Sean McGrath
CMPE 151, Spring 2015
June 11, 2015

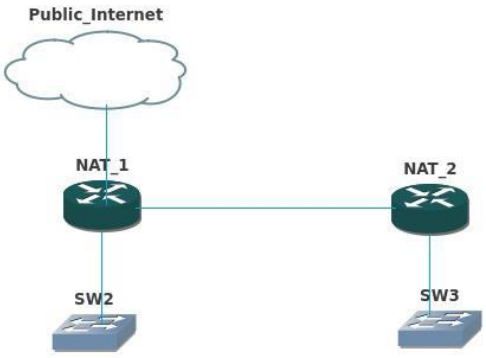
CMPE 151 Project Paper—Double NAT

Introduction

For my project I covered the topic of double Network Address Translation (NAT). My motivation for studying this topic was to gain a better understanding of NAT which was introduced to me in CMPE 150 and understand this particular NAT configuration that is seen as troublesome to the performance of a private network, and which can also be limiting to the capability to communicate between the various private subnets. The goal of the lab which I developed was to achieve a greater understanding of the address translation process in such a network, observe the performance impact caused by a second NAT router, and discover instances where double NAT can be configured in such a way which limits the ability for hosts to communicate within the private network.

Overview of Double NAT

Before outlining the lab, we will cover the technology involved. A double NAT is a network configuration in which a NAT router is itself connected to a private interface of another NAT router. We show an example topology below:



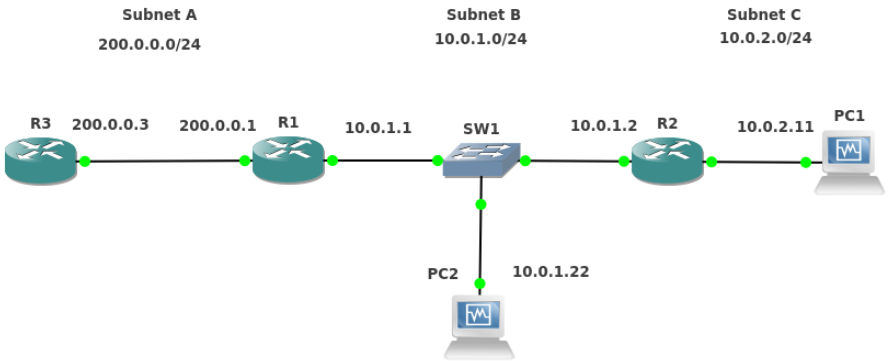
The result of this topology is that in order for any host on Router 2's private interface to communicate with the public Internet, the source address of that host must be translated twice, once by Router 2 and again by Router 1. This process also segregates subnets sitting behind Router 2 from those private subnets a layer above behind Router 1. In practice, such a topology is usually implemented unknowingly in a home or small business environment, most often caused by connecting a wireless router running NAT to a modem provided by your ISP. The greatest concern with double NAT is the impact on performance, the ability for private subnets to effectively communicate, and achieving peer-to-peer communication between hosts whose IP addresses are being translated by NAT. In my lab I do not look into peer-to-peer networking over

NAT, instead focusing on performance and how one can compromise the communication within a private network as would be the key administrative concerns in practice since it is assumed that peer-to-peer communication is handled by the application.

Overview of Lab

Topology 1:

In my first topology, I set up a double NAT network as shown below:



It should be noted that all routing configuration was done statically, and that Port Address Translation (PAT) was implemented to achieve address translation of multiple hosts on a subnet, mapping their private addresses to a single public IP address. This was done by means of an access list and Cisco's *overload* command on routers 1 and 2. Further detail of my PAT implementation is included in the later section on netref content.

In this step of the lab the goal is to achieve NAT configuration and communication between each subnet, as well as analyze added delay caused by R2's NAT process. We illustrate the translation process below via packet captures of *pings* from each PC to R3:

Pinging R3 from PC2:

Capture on R1's private interface

2	0.431554	10.0.1.22	200.0.0.3	ICMP	98	Echo (ping) request id=0x3c17, seq=1/256, ttl=64
3	0.454504	200.0.0.3	10.0.1.22	ICMP	98	Echo (ping) reply id=0x3c17, seq=1/256, ttl=254
4	1.436752	10.0.1.22	200.0.0.3	ICMP	98	Echo (ping) request id=0x3c17, seq=2/512, ttl=64
5	1.465363	200.0.0.3	10.0.1.22	ICMP	98	Echo (ping) reply id=0x3c17, seq=2/512, ttl=254

Capture on R1's public interface

3	1.845607	200.0.0.1	200.0.0.3	ICMP	98	Echo (ping) request id=0x3c17, seq=1/256, ttl=63
4	1.855670	200.0.0.3	200.0.0.1	ICMP	98	Echo (ping) reply id=0x3c17, seq=1/256, ttl=255
5	2.856579	200.0.0.1	200.0.0.3	ICMP	98	Echo (ping) request id=0x3c17, seq=2/512, ttl=63
6	2.866639	200.0.0.3	200.0.0.1	ICMP	98	Echo (ping) reply id=0x3c17, seq=2/512, ttl=255

Note that the source address 10.0.1.22 is translated to 200.0.0.1 by R1.

