

Sai Pavan Phaneendra Vaddi

Saipavan2930@gmail.com

9515952930

Batch No: 129

CASE STUDY

Create two VPCs in different regions and connect the two vpc's using peering.

Step 0: Login to aws account with the credentials.

Step 1: Create a VPC in Ohio Region

The screenshot shows the AWS Management Console interface for creating a new VPC in the Ohio region. The page is titled 'Create VPC' and includes a breadcrumb trail: VPC > Your VPCs > Create VPC. Below the title, there is a brief description of a VPC and a 'Preview' section.

VPC settings

- Resources to create:** Two options are available: 'VPC only' (unselected) and 'VPC and more' (selected).
- Name tag auto-generation:** A checkbox labeled 'Auto-generate' is checked. The text field below it contains 'ohioregion'.
- IPv4 CIDR block:** The text field contains '10.0.0.0/24'. To the right, it indicates '256 IPs'. A note below states: 'CIDR block size must be between /16 and /28.'
- IPv6 CIDR block:** Two options are available: 'No IPv6 CIDR block' (selected) and 'Amazon-provided IPv6 CIDR block' (unselected).
- Tenancy:** A dropdown menu is set to 'Default'.
- Number of Availability Zones (AZs):** Three radio buttons are present, with '2' selected.

Preview

The preview section shows a diagram of the VPC configuration. It includes a box for the VPC named 'ohioregion-vpc'. This VPC is connected to four subnets: two in the 'us-east-2a' availability zone ('ohioregion-subnet-public1-us-east-' and 'ohioregion-subnet-private1-us-east-') and two in the 'us-east-2b' availability zone ('ohioregion-subnet-public2-us-east-' and 'ohioregion-subnet-private2-us-east-'). These subnets are then connected to three route tables: 'ohioregion-rtb-public', 'ohioregion-rtb-private1-us-east-2a', and 'ohioregion-rtb-private2-us-east-2b'.

aws

Services

Search

[Alt+S]

Ohio

salvaddi

two AZs for high availability

123

Customize AZs

Number of public subnets

Info

The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

02

Number of private subnets

Info

The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

024

Customize subnets CIDR blocks

NAT gateways (\$)

Info

Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway.

NoneIn 1 AZ1 per AZ

VPC endpoints

Info

Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

NoneS3 Gateway

DNS options

Info

☒ Enable DNS hostnames

☒ Enable DNS resolution

Additional tags

Cancel

Create VPC

aws

Services

Search

[Alt+S]

