********************************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ********************************************************************************************************************************
changed: [127.0.0.1] => (item=tenant3br_3)

PLAY RECAP ********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
l3 veth ceation
sudo ovs-vsctl add-port tenant3br_3 t3_3
/home/ece792
(3, 3)
3
ovs_l2_net: t3br
Running the playbook

PLAY [Patience is virtue - Bridge and Network creation] ***********************

TASK [Gathering Facts] ********************************************************

TO PASSSSSSSS NS_tenant3_3
Network ID is 185.0.0.0/16


Subnet ID -> 185.0.1.20/30
Broadcast ID -> 185.0.1.23
185.0.1.21
185.0.1.22


[['185.0.1.21', '185.0.1.22']]
[['185.0.1.21', '185.0.1.22'], ['185.0.129.21', '185.0.129.22']]
ip link add ns_proj type veth peer name NS_tenant3_3
ip link set ns_proj netns NS_tenant3_3
ip link set NS_tenant3_3 netns proj1
ip netns exec NS_tenant3_3 brctl addbr BR_NS
ip netns exec NS_tenant3_3 ip link set ns_proj up
ip netns exec proj1 ip link set NS_tenant3_3 up
ip netns exec NS_tenant3_3 ip addr add 185.0.1.22/30 dev ns_proj
ip netns exec proj1 ip addr add 185.0.1.21/30 dev NS_tenant3_3
sh vxlan-create.sh NS_tenant3_3 42 ns_proj vxlan0 185.0.129.22 185.0.1.21 BR_NS
File Deleted
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x92a9b8d0 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
```

```
changed: [localhost] => (item=tenant3br_3)
 [WARNING]: Consider using 'become', 'become_method', and 'become_user' rather
than running sudo

TASK [/home/ece792/vpc/ansible/roles/test_role : Bringin up the bridge] ********
changed: [localhost] => (item=tenant3br_3)

TASK [/home/ece792/vpc/ansible/roles/test_role : debug] ***********************
ok: [localhost] => {
    "result_bridge.stdout_lines": "VARIABLE IS NOT DEFINED!"
}

TASK [/home/ece792/vpc/ansible/roles/test_role : Generation of XML file (L2 brige)] ***
changed: [localhost] => (item={u'name': u'tenant3br_3', u'network': u'tenant3br_3'})

TASK [/home/ece792/vpc/ansible/roles/net_define : Defining the networks!] ******
changed: [localhost] => (item=tenant3br_3)

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_3)

PLAY RECAP ********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 3
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_3 ip a add 12.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_3 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=12.0.0.2,12.0.0.253,12h --dhcp-option=3,12.0.0.1
0
0
Running the play for tenant3vm1_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
```

```
ece792@t12_vm7: ~/vpc

TASK [/home/ece792/vpc/ansible/roles/net_define : Starting the networks!] ******
changed: [localhost] => (item=tenant3br_3)

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 3
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_3 ip a add 12.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_3 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=12.0.0.2,12.0.0.253,12h --dhcp-option=3,12.0.0.1
0
0
Running the play for tenant3vm1_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_3 defined from /etc/libvirt/qemu/tenant3vm1_3.xml

0
Domain tenant3vm1_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_3 2 3
```

```
ece792@t12_vm7: ~/vpc
localhost                  : ok=7    changed=5    unreachable=0    failed=0

0
root@t12_vm6:/home/ece792/vpc# pwd
/home/ece792/vpc
root@t12_vm6:/home/ece792/vpc# sudo python ./hyp2_veth.py 3 3
VETH FUNCTION ON THEIR END??????
0
0
0
0
0
0
0
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_3 ip a add 12.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_3 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=12.0.0.2,12.0.0.253,12h --dhcp-option=3,12.0.0.1
0
0
Running the play for tenant3vm1_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_3 defined from /etc/libvirt/qemu/tenant3vm1_3.xml

0
Domain tenant3vm1_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_3 2 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_3 2 3
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
```

```
ece792@t12_vm7: ~/vpc
0
0
0
0
out of the stupid veth
You asked for more, give us time to bring your connections up!
Veth connection for VXLAN Creation
sudo ip netns exec Tenant3_SG_3 ip a add 12.0.0.1/24 dev t3r
sudo ip netns exec Tenant3_SG_3 dnsmasq --interface=t3r --except-interface=lo --bind-interfaces --dhcp-range=12.0.0.2,12.0.0.253,12h --dhcp-option=3,12.0.0.1
0
0
Running the play for tenant3vm1_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_3 defined from /etc/libvirt/qemu/tenant3vm1_3.xml

0
Domain tenant3vm1_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_3 2 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_3 2 3
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
```

```
ece792@t12_vm7: ~/vpc
Running the play for tenant3vm1_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_3 defined from /etc/libvirt/qemu/tenant3vm1_3.xml

0
Domain tenant3vm1_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_3 2 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_3 2 3
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm3_3 defined from /etc/libvirt/qemu/tenant3vm3_3.xml

0
Domain tenant3vm3_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_3 4 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x92aa2160 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_3 4 3
```

```
ece792@t12_vm7: ~/vpc
TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm1_3 defined from /etc/libvirt/qemu/tenant3vm1_3.xml

0
Domain tenant3vm1_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm2_3 2 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (EOF received) (open) window=2097152 in-buffer=206 -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 1 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm2_3 2 3
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x92aa24a8 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm3_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************************
ok: [127.0.0.1]

