RCOMP - Redes de Computadores (Computer Networks)

2024/2025

Theoretical-practical lesson 07

- Network Address Translation (NAT).
- Static and dynamic NAT.
- Network Address Port Translation (NAPT).

Network Address Translation (NAT)

NAT refers to a technique under which packets' source and destination addresses are automatically changed in intermediate nodes, usually routers. This is done with specific purposes as we will see ahead.

It's somewhat peculiar we are now talking about manipulating packets' source and destination addresses while in the previous lesson we were so keen on blocking IP spoofing.

Nevertheless, although we will be changing addresses, unlike a hacker using IP spoofing, we will not be using forged addresses.

In a networking infrastructure, each node must have a unique address and the internet is no exception. If you have an internet connection at home, perhaps you have already wondered if each connected device at your home has a unique address on the internet. The answer is **no**.

There wouldn't be enough addresses in the IPv4 32-bits address space. IPv6 could solve this, but for now, is only partially deployed over the internet.

Some of the most significant applications of NAT can attenuate this IPv4 address space exhaustion issue.

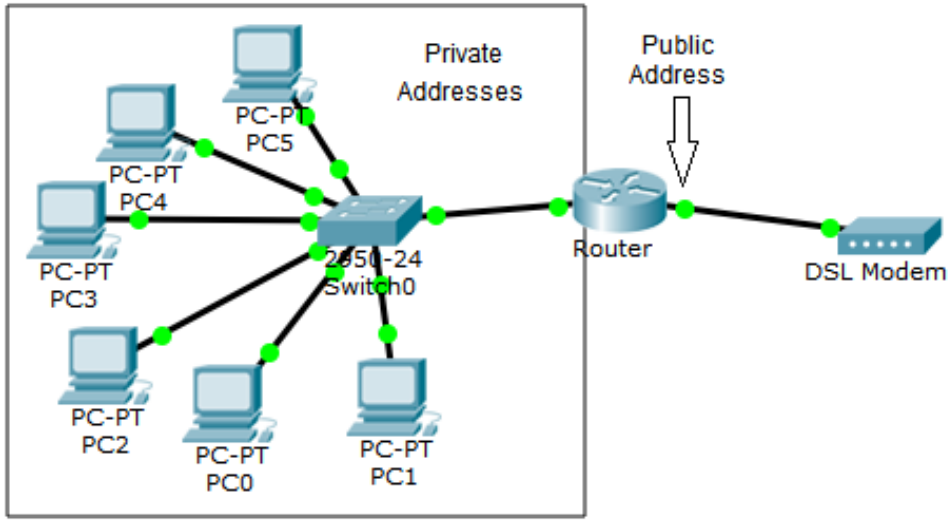
Private IPv4 addresses

We already know some blocks of addresses from the IPv4 address space are reserved for private use, as you may recall, they are **10.0.0.0/8**, **172.16.0.0/12**, and **192.168.0.0/16**.

Unlike public addresses, private addresses are not recognised over the internet (these addresses do not exist on the internet). They can't be used by clients as source addresses, because even if requests reach servers, the server reply will never reach back the client. Apparently private addresses can only be used for traffic within local networks with no connection to the internet.

Here is where NAT can play a role, ultimately NAT will allow a big number of nodes using private addresses to be hidden behind a single public address shared by all those private nodes.

For now, we'll start with some other NAT applications.



SNAT, DNAT and the NAT table

Address changing can be enforced on a packet's source address, this is called source NAT (SNAT), or on the packet's destination address, this is called destination NAT (DNAT).

NAT must operate transparently for end nodes, namely the packet's source node and the destination node can´t become aware NAT is being used. This is guaranteed by applying one translation in one direction and the contrary translation on the opposite direction.

A NAT table contains addresses equivalencies between two sides, say side A and side B:

The NAT table alone is insufficient for NAT to operate, you must also declare in which direction is SNAT applied (implicitly DNAT is applied in the opposite direction).

Alternatively, you can declare in which direction is DNAT applied (implicitly SNAT is applied in the opposite direction).

NAT Table	
Side A	**Side B**
X1.X1.X1.X1	X2.X2.X2.X2
X3.X3.X3.X3	X4.X4.X4.X4

NAT operation

Assuming we have this NAT table and we set SNAT to be applied when a packets travel from side A to side B (implicitly DNAT is applied when a packet travels from side B to side A).