Ohio

salvaddi

VPC

>

Your VPCs

>

Create VPC

>

Create VPC resources

Create VPC workflow

Success

Details

✔ Create VPC: vpc-0c83a639f5ba9db84

✔ Enable DNS hostnames

✔ Enable DNS resolution

✔ Verifying VPC creation: vpc-0c83a639f5ba9db84

✔ Create subnet: subnet-01647884e25333c01

✔ Create subnet: subnet-0cd0204d743463b3b

✔ Create subnet: subnet-031508c158b642ea6

✔ Create subnet: subnet-094e2dd0331412b4a

✔ Create internet gateway: igw-0688619aed706b908

✔ Attach internet gateway to the VPC

✔ Create route table: rtb-08341cbe906ec0d5d

✔ Create route

✔ Associate route table

✔ Associate route table

✔ Create route table: rtb-0efa4ce15d1458fc8

✔ Associate route table

✔ Create route table: rtb-0d1cc4e6c37f55fd9

✔ Associate route table

✔ Verifying route table creation

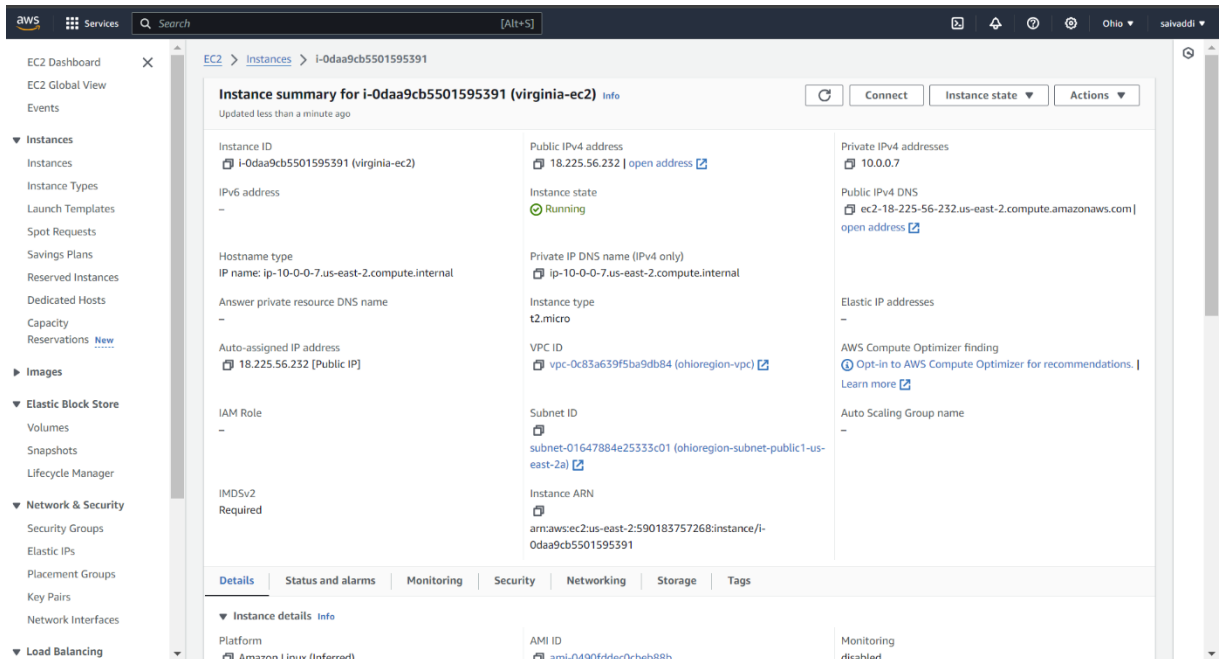
View VPC

https://us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-...

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 2:

1. Create a EC2 with custom VPC (which is created above) in Ohio Region with Amazon Linux.



2. Install a nginx in EC2 and write the file index.html in /usr/share/nginx/html.

Command: `yum install nginx -y`

```
sat@PAVAN-PC: ~/Desktop (pavan)
$ ssh -i "ohiokey.pem" ec2-user@ec2-18-225-56-232.us-east-2.compute.amazonaws.com
The authenticity of host 'ec2-18-225-56-232.us-east-2.compute.amazonaws.com (18.225.56.232)' can't be established.
ED25519 key fingerprint is SHA256:1h3IT0do4LHQShf4fwKItSJEbatCaj9vTo+FygdRmo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-225-56-232.us-east-2.compute.amazonaws.com' (ED25519) to the list of known hosts.

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-10-0-0-7 ~]$ sudo -i
[root@ip-10-0-0-7 ~]# yum update -y && yum install nginx -y
Last metadata expiration check: 0:14:09 ago on Wed Sep 4 06:59:45 2024.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 0:14:09 ago on Wed Sep 4 06:59:45 2024.
Dependencies resolved.

=====
Package                               Architecture    Version                               Repository    Size
Installing:
nginx                                  x86_64          1:1.24.0-1.amzn2023.0.2              amazonlinux   32 k
Installing dependencies:
generic-logos-httpd                   noarch          18.0.0-12.amzn2023.0.3                amazonlinux   19 k
gperftools-libs                       x86_64          2.9.1-1.amzn2023.0.3                  amazonlinux   308 k
libunwind                             x86_64          1.4.0-5.amzn2023.0.2                  amazonlinux   66 k
nginx-core                             x86_64          1:1.24.0-1.amzn2023.0.2                amazonlinux   586 k
nginxfilesystem                       noarch          1:1.24.0-1.amzn2023.0.2                amazonlinux   9.1 k
nginx-mimetypes                       noarch          2.1.49-3.amzn2023.0.3                  amazonlinux   21 k
Transaction Summary
Install 7 Packages
Total download size: 1.0 M
Installed size: 3.4 M
Downloading Packages:
(1/7): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.rpm                295 kB/s | 19 kB  00:00
(2/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm                    3.8 MB/s | 308 kB 00:00
(3/7): nginx-1.24.0-1.amzn2023.0.2.x86_64.rpm                             1.8 MB/s | 32 kB  00:00
(4/7): libunwind-1.4.0-5.amzn2023.0.2.x86_64.rpm                          782 kB/s | 66 kB  00:00
(5/7): nginxfilesystem-1.24.0-1.amzn2023.0.2.noarch.rpm                   501 kB/s | 9.1 kB 00:00
(6/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm                    1.1 MB/s | 21 kB  00:00
(7/7): nginx-core-1.24.0-1.amzn2023.0.2.x86_64.rpm                        15 MB/s | 586 kB 00:00
Total
-----
5.7 MB/s | 1.0 MB 00:00
Running transaction check
```

3. Write "This is Ohio server" in index.html.

Step 3: Create a VPC in N.Virginia region.

Create VPC [Info](#)

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.

VPC settings

Resources to create [Info](#)
Create only the VPC resource or the VPC and other networking resources.

☐ VPC only ☒ VPC and more

Name tag auto-generation [Info](#)
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

☒ Auto-generate

IPv4 CIDR block [Info](#)
Determine the starting IP and the size of your VPC using CIDR notation.

256 IPs
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
☒ No IPv6 CIDR block
☐ Amazon-provided IPv6 CIDR block

Tenancy [Info](#)
Default

Number of Availability Zones (AZs) [Info](#)
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.

1 2 3

Preview

VPC [Show details](#)
Your AWS virtual network

Subnets (4)
Subnets within this VPC

us-east-1a

- virginiaregion-subnet-public1-us-
- virginiaregion-subnet-private1-us-

us-east-1b

- virginiaregion-subnet-public2-us-
- virginiaregion-subnet-private2-us-

Route tables (3)
Route network traffic to resources

- virginiaregion-rtb-public
- virginiaregion-rtb-private1-us-east-1a
- virginiaregion-rtb-private2-us-east-1b

two AZs for high availability

1 2 3

► **Customize AZs**

Number of public subnets [Info](#)
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 2

Number of private subnets [Info](#)
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

0 2 4

► **Customize subnets CIDR blocks**

NAT gateways (\$) [Info](#)
Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway.

None In 1 AZ 1 per AZ

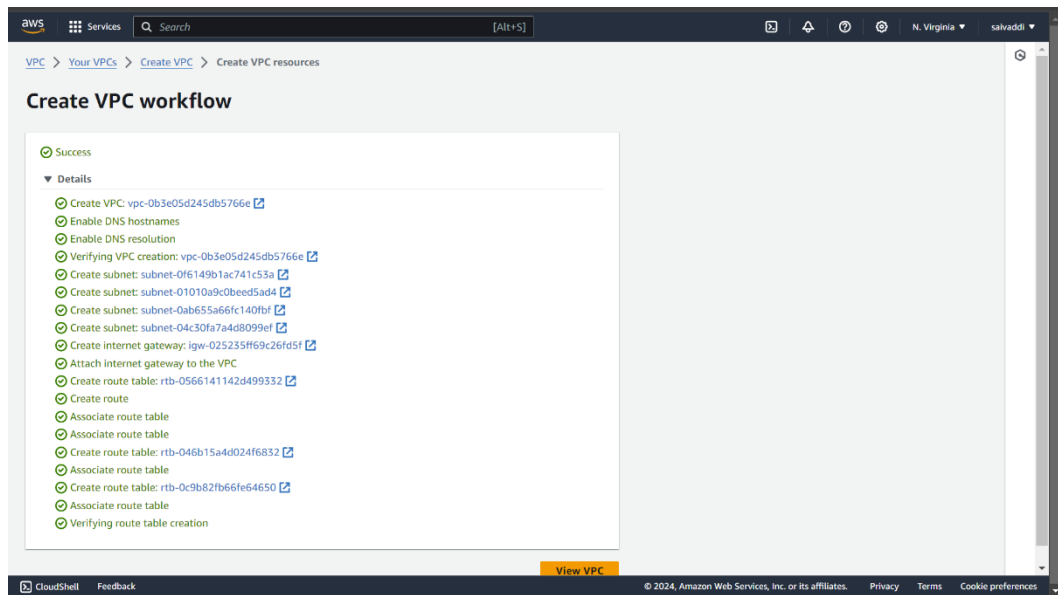
VPC endpoints [Info](#)
Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