TASK [/home/ece792/vpc/ansible/roles/gen_vm : Generating XML for the image] ****************************************************************************************************************************************
changed: [127.0.0.1]

PLAY RECAP *********************************************************************************************************************************************************************************************************
127.0.0.1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

0
Domain tenant3vm3_3 defined from /etc/libvirt/qemu/tenant3vm3_3.xml

0
Domain tenant3vm3_3 started

0
VM on creation on the other hypervisor!!
bash /home/ece792/vpc/runner.sh 3 tenant3vm4_3 4 3
Success-pt1!
<paramiko.ChannelFile from <paramiko.Channel 0 (closed) -> <paramiko.Transport at 0x92aa2160 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
bash /home/ece792/vpc/runner_pt2.sh 3 tenant3vm4_3 4 3
Success-pt2!
<paramiko.ChannelFile from <paramiko.Channel 1 (closed) -> <paramiko.Transport at 0x92aa2160 (cipher aes128-ctr, 128 bits) (active; 0 open channel(s))>>>
Running the play for tenant3vm5_3 creation!

PLAY [Patience is virtue pt2 - VM Creation!] ************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************************
```

```
0
ece792@t12_vm7:~/vpc$ cat /var/log/syslog | grep "DHCPACK(t"
Nov 18 07:03:24 t12_vm7 dnsmasq-dhcp[23824]: DHCPACK(t3r) 10.0.0.231 52:54:00:8a:3d:be
Nov 18 07:03:39 t12_vm7 dnsmasq-dhcp[23824]: DHCPACK(t3r) 10.0.0.60 52:54:00:dd:31:94
Nov 18 07:04:00 t12_vm7 dnsmasq-dhcp[23824]: DHCPACK(t3r) 10.0.0.127 52:54:00:c9:1c:6e
Nov 18 07:04:20 t12_vm7 dnsmasq-dhcp[23824]: DHCPACK(t3r) 10.0.0.185 52:54:00:02:b0:83
Nov 18 07:04:38 t12_vm7 dnsmasq-dhcp[23824]: DHCPACK(t3r) 10.0.0.31 52:54:00:e8:68:f7
Nov 18 07:05:34 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.201 52:54:00:70:d6:2b
Nov 18 07:05:52 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.149 52:54:00:d9:c2:7a
Nov 18 07:06:01 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.149 52:54:00:d9:c2:7a
Nov 18 07:06:06 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.24 52:54:00:68:2a:62
Nov 18 07:06:35 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.53 52:54:00:9c:29:ce
Nov 18 07:06:50 t12_vm7 dnsmasq-dhcp[28190]: DHCPACK(t3r) 11.0.0.225 52:54:00:6a:c2:09
Nov 18 07:07:50 t12_vm7 dnsmasq-dhcp[32727]: DHCPACK(t3r) 12.0.0.229 52:54:00:e3:a8:72
Nov 18 07:08:15 t12_vm7 dnsmasq-dhcp[32727]: DHCPACK(t3r) 12.0.0.13 52:54:00:3c:07:50
Nov 18 07:08:35 t12_vm7 dnsmasq-dhcp[32727]: DHCPACK(t3r) 12.0.0.201 52:54:00:96:66:c5
Nov 18 07:08:54 t12_vm7 dnsmasq-dhcp[32727]: DHCPACK(t3r) 12.0.0.144 52:54:00:52:44:36
Nov 18 07:09:09 t12_vm7 dnsmasq-dhcp[32727]: DHCPACK(t3r) 12.0.0.59 52:54:00:70:93:9a
ece792@t12_vm7:~/vpc$
```

```
 180   ref-VM10                       running
 181   tenant3vm1_1                   running
 182   tenant3vm3_1                   running
 183   tenant3vm5_1                   running
 184   tenant3vm1_2                   running
 185   tenant3vm3_2                   running
 186   tenant3vm5_2                   running
 187   tenant3vm1_3                   running
 188   tenant3vm3_3                   running
 189   tenant3vm5_3                   running
```

```
 138   ref-VM20                          running
 139   tenant3vm2_1                      running
 140   tenant3vm4_1                      running
 141   tenant3vm2_2                      running
 142   tenant3vm4_2                      running
 143   tenant3vm2_3                      running
 144   tenant3vm4_3                      running

