Lab 6: Setting up Network Address Translation (NAT) Server

East Tennessee State University

CSCI 4417/5417: Introduction to System Administration

Spring 2016

Pramod Nepal

Purpose

The focus of this lab was to learn the setup NAT Server to use as a router for a computer configured using private IP address. After the setup the computer with the private IP would be able to download packages from the internet.

Materials

- Lab Instructions
- PuTTY
- Two Ubuntu Instances

Procedure and Results

Instance and Security Setup

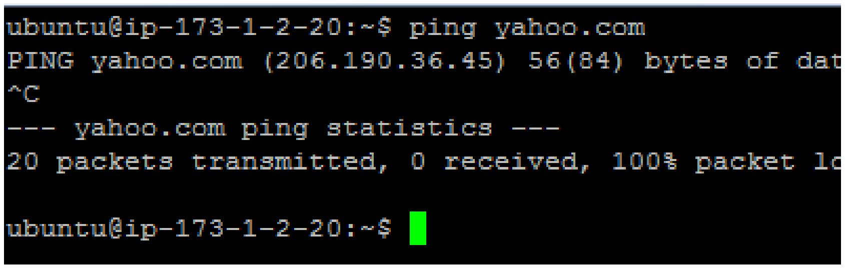
In the EC2 console as a way to protect unintended modification a new image was created using 'Image->Create Image' right-click menu option on the Ubuntu instance. After creating an image, the instance was renamed to UbuntuNAT. The instance ID of this image was copied for later use.

A new Ubuntu instance was created and named lastnameUbuntu-NAT-AMI. In the VPC dashboard the private subnet created during Test 1 was selected. 'Subnet Actions' and 'Modify Auto-Assign Public IP' were clicked to make sure 'Enable auto-assign Public IP' was unchecked. The lastnameVPC was selected after clicking on 'Route Tables'. Next, 'Subnet Associations' was clicked and edited to make sure private subnet was unselected. After this, a new route table named lastnamePrivateRT was created. During the creation existing VPC created in previous labs was selected. In the routes a new route of 0.0.0.0/0 was created and the target was selected as the instance ID of the UbuntuNAT created before. The routes showed 'Black Hole' status for instance that were not currently running. This route table was associated with the new subnet in Subnet Associations tab. In the Subnet Associations tab lastnamePrivateSubnet was checked after clicking the Edit button. To make sure that all commmunication passed through the NAT server the UbuntuNAT instance was selected in the EC2 console and under Networking option in right-click menu, 'Change Source/Dest Check' was clicked and disable button was clicked. The new Ubuntu instance was given a private IP address of 173.1.2.20 during its creation. With this IP address the Ubuntu instance would be able to communicate with other computers in the same network within a given IP range. A new Security Group was created for the NAT named lastnameNATSG. HTTP and HTTPS ports were enabled. These ports were opened for the new instance to be able to run 'apt-get update' command. The new instance was launched using the 4417key file created in earlier labs. The new security group was associated with the main VPC by clicking on the Security Groups and selecting lastnameNATSG. The Group ID was copied and then under lastnameSG, an inbound rule was added for 'All traffic' from 'Custom IP' and the 'Security Group ID' was pasted for the new security group. The NAT server instance was also started along with the new Ubuntu instance.

Configuring NAT Server

The Ubuntu NAT server was connected using PuTTY (over ssh). In case if the server printed 'Could not resolve…' message the hostname could be added to /etc/hosts following "<IP-ADDRESS> lastnameUbuntu01" format. To connect to the new Ubuntu instance the 4417key.pem file was copied following the procedures used in User Management lab to copy the Bash Shell script. A permission of 600 was set for the 4417key.pem file. This ensures that only the user is able to read+write into the file. The private instance was connected using "ssh –i '4417key.pem' ubuntu@173.1.2.20' command. Since the new instance has a private IP of 173.1.2.20 and no public IP, it was not connected to the internet. In other words a 'ping yahoo.com' would not ping Yahoo (see Fig. 1).