None S3 Gateway

DNS options [Info](#)
☒ Enable DNS hostnames
☒ Enable DNS resolution

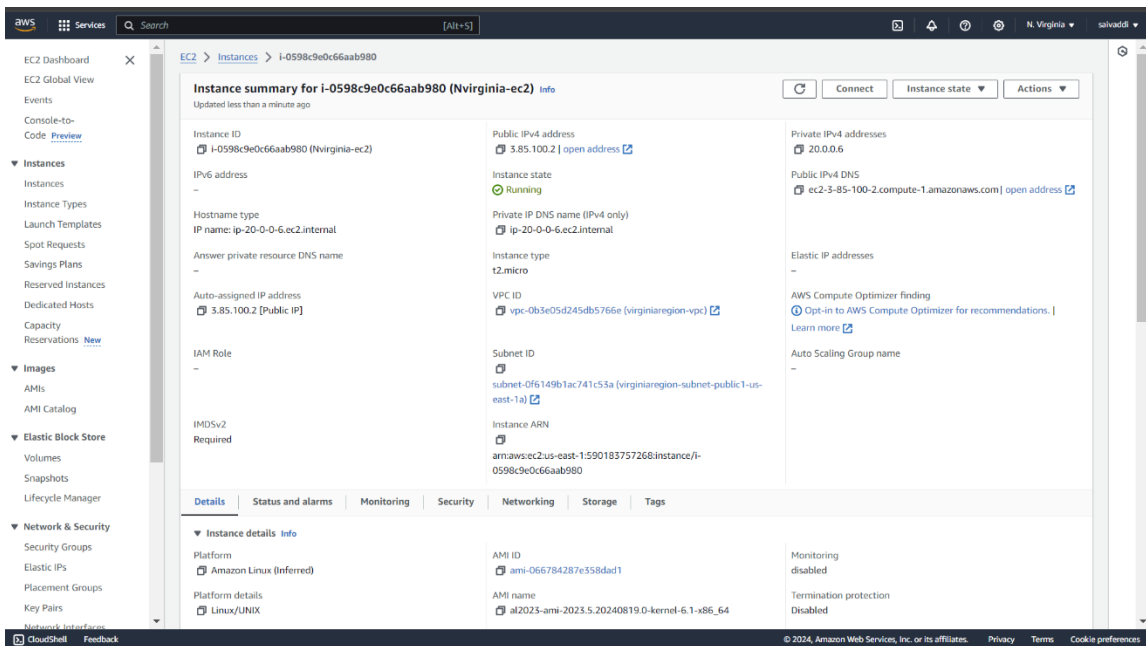
► **Additional tags**

Cancel **Create VPC**



Step 4:

1. Create a EC2 with custom VPC (which is created above) in N.Virginia Region with Amazon Linux.



2. Install a nginx in EC2 and write the file index.html in /usr/share/nginx/html.
3. Write "This is Virginia server" in index.html.

Step 5:

1. Create a peering connection in Ohio Region (VPC -> Peering Connection).

Create peering connection

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately.

Peering connection settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.

