1. When to use Elastic IP over Public IP

Public ips are allotted by AWS to all the “auto-assign ip” enabled instances , but whenever the instances are stopped and started, a new ip is given. However, if we wish to make our instance permanently available on a single ip (for long term projects) we should use elastic ips. Elastic ips are specially allocated ips by aws which is given to you only and it won’t be taken away even if the instance is stopped and started again.

2. Valid IP Ranges for LAN, Implication of using Public IP ranges for Private Network.

There are three ranges of addresses that can be used in a private network:

- 10.0. 0.0 – 10.255. 255.255.
- 172.16. 0.0 – 172.31. 255.255.
- 192.168. 0.0 – 192.168. 255.255.

If you use public addresses on your private network, t

3. List down the things to keep in mind while VPC peering.

1. Transitive VPC peering is not allowed. If VPC A is peered VPC B and VPC B is peered with VPC C then it doesn’t mean that VPC A is peered with VPC C, you need to explicitly peer them.
2. VPC’s with overlapping IP’s can’t be peered.
3. It is not possible to create a VPC peering connection between VPCs present in different regions.
4. Only one VPC peering connection is possible between two VPCs at a time
5. In case of following VPC peering connections, it is not allowed to extend the peering

4. CIDR of a VPC is <u>10.0.0.0/16</u>, if the subnet mask is /20 calculate the number of subnets that could be created from the VPC. Also find the number of IP in subnet.

Subnet mask of /20 means first 20 bits of the ip address are network bits and the rest are host bits
Number of host bits = 32 - 20 = 12
Number of Ip’s possible = 2^host bits - 2 = 2^12 - 2
Number of subnets possible = 2^4 = 16

5. Differentiate between NACL and Security Groups.

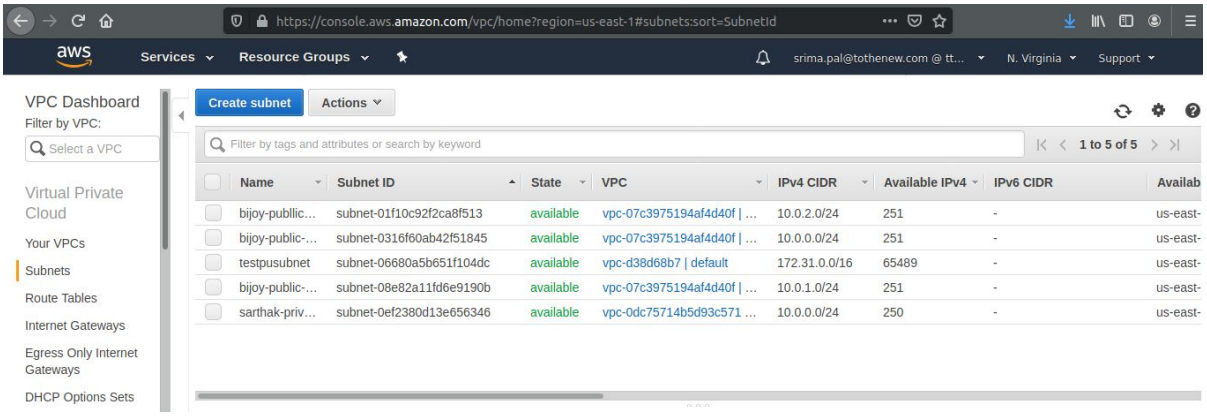
Sno	Security Groups	Network ACL's
1.	Supports Allow rules only (implicit deny) You cannot deny a certain IP address from establishing a connection.	You can both allow and deny rules.
2.	Stateful: This means any changes applied to an incoming rule will be automatically applied to the outgoing rule.	Stateless: This means any changes applied to an incoming rule will not be applied to the outgoing rule.
3.	Security groups are tied to an instance.	Network ACL are tied to the subnet.
4.	All rules in a security group are applied.	Rules are applied in their order (the rule with the lower number gets processed first)