root@t12_vm6:/home/ece792/vpc#
```

Section 2 - Functional Features

#1 Broker server based load-balancing

What is broker server based load-balancing?

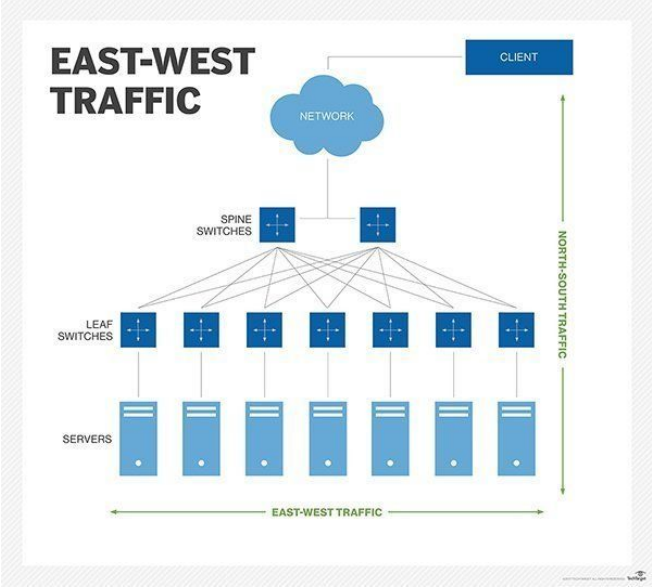
When a website domain name is entered in a web-browser of a host, the host is already aware of it's local broker server. The broker server receives a publish/subscribe request for content and returns the edge server to which the data has to be published or subscribed from.

What is DNS based load-balancing in terms of edge computing?

- When it comes to CDN, the broker server sends a response of one of the edge CDN servers from a pool of edge CDN servers to the subscribe request initiator.
- Considering that the broker server response to different clients are different CDN edge servers, load-balancing is achieved.

#2 East to West Traffic

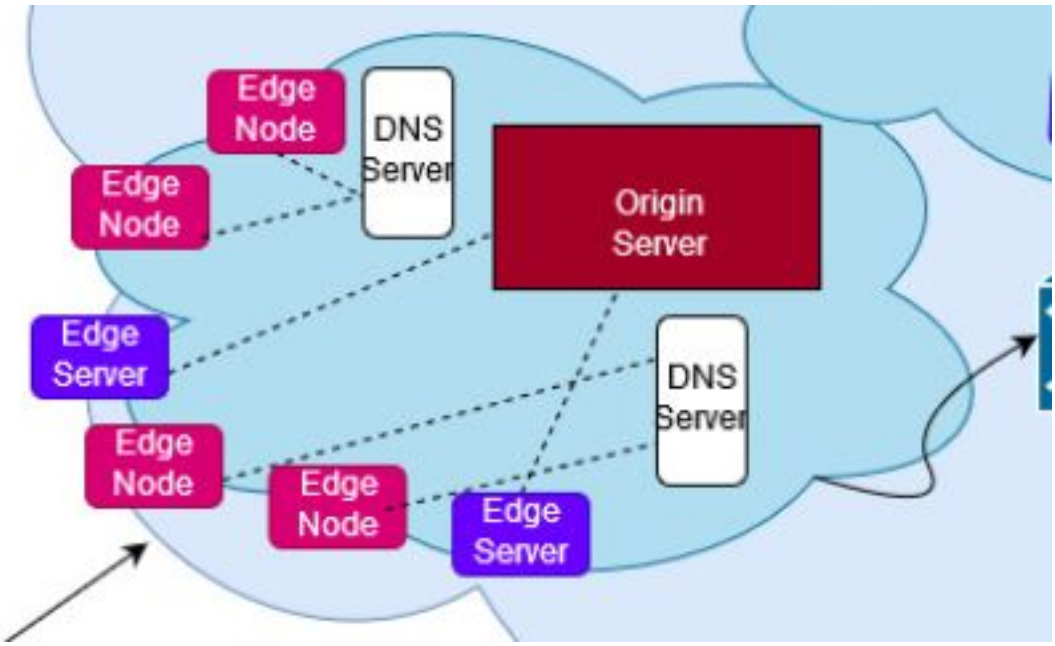
We also plan to implement east to west traffic that depicts communication between nodes in a local area network that are normally placed horizontally inside a datacenter. This type of communication is going to be shown in our project by the communication between two edge servers. The following figure shows servers in a datacenter which apply east to west traffic by traversing through the leaf switches and spine switches.



What have we implemented so far?

While implementing edge as a service, we assume that our customer, ie Netflix, has subscribed for a content delivery network. Here every VM of our tenant is an edge node, that is allowed to ask for a subscribe or publish request of the data created by some client. Our model includes two hypervisors which form the VPC for all our tenants, and these hypervisors can be assumed to be at geographically different locations. To our customer, the higher most layer of the

topology is the subnet gateway, and hence we place our origin server for our customer at the common single Router provided to each tenant. In such a case, we produce an edge server that can cache data at the nearest point to our edge devices. Hence we provide an edge server for each subnet of the tenant. In order to resolve which content must be cached at which edge server, we use a broker server that resolves all requests, making our model a content based DNS routing. As shown below, the flow of the feature is shown:



Mechanism of edge computing:
We have assumed the file format for the content to be an rfc file. And we take into account a base assumption that every subnet of a VM prefers to use a certain subset of rfc files more commonly than others. Hence as given below, we have assumed:

Rfc (1-10)- Subnet 1
Rfc(11-20)-Subnet 2
Rfc(21-30)-Subnet 3
So this way, based on what file is being published, the correspondingly matching Edge server of the preferred subnet is given back to the edge node from the broker server to publish into. This is saved as a cache file into the edge server, as the client pushes the data. The data is also published into a corresponding edge server on another hypervisor which also prefers that data. Finally publishing this data into an origin server is a default step, as the origin server symbolises the data center which holds all data. Now that this new data is cached into an edge server, the mapping of content to edge server is saved in a file at all the broker servers to keep track of content-edge server mapping.

In case of a subscribe action, similar to a publishing action, the broker server which receives the request will check which edge server to be forwarded. This broker server acts as a broker or an authoritative DNS, and uses the logic of preferring its own local edger server to retrieve the data over an edger server that is on another hypervisor. Hence it redirects the edge node to

contact an edge server -which has the requested file as its preferred file caching. This way we implement east to west traffic, where an edge server that is not located in its local network is preferred over another edge server within its network.

To implement this we have used python as the tool. We have a dhcp_helper.py file which is called in the main.py , and it is used to IP addresses that will be assigned to each edge server and broker server in a subnet.
We arbitrarily fix the last few addresses for these servers.
255.255.255.254 - broker server address for every subnet given by the
255.255.255.253- 255.255.255.252 -Edge server

