Connecting to private Ec2 Instances using Systems Manager

Projects Descriptions:

- **Leverage the Power of AWS Systems Manager:**

  Use AWS Systems Manager to establish connections between instances in the private subnet without the need for a keypair

- **Utilize the Architecture From Our Previous Project:**

  Incorporate the architecture we previously built. For reference, please check our previous post: Previous Project Post: https://www.linkedin.com/posts/rogmer-bulaclac_project-file-activity-7144285021472796672-Cvm4?utm_source=share&utm_medium=member_desktop

- **Reference Links for This Project:**

  1. *https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/connect-to-an-amazon-ec2-instance-by-using-session-manager.html*
  2. *https://dev.to/aws-builders/connecting-to-private-ec2-instances-using-systems-manager-a-hands-on-guide-33m*

- **Creation of Instances, IAM Roles, and VPC Endpoints Using Terraform:**

  In this project, utilize Terraform to create instances, IAM roles, and VPC endpoints.

Prerequiestes:

1. **AWS** Acccount
2. AWS ACCESS KEY and SECRET ACCESS KEY with administrator ACCESS
3. Terraform Must be Installed
4. GIT must be Installed
5. Text Editor git

Instructions:

1. Clone project repository:

git clone -b terraform https://github.com/robudexIT/awsdevopsproject.git

2. cd to terraform project directory:

   cd awsdevopsproject/terraform/vpcpeering/

3. Open a**wsdevopsproject/terraform/vpcpeering/main.tf** with your chosen text editor. Uncomment the **tokyo_ec2** and **osaka_ec2** modules, and save the file.

```
# module "tokyo_ec2" {
#     source = "./ec2"
#     region = "ap-northeast-1"
#     instance_name = "tokyo_instance"
#     subnet_id = module.tokyo_vpc.private_subnet_id
#     instance_sg_id = module.tokyo_vpc.private_sg_id
#    #  iam_instance_profile = module.iam.instance_profile_name
# }

# module "osaka_ec2" {
#     source = "./ec2"
#     providers = {
#     aws = aws.osaka
#     }
#     region = "ap-northeast-3"
#     instance_name = "osaka_instance"
#     subnet_id = module.osaka_vpc.private_subnet_id
#     instance_sg_id = module.osaka_vpc.private_sg_id
#    #  iam_instance_profile = module.iam.instance_profile_name
# }
```

REMOVE THIS

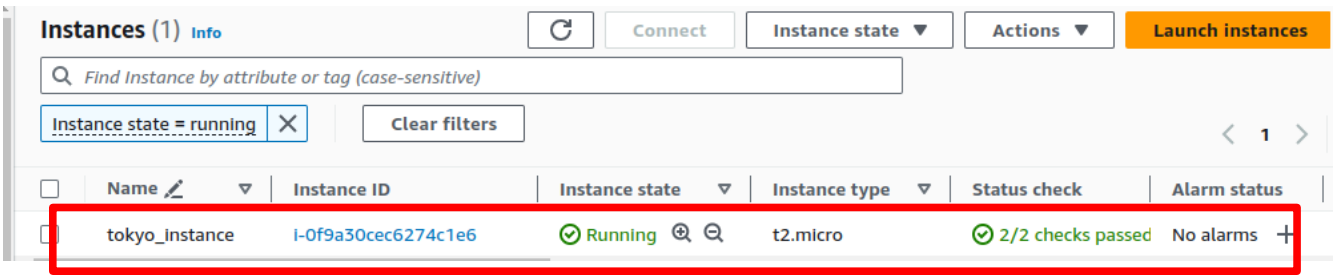
```
module "tokyo_ec2" {
    source = "./ec2"
    region = "ap-northeast-1"
    instance_name = "tokyo_instance"
    subnet_id = module.tokyo_vpc.private_subnet_id
    instance_sg_id = module.tokyo_vpc.private_sg_id
   #  iam_instance_profile = module.iam.instance_profile_name
}

module "osaka_ec2" {
    source = "./ec2"
    providers = {
    aws = aws.osaka
    }
    region = "ap-northeast-3"
    instance_name = "osaka_instance"
    subnet_id = module.osaka_vpc.private_subnet_id
    instance_sg_id = module.osaka_vpc.private_sg_id
   #  iam_instance_profile = module.iam.instance_profile_name
}
```