6. Implement a 2-tier vpc with following requirements:

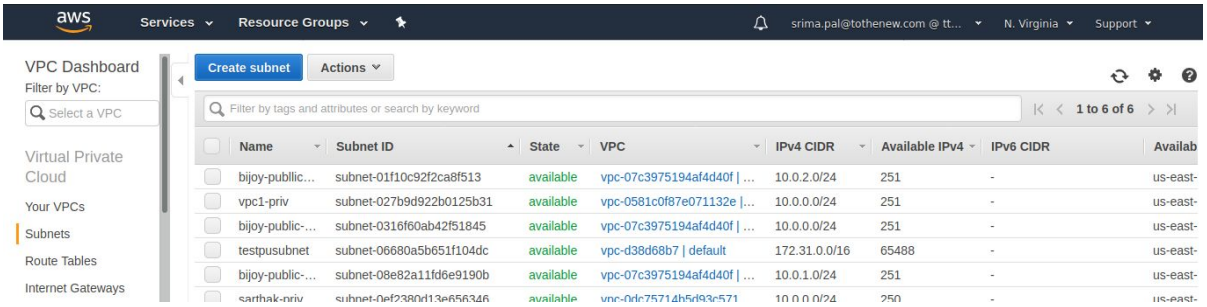
1. Create a private subnet, attach NAT, and host an application server(Tomcat)

2. Create a public subnet, and host a web server(Nginx), also proxypass to Tomcat from Nginx

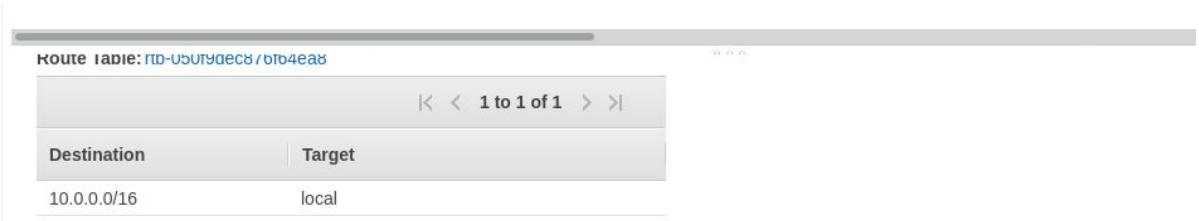
Step 1. Go to VPC, Subnets and then create subnets.



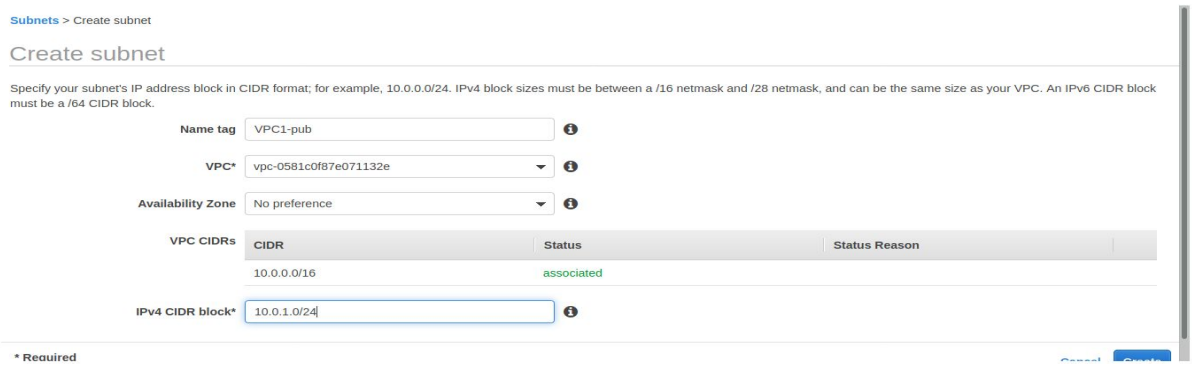
Step 2. Private Subnet Created by giving the subnet range to be 10.0.0.0/24