Pinging R3 from PC1 is similar, with one added translation observed:

Capture on R2's private interface

5	20.545926	10.0.2.11	200.0.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=1/256, ttl=64
6	20.592278	200.8.0.3	10.0.2.11	ICMP	98	Echo (ping) reply id=0xda16, seq=1/256, ttl=253
7	21.561705	10.0.2.11	200.0.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=2/512, ttl=64
8	21.603735	200.8.0.3	10.0.2.11	ICMP	98	Echo (ping) reply id=0xda16, seq=2/512, ttl=253

Capture on R1's private interface

3	12.451662	10.0.1.2	200.0.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=1/256, ttl=63
4	12.481943	200.8.0.3	10.0.1.2	ICMP	98	Echo (ping) reply id=0xda16, seq=1/256, ttl=254
5	13.462919	10.0.1.2	200.0.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=2/512, ttl=63
6	13.493386	200.8.0.3	10.0.1.2	ICMP	98	Echo (ping) reply id=0xda16, seq=2/512, ttl=254

Capture on R1's public interface

3	3.872189	200.8.0.1	200.8.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=1/256, ttl=62
4	3.882264	200.8.0.3	200.8.0.1	ICMP	98	Echo (ping) reply id=0xda16, seq=1/256, ttl=255
5	4.883511	200.8.0.1	200.8.0.3	ICMP	98	Echo (ping) request id=0xda16, seq=2/512, ttl=62
6	4.893582	200.8.0.3	200.8.0.1	ICMP	98	Echo (ping) reply id=0xda16, seq=2/512, ttl=255

The key take away we observe in this step of the lab is that with PC2 and R2 connected to the same interface on R1, communication is possible between PC2 and PC1 since R2 will never have two interfaces with the same IP prefix. As long as the routers each have a route to Subnet B and Subnet C, any host on these two subnets can easily communicate.

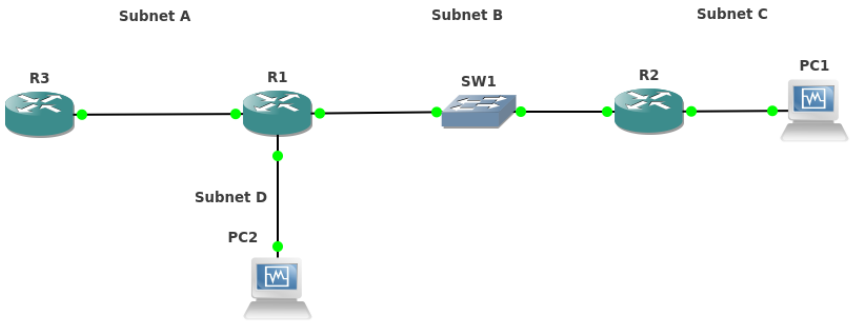
Analyzing Transmission Delay:

The process for observing and analyzing added delay caused by our second NAT router was simply to simultaneously *ping* R3 from PC1 and PC2 and average the observed transmission times reported on each host. We averaged the transmission times of 20 ICMP messages sent by each host and found the difference, as shown in the table below:

	PC1	PC2
	65.6	69.9
	69.3	29.9
	49.9	31.9
	47.9	24.4
	43.2	56.3
	51.7	20.7
	67.6	31.4
	46.8	27.7
	48.4	23.2
	46.9	27
	43.1	30.8
	55.8	29.2
	45.6	28.1
	45.1	31
	84.3	36.8
	44.8	30.2
	42.1	54
	79.6	30.3
	43.8	27.9
	48.5	24.4
AVG	53.5	33.255
PC1-PC2	20.245	

We found the average added delay by transmitting through a second NAT router to be about 20 milliseconds, slightly greater than we anticipated, but still not an added amount of delay to significantly impact the performance of the network.

Topology 2:



	Subnet A	Subnet B	Subnet C	Subnet D
IP prefix	200.0.0.0/24	10.0.1.0/24	10.0.2.0/24	10.0.2.0/24
R3	200.0.0.3			
R1	200.0.0.1	10.0.1.1		10.0.2.1
R2		10.0.1.2	10.0.2.2	
PC1			10.0.2.11	
PC2				10.0.2.22

In our second topology, we attach PC2 to a second inside NAT interface on R1, Subnet D, and give this subnet the same IP prefix as Subnet C. This is such an addressing configuration which results in Subnet D and Subnet C being unable to communicate with each other. This is because as PC2 tries to *ping* PC1, it assumes that they are both on the same subnet and issues an ARP Broadcast to determine PC1's MAC address, which of course fails.

