Step by Step Configuration of Dynamic NAT

Unlike static NAT that provides a permanent mapping between an internal address and a specific public address, dynamic NAT maps private IP addresses to public addresses. Dynamic NAT uses a pool of public addresses and assigns them on a first-come, first-served basis.

When a host with a private IP address requests access to the Internet, dynamic NAT chooses an IP address from the pool that is not already in use by another host. Dynamic NAT is useful when fewer addresses are available than the actual number of hosts to be translated.
When configuring dynamic NAT, you need an ACL to permit only those addresses that are to be translated.

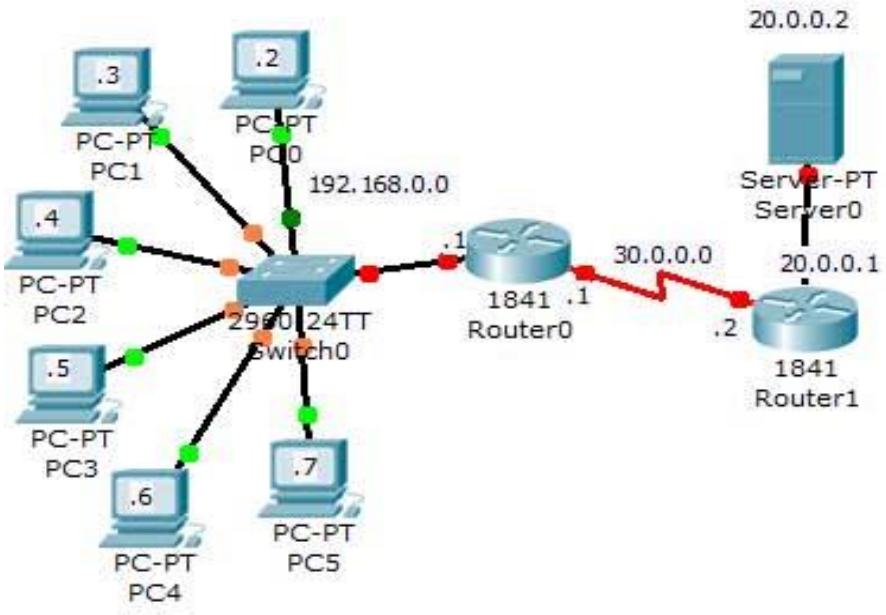
Access Control Lists (ACLs).

Access Control List (ACL) are filters that enable you to control which routing updates or packets are permitted or denied in or out of a network. They are specifically used by network administrators to filter traffic and to provide extra security for their networks. This can be applied on routers (Cisco).

ACLs provide a powerful way to control traffic into and out of your network; this control can be as simple as permitting or denying network hosts or addresses. You can configure ACLs for all routed network protocols.

The most important reason to configure ACLs is to provide security for your network. However, ACLs can also be configured to control network traffic based on the TCP port being used.

With dynamic NAT, you must manually define two sets of addresses on your address translation device. One set defines which inside addresses are allowed to be translated (the local addresses), and the other defines what these addresses are to be translated to (the global addresses).



In this example our internal network is using 192.168.0.0 network. We have five public ip address 50.0.0.1 to 50.0.0.5 to use. **Router1(1841 Router0)** is going to be NAT device. Double click on **Router1(1841 Router0)** and configure it as given below

```
Router>enable
Router#configure terminal
Router(config)#hostname R1
R1(config)#interface fastethernet 0/0
R1(config-if)#ip address 192.168.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip address 30.0.0.1 255.0.0.0
R1(config-if)#clock rate 64000
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 serial 0/0/0
R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
R1(config)#ip nat pool test 50.0.0.1 50.0.0.5 netmask 255.0.0.0
R1(config)#ip nat inside source list 1 pool test
R1(config)#interface fastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
R1(config)#exit
```