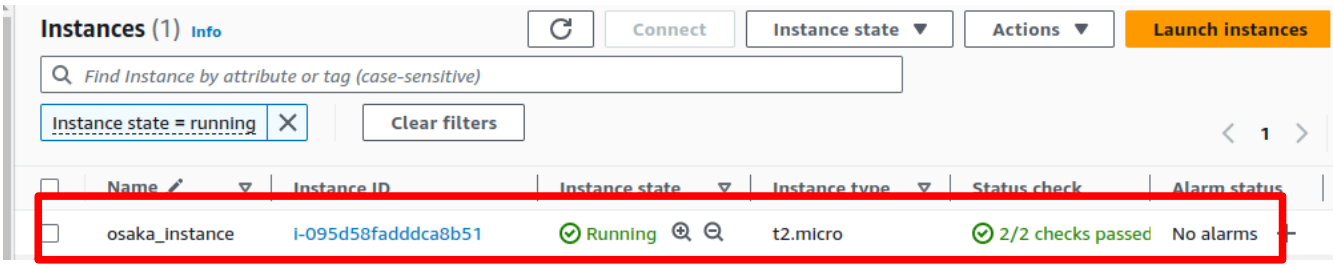
MUST BE LIKE THIS

4. **Run these commands:**

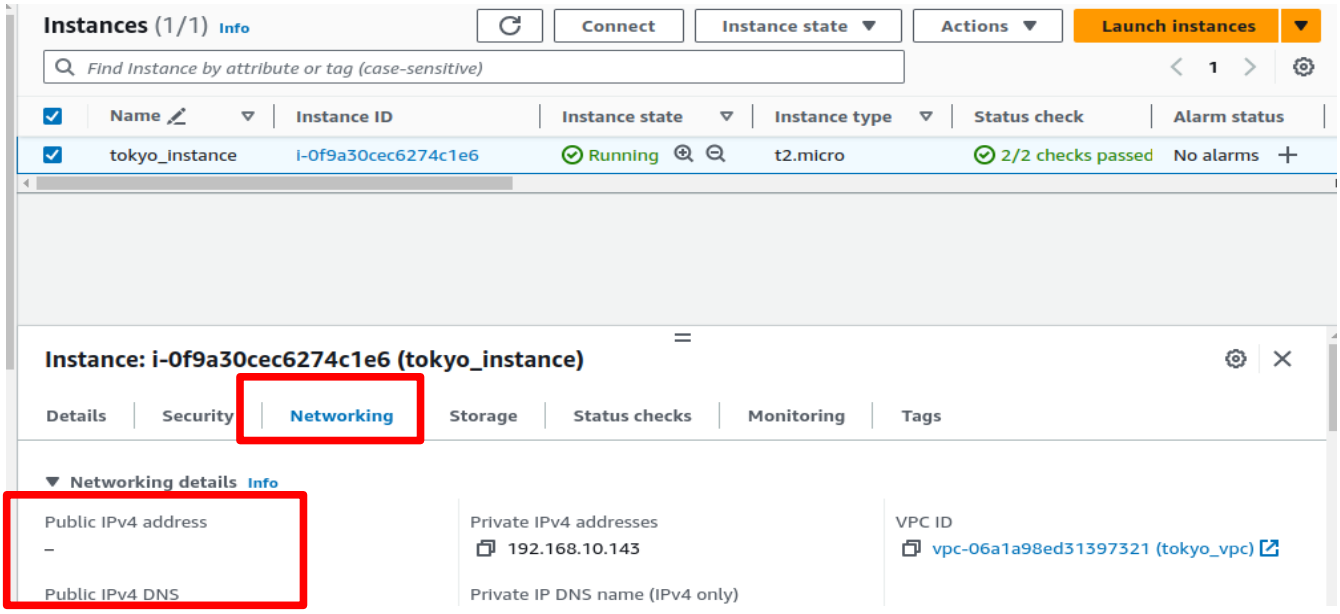
- terraform init
- terraform validate
- terraform plan
- terraform apply – auto-approve

5. When Terraform completes its execution, open the AWS console in two separate tabs. In one tab, navigate to the **Tokyo region**, and in the other tab, navigate to the **Osaka region**. Once there, go to the EC2 section in each region and verify if the **instance's** have been successfully created.