We have 3 python files running, one main.py that runs on every VM and two other python files that do publish and subscribe.

Section 3 - Management Features

#1 Performance Management

Feature description:

Performance management examines and monitors the current network efficiency and plans ahead for future changes or upgrades. While constantly monitoring the health of the network and searching for trends, network parameters are tracked and logged; these include data transmission rate (throughput), error rates, downtime/uptime, use-time percentages and response time to user and automated inputs or requests.

Implementation logic:

1. Collect metrics from each edge node using a python script and a networking module (Netmiko). Example metrics – Amount of storage available, Uptime, link flap, amount of traffic passed.

This helps the customer to make a calculated decision if there is a larger capacity requirement, or to decide if higher storage is required at the edge nodes.

2. Collected metrics are pushed to InfluxDB which records data with respect to time. These metrics can be used to display graph trends on Grafana.
3. InfluxDB runs on the management VM.

#2 Configuration Management

Feature description:

Configuration management involves the collecting and storing various configuration data, preferably in an easily accessible database, simplifying configuration procedures for each network device.

Implementation logic

1. Collect current configuration details (ex: mem, cpu, ram) and store them in a simple database such as MongoDB. MongoDB uses a NoSQL DB which has JSON like mapping which makes it easily readable. Customers can query this DB to get info about the current configuration of their edge CDN servers. (edge nodes)
2. Develop RESTful APIs/CLI methods and expose them to the customer to make configuration easy.

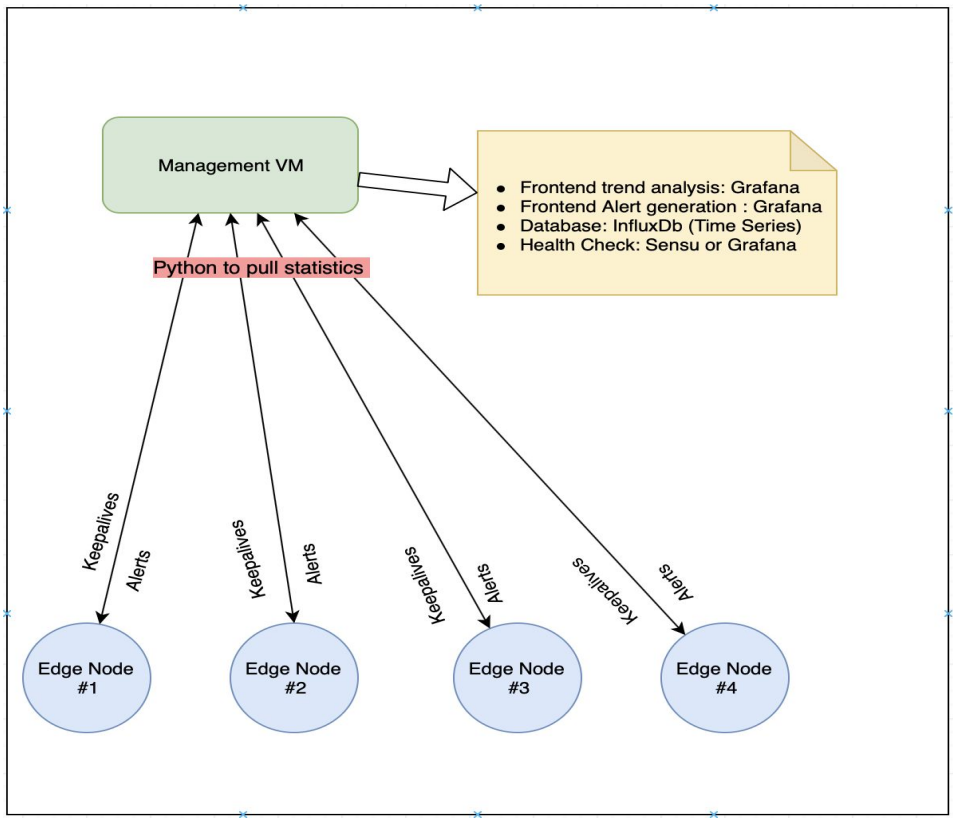
Possible APIs include:

- Configure the number of hosts an edge CDN server must handle at any given time
- Prioritize the catching of data
- Prioritize our customer's customer. (In case of paid or unpaid)

3. MongoDB runs on the management VM.

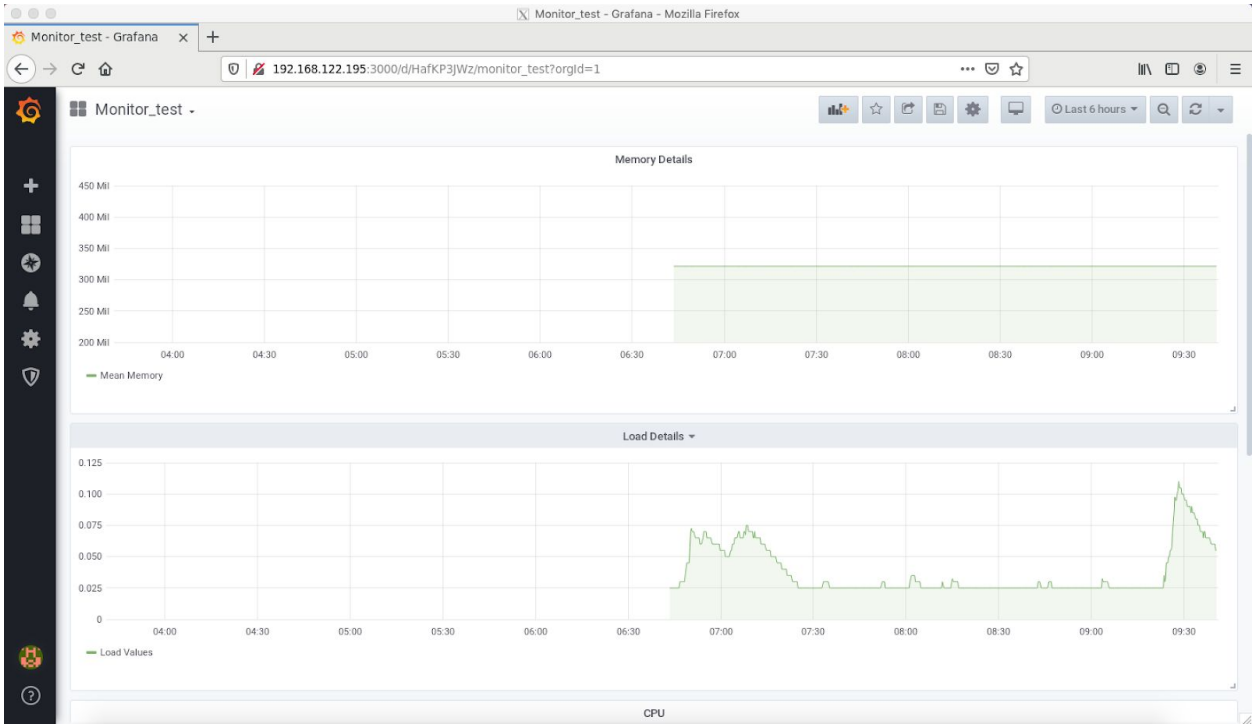
What have we implemented so far?

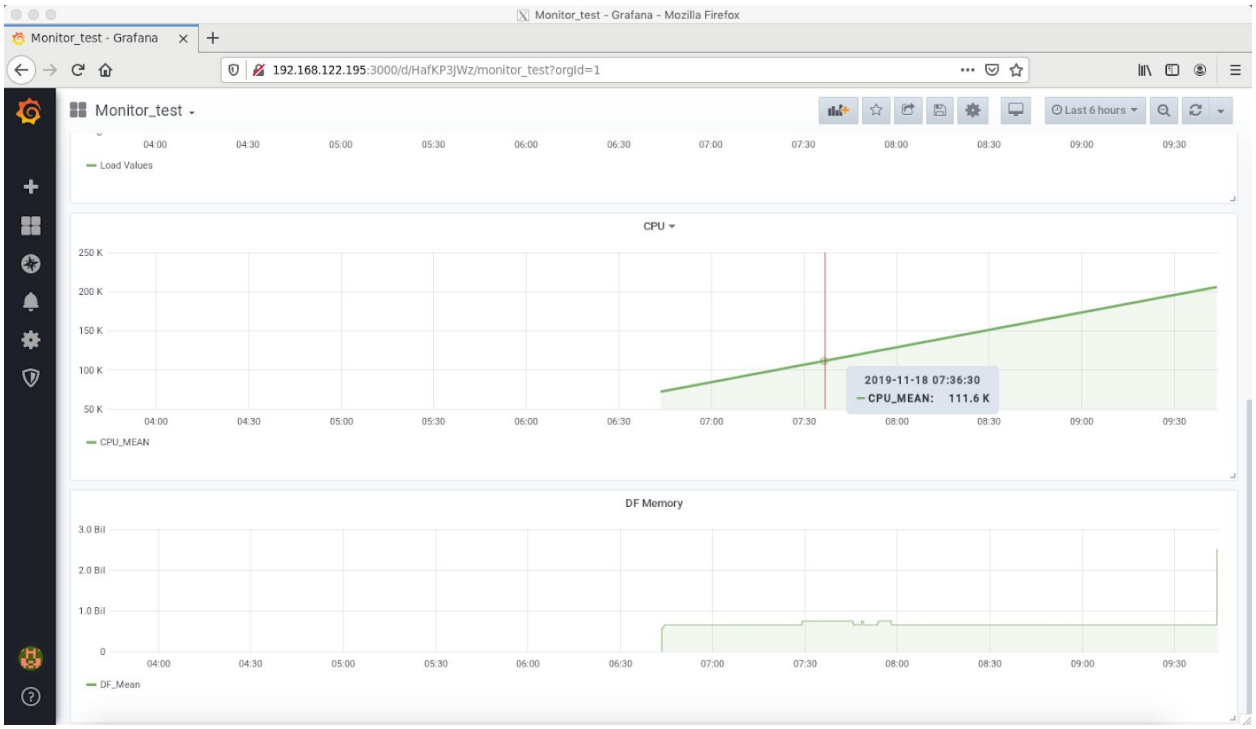