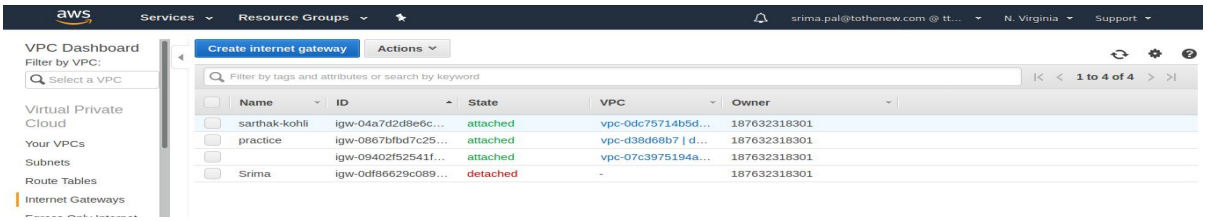
Step 3. We see that the route table(default) of the private subnet has no internet gateway attached



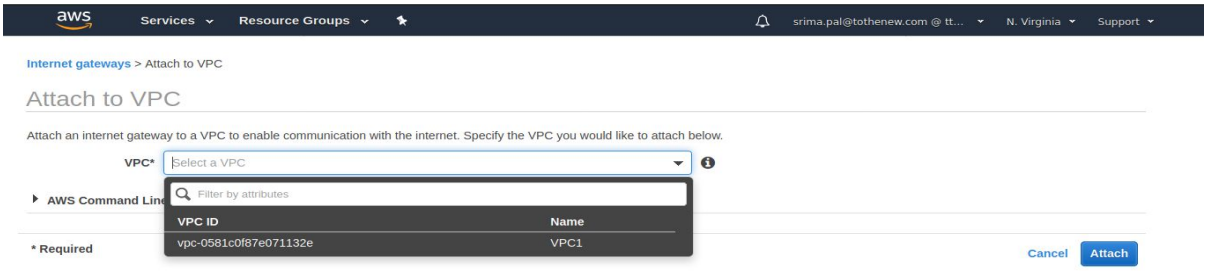
Step 4. Create a public subnet



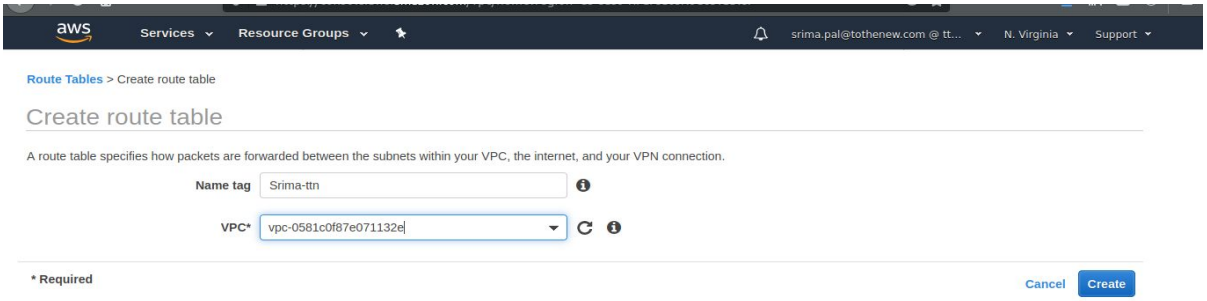
Step 5. Make an internet gateway



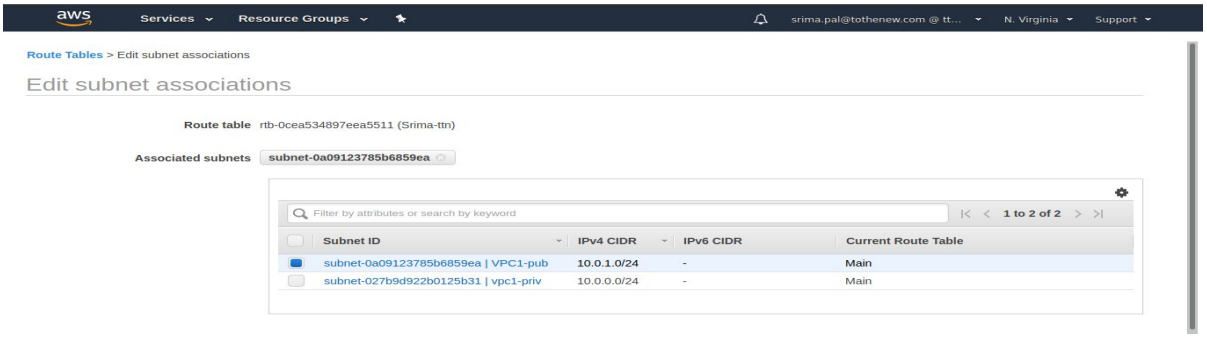