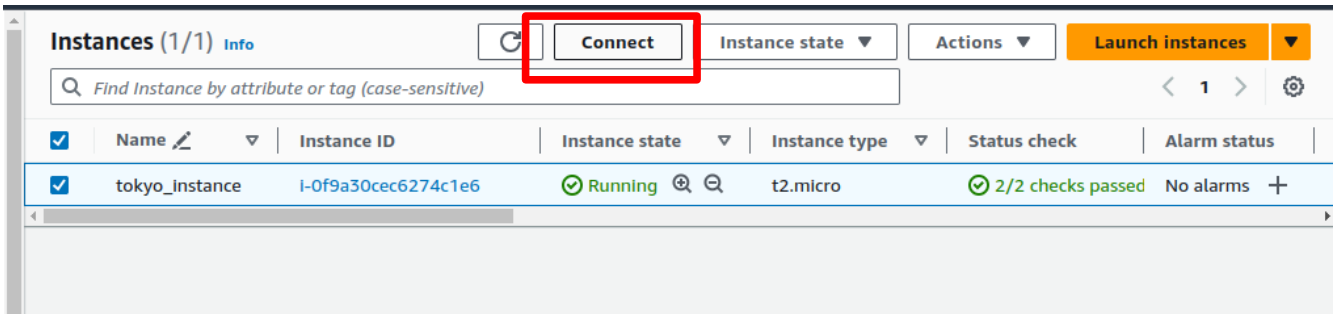


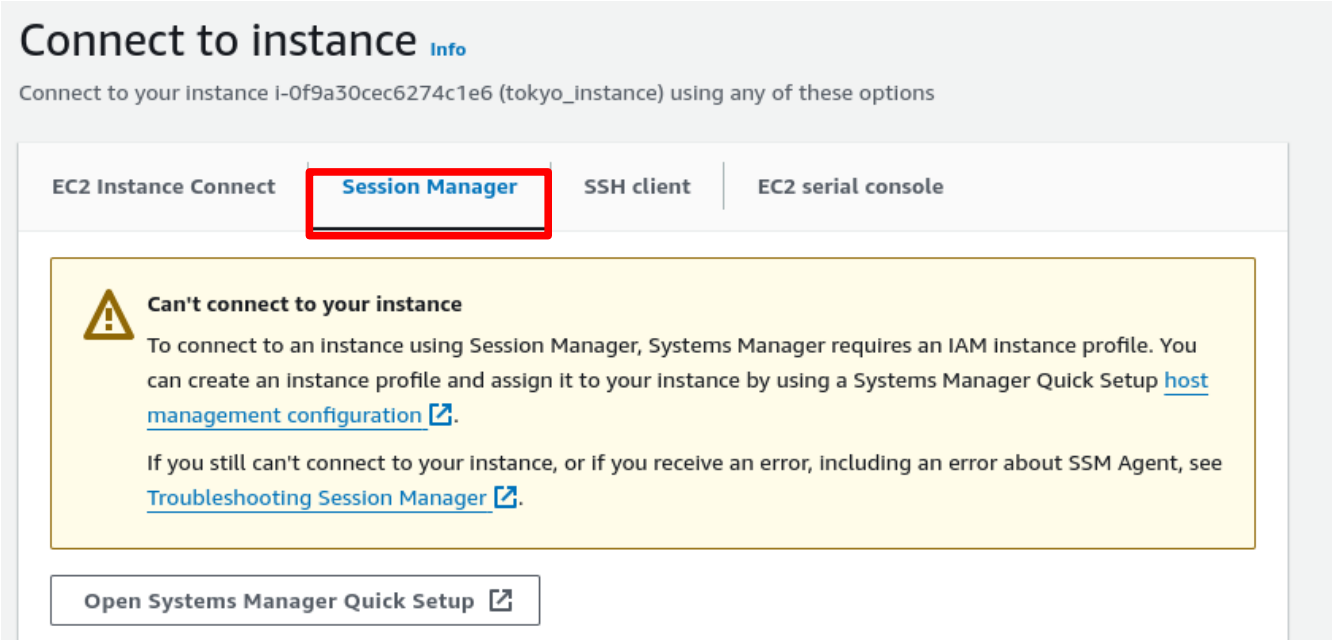


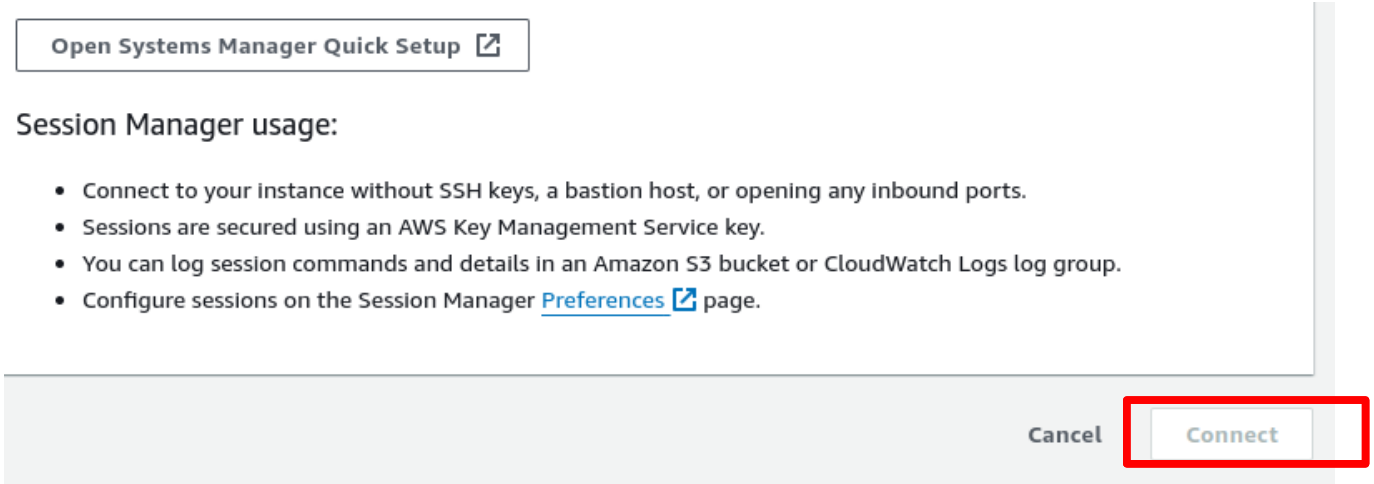
6. Navigate to '**Instance**,' then click on '**Networking**.' Verify that no public **IP is assigned** and confirm that the **public DNS** remains unset