Figure 1: Failed to ping

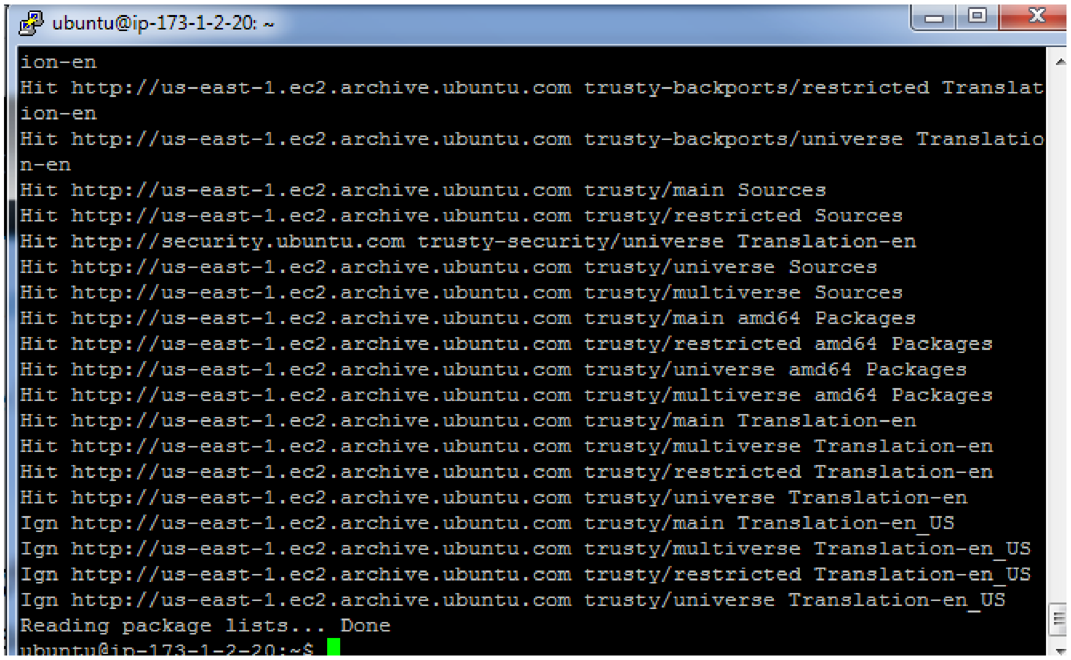


```
ubuntu@ip-173-1-2-20:~$ ping yahoo.com
PING yahoo.com (206.190.36.45) 56(84) bytes of dat
^C
--- yahoo.com ping statistics ---
20 packets transmitted, 0 received, 100% packet lo

ubuntu@ip-173-1-2-20:~$
```

Back into the NAT server IP forwarding was enabled by configuring /etc/sysctl.conf and enabling 'net.ipv4.ip_forward=1' line. This setting was enabled with 'sudo sysctl –p' command. Net IP masquerading (forwarding) was enabled by executing 'sudo iptables -t nat -A POSTROUTING -o eth0 -s 173.1.0.0/16 -j MASQUERADE' command. After this the new Ubuntu instance was able to connect to the internet through the NAT server. This was tested by login into the new instance and by pinging to 'yahoo.com'. Now the new instance was able to update the package metadata with 'sudo apt-get update' command. For routing to be persistent above IP forwarding command was added to /etc/rc.local in the NAT server and restarted. After re-login to the NAT server and ssh to new Ubuntu instance, it was able to run the previous internet facing commands like ping and apt-get (see Fig. 2).

Figure 2: Instance connected to internet



```
ion-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty-backports/restricted Translat
ion-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty-backports/universe Translatio
n-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/main Sources
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/restricted Sources
Hit http://security.ubuntu.com trusty-security/universe Translation-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/universe Sources
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/multiverse Sources
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/main amd64 Packages
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/restricted amd64 Packages
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/universe amd64 Packages
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/multiverse amd64 Packages
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/main Translation-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/multiverse Translation-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/restricted Translation-en
Hit http://us-east-1.ec2.archive.ubuntu.com trusty/universe Translation-en
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/main Translation-en_US
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/multiverse Translation-en_US
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/restricted Translation-en_US
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/universe Translation-en_US
Reading package lists... Done
ubuntu@ip-173-1-2-20:~$
```

Observations

In this lab the way to create an image was practiced. It was done to make sure the instance could be restored if something went wrong while using it. To disable EC2 auto-assign public IP to the created new instance 'Enable auto-assign Public IP' was disabled. Thus this instance was made completely private and the way it could connect to the internet was only by connecting to the NAT server. The 'Change Source/Dest Check' was disabled so that the request response from and to the internet would flow to the private instance without any checks. In the Security Group HTTP and HTTPS ports were enabled, so that the Ubuntu computer under NAT could run apt-get command. The key file was copied to the NAT server so that one could ssh into the private server to see if they could connect to internet. To create a NAT server two commands needed to be run. These commands enabled port forwarding. Once the masquerading command worked, it was made persistent by adding it to /etc/rc.local file. Commands on this file get executed at each boot. As the instance connected to the NAT server has private IP address an outside attacker would not be able to connect to it. This in a way protects the instance from direct attack and the organizations information is safe to a certain extent. Even the instances under private subnet are not isolated from other Internet attacks. Since the instances can receive and send packets they are vulnerable for attacks done by sending malicious packets. To make the private instance more secure different solutions like Intrusion Detection System, Antivirus software can be used. In a traditional data center asymmetric data flow needs to be monitored. As the data center evolves the security solutions also needs to be adaptive. In a cloud-based solution security rules are provided by the provider, which can be a wrapper for securing different resources. As the data center evolves new algorithms could be written to adapt to changes. In a production environment to enhance security AWS Security Groups needs to be more restrictive. It should not have any unnecessary IP or port open other than required. Instances needs to have private IP unless they need access from outside. The public/private instances that were created on this lab can also be configured while setting up a new VPC instance.