Step 6. Attach internet gateway to VPC



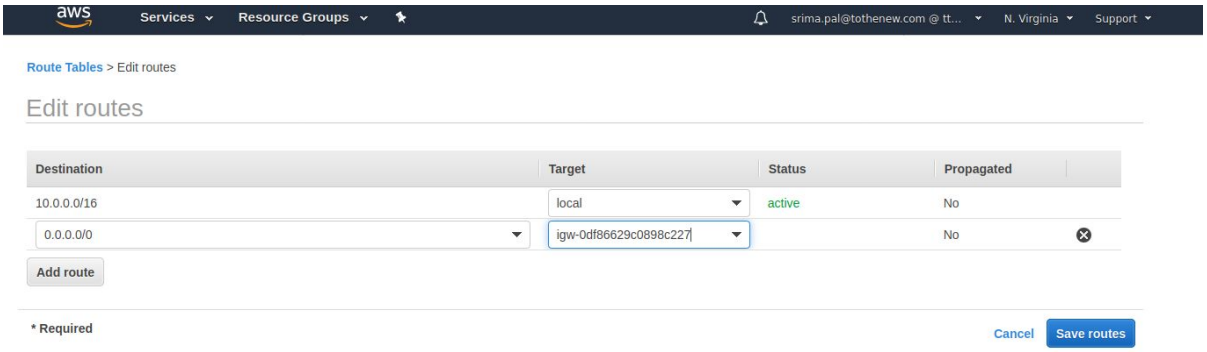
Step 7.Create a route table



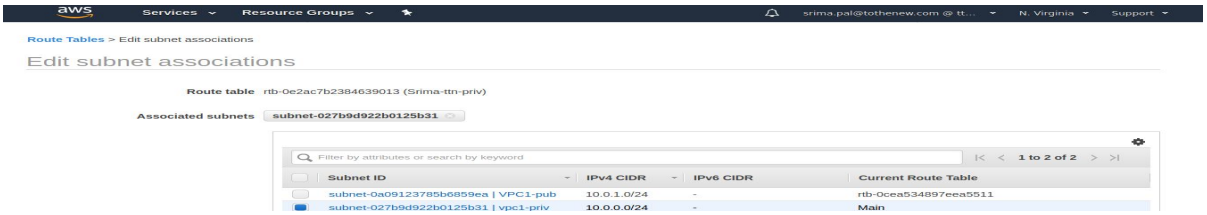
Step 8. Attach route table to the public subnet



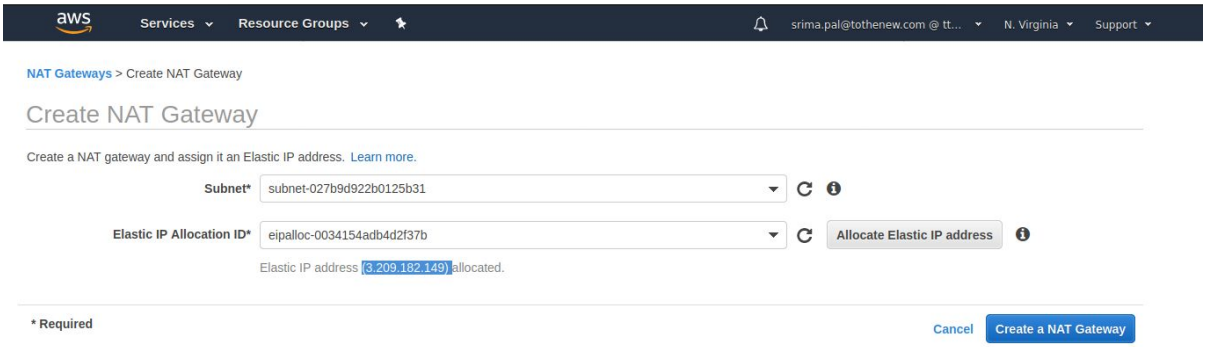
Step 7. Attach internet gateway to the route table of public subnet