Select a local VPC to peer with

VPC ID (Requester)

VPC CIDRs for vpc-0c83a639f5ba9db84 (ohioregion-vpc)

CIDR	Status	Status reason
10.0.0.0/24	Associated	-

Select another VPC to peer with

Account
☒ My account
☐ Another account

Region

Select another VPC to peer with

Account
☒ My account
☐ Another account

Region
☐ This Region (us-east-2)
☒ Another Region

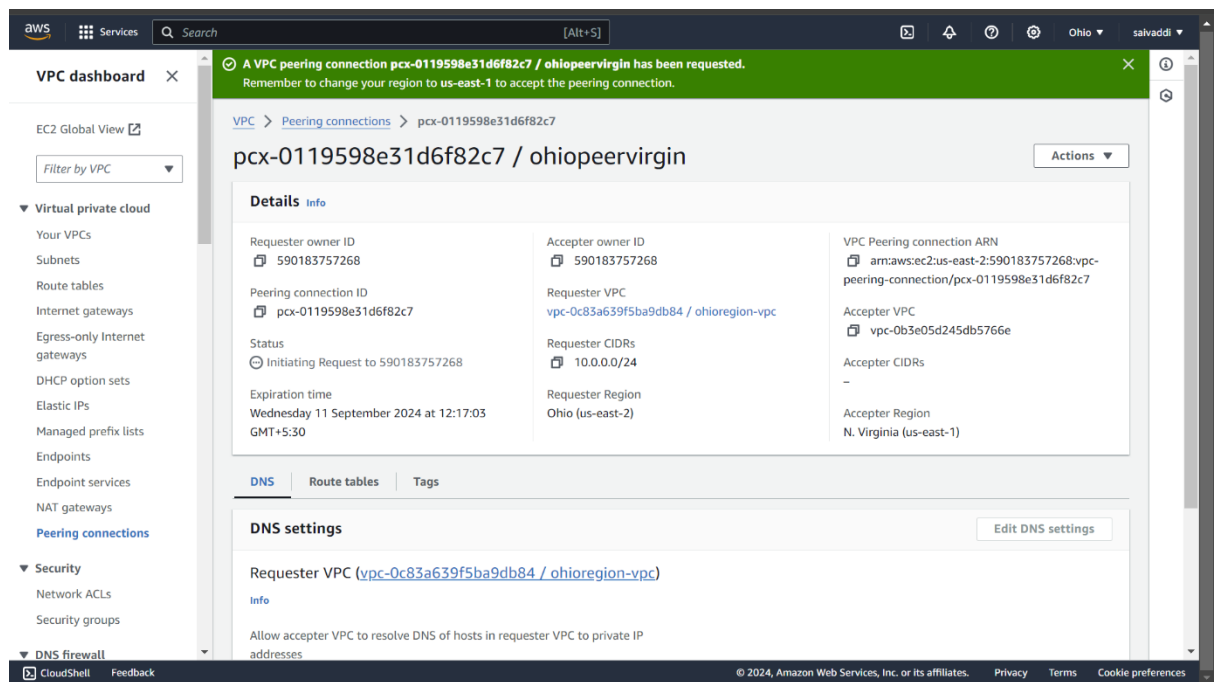
VPC ID (Acceptor)