Model architecture:



As shown in the model above, every edge node sends its data to the management VM , where an admin can monitor the performance of the system. The collectd daemon packages are installed in every VM on the hypervisor. The .conf file pushes the collected data through the management interface, ie. the proj_virbr0. The influxDB, the time series database is installed in the hypervisor listens through the port 25826 for the UDP packets and this port pushes the data into the management network's collectd. Grafana is installed and listens to the port at localhost:3000.

Following are the screenshots captured of memory, load and cpu usage at the Hypervisor 1- which includes performance and configuration management:





#3 Fault Management

Feature description

The goals and objectives include early fault recognition, isolation of negative effects, fault correction and logging of the corrections to assist in improvement.

Implementation logic

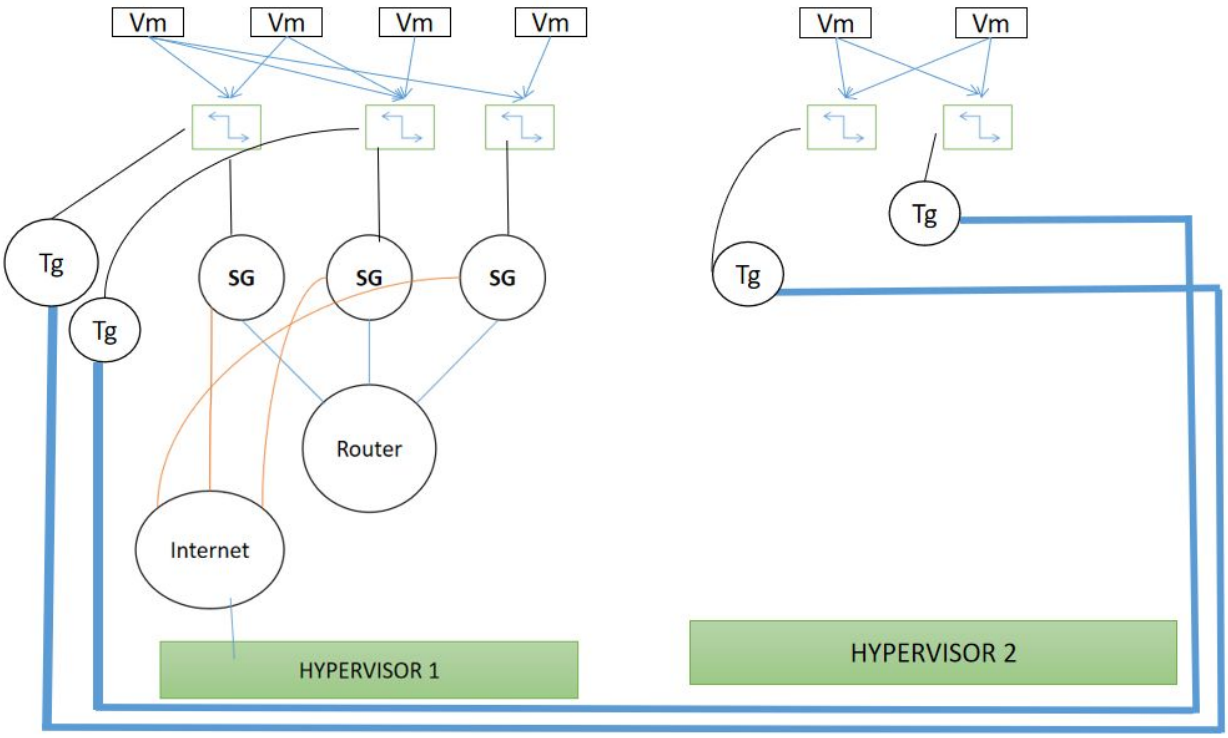
1. Use an agent- based health monitoring service such as Sensu to generate keepalives from the edge CDN nodes.

2. Use Sensu to perform health checks on the disk, memory and other parameters. Generate alerts and log them.
3. Extend features of Sensu by writing custom checks. For example, a check to see if the number of hosts served by the edge CDN. If that exceeds the specified threshold, an alert is generated.
4. Alerts can be classified as Warnings or Critical. Alerts can be sent via a slack notification or an email or just on the frontend.
5. Sensu runs on the management vm on any arbitrary port from the usable range.

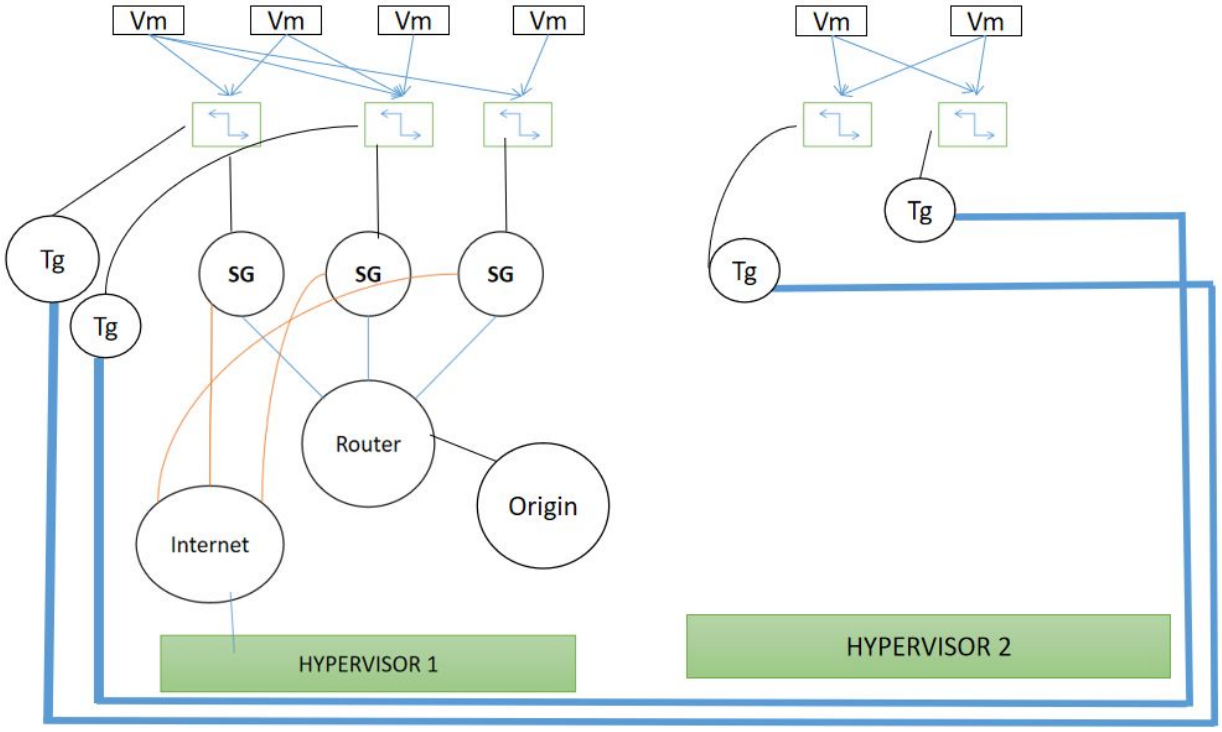
Milestone 3 Progress Log

Section 1 - VPC Containerised

We have containerised all the VMs! So Subnet gateways, gateway routers, Tunnel gateways, Internet namespace, are all containers now. Each container of the tenant is attached to those only those bridges that is specified for a specific subnet (as given by the tenant).



Finally, the Origin server is also made (which will be used as a part of management and functional features) as shown in the figure below:



Northbound part of our Project:

Json input for our main file (Part 1):

```
{
        "subnet_nos":"3",
        "subnet_dict": {"1":"10.0.0.0/24", "2":"11.0.0.0/24", "3":"12.0.0.0/24"},
        "vm_nos":"2",
        "new_dict": {"1": ["1","2","3"], "2": ["1","2"]},
        "Internet": "yes",
        "Management": "yes"
}
```

Json input for origin creation- input -> includes subnets with edge servers (Part 2):

```
{
        "Subnet_IPs": ["10.0.0.0/24","11.0.0.0/24","12.0.0.0/24"]
}
```

Json input for our management feature- List of Edge servers to be monitored (Part 1):

```
{
        "Container_name": ["T_132_E1","T_132_E2"]
}
```

Json input for our management feature- Name of Grafana server(Part 2)

```
{
                "Container_name": "Origin139",
}
```

Json input for origin server- accounting feature for edge servers

```
{
```

```
        "Container_name": ["T_132_E1","T_132_E2"]
}
```

Logic layer:
We use a main.py - python file that begins the process of VPC creation calling python subscripts in the order of creation as described in the VPC Flow - milestone 2 progress section.
Some parts of the southbound script are still a part of main.py.

Southbound part of our logic:
It consists of multiple python files tasked to build different parts of our architecture.
Internet_L3_Connectivity.py : sets up namespaces for internet connectivity
Vxlan_create.py : Vxlan creation
Ovs_bridge creation : part of the main.py
Docker container Creation: Creates tenant VMs
Origin_Server.py : Creates the Origin server, for every tenant (fixed IP 24.16.8.10), there exists only one origin server

Section 2 -Management Feature:

Feature 1:
Self Healing- Kubernetes:
We have a bash script that runs every 30 seconds which checks for active containers and if it detects a change in the list of active containers, the containers which have gone down will be started again. The crontab time-scheduler must be called and started to get this feature.

```
ssh
ece792@t12_vm7:~/vpc/management/useless_checker$ cat output.out
We died! []
ece792@t12_vm7:~/vpc/management/useless_checker$ cat output.out
ece792@t12_vm7:~/vpc/management/useless_checker$ cat output.out
We died! ['Ram_int2']
Spawning edge node Ram_int2
ece792@t12_vm7:~/vpc/management/useless_checker$ cat output.out
We died! ['Ram_int2']
Spawning edge node Ram_int2
ece792@t12_vm7:~/vpc/management/useless_checker$ sudo docker attach Ram_int2
[sudo] password for ece792:
root@470688d251af:/# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qle
n 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: gre0@NONE: <NOARP> mtu 1476 qdisc noop state DOWN group default qlen 1000
    link/gre 0.0.0.0 brd 0.0.0.0
3: gretap0@NONE: <BROADCAST,MULTICAST> mtu 1462 qdisc noop state DOWN group default q
len 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
4: erspan0@NONE: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default q
len 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
5589: eth0@if5590: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
group default
    link/ether 02:42:bc:00:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 188.0.0.5/16 brd 188.0.255.255 scope global eth0
       valid_lft forever preferred_lft forever
root@470688d251af:/#
```

