CSE 4512 [Computer Networks Lab]

Lab # 11

1. Objectives:

- Describe the concept of Network Address Translation (NAT)
- Explain different types of NAT configuration
- Implement NAT in a given topology

2. Theory:

[For the commands, the **`bold`** ones are *constant keywords* which should be exactly same. `Non-bold` ones are the configurable options.]

Network Address Translation (NAT):

You already know from your theory lectures that IPv6 was born partly due to the address space exhaustion of IPv4. One great technique that was the key for survival of IPv4 is this **NAT**. If not for NAT, IPv4 would be long gone by now. And that gave the world some time to adopt IPv6 in mass scale. In this lab, you'll learn about this special technique called NAT.

Basically, the idea of NAT is there'll be a set of IP addresses for the hosts in the internal network and to the outside world those internal hosts will be exposed using a different set of IP addresses. You know that each host is recognized through its IP address on the internet. To conform with this, each host connected to the internet would have to have a unique IP which readily becomes close to impossible considering there are billions of connected devices.
NAT allows you to assign arbitrary IP addresses to your internal hosts where these addresses are only locally significant i.e., these are locally unique. Then in the edge or gateway router of the network you'll have one or a set of IP addresses which are globally unique. That edge or gateway router will do the conversion/translation from the globally unique address to the locally unique ones or vice-versa. Outside world won’t know the actual IP addresses of internal hosts which provides an extra layer of security. Moreover, your organization can buy only a handful of global IP addresses from the ISP but can use

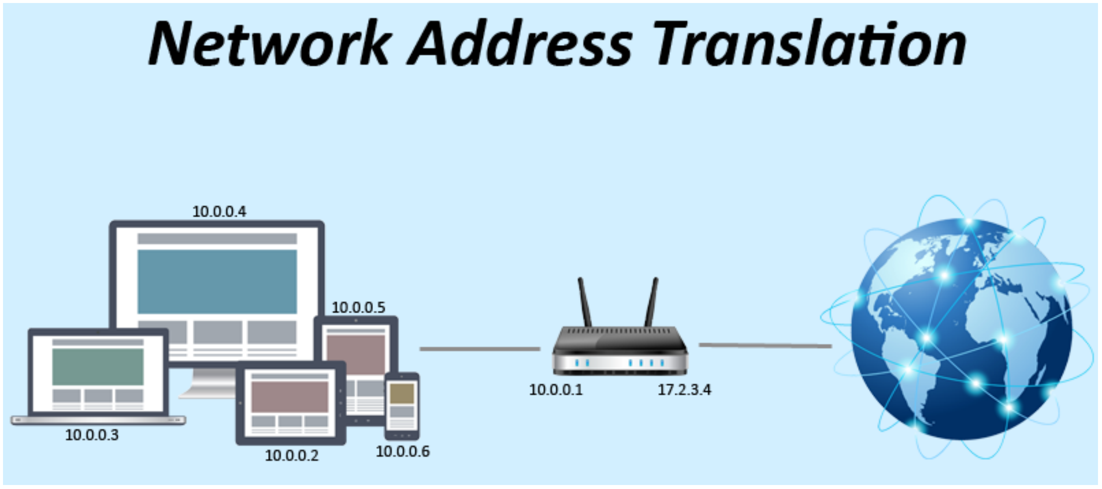


Figure 1: NAT

those with much greater number of hosts through NAT. The following figure summarizes what we just talked about.

Now that we know the basics of NAT, lets get technical. There are three types of NAT that you can define in Cisco devices: Static, Dynamic and Overloaded or Port Address Translation (PAT).

Static NAT

It allows one-to-one mapping between local and global addresses. You will have to configure one global IP address for each of internal hosts that you want NAT to translate. The command to enable the static translation is as follows:

```
Router(config)# ip nat inside source static local_ip global_ip
```

After you've specified the translation, you'll need to do two things. First, you need to specify the *inside* interface. The inside interface denotes that the hosts connected to this interface will have their IPs translated to the global one. Second, you need to specify the *outside* interface. Outside interface denotes that through this interface the translated packets will go out to the world. There can be more than one inside or outside interfaces. After you specify these interfaces, NAT will start the specified translations. You need to specify these *inside* and *outside* interfaces for the other two NAT types also. Following are the commands to specify the inside and outside interfaces:

```
Router(config-if)# ip nat inside
Router(config-if)# ip nat outside
```