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

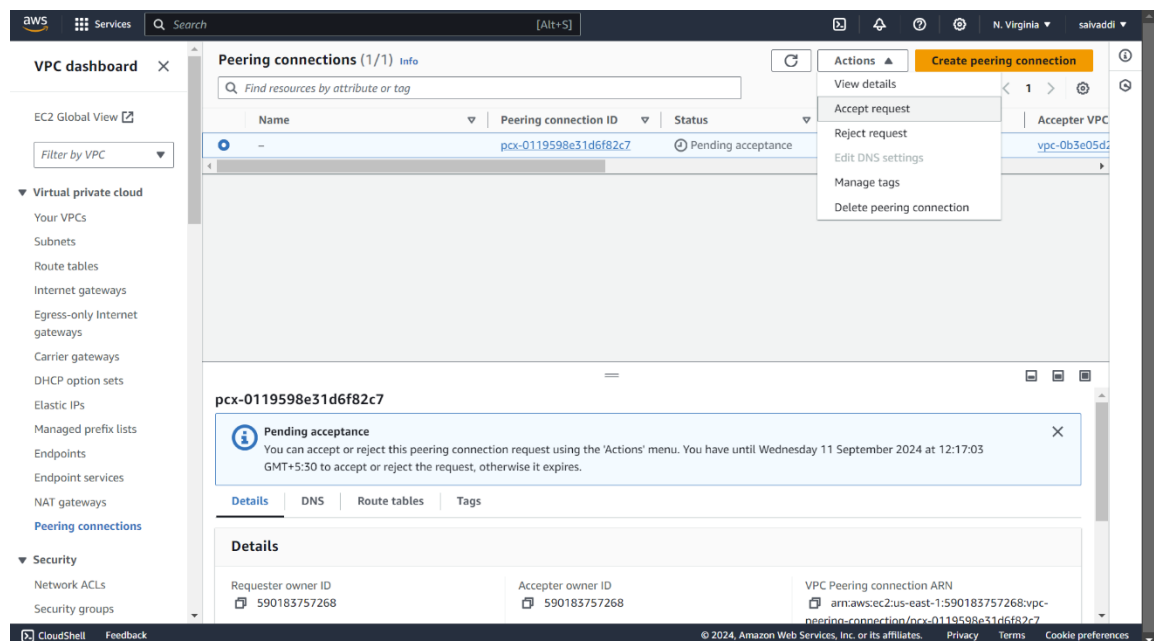
Key: Value - optional:

You can add 49 more tags.

2. After successfully creation of peering connection in Ohio region Request sent to N.Virginia Region.



3. Now Accept the Peering Connection in N.Virginia Region.



Step 6: Associate the peering connection in public route table in both regions.
(VPC -> Route Table -> Edit routes).

In Ohio Region,

Destination	Target	Status	Propagated
10.0.0.0/24	local	Active	No
0.0.0.0/0	Internet Gateway	Active	No
20.0.0.0/24	Peering Connection	-	No

In N.Virginia Region,

Destination	Target	Status	Propagated
20.0.0.0/24	local	Active	No
0.0.0.0/0	Internet Gateway	Active	No
10.0.0.0/24	Peering Connection	-	No

Step 7: Check the Peering Connection is established or not by pasting the ipv4 of ohio region EC2 in N.Virginia Region EC2 (vice-versa).

command: curl ipv4

```
root@ip-10-0-0-7:/usr/share/nginx/html
[root@ip-10-0-0-7 ~]# cd /usr/share/nginx/html
[root@ip-10-0-0-7 html]# rm index.html
rm: remove regular file 'index.html'? yes
[root@ip-10-0-0-7 html]# vi index.html
[root@ip-10-0-0-7 html]# sytemctl nginx.html
-bash: sytemctl: command not found
[root@ip-10-0-0-7 html]# systemctl nginx.html
Unknown command verb nginx.html.
[root@ip-10-0-0-7 html]# systemctl restart nginx
[root@ip-10-0-0-7 html]# curl 10.0.0.7
This is ohio server ec2
[root@ip-10-0-0-7 html]# curl 20.0.0.6
this is virginia server
[root@ip-10-0-0-7 html]#
```

```
root@ip-20-0-0-6:/usr/share/nginx/html
[ec2-user@ip-20-0-0-6 ~]$ sudo -i
[root@ip-20-0-0-6 ~]# pwd
/root
[root@ip-20-0-0-6 ~]# cd /usr/share/nginx/html
[root@ip-20-0-0-6 html]# curl 20.0.0.6
this is virginia server
[root@ip-20-0-0-6 html]# curl 10.0.0.7
This is ohio server ec2
[root@ip-20-0-0-6 html]#
```