Configure and Verify NAT for Given Network Requirements and Troubleshooting ACLs

Your Name: ­­­­­­­­­­­­­­­­­­­­­­­­Shivani Patel

DSI Number: ­­­­­­­­­­­­­­­­­­­­­­­­D40352579

NETW204

DeVry University

Date: ­­­­­­­­­­­­­­­­­­­­­­­­08/23/2015

Professor Name: ­­­­­­­­­­­­­­­­­­­­­­­­Jayson Sayers

Table of Contents

[Introduction 3](#_Toc396564211)

[Procedure 3](#_Toc396564212)

[Results 3](#_Toc396564213)

[Application and Analysis 4](#_Toc396564214)

[Questions 4](#_Toc396564215)

[Conclusion and Recommendation 4](#_Toc396564216)

[Lessons Learned 4](#_Toc396564217)

[References 4](#_Toc396564218)

# Introduction

Network address translation is an important service that saves IP addresses by using one single public IP address for multiple local IP address, we will learn how to create a pool of usable public IP addresses and configure NAT statically and dynamically. Also, Access Control Lists are exactly what the name means, there are security measures that control which devices get access to the network and it defines what their privileges are. To implement an ACL is not too difficult, but to troubleshoot ACLs and make sure the lists are permitting the correct users to gain access to the network, that is another lesson in itself. This lab we were required to take the given information and troubleshoot the ACLs.

# Procedure

To enable NAT we needed to first define the interfaces as facing “in” the network or facing “out” of the network. This was done by going into interface configuration mode for the specific interfaces and using the #ip nat inside, and #ip nat outside commands.

We then created a static NAT inside using the #ip nat inside source static {ip add/translated ip add} to the server. To test configurations we have to turn the interfaces off.

Helpful commands are #show ip nat translations, and #show users.

To configure dynamically we had to disable the static configurations with the #no ip nat inside source static {Ip add/translated ip add}. You can also use #clear ip nat translation if you do not want to delete child entries.

We have to create a pool of address first with the #ip nat pool {poolname/ ip add|range netmask}.

We created an access list so we can configure the NAT inside the access list. Once completed, we defined the dynamic NAT translation with the command #ip nat inside source list {list name} pool {poolname}.

For PAT we simply needed an access list to match the traffic, and the PAT translation command. The suffix “overload” is attached to the #ip nat inside source list {acl id} {interface id} overload perhaps to signify the extended ACL and that fact that it is a PAT not a NAT.

I actually don’t remember my entire procedure for the second part of the lab, I just remember I had to go to the week 6 lab a lot to try and see how we used ACL commands there. I don’t think there was a right way to troubleshoot this lab, just a long way and a short way to troubleshoot the lab and I think I went the long way. I ended up clearing the ACLs and just reconfiguring everything. I made sure all of the lists were using the correct wildcard masks because for the longest time I was using subnet masks.

# Results

I feel as if I was really proficient with the labs this week. This week we learned how to configure NAT and PAT inside and outside the network. Then there was a part where we had to manually go in and check to see if the ACLs were done correctly and had to delete them and rewrite some of the configurations. