DNS-as-a-service

Linux Networking Project - Fall 2018

Containerized Solution of DNS-as-a-Service

Functional Documentation of Infrastructure automation of Containerized Architecture:

VPC CRUD:

- **Create:** Tenant will upload a yaml file with VPC fields: VPC Name and in return we will provide tenant VPC id of newly created VPC.
  - *Command:*

    ```
    sudo ansible-playbook create_vpc.yaml
    ```

  - Requirements: `vpc_vars.yaml` file with tenant id and required parameters.
- **Read:** List down all the VPC ids for a tenant.
  E.g. `sudo cat /home/ece792/tenants/t7/vpc_db.json`.
- **Update:** There couldn't be any updates as for VPC.
- **Delete:** Remove the VPC for the tenant, which may force to destroy deletion of all VMs running inside the VPC. (automated as python script)

Subnet CRUD:

- **Create:** Tenant can create as many subnets inside the VPC. The only requirement to create subnet is that, tenant should have created at least one VPC in our infrastructure.
  - *Command:*

    ```
    sudo ansible-playbook create_subnet.yaml
    ```

  - Requirements: `subnet_vars.yaml` file with tenant id and required parameters.
- **Read:** List down all the subnets for a tenant.
  E.g. `sudo cat /home/ece792/tenants/t7/subnet_db.json`.

- **Update:** Tenant can change the subnet CIDR block by logging into his controller VM and run the dhcp server again on that interface. But in the practice it is not recommended. Instead, create a new subnet. (manual task)
- **Delete:** Shutoff all the containers connected to this subnet and delete subnet L2 bridge. (automated as python script).

DNS Server CRUD:

- **Create:** Create DNS Server
  - *Command:*

    ```
    sudo ansible-playbook create_dns.yaml
    ```

  - For availability, create two DNS servers. Run the above scripts twice with different dns vm names.
- **Read:** Each tenant's database of DNS servers is maintained inside the tenants folder in hypevisor.
  E.g. `/home/ece792/tenants/t2/dns_db.json`
- **Delete:** Destroying the DNS server VM. (automated as python script)

Please note the following:

1. Create DNS at VPC Level:

- Defined the level as 'vpc' in yaml file.

2. Create DNS at Subnet Level:

- Defined the level as 'subnet' in yaml file.

The ansible script create_dns.yaml file will take care of the logic, automation of DNS configurations and adding the A, NS records entries in the hierarchy.

Containerized Server Instance CRUD:

- **Create:**
  - Tenant / User will require to fill up a form or json / yaml file which has creation parameters like: vcpu, memory, disk size, OS type, VPC id and as result we will create VM for the user, assign IP.
  - *Commands:*

    ```
    sudo ansible-playbook create_server_instance.yaml
    ```

    - Requirements: `instance_vars.yaml` file with tenant id and required parameters.
    - Automation:
        - Completely automated server containers instance spin up, attach with subnet and will also get ip address from dhcp server from subnet as well as controller network.
        - It will also define and update nameserver in the container.
        - Provider adming able to ssh to this new vm without any ssh keypair generation manual steps.
- **Read:** Each tenant's database is maintained inside the tenants folder in hypevisor. E.g. `/home/ece792/tenants/t7/vm_db.json`. Here, the only manual task required is to fill in the VM's ip address as domifaddr is not showing up the ip assigned by DHCP server inside namespace.
- **Update:** Tenant / user should be able to update the VPC id, i.e. putting the existing VM, add interfaces. Tenant is able to terminate, restart VM. (Manual task)
- **Delete:** Destroying the VM and free up memory. (automated as python script)

Provider / Developer Guide for Hypervisor Infrastructure:

- Make sure there is `provider_ns` network namespace is created in the linux box machine (hypervisor).
- Create Controller Container which has the private key as provider to ssh to all instances of the running docker containers for magement features.
- Create Hypervisor DNS container using the automation script:
    - *Command:*

    ```
    sudo ansible-playbook provider_configuration/hv_dns.yaml
    ```

Docker Images:

Docker Images types we have used:

1. centos image: For client server containers with CentOS.
2. ubuntu image: For client server containers with Ubuntu OS.
3. Bind CentOS Image: Used for creating DNS server containers inside the VPC and subnet.
4. Hypervisor Bind CentOS Image: For creating Hypervisor bind containers.

Deletion of VPC, Subnet, VM instance and DNS instance:

- We have written a single python script for deletion of VPC, subnet and VM instnaces.
- Run command:

`sudo python delete_script.py`
and then it will ask user which VPC, subnet and VM instance to delete. So, it is totally interactive deletion script.

Clients:

It is assumed that client will perform following steps in order to build their infrastructure.

- Will upload file first for VPC creation `vpc_vars.yml`
- Upload file for subnet creation `subnet_vars.yml`
- Upload file for dns in his VPC creation `dns_vars.yml`
- Upload file for guest vm or server vm inside subnet and vpc creation `instance_vars.yml`

Prerequisties for ansible scripts:

- In the hypervisor, you should have dns.img and centos7-minimal.img in images folder, which has preconfigured dhcp client and dns images has bind9, firewall settings pre-configured.

Notes:

- All the ansible scripts / tasks written are idempotent meaning if a client runs the same parameters, our end system won't be affected.
- All the duplication checks are done in the playbook and accordingly tasks are run.
- We are storing database as file and each tenant has different folder inside `tenants` directory. (Also attached sample tenants database with code.)

VM architecture:

- Documentation: ./README_Containers.md
- Codes inside this folder for VM.

References:

- Bind CentOS Docker Image: https://github.com/CentOS/CentOS-Dockerfiles/tree/master/bind/centos7
- Kube DNS Feature: https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/
- AWS VPC Subnets: https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Subnets.html
- Cloudflare DNS: https://www.cloudflare.com/dns/
- Amazon Route 53: https://aws.amazon.com/route53/
- NS1 DNS: https://ns1.com/products

- Google Cloud DNS: https://cloud.google.com/dns/
- Azure DNS: https://azure.microsoft.com/en-us/services/dns/
- Rackspace Cloud DNS: https://www.rackspace.com/cloud/dns
- https://www.siteground.com/kb/private_dns/