```
ece792@t12_vm6:~/vpc/management/useless_checker$ cat output.out
We died! []
ece792@t12_vm6:~/vpc/management/useless_checker$ cat output.out
ece792@t12_vm6:~/vpc/management/useless_checker$ cat output.out
ece792@t12_vm6:~/vpc/management/useless_checker$ cat output.out
We died! ['keen_kare']
Spawning edge node keen_kare
ece792@t12_vm6:~/vpc/management/useless_checker$ sudo docker attach keen_kare
root@08a36c170585:/# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: gre0@NONE: <NOARP> mtu 1476 qdisc noop state DOWN group default qlen 1000
    link/gre 0.0.0.0 brd 0.0.0.0
3: gretap0@NONE: <BROADCAST,MULTICAST> mtu 1462 qdisc noop state DOWN group default qlen 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
4: erspan0@NONE: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default qlen 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
2813: eth0@if2814: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:bd:00:00:53 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 189.0.0.83/16 brd 189.0.255.255 scope global eth0
       valid_lft forever preferred_lft forever
root@08a36c170585:/# read escape sequence
ece792@t12_vm6:~/vpc/management/useless_checker$
```

Feature 2:

Accounting:

This feature allows the tenant to monitor the details of CPU and Memory usage by the Edge servers where RFC files are cached. This display of details, gives the tenant the ability to optimally allot more edge servers if required.
The following Northbound json file exists in every origin server, so the tenant can view it accordingly.

```
ssh
root@a35dad154df3:~# python3 main.py
T_132_E1 T_132_E2
docker stats --no-stream --format "table {{.Container}} {{.CPUPerc}} {{.MemUsage}}" T_132_E1 T_132_E2
Grabbing info

 **************************
b'CONTAINER CPU % MEM USAGE / LIMIT'
b'T_132_E1 0.00% 58.45MiB / 23.54GiB'
b'T_132_E2 0.19% 37.97MiB / 23.54GiB'
root@a35dad154df3:~# ls
input_data.json  main.py
root@a35dad154df3:~# cat input_data.json
{
    "Container_name": ["T_132_E1","T_132_E2"]
}
root@a35dad154df3:~#
```

Section 3 - Functional Feature:

Feature description:

Our feature places more control over the customer's requirement.

- The customer can choose the number of the edge servers they want. Post, the creation of the VPC, a file is presented to the customer through which the number of edge servers they request are created.

Northbound input:

```
ece792@t12_vm7:~/vpc$ cat ServerCreation.txt
{"No of edge servers in Location 1 for RFC range 1-10": 1,
"No of edge servers in Location 1 for RFC range 11-20": 2,
"No of edge servers in Location 1 for RFC range 21-30": 1,
"No of edge servers in Location 2 for RFC range 1-10": 1,
"No of edge servers in Location 2 for RFC range 11-20": 1,
"No of edge servers in Location 2 for RFC range 21-30": 1}
ece792@t12_vm7:~/vpc$
```

- Each location refers to a hypervisor. The edge servers across two locations are connected with the help of VXLAN tunneling.
- The "Create_Servers.py" code will create the edge servers and broker servers that connect to tenant bridges.

- We have placed two broker servers, one for each hypervisor. All the customers residing in Hypervisor 1 are redirected to Broker server in Hypervisor 1. All the customers residing in Hypervisor 2 are redirected to Broker server in Hypervisor 2.
- As per the above input to the text file, 1 edge server will be attached to tenant bridge 1 in hypervisor 1. 2 edge servers will be attached to tenant bridge 2 in hypervisor 1. 1 edge server will be attached to tenant bridge 3 in hypervisor 1. 1 edge server will be attached to tenant bridge 1 in hypervisor 2. 1 edge server will be attached to tenant bridge 2 in hypervisor 2. 1 edge server will be attached to tenant bridge 3 in hypervisor 2.
- After the creation of edge servers and broker servers the customer can perform two operations:
    - Publish
    - Subscribe
- The following is the northbound input for the above operations:

```
{"Type of Message (Publish/Subscribe)":"Publish",
"RFC number":"12",
"Algorithm 1-Closest Server (True/False)":"True",
"Algorithm 2-Fixed Server (True/False)":"False", "Fixed Server IP":"12.0.0.252",
}
```

- The customer has the option to choose between the algorithms that he/she pleases to.
- The inputs to the JSON file are used in the logic layer to call the respective scripts that carry on the functions requested. (“Logiccode.py”)

```
import json
import os
f1 = open("Client_file",'r')
f2 = json.loads(f1.read())
print(type(f2))

rfc_id = f2["RFC number"]

if f2["Type of Message (Publish/Subscribe)"]=='Publish':
    os.system("python3 dns_pub_host.py")
    print("python3 dns_pub_host.py rfc_"+rfc_id)


elif f2["Type of Message (Publish/Subscribe)"]=='Subscribe':
    if f2["Algorithm 1-Closest Server (True/False)"]=='True':
        #os.system("python3 subscribe.py")
        print("python3 subscribe.py "+rfc_id)
    elif f2["Algorithm 2-Fixed Server (True/False)"]=='True':
        Server_IP=f2["Fixed Server IP"]
        #os.system("python3 subscribe.py "+str(Server_IP))
        print("python3 subscribe_fs.py "+rfc_id+ " " +str(Server_IP))
    elif f2["Algorithm 3-Least busy server (True/False)"]=='True':
        #os.system("python3 subscribe.py")
        print("python3 subscribe_lb.py "+rfc_id)
```