7. Click the **connect** button the select **Session Manager.**







As you notice, the **connect** button is disable

8**.** Reopen **awsdevopsproject/terraform/vpcpeering/main.tf** with your chosen text editor. Uncomment the **iam, tokyo_vpc_endpoint, and osaka_vpc_endpoint** modules, and save the file.

```
# module "iam" {
#   source = "./iam"
# }


# module "tokyo_vpc_endpoint" {
#    source = "./vpc_endpoint"
#    region = "ap-northeast-1"
#    vpc_id = module.tokyo_vpc.vpc_id
#    security_group_id = module.tokyo_vpc.private_sg_id
#    subnet_id = module.tokyo_vpc.private_subnet_id
#    route_table_id = module.tokyo_vpc.private_rt_id
# }

# module "osaka_vpc_endpoint" {

#    source = "./vpc_endpoint"
#    providers = {
#       aws = aws.osaka
#    }
#    region = "ap-northeast-3"
#    vpc_id = module.osaka_vpc.vpc_id
#    security_group_id = module.osaka_vpc.private_sg_id
#    subnet_id = module.osaka_vpc.private_subnet_id
#    route_table_id = module.osaka_vpc.private_rt_id

# }

# output "instance_profile_name"  {
#     value = module.iam.instance_profile_name
# }
```

REMOVE THIS

MUST BE LIKE THIS

```
module "iam" {
  source = "./iam"
}


module "tokyo_vpc_endpoint" {
   source = "./vpc_endpoint"
   region = "ap-northeast-1"
   vpc_id = module.tokyo_vpc.vpc_id
   security_group_id = module.tokyo_vpc.private_sg_id
   subnet_id = module.tokyo_vpc.private_subnet_id
   route_table_id = module.tokyo_vpc.private_rt_id
}

module "osaka_vpc_endpoint" {

   source = "./vpc_endpoint"
   providers = {
      aws = aws.osaka
   }
   region = "ap-northeast-3"
   vpc_id = module.osaka_vpc.vpc_id
   security_group_id = module.osaka_vpc.private_sg_id
   subnet_id = module.osaka_vpc.private_subnet_id
   route_table_id = module.osaka_vpc.private_rt_id

}

output "instance_profile_name"  {
    value = module.iam.instance_profile_name
}
```

9. **Run these commands:**

- terraform init
- terraform validate
- terraform plan
- terraform apply – auto-approve

```
(base) robudex@robudex-Dell-System-Vostro-3360:~/SBTPHPROJECTS/awsdevopsproject/terraform/vpcpeering$ terraform init

Initializing the backend...
Initializing modules...
- osaka_vpc_endpoint in vpc_endpoint
- tokyo_vpc_endpoint in vpc_endpoint

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.31.0
```

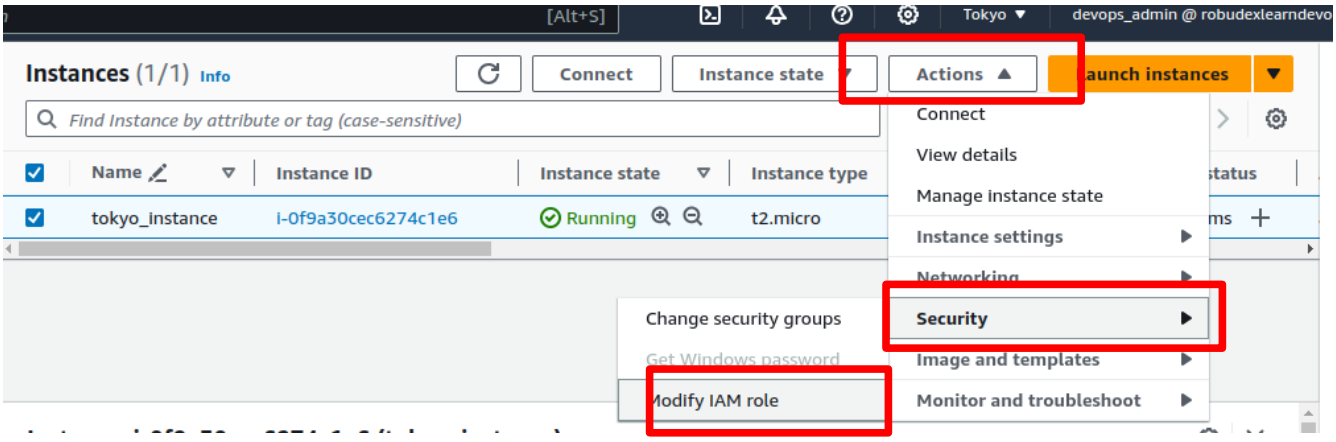
```
(base) robudex@robudex-Dell-System-Vostro-3360:~/SBTPHPROJECTS/awsdevopsproject/terraform/vpcpeering$ terraform validate

  Warning: Reference to undefined provider

    on main.tf line 13, in module "osaka_vpc":
    13:     aws = aws.osaka

  There is no explicit declaration for local provider name "aws" in module.osaka_vpc, so Terraform is assuming you mean to pass a
  configuration for "hashicorp/aws".

  If you also control the child module, add a required_providers entry named "aws" with the source address "hashicorp/aws".

  (and 2 more similar warnings elsewhere)

Success! The configuration is valid, but there were some validation warnings as shown above.
```