Dynamic NAT

This type of NAT establishes a mapping between a local address and a pool of global addresses. For a single local address, a global IP address will be selected from the pool dynamically. When not in use, the assigned global IP will be released after a certain time-out period so that other hosts can re-use it. This is more convenient than the static one as you don't need to manually configure every mapping. For configuring dynamic NAT, first you need to *create an access list permitting the **local addresses*** to get translated. Then you have to specify the pool of global IP addresses from where the IPs will be allocated. The pool is a range of IP addresses in a given network where the subnet mask will specify the corresponding network portion. The command to specify the pool is as follows:

```
Router(config)# ip nat pool POOL_NAME start_ip end_ip netmask
subnet_mask
```

Then you need to establish the relation between the earlier defined *access list* and *nat pool* through the following command:

```
Router(config)# ip nat inside source list access_list_number pool
POOL_NAME
```

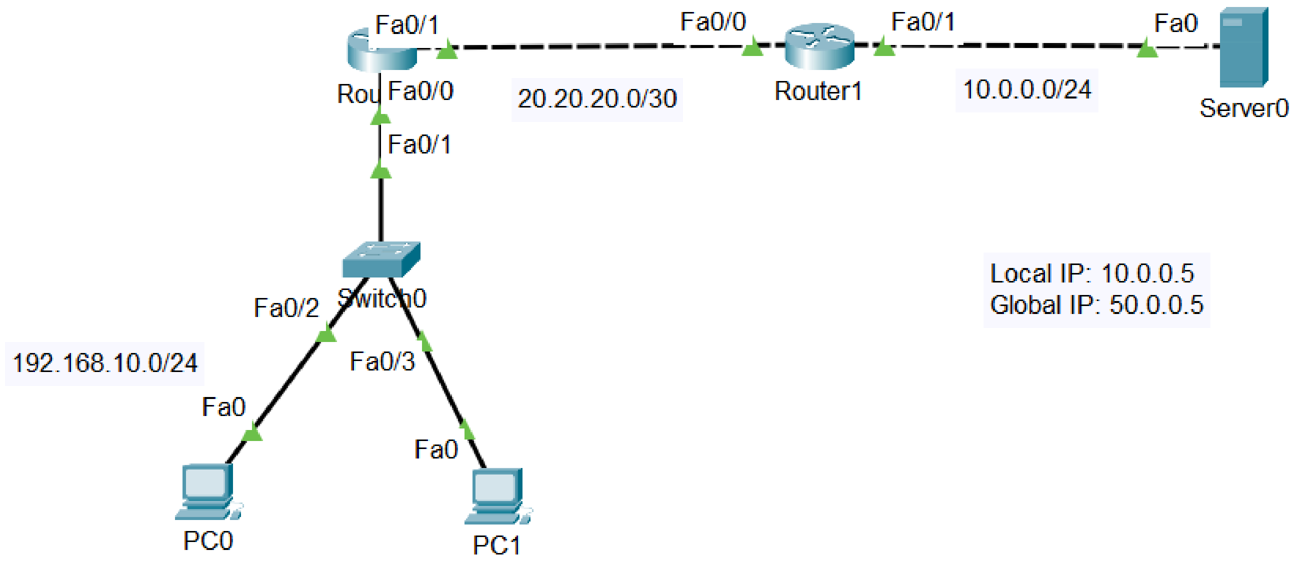
After that, like the static NAT, you have to specify the *inside* and *outside* interfaces.

Port Address Translation (PAT)

For dynamic NAT, in the worst case, you'd need as many global IP addresses as the internal hosts. This is not plausible in most circumstances where you've limited global IP and hundreds of local hosts. Its where PAT comes in. PAT establishes many-to-one mapping between multiple local hosts and a single global IP address. It uses the Port (TCP/UDP port) information to distinguish between different internal hosts and assign a single global IP to all those addresses thus greatly conserving the global address pool. The configuration of PAT is almost similar to dynamic NAT except you just have to add the **overload** keyword at the very end while specifying the relation between access list and nat pool. The command format is below:

```
3. Router(config)# ip nat inside source list access_list_number pool
                          POOL_NAME overload
```

4. Configure NAT:



I. Configure Router0 Interfaces

```
Router(config)# int fa0/0
Router(config-if)# ip address 192.168.10.1 255.255.255.0
Router(config-if)# no shutdown
Router(config)# int fa0/1
Router(config-if)# ip address 20.20.20.1 255.255.255.252
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# ip route 50.0.0.0 255.255.255.0 20.20.20.2
Router# copy running-config startup-config
```