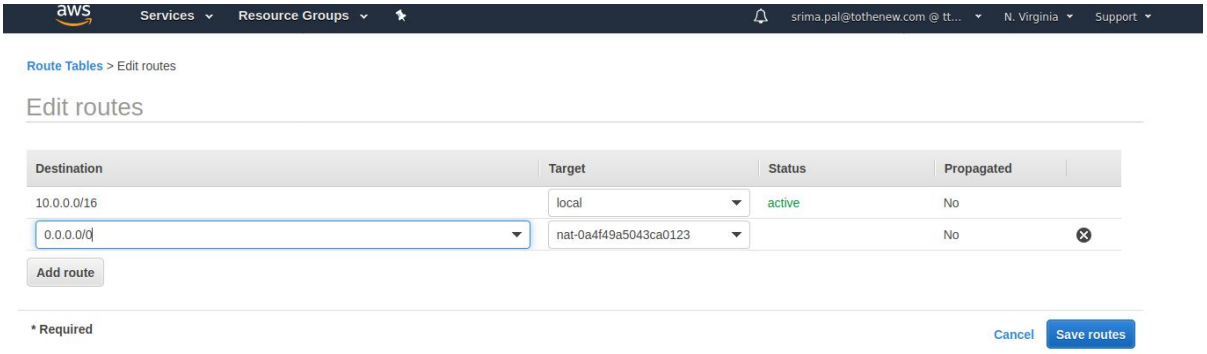
Step 8. Create another route table and attach it to the Private subnet