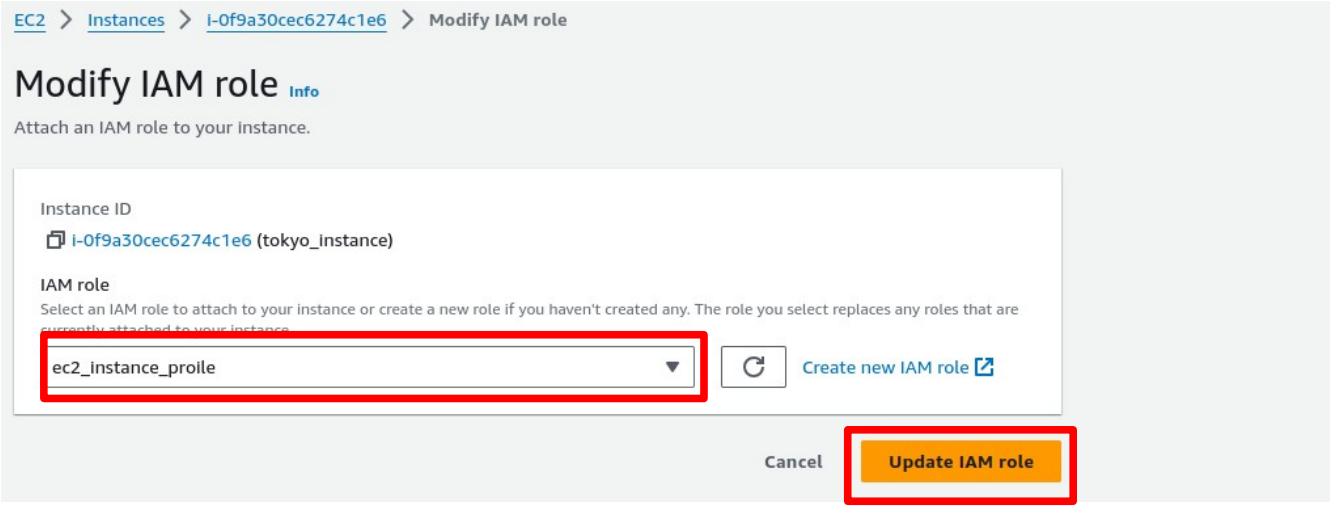
```
(base) robudex@robudex-Dell-System-Vostro-3360:~/SBTPHPROJECTS/awsdevopsproject/terraform/vpcpeering$ terraform plan
module.tokyo_vpc.data.aws_availability_zones.available: Reading...
module.tokyo_vpc.aws_vpc.company_vpc: Refreshing state... [id=vpc-06a1a98ed31397321]
module.osaka_vpc.data.aws_availability_zones.available: Reading...
module.osaka_vpc.aws_vpc.company_vpc: Refreshing state... [id=vpc-0b9940c62c7435fb4]
module.tokyo_vpc.data.aws_availability_zones.available: Read complete after 1s [id=ap-northeast-1]
module.osaka_vpc.data.aws_availability_zones.available: Read complete after 0s [id=ap-northeast-3]
module.osaka_vpc.aws_subnet.private_subnet: Refreshing state... [id=subnet-0be608ee5909fec00]
module.osaka_vpc.aws_internet_gateway.igw: Refreshing state... [id=igw-0e47b5bf0d4a3c432]
module.osaka_vpc.aws_subnet.public_subnet02: Refreshing state... [id=subnet-03d0e92b83951abec]
module.osaka_vpc.aws_subnet.public_subnet01: Refreshing state... [id=subnet-0a9b3b9d81aedd7b1]
module.osaka_vpc.aws_route_table.private_rt: Refreshing state... [id=rtb-02de27564357402bd]
module.osaka_vpc.aws_security_group.public_sg: Refreshing state... [id=sg-04f47dd5e72b258fb]
module.tokyo_vpc.aws_subnet.public_subnet02: Refreshing state... [id=subnet-07eff4042102132c5]
module.tokyo_vpc.aws_subnet.public_subnet01: Refreshing state... [id=subnet-0cadba55c80ffcf2b]
```

```
Plan: 14 to add, 2 to change, 0 to destroy.

Changes to Outputs:
  + instance_profile_name           = "ec2_instance_proile"

  Warning: Reference to undefined provider

    on main.tf line 13, in module "osaka_vpc":
    13:     aws = aws.osaka

  There is no explicit declaration for local provider name "aws" in module.osaka_vpc, so Terraform is assuming you mean to pass a
  configuration for "hashicorp/aws".

  If you also control the child module, add a required_providers entry named "aws" with the source address "hashicorp/aws".

  (and 2 more similar warnings elsewhere)
```

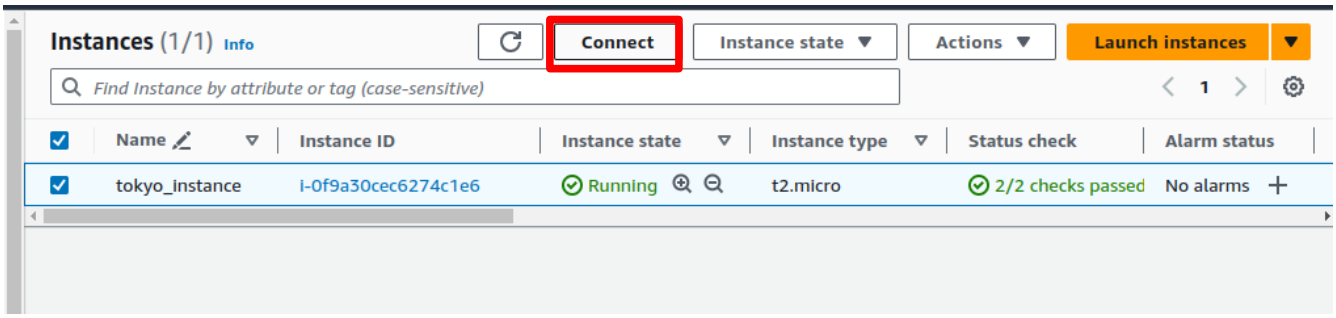
```
  If you also control the child module, add a required_providers entry named "aws" with the source address "hashicorp/aws".

  (and 2 more similar warnings elsewhere)

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

instance_profile_name = "ec2_instance_proile"
osaka_private_route_table_id = "rtb-02de27564357402bd"
osaka_private_security_group_id = "sg-06fd4e60d435dbb36"
osaka_private_subnet_id = "subnet-0be608ee5909fec00"
osaka_vpc_id = "vpc-0b9940c62c7435fb4"
tokyo_private_route_table_id = "rtb-040a1ad4c35b615db"
tokyo_private_security_group_id = "sg-084211f1dae196292"
tokyo_private_subnet_id = "subnet-0579e1ee30dddb339"
tokyo_vpc_id = "vpc-06a1a98ed31397321"
```