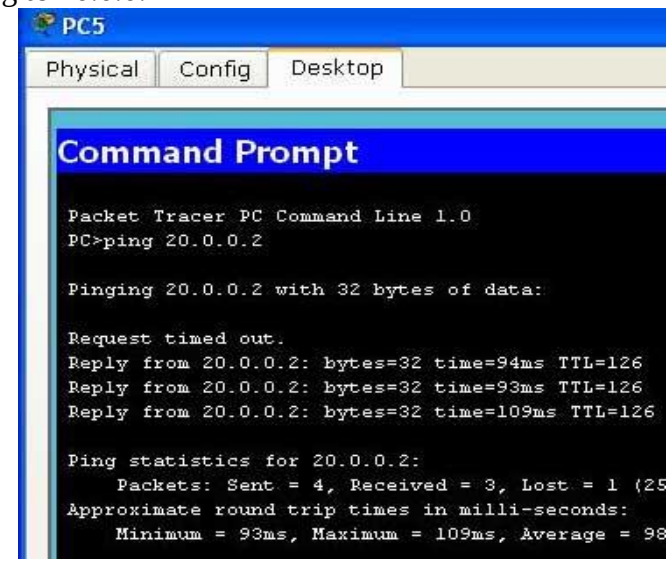
Now double click on R2(1841 Router1) and configure it as given below

```
Router>enable
Router#configure terminal
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 30.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#ip route 0.0.0.0 0.0.0.0 serial 0/0/0
Router(config)#hostname R2
```

For testing of NAT go R1 and enable debug for NAT from privilege mode

```
R1#debug ip nat
```

Now go on pc and ping to 20.0.0.2



When ICMP ping packet reach to R1. It examines its source address against the access list 1. As this packet is generated from the network of 192.168.0.0 so it will pass the access list. Now router will check NAT pools for free address to translate with this address. Which you can check in the output of debag command in R1

```
IP NAT debugging is on
NAT: s=192.168.0.7->50.0.0.1, d=20.0.0.2[1]
NAT*: s=20.0.0.2, d=50.0.0.1->192.168.0.7[1]
NAT: s=192.168.0.7->50.0.0.1, d=20.0.0.2[1]
NAT*: s=20.0.0.2, d=50.0.0.1->192.168.0.7[1]
NAT: s=192.168.0.7->50.0.0.1, d=20.0.0.2[1]
NAT*: s=20.0.0.2, d=50.0.0.1->192.168.0.7[1]
NAT: s=192.168.0.7->50.0.0.1, d=20.0.0.2[1]
NAT*: s=20.0.0.2, d=50.0.0.1->192.168.0.7[1]
```

As you can see in output 192.168.0.5 is translate with 50.0.0.1 before leaving the router.

Now check for web access from any client pc

In real life its best practices to turn off debug after testing so go on Router 1 and turn off debug mode.

```
R1#no debug ip nat IP NAT debugging is off R1#
```

Explanation:

```
ip route 0.0.0.0 0.0.0.0 serial 0/0/0
```

The default route is the IP address of the next hop when no other routes are known.

To configure the default route to be 192.168.1.1:

```
config t ip route 0.0.0.0 0.0.0.0 192.168.1.1
```

An interface can be used as an alternative to and IP address. To use serial0/0 for destinations not in the routing table, use:

```
ip route 0.0.0.0 0.0.0.0 serial 0/0
```

Basics of Cisco IP access control lists

If you're managing anything bigger than a home network, you probably have a router or two in your network closet. These routers represent the first line of protection for your network from the rest of the world. By implementing

access control lists (ACLs), you can enable or deny the flow of data to and from your network. In this article, we'll discuss setting up an Inbound ACL on your border router (the one that connects you to the Internet) to provide some basic security for your internal network.

Cisco IP ACLs can be divided into two basic types: Standard (sometimes called basic) and Extended. Both lists have a standard format used for their entry and a standard numbering scheme designed to help you differentiate which list is applied to which interface.