3	17.661092	CadmusCo_c4:88:b0	Broadcast	ARP	42	Who has 10.0.2.11? Tell 10.0.2.22
4	18.660738	CadmusCo_c4:88:b0	Broadcast	ARP	42	Who has 10.0.2.11? Tell 10.0.2.22
5	19.665678	CadmusCo_c4:88:b0	Broadcast	ARP	42	Who has 10.0.2.11? Tell 10.0.2.22

The key detail in this part of the lab is that the two hosts can communicate if a static route to each is implemented in both hosts and the routers. We show our routing table entries for R1 and R2 below:

```
C    200.0.0.0/24 is directly connected, FastEthernet0/0
     10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
S       10.0.2.11/32 [1/0] via 10.0.1.2
C       10.0.2.0/24 is directly connected, FastEthernet2/0
C       10.0.1.0/24 is directly connected, FastEthernet1/0
R1#
```

```
S    200.0.0.0/24 [1/0] via 10.0.1.1
     10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C       10.0.2.0/24 is directly connected, FastEthernet1/0
C       10.0.1.0/24 is directly connected, FastEthernet0/0
S       10.0.2.22/32 [1/0] via 10.0.1.1
R2#show ip nat translations
```

Notice now that the routers know a route to 10.0.2.11 and 10.0.2.22, so PC1 and PC2 can communicate even though their subnets have the same IP prefix.

The final stage of the lab was to connect a second PC, PC3, to Subnet D and give it the exact same IP address as PC1. Obviously, now PC2 could not communicate directly with PC1 because any ICMP message with the destination address 10.0.2.11 was transmitted to PC3. What was interesting for us to observe, however, was that PC1 and PC2 could communicate if and only if PC1 initiated the message. This is because R2 is still translating PC2's IP to its own, for which PC1 has a route to. Evidence of this process is seen via packet capture on R1's Subnet D interface. Observe that the source address is 10.0.1.2, the address of R2, *not* PC1. PC2 has no idea that it is really communicating with PC1.

1	0.000000	cc:00:6f:97:00:20	cc:00:6f:97:00:20	LOOP	60	Reply
2	5.447980	10.0.1.2	10.0.2.22	ICMP	98	Echo (ping) request id=0x2517, seq=1/256, ttl=62
3	5.451736	CadmusCo_c4:88:b0	Broadcast	ARP	42	Who has 10.0.2.1? Tell 10.0.2.22
4	5.518657	cc:00:6f:97:00:20	CadmusCo_c4:88:b0	ARP	60	10.0.2.1 is at cc:00:6f:97:00:20
5	5.519202	10.0.2.22	10.0.1.2	ICMP	98	Echo (ping) reply id=0x2517, seq=1/256, ttl=64
6	6.417684	10.0.1.2	10.0.2.22	ICMP	98	Echo (ping) request id=0x2517, seq=2/512, ttl=62
7	6.418128	10.0.2.22	10.0.1.2	ICMP	98	Echo (ping) reply id=0x2517, seq=2/512, ttl=64

Conclusions

From the exercises performed in the lab, we can conclude that double NAT is not necessarily a bad topology because, if carefully implemented, it will result in effective communication between all internal, private subnets, and between each private subnet with the public Internet. We also observed that there is an added delay of approximately 20 milliseconds resulting from going through a second NAT router. Although this is less than ideal, it is only a minimal negative effect on performance, and is not significant enough to render double NAT an unsuitable configuration.

We also observed that the only addressing problems which compromise communication between the various private subnets are a result of one of the inside interfaces on R1 having the same IP prefix as an inside interface on R2. As long as the person(s) setting up the network are aware of this hazard, the double NAT can be implemented to allow for total internal communication, or even exploited as an easy means of segregating different parts of the network. The real problems with a double NAT topology, at least regarding communication within the private network, are most often a result of people being unaware that a double NAT has been configured.

Problems Encountered

The only problem I ran into was implementing Port Address Translation on the Cisco routers. This was easily resolved once I found this reference from Cisco: http://www.cisco.com/c/en/us/support/docs/ip/network-address-translation-nat/13772-12.html How I implemented PAT is detailed in the following section

Netref Content

Port Address Translation (PAT):

Port Address Translation (PAT) is a means of mapping multiple inside IP addresses to a single outside address, most often the address of the router's outside interface. Cisco refers to PAT as NAT overloading, and the *overload* IOS command is used to achieve PAT. The following IOS commands are useful in achieving proper PAT configuration:

ip access-list standard <List_Name>

This creates a standard ip access-list named <*List_Name*>

Permit <IP_prefix> <MASK>

This adds all IP addresses with the designated prefix and mask to the access list

ip nat inside source list <List> interface <Interface> overload

This command instructs the router to translate the source address of any packet received on the inside interface permitted by access-list *List* to the address of the router's outside interface *Interface*. Translations are overloaded, allowing multiple inside devices to be translated to the same IP address.

Example: Configuring PAT on a router who's inside interface has an IP prefix of 10.0.1.0/24 and translating any such address to the IP address of the router's outside interface fastEthernet 0/1:

```
conf t
ip access-list standard NAT
permit 10.0.1.0 0.0.0.255
exit
ip nat inside source list NAT interface fastEthernet 0/1 overload
end
```