10. When Terraform completes its execution, open the AWS console in two separate tabs. In one tab, navigate to the **Tokyo region**, and in the other tab, navigate to the **Osaka region**. Once there, go to the EC2 section in each region. Then Modify IAM role



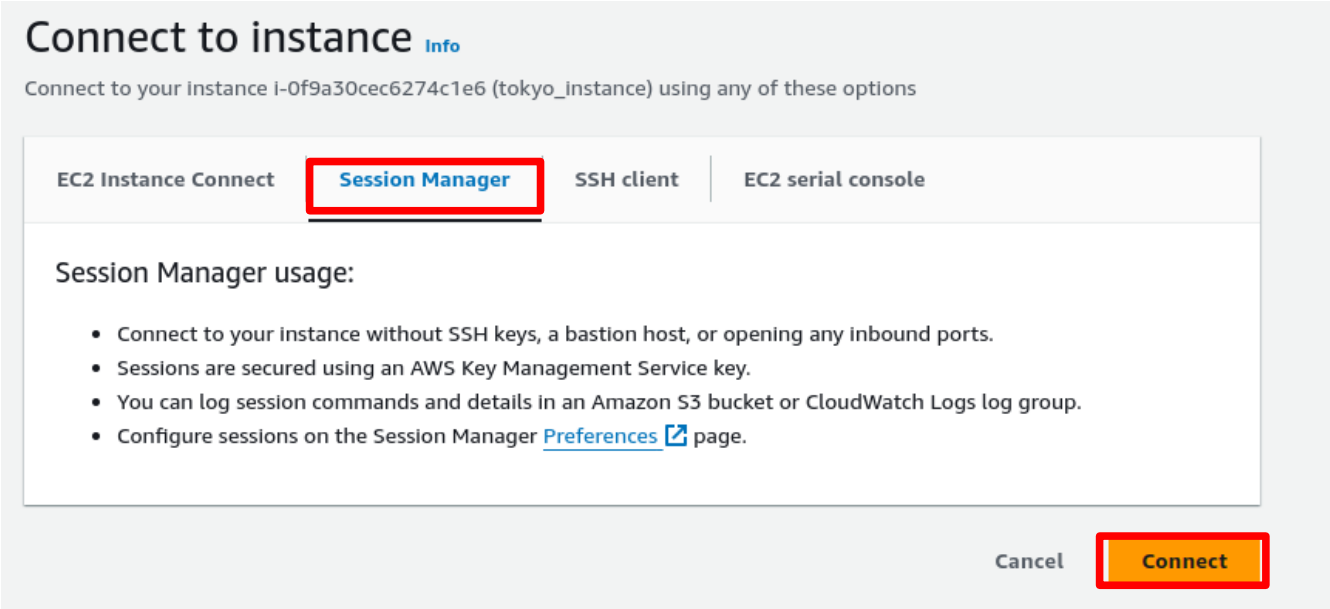


Click connect



Select **Session Manager-**> Click **Connect**

Note: *Sometimes, the newly updated IAM Role may take some time to take effect. If you don't see a similar result on the screenshot below, refresh the page until it appears as shown in the screenshot.*



Do similar steps on Osaka Region **EC2 Instance**.

11. Testing connectivy and Install packages like **httpd**

Instance Tokyo and Osaksa Instance type (**ifconfig**)

```
sh-4.2$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 192.168.10.143  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::4e3:2fff:fec1:b601  prefixlen 64  scopeid 0x20<link>
        ether 06:e3:2f:c1:b6:01  txqueuelen 1000  (Ethernet)
        RX packets 249056  bytes 353692943 (337.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 110040  bytes 7101236 (6.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 48  bytes 3888 (3.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 48  bytes 3888 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sh-4.2$
```

```
sh-4.2$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 172.16.10.99  netmask 255.255.255.0  broadcast 172.16.10.255
        inet6 fe80::ca0:91ff:fe25:6e74  prefixlen 64  scopeid 0x20<link>
        ether 0e:a0:91:25:6e:74  txqueuelen 1000  (Ethernet)
        RX packets 1681  bytes 260588 (254.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2168  bytes 256064 (250.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 48  bytes 3888 (3.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 48  bytes 3888 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sh-4.2$
```