A word of caution: Before you jump into creating ACLs for your routers, know that all **ACLs have an implicit "deny all" statement** at the end of each list. You do not have to create this entry and it will not appear when you display the list. However, by default, it is the last entry on every list. That means that if you don't explicitly permit traffic to or from your network, it is denied as a rule when you apply the list. Also, make things easy on yourself and instead of working directly from the Cisco IOS command line, first type out your ACLs in Windows Notepad. Just make sure you restrict access to this file. It should not be public knowledge and should never be shared with anyone outside your network.

A Standard IP ACL controls the flow of information based on network addresses. These lists are created and applied to an interface as either inbound or outbound packet filters. The syntax follows this format:
access-list [list number] [permit | deny] [source address] [wildcard-mask] [log]

- List Number = A number from 1 to 99 (think of it as the name of the list)
- Permit | Deny = Whether to permit or deny this packet of information
- Source Address = The address of the machine from which the packet originates
- Wildcard Mask = The network mask to use with the source address (Cisco masks are a little different; 0=octet must match exactly; 255=octet is not significant or doesn't matter)
- Log = Whether to log this entry to the console (if logging is enabled)

A sample inbound ACL would be:
access-list 1 deny 10.0.0.0 0.255.255.255 log
access-list 1 deny 172.16.0.0 0.15.255.255 log
access-list 1 deny 192.168.0.0 0.0.255.255 log
access-list 1 deny 127.0.0.1 0.0.0.0 log
access-list 1 deny 10.1.10.1 [replace with your IP address] 255.255.255.0 [replace with your subnet mask] log

access-list 1 permit 0.0.0.0 255.255.255.255 log

In this example, entries 1, 2, and 3 deny the non-routable (private) IP addresses for each network class as defined by RFC 1597.

Now let's look at how the entries differ and what they specifically deny or permit:

- Entry 1—"access-list 1 deny 10.0.0.0 0.255.255.255 log"—is a deny statement for packets with an originating address of 10.0.0.0 to 10.255.255.255. Rather than having to make 16 million entries to block each Class A IP address, I use the mask 0.255.255.255. This tells the router to match the first octet of 10 and disregard the other three octets.
- Entry 2—"access-list 1 deny 172.16.0.0 0.15.255.255 log"—is a deny statement for packets with an originating address of 172.16.0.0 to 172.31.255.255. Once again, rather than make 1 million entries to block each Class B IP address, I use the mask 0.15.255.255. This tells the router to match the first octet of 172 and to match the second octet of 16 with a range of 15 additional networks and to disregard the third and fourth octet.
- Entry 3—"access-list 1 deny 192.168.0.0 0.0.255.255 log"—is a deny statement for packets with an originating address of 192.168.0.0 to 192.168.255.255. Instead of making 65,000 entries to block each Class C IP address, I use the mask 0.0.255.255. The router will match the first two octets exactly and disregard the last two octets.
- Entry 4—"access-list 1 deny 127.0.0.1 0.0.0.0 log"—is a deny statement for packets with an originating address of 127.0.0.1, which is the hardware loop-back address of any Ethernet adapter. You could leave off the mask because a Standard IP ACL assumes a mask of 0.0.0.0 if none is specified.
- Entry 5—"access-list 1 deny 10.1.10.1 [replace with your IP address] 255.255.255.0 [replace with your subnet mask] log"—will deny anyone from externally spoofing your network.
- Entry 6—"access-list 1 permit 0.0.0.0 255.255.255.255 log"—is a permit entry to allow packets that were not previously rejected to enter your network.

Note

Remember that by default, all Cisco access control lists have an implicit "deny all" statement at the end of each list. You do not have to create this entry, and it will not appear when you display the list. However, this means

that if you apply an empty ACL to an interface, then no traffic will pass through that interface.