NAT Table SNAT from side A to side B	
Side A	**Side B**
X1.X1.X1.X1	X2.X2.X2.X2
X3.X3.X3.X3	X4.X4.X4.X4

What this configuration means is:

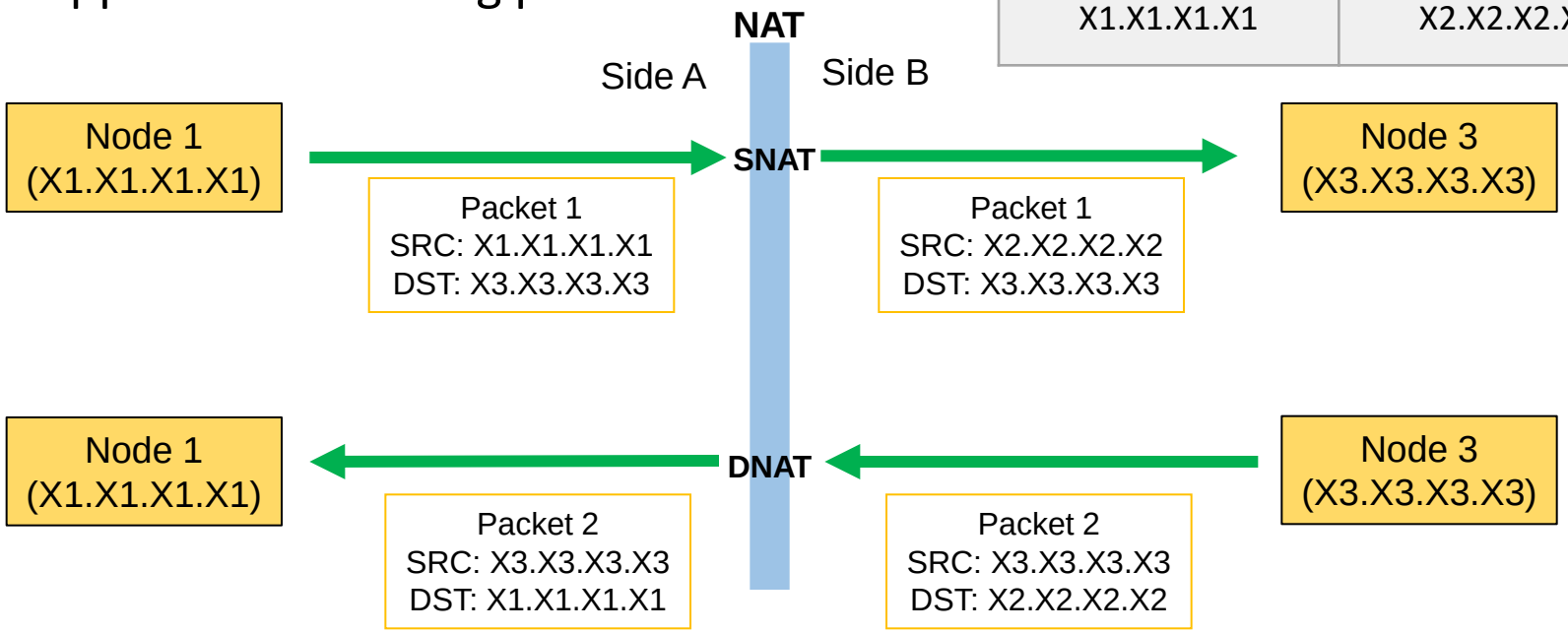
- When a packet travels from side A to side B, if the packet's **source address** matches an entry on the table's A side, it will be translated do the corresponding address on the table's B side (SNAT).
- When a packet travels from side B to side A, if the packet's **destination address** matches an entry on the table's B side, it will be translated do the corresponding address on the table's A side (DNAT).

This is the general principle of NAT operation, and it can be used for several purposes.

NAT operation

Using this NAT table, and SNAT being applied from side A to side B. Let´s see what happens for matching packets:

NAT Table SNAT from side A to side B	
Side A	**Side B**
X1.X1.X1.X1	X2.X2.X2.X2



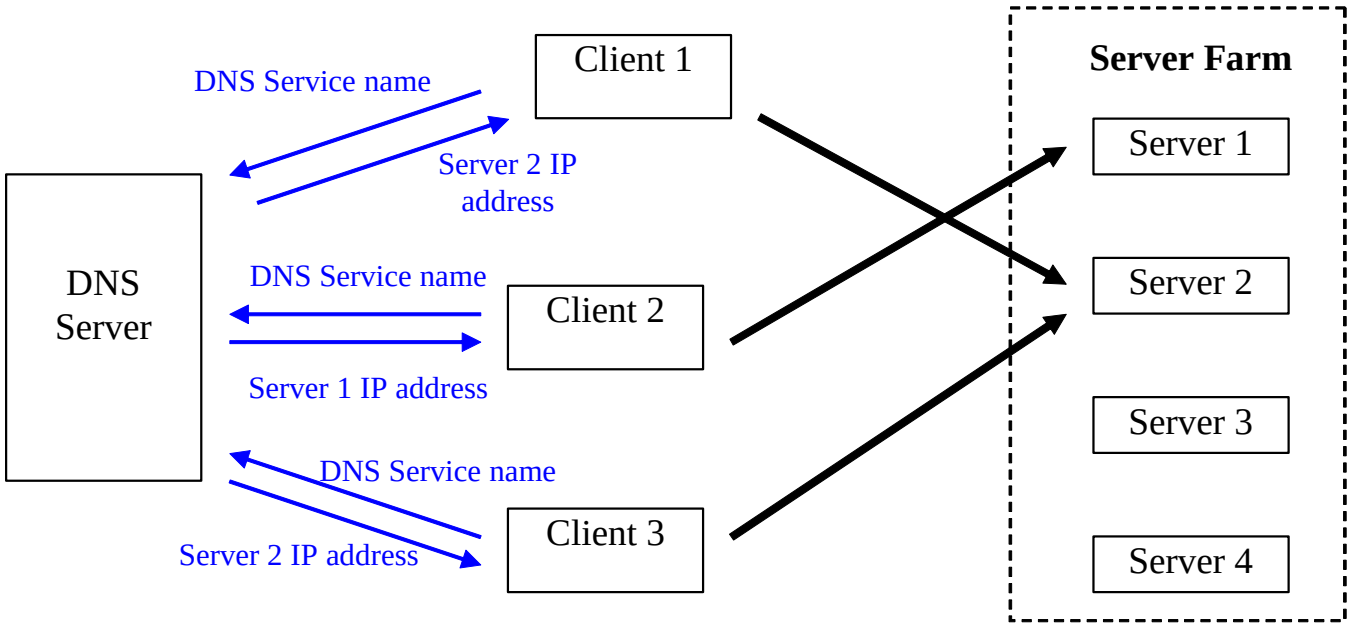
Under the point of view of Node 1 and Node 3 everything looks pretty normal, and that's key. Also:

- Side A source addresses are hidden and not seen by side B.
- Side A nodes are reachable by side B only if side B NAT table addresses are used as destination addresses.

Servers Load Balancing with DNS

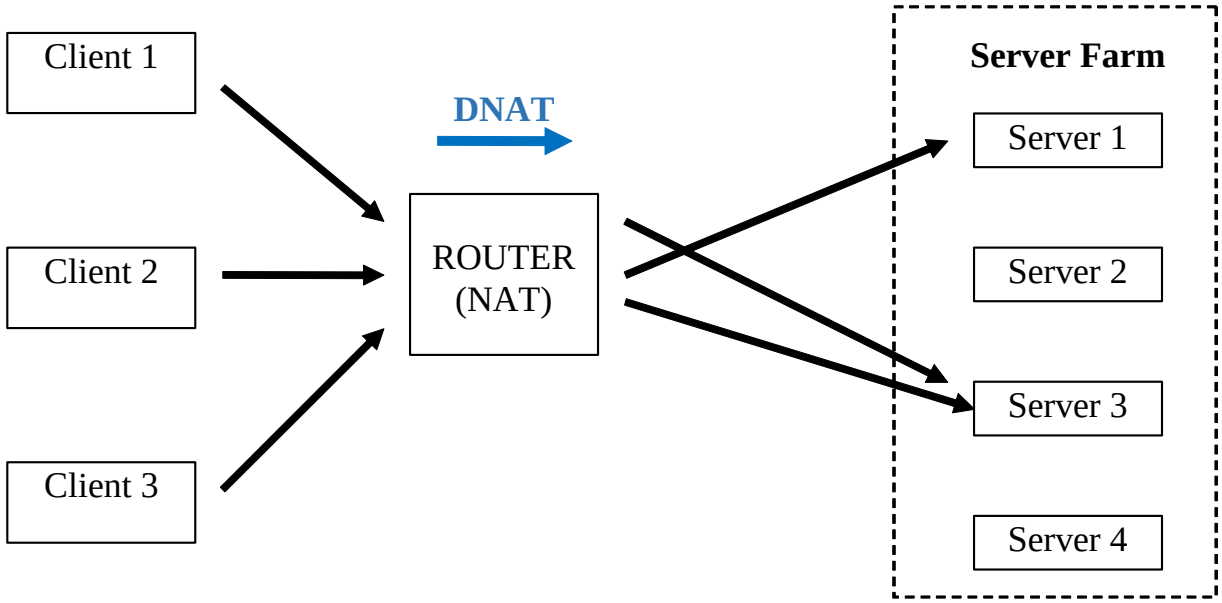
When a single server is insufficient to handle all clients, we must build a servers' cluster (also known as servers farm) and distribute the clients among the available servers.

There are several approaches for distributing clients by the available servers. One is using DNS: the service has a single DNS name, however, when clients request the name resolution to attain the corresponding IP address, they don't get all the same IP address for the service.