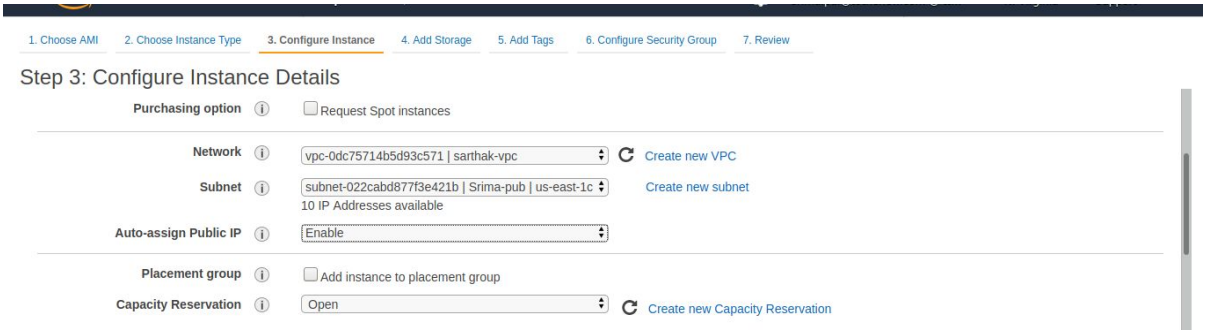
Step 9. Create a NAT Gateway

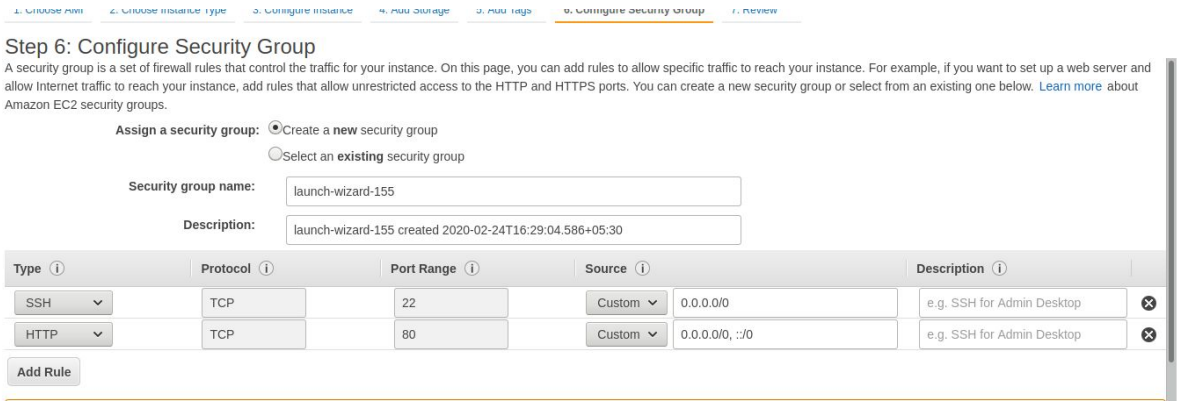


Step 10. Attach NAT Gateway to private subnet



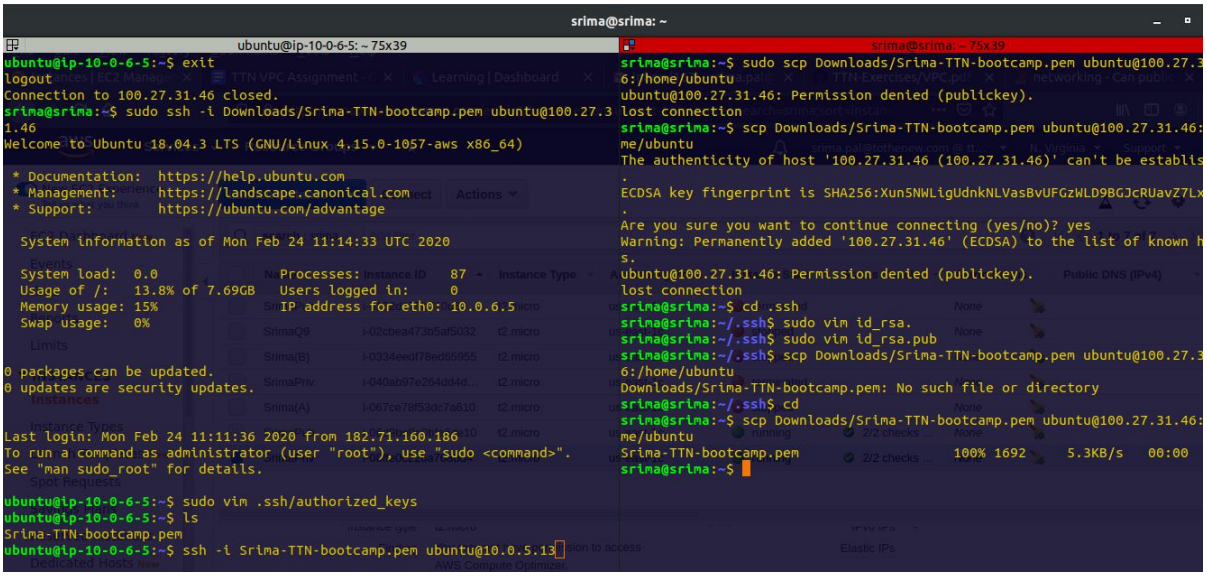
Step 11. Launch an instance in public subnet.





Step 12. Launch an instance in private subnet, disabling the auto assign ip feature and allowing custom tcp(port 8080) in security group .