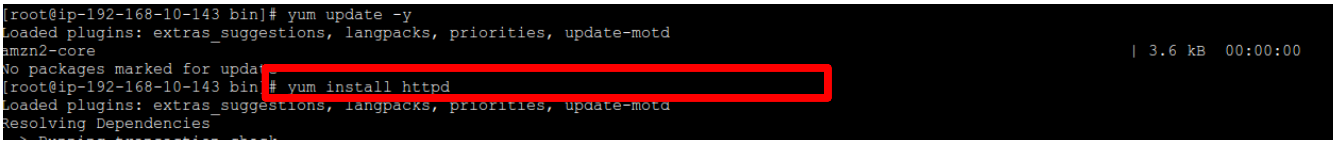
Now, ping the IP address of the Osaka instance from the Tokyo instance, and vice versa

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 192.168.10.143  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::4e3:2fff:fec1:b601  prefixlen 64  scopeid 0x20<link>
        ether 06:e3:2f:c1:b6:01  txqueuelen 1000  (Ethernet)
        RX packets 249753  bytes 353835183 (337.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 110804  bytes 7247231 (6.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 48  bytes 3888 (3.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 48  bytes 3888 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sh-4.2$ ping 172.16.10.99
PING 172.16.10.99 (172.16.10.99) 56(84) bytes of data.
64 bytes from 172.16.10.99: icmp_seq=1 ttl=255 time=8.41 ms
64 bytes from 172.16.10.99: icmp_seq=2 ttl=255 time=8.50 ms
64 bytes from 172.16.10.99: icmp_seq=3 ttl=255 time=8.54 ms
64 bytes from 172.16.10.99: icmp_seq=4 ttl=255 time=8.47 ms
64 bytes from 172.16.10.99: icmp_seq=5 ttl=255 time=8.49 ms
64 bytes from 172.16.10.99: icmp_seq=6 ttl=255 time=8.53 ms
^C
--- 172.16.10.99 ping statistics ---
```

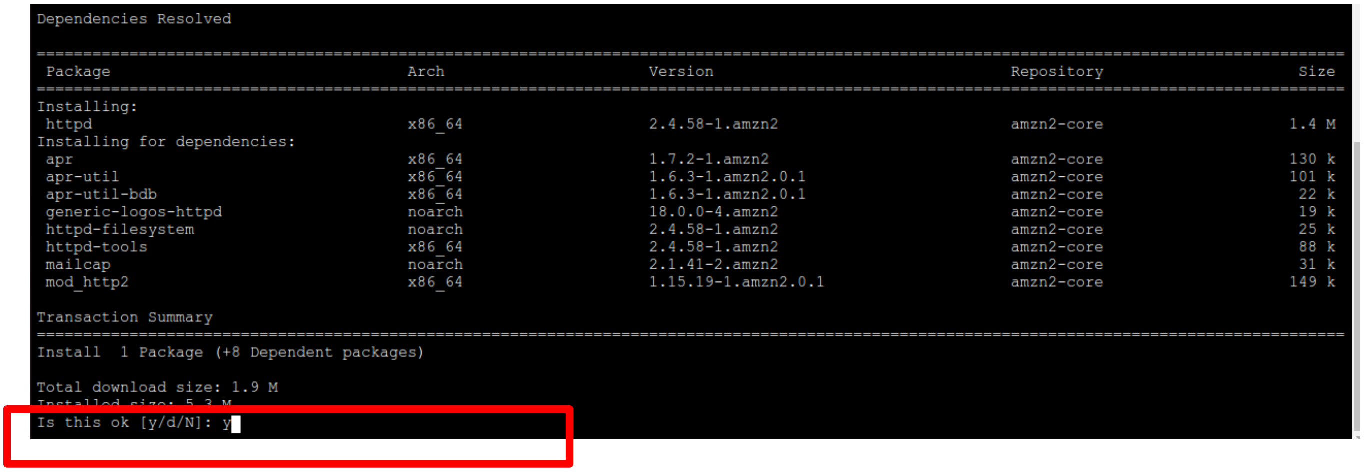
```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 172.16.10.99  netmask 255.255.255.0  broadcast 172.16.10.255
        inet6 fe80::ca0:91ff:fe25:6e74  prefixlen 64  scopeid 0x20<link>
        ether 0e:a0:91:25:6e:74  txqueuelen 1000  (Ethernet)
        RX packets 1855  bytes 297542 (290.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2339  bytes 289157 (282.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 48  bytes 3888 (3.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 48  bytes 3888 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sh-4.2$ ping 192.168.10.143
PING 192.168.10.143 (192.168.10.143) 56(84) bytes of data.
64 bytes from 192.168.10.143: icmp_seq=1 ttl=255 time=7.81 ms
64 bytes from 192.168.10.143: icmp_seq=2 ttl=255 time=7.81 ms
64 bytes from 192.168.10.143: icmp_seq=3 ttl=255 time=7.80 ms
64 bytes from 192.168.10.143: icmp_seq=4 ttl=255 time=7.76 ms
64 bytes from 192.168.10.143: icmp_seq=5 ttl=255 time=7.85 ms
64 bytes from 192.168.10.143: icmp_seq=6 ttl=255 time=8.21 ms
^C
--- 192.168.10.143 ping statistics ---
```

Now Let’s try to install **httpd** package

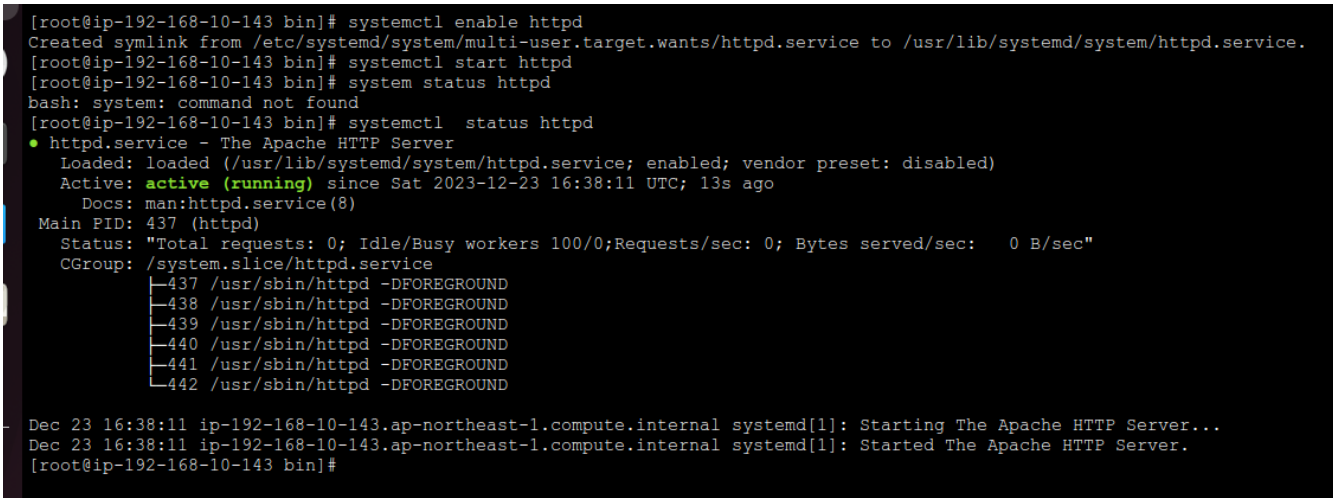
```
sh-4.2$
sh-4.2$
sh-4.2$ sudo su
[root@ip-192-168-10-143 bin]# yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
No packages marked for update
[root@ip-192-168-10-143 bin]# yum install httpd
```