Servers Load Balancing with NAT

Another alternative for clients' distribution between available servers is DNAT, now we have a single IP address for the service, however, the service IP address doesn't belong to any real server, it belongs to the NAT router. The NAT router will change the clients' requests destination addresses to match real servers.



The NAT device (router) may use several criteria for clients' distribution:

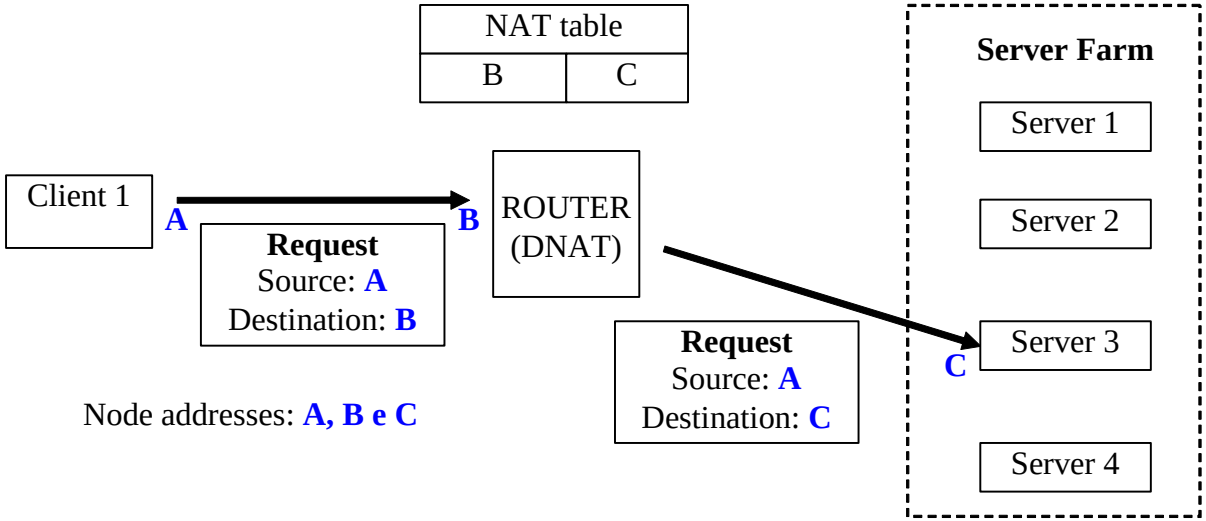
- Simple round-robin.
- Maintain the same number of clients per server.
- Maintain the same load per server.

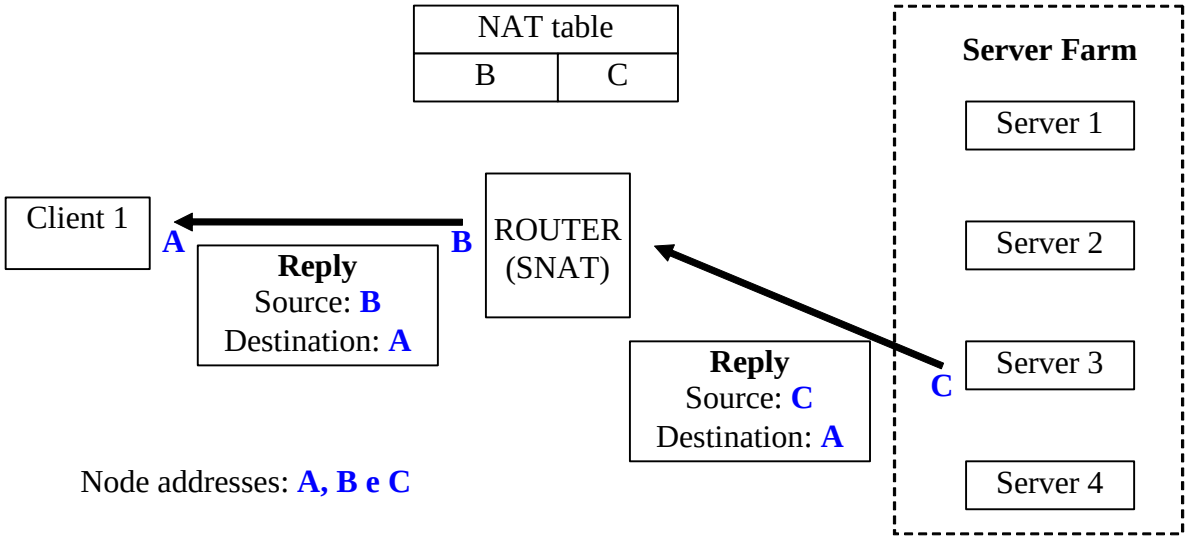
And of course, do the above while excluding offline servers.

Servers Load Balancing - NAT

Analysing a single request and corresponding reply:

Because real servers' IP addresses are hidden from clients, the servers cluster may even be on a private network.





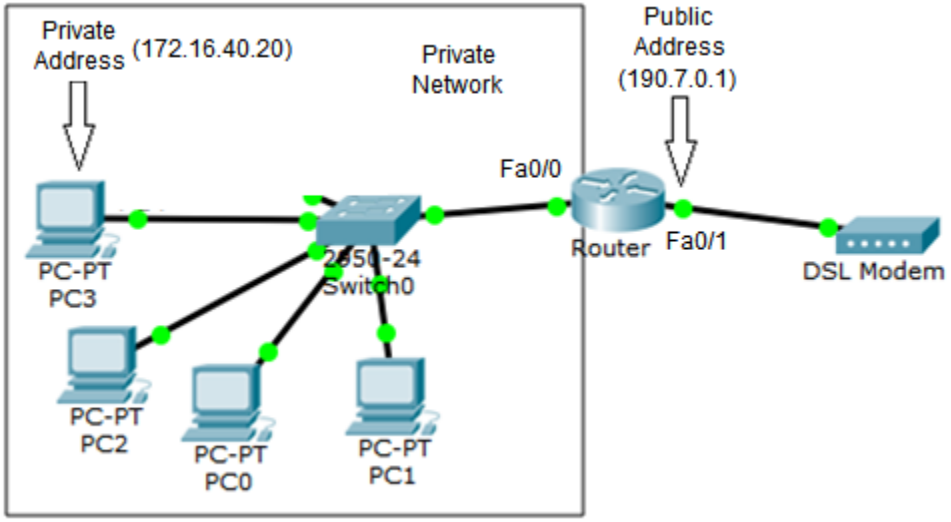
From the point of view of Client 1, the request was sent to address B and the reply is received from address B, so, everything is normal.

Private networks – static NAT

We have already seen NAT has the effect of hiding source addresses of nodes in the side from which incoming traffic has SNAT applied. This may be used to provide internet access for private networks.

If we want a private node to be able to access the internet, we can configure NAT to translate its source address (SNAT) to a public address. Let's take the example on the next image with a Cisco router, imagine we want **PC3** to be able to access the internet.

In Cisco routers sides A and B are called **inside** and **outside**. One thing we must always do, is declare to which side each router's interface belongs to. We can, for instance, use **inside** for the private network.




```
interface Fa0/0
ip nat inside

interface Fa0/1
ip nat outside
```

Now we can add a static NAT translation to the NAT table:

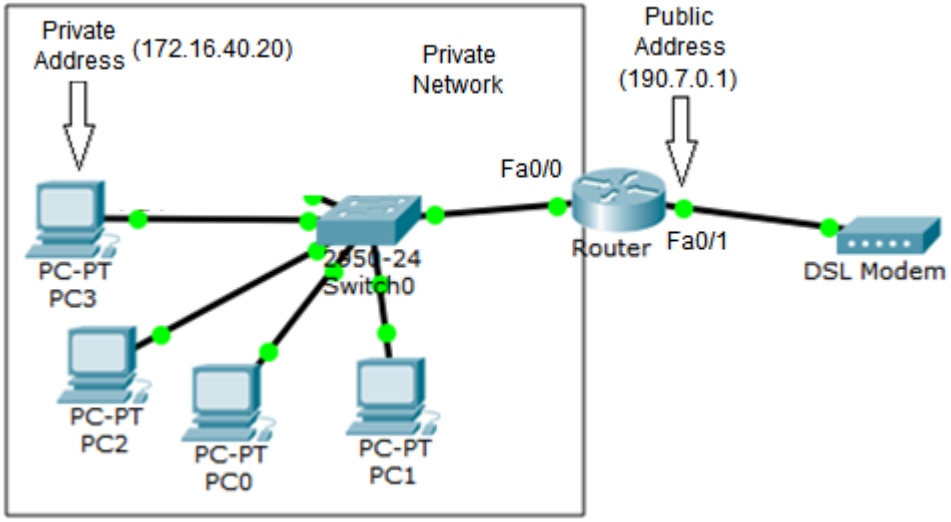
```
(config)#ip nat inside source static 172.16.40.20 190.7.0.1
```

Private networks – static NAT

```
(config)#ip nat inside source static 172.16.40.20 190.7.0.1
```

This command adds a static (permanent) line to the NAT table:

NAT Table SNAT from inside to outside (DNAT from outside to inside)	
Inside	**Outside**
172.16.40.20	190.7.0.1



Static NAT is most useful for providing access from the internet to servers connected to a private network. In the above configuration, we can see that any access from the internet to address 190.7.0.1 is going to be redirected (DNAT) to the private node 172.16.40.20.

For private clients' access to the internet, static NAT has some major drawbacks. First, we need one public address permanently allocated for each private address. There's also a security issue, the client node will be exposed to all sorts of attacks from the internet directed to the public address.

Private networks – dynamic NAT

With dynamic NAT, NAT table entries are not permanent, they are created by the router on demand. Basically, we let the router manage a set of public IP addresses (addresses pool) and assigned them to clients as needed.

The number of addresses in the pool, limits the maximum number of private clients being able to communicate with the internet **at the same time**.

Once the pool is exhausted no further clients will be able to communicate, however, in dynamic NAT, entries in the table are temporary, if there's no traffic from a client during some amount of time, the entry is removed, and the assigned public IP address is free to be assigned to another client. In a Cisco router we can define an addresses pool for later use by NAT with the command:

```
ip nat pool POOL-NAME FIRST-IP LAST-IP netmask NETWORK-MASK
```

Unlike the static NAT configuration we have already seen, dynamic NAT uses more information about transactions, each NAT table entry stores information about the remote node being accessed and upper layer protocols. Packets will match a table entry only if they match all characteristics recorded in that table entry.

Private networks – dynamic NAT

NAT table (dynamic entries)		
Inside local	Outside local	Outside remote
192.168.10.5 – ICMP:2234	190.10.7.8 – ICMP:10226	120.1.72.235 – ICMP:10226
192.168.10.200 – UDP:2288	190.10.7.9 – UDP:45611	182.67.20.1 – UDP:34
172.17.0.235 – TCP:45002	190.10.7.6 – TCP:25516	120.50.7.3 – TCP:80

The image above show dynamic NAT entries in a NAT table, they are created by the router as requests from inside clients travel to the internet. The point is, each entry will only match traffic with those precise characteristics.

Take for instance the last line. This line exists because the internal node 172.17.0.235 has opened a TCP connection using local port number 45002 and connected to outside node 120.50.7.3 port number 80.

This exposes almost nothing the inside source node to outside attacks. It will only match with TCP segments coming from port number 80 of node 120.50.7.3 that are being sent to port number 25516 of address 190.10.7.9. No other traffic is able to reach the private node 172.17.0.235.

Private networks – dynamic NAT

In a Cisco router we can enable dynamic NAT by using the command:

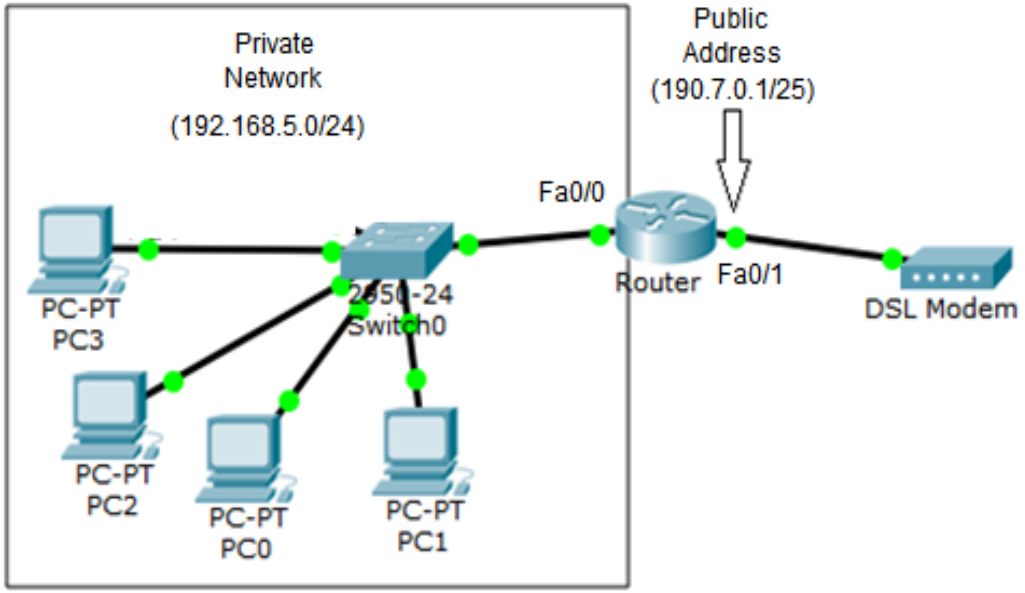
```
ip nat inside source list ACL-ID pool POOL-NAME
```

This will enable dynamic NAT by applying SNAT to traffic traveling from inside interfaces to outside interfaces, as far as that traffic matches (is permitted by) the ACL-ID access list, outside addresses will be allocated from the POOL-NAME addresses pool. ACL-ID can be either a number or a name, usually a standard ACL is appropriate.

Let's take an example on Cisco IOS:

We want to configure dynamic NAT to provide internet access for the private network.