Ssh into the public instance and access the private instance through the public instance



Step 13. Install tomcat on private instance

```
ubuntu@ip-10-0-5-13:~$ sudo systemctl start tomcat9
ubuntu@ip-10-0-5-13:~$ sudo systemctl status tomcat9
● tomcat9.service - Apache Tomcat 9 Web Application Server
   Loaded: loaded (/lib/systemd/system/tomcat9.service; enabled; vendor pre
   Active: active (running) since Mon 2020-02-24 11:19:53 UTC; 1min 9s ago
     Docs: https://tomcat.apache.org/tomcat-9.0-doc/index.html
 Main PID: 3835 (java)
    Tasks: 34 (limit: 1152)
   CGroup: /system.slice/tomcat9.service
           └─3835 /usr/lib/jvm/default-java/bin/java -Djava.util.logging.co

Feb 24 11:19:56 ip-10-0-5-13 tomcat9[3835]: OpenSSL successfully initialize
Feb 24 11:19:56 ip-10-0-5-13 tomcat9[3835]: Initializing ProtocolHandler ["
Feb 24 11:19:56 ip-10-0-5-13 tomcat9[3835]: Server initialization in [2,814
Feb 24 11:19:57 ip-10-0-5-13 tomcat9[3835]: Starting service [Catalina]
Feb 24 11:19:57 ip-10-0-5-13 tomcat9[3835]: Starting Servlet engine: [Apach
Feb 24 11:19:57 ip-10-0-5-13 tomcat9[3835]: Deploying web application direc
Feb 24 11:20:01 ip-10-0-5-13 tomcat9[3835]: At least one JAR was scanned fo
Feb 24 11:20:01 ip-10-0-5-13 tomcat9[3835]: Deployment of web application d
Feb 24 11:20:01 ip-10-0-5-13 tomcat9[3835]: Starting ProtocolHandler ["http
Feb 24 11:20:01 ip-10-0-5-13 tomcat9[3835]: Server startup in [4,669] milli
lines 1-19/19 (END)
```

Step 14. Install nginx on public instance

```
ubuntu@ip-10-0-6-5: ~ 75x39
ubuntu@ip-10-0-6-5:~$ sudo systemctl start nginx
ubuntu@ip-10-0-6-5:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor prese
   Active: active (running) since Mon 2020-02-24 11:22:10 UTC; 53s ago
     Docs: man:nginx(8)
 Main PID: 2767 (nginx)
    Tasks: 2 (limit: 1152)
   CGroup: /system.slice/nginx.service
           ├─2767 nginx: master process /usr/sbin/nginx -g daemon on; maste
           └─2769 nginx: worker process

Feb 24 11:22:10 ip-10-0-6-5 systemd[1]: Starting A high performance web ser
Feb 24 11:22:10 ip-10-0-6-5 systemd[1]: nginx.service: Failed to parse PID
Feb 24 11:22:10 ip-10-0-6-5 systemd[1]: Started A high performance web serv
lines 1-13/13 (END)
```

Step 15. Create file

```
ubuntu@ip-10-0-6-5: /var/www/html 75x39
<html>
        <head>
                <title>Nginx</title>
        </head>
        <body>
                <p>NGINX</p>
        </body>
</html>
```

Abc.com

```
ubuntu@ip-10-0-6-5: /etc/nginx/sites-available 75x39
server {
        listen 80;
        server_name abc.com;

        root /var/www/html;
        index index.html;
        location / {
        proxy_pass http://10.0.5.13:8080/;
        }

}
```

Step 16. Proxy Passed

```
ubuntu@ip-10-0-6-5:/etc/nginx/sites-available$ curl abc.com
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <title>Apache Tomcat</title>
</head>

<body>
<h1>It works !</h1>

<p>If you're seeing this page via a web browser, it means you've setup Tomc
at successfully. Congratulations!</p>

<p>This is the default Tomcat home page. It can be found on the local files
ystem at: <code>/var/lib/tomcat9/webapps/ROOT/index.html</code></p>

<p>Tomcat veterans might be pleased to learn that this system instance of T
omcat is installed with <code>CATALINA_HOME</code> in <code>/usr/share/tomc
at9</code> and <code>CATALINA_BASE</code> in <code>/var/lib/tomcat9</code>.
```