You probably noticed that I put *log* at the end of every entry. I do this because every packet "walks down the ACL" until the router finds an entry it can apply to the packet. Some packets will be rejected by the first entry, and some packets will make it all the way to the last entry before they meet an entry that can be applied to them. The log comes in handy when you want to speed up processing of packets. After your ACL is running for a few minutes, you can type the command *show ip access-list 1*. You'll then see something like this:
access-list 1 deny 10.0.0.0 0.255.255.255 log
access-list 1 deny 172.16.0.0 0.15.255.255 log
access-list 1 deny 192.168.0.0 0.0.255.255 log
access-list 1 deny 127.0.0.1 0.0.0.0 log (8 matches)
access-list 1 deny 10.1.10.1 [replace with your IP address] 255.255.255.0 [replace with your subnet mask] log (87 matches)
access-list 1 permit 0.0.0.0 255.255.255.255 log (4320 matches)

As you can see, this says that entry 4 denied eight packets, and entry 5 denied 87 packets. Now to save processing time and speed up the flow of packets into your network, rearrange your ACL to move entry 5 to the top of list, followed by entry 4. A simple rule to follow is to have your most specific deny and permit statements at the top of the list, followed by the most active general entries.

Finally, let's apply the ACL that we created:

1. Open the ACL you typed in Notepad.
2. Copy the contents of the list to the Clipboard.
3. Use a terminal emulator such as HyperTerminal to Telnet into your router.
4. Log on to your router.
5. Enter privileged exec mode.
6. Type *config terminal*.
7. Paste your ACL to the command line and press [Enter].
8. Go to the interface you want to apply your ACL to (e.g., interface e0).
9. Type *ip access-group 1 in*. To remove an ACL, type *no ip access-group 1 in*.

Your list is now entered and applied to an interface. Check it periodically

with the *show ip access-list 1* command and make adjustments as necessary.

Wildcard masks are the inverse of the subnet, the 0's are significant and the 1's are not significant. If you compare the source address to the wildcard mask, the source address bits that match up with the 0's specifies the network (only one) and the source address bits that match the 1's are all the hosts to which the list applies.

Wildcard Examples

Source	Wildcard	Host Range	Matches
192.168.1.55	0.0.0.0	192.168.1.55	**One host**
192.168.1.0	0.0.0.255	192.168.1.1 - 192.168.1.255	Last octet (**one network**)
192.168.0.0	0.0.255.255	192.168.0.1 - 192.168.255.255	Last two octets (**entire 192.168.0.0 network**)
142.110.16.0	0.0.7.255	142.110.16.1 - 142.110.23.255	Last octet and right 3 bits of second from right octet (**part of the network**)
172.16.32.0	0.0.31.255	172.16.32.1 - 172.16.63.255	Last octet and right 5 bits of second from right octet (**part of the network**)

Example for Standard IP Access Lists

```
Router(config)#access-list 1 deny host 192.168.1.4
Router(config)#access-list 1 permit 0.0.0.0 255.255.255.255
Router(config)#int e0
Router(config-if)#ip access-group 1 out
```

This access list allows traffic from 192.168.1.4 to enter the router, but the access list denies it from exiting on interface Ethernet 0. The deny statement uses the default wildcard mask of 0.0.0.0 (i.e. all bits are significant and it only applies to one host). The 0.0.0.0 255.255.255.255 can be replaced with the word `any`. The list is applied to the outbound of one interface as opposed to the inbound. This will prevent the host being blocked from other networks on the router that might not have been intended since traffic from 192.168.1.4 can enter the router and be switched to other networks on interfaces other than ethernet 0. You should apply the standard IP access list as close to the destination network as possible, or you could inadvertently block access to portions of your network. Use the command `show access-lists` to see the access lists on your router. For just IP access lists use the command `show ip access-list`.

To remove the access list use the command `no access-list [list #]`. Use the `show ip interface` and the `show interface` commands to verify that an access list has been successfully applied to an interface.