Public addresses available for the router are from 190.7.0.10 up to 190.7.0.20.

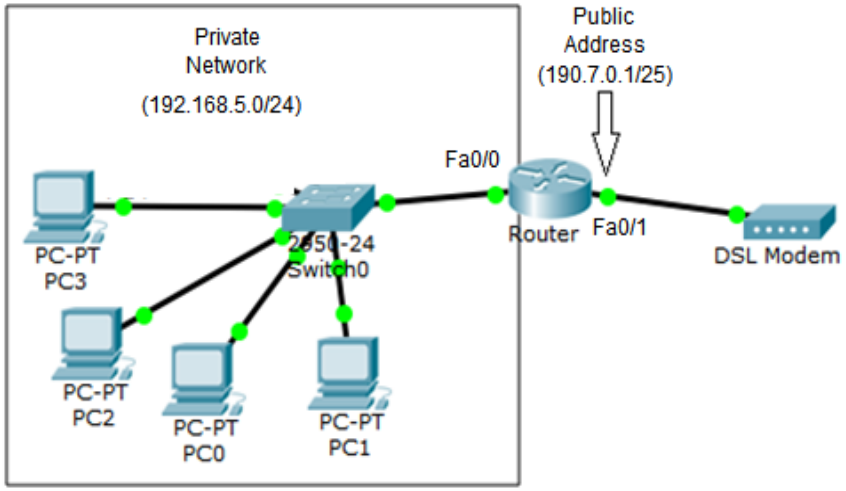


Private networks – dynamic NAT

```
interface Fa0/0
ip nat inside

interface Fa0/1
ip nat outside

no access-list 10
access-list 10 permit 192.168.5.0 0.0.0.255
```




```
ip nat pool PUB 190.7.0.10 190.7.0.20 netmask 255.255.255.128

ip nat inside source list 10 pool PUB
```

A maximum of 11 private network nodes will be able to communicate at the same time, but this could be dramatically changed just by adding **overload** to the NAT command:

```
ip nat inside source list 10 pool PUB overload
```

Overload argument says the router is allowed to assign the same outside IP address to several inside clients at the same time.

This is called NAPT (Network Address and Port Translation).

NAPT (Network Address and Port Translation)

Overload is possible in dynamic NAT because we are storing much more information in the NAT table than with static NAT. Now, let's check the worst-case scenario in the following table: several private clients accessing the same internet service:

NAT table (dynamic entries with overload)		
Inside local	Outside local	Outside remote
192.168.10.5 – TCP:42234	190.10.7.8 – TCP:42234	120.50.7.3 – TCP:80
192.168.10.6 – TCP:45002	190.10.7.8 – TCP:45002	120.50.7.3 – TCP:80
192.168.10.7 – TCP:45002	190.10.7.8 – TCP:45003	120.50.7.3 – TCP:80

All these three inside clients are accessing the same service, yet only one outside address is used (overload). Nevertheless, NAT can still identify to which inside client traffic is directed, because the outside data recorded for each table entry is always different. The router must ensure this.

We can see that by default, this router, is using the same outside source port the client is using, however, on the last line, it was forced to change that because it would lead to a repeated line on the outside data.

This is how NAPT operates. And what about packets without port numbers like ICMP? For ICMP a unique message identifier is used, some protocols are difficult to use with NAPT, for instance GRE (Generic Routing Encapsulation).

Dynamic NAPT configuration

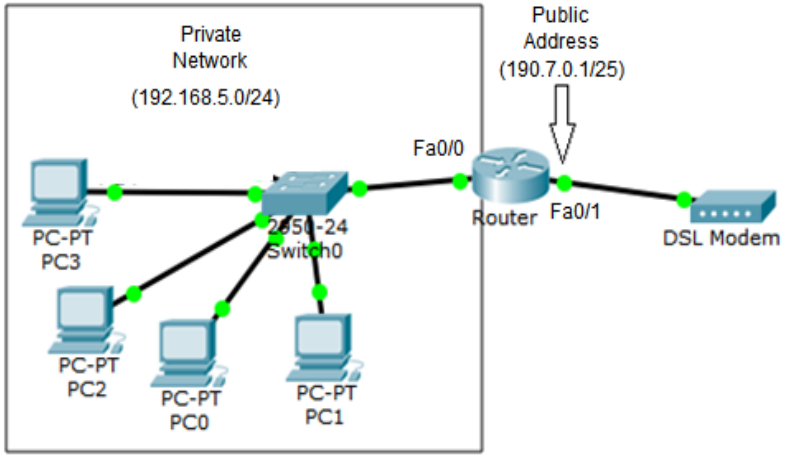
Configuring dynamic NAPT in a Cisco router is just a matter of adding the overload argument to the **ip nat inside** command, when overload is used the provided address pool may be as short as a single address.

If only one outside address is to be used, it will usually be the router's own address. In this case we can use a configuration syntax where there's no need to declare the addresses pool.

```
interface Fa0/0
ip nat inside

interface Fa0/1
ip nat outside

no access-list 10
access-list 10 permit 192.168.5.0 0.0.0.255
ip nat inside source list 10 interface Fa0/1 overload
```



Instead of an address pool, we provide the name of the outside interface, that interface's address will be used as the single outside address for all communications.

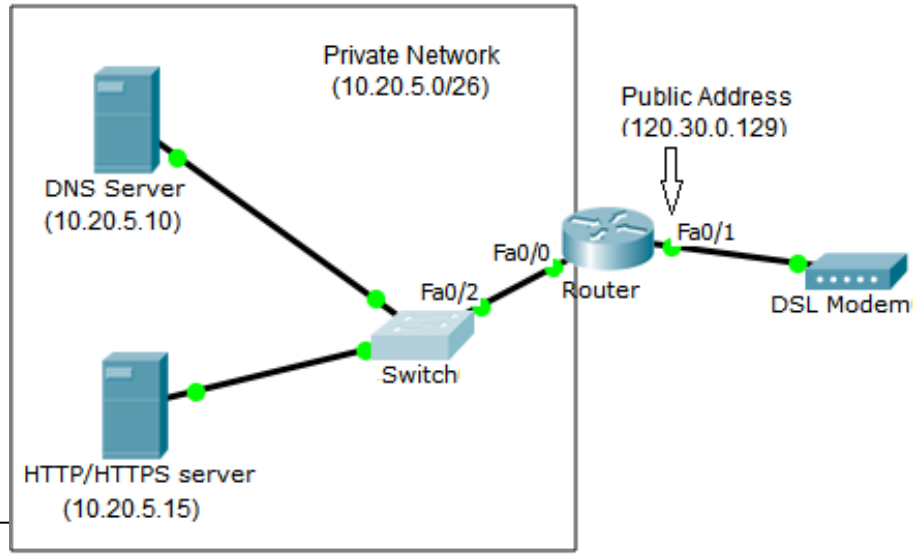
Static NAPT configuration

Static NAPT in a Cisco router is similar to static NAT, the difference is, now the static rules will identify protocols and port numbers:

ip nat inside source static PROTOCOL INSIDE-ADDRESS INSIDE-PORT OUTSIDE-ADDRESS OUTSIDE-PORT

PROTOCOL can be either **udp** or **tcp**.

In this example we have two servers in a private network, and we want them to be accessible from the internet at the router 120.30.0.129 public address. This is a classical case for static NAT.




```
interface Fa0/0
ip nat inside

interface Fa0/1
ip nat outside

ip nat inside source static udp 10.20.5.10 53 120.30.0.129 53
ip nat inside source static tcp 10.20.5.10 53 120.30.0.129 53
ip nat inside source static tcp 10.20.5.15 80 120.30.0.129 80
ip nat inside source static tcp 10.20.5.15 443 120.30.0.129 443
```

Notice this problem can't be solved with simple static NAT, static NAPT is required.

Cisco IOS NAT commands and Packet Tracer

In a real Cisco router, there are several ways to enforce NAT with the **ip nat** command:

Command	What it does
ip nat inside source ...	Apply SNAT to packets traveling from inside to outside. Apply DNAT to packets traveling from outside to inside.
ip nat outside source ...	Apply SNAT to packets traveling from outside to inside. Apply DNAT to packets traveling from inside to outside.
ip nat source ...	Apply SNAT (and DNAT in the opposite direction) without defining inside and outside interfaces. The traffic to with SNAT is applied is defined by an ACL.
ip nat inside destination ...	Apply DNAT to packets traveling from outside to inside. Apply SNAT to packets traveling from inside to outside. This configuration is used for server load balancing when we have a servers' cluster located inside.

On Packet Tracer, only the first two are available, they look like the same, only switching sides. The two alternatives may, however, be required if we want to implement on the same router different SNAT configurations in each direction.

Security on private networks

Generally speaking, private networks with dynamic NAT are awesome from the security point of view. Private nodes are hidden and protected from external access. As we have seen, dynamic NAT tables only allow temporarily access, and only for sessions started by the private client node.

There is, however, one issue. Network administrators are legally obliged to be able to tell authorities which local user is blameable for any access to internet servers.

The problem is, authorities will provide a source IP address used on the access and ask us who the user was. If we are using NAPT, the answer may be difficult to get because all users are sharing the same public address on internet accesses.

We could log all NAT translations (that will be a massive amount of data), but even so, if authorities are not able to provide us the source port number used on the access it could be difficult to pinpoint the accountable user.