```
[root@ip-192-168-10-143 bin]# yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                                                    | 3.6 kB  00:00:00
No packages marked for update
[root@ip-192-168-10-143 bin]# yum install httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
```



```
Dependencies Resolved

=====================================================================================================================================
 Package                             Arch                     Version                             Repository                   Size
=====================================================================================================================================
Installing:
 httpd                               x86_64                   2.4.58-1.amzn2                      amzn2-core                  1.4 M
Installing for dependencies:
 apr                                 x86_64                   1.7.2-1.amzn2                       amzn2-core                  130 k
 apr-util                            x86_64                   1.6.3-1.amzn2.0.1                   amzn2-core                  101 k
 apr-util-bdb                        x86_64                   1.6.3-1.amzn2.0.1                   amzn2-core                   22 k
 generic-logos-httpd                 noarch                   18.0.0-4.amzn2                      amzn2-core                   19 k
 httpd-filesystem                    noarch                   2.4.58-1.amzn2                      amzn2-core                   25 k
 httpd-tools                         x86_64                   2.4.58-1.amzn2                      amzn2-core                   88 k
 mailcap                             noarch                   2.1.41-2.amzn2                      amzn2-core                   31 k
 mod_http2                           x86_64                   1.15.19-1.amzn2.0.1                 amzn2-core                  149 k

Transaction Summary
=====================================================================================================================================
Install  1 Package (+8 Dependent packages)

Total download size: 1.9 M
Installed size: 5.3 M
Is this ok [y/d/N]: y
```



```
[root@ip-192-168-10-143 bin]# systemctl enable httpd
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
[root@ip-192-168-10-143 bin]# systemctl start httpd
[root@ip-192-168-10-143 bin]# system status httpd
bash: system: command not found
[root@ip-192-168-10-143 bin]# systemctl  status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Sat 2023-12-23 16:38:11 UTC; 13s ago
     Docs: man:httpd.service(8)
 Main PID: 437 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─437 /usr/sbin/httpd -DFOREGROUND
           ├─438 /usr/sbin/httpd -DFOREGROUND
           ├─439 /usr/sbin/httpd -DFOREGROUND
           ├─440 /usr/sbin/httpd -DFOREGROUND
           ├─441 /usr/sbin/httpd -DFOREGROUND
           └─442 /usr/sbin/httpd -DFOREGROUND

Dec 23 16:38:11 ip-192-168-10-143.ap-northeast-1.compute.internal systemd[1]: Starting The Apache HTTP Server...
Dec 23 16:38:11 ip-192-168-10-143.ap-northeast-1.compute.internal systemd[1]: Started The Apache HTTP Server.
[root@ip-192-168-10-143 bin]#
```

12. This concludes the project. Once done, don't forget to terminate all resources created by Terraform. In your terminal, type the command:

- **terraform destroy -auto-approve**