II. Configure Router1 Interfaces

```
Router(config)# int fa0/1
Router(config-if)# ip address 10.0.0.1 255.255.255.0
Router(config-if)# no shutdown
Router(config)# int fa0/0
Router(config-if)# ip address 20.20.20.2 255.255.255.252
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# ip route 192.168.10.0 255.255.255.0 20.20.20.1
Router# copy running-config startup-config
```

III. **Configure PC0**

```
IP: 192.168.10.5
Mask: 255.255.255.0
Gateway: 192.168.10.1
```

IV. **Configure PC1**

```
IP: 192.168.10.10
Mask: 255.255.255.0
Gateway: 192.168.10.1
```

V. **Configure Server0**

```
IP: 10.0.0.05
Mask: 255.255.255.0
Gateway: 10.0.0.1
```

VI. **Enable static NAT insider Router1**

```
Router(config)# ip nat inside source static 10.0.0.5 50.0.0.5
Router(config)# int fa0/1
Router(config-if)# ip nat inside
Router(config)# int fa0/0
Router(config-if)# ip nat outside
Router# copy running-config startup-config
```

VII. **Verify NAT**

```
Router# show ip nat translations
Router# show ip nat statistics
```

5. Tasks:

I. You will configure *dynamic NAT and PAT* following the instructions given in the task. The task description for this task is provided in the pdf ***Task-1_configure-NAT***. You're provided a .pka file for this task.

Packet Tracer - Configure PAT

Objectives

Part 1: Configure Dynamic NAT with Overload

Part 2: Verify Dynamic NAT with Overload Implementation

Part 3: Configure PAT using an Interface

Part 4: Verify PAT Interface Implementation

Part 1: Configure Dynamic NAT with Overload

Step 1: Configure traffic that will be permitted.

On **R1**, configure one statement for ACL 1 to permit any address belonging to 172.16.0.0/16.

```
R1(config)# access-list 1 permit 172.16.0.0 0.0.255.255
```

Step 2: Configure a pool of address for NAT.

Configure **R1** with a NAT pool that uses the two useable addresses in the 209.165.200.232/30 address space.

```
R1(config)# ip nat pool ANY_POOL_NAME 209.165.200.233 209.165.200.234 netmask
255.255.255.252
```

Step 3: Associate ACL 1 with the NAT pool and allow addresses to be reused.

```
R1(config)# ip nat inside source list 1 pool ANY_POOL_NAME overload
```

Step 4: Configure the NAT interfaces.

Configure **R1** interfaces with the appropriate inside and outside NAT commands.

```
R1(config)# interface s0/1/0
R1(config-if)# ip nat outside
R1(config-if)# interface g0/0/0
R1(config-if)# ip nat inside
R1(config-if)# interface g0/0/1
R1(config-if)# ip nat inside
```

Part 2: Verify Dynamic NAT with Overload Implementation

Step 1: Access services across the internet.

From the web browser of each of the PCs that use **R1** as their gateway (**PC1**, **L1**, **PC2**, and **L2**), access the web page for **Server1**.

Were all connections successful?

Step 2: View NAT translations.

View the NAT translations on **R1**.

```
R1# show ip nat translations
```

Notice that all four devices were able to communicate, and they are using just one address out of the pool. PAT will continue to use the same address until it runs out of port numbers to associate with the translation. Once that occurs, the next address in the pool will be used. While the theoretical limit would be 65,536 since the port number field is a 16 bit number, the device would likely run out of memory before that limit would be reached.

Part 3: Configure PAT using an Interface

Step 1: Configure traffic that will be permitted.

On **R2**, configure one statement for ACL 2 to permit any address belonging to 172.17.0.0/16.

Step 2: Associate ACL 2 with the NAT interface and allow addresses to be reused.

Enter the **R2** NAT statement to use the interface connected to the internet and provide translations for all internal devices.

```
R2(config)# ip nat inside source list 2 interface s0/1/1 overload
```

Step 3: Configure the NAT interfaces.

Configure **R2** interfaces with the appropriate inside and outside NAT commands.

Part 4: Verify PAT Interface Implementation

Step 1: Access services across the internet.

From the web browser of each of the PCs that use **R2** as their gateway (**PC3**, **L3**, **PC4**, and **L4**), access the web page for **Server1**.

Were all connections successful?

Step 2: View NAT translations.

View the NAT translations on **R2**.

Step 3: Compare NAT statistics on R1 and R2.

Compare the NAT statistics on the two devices.

Why doesn't **R2** list any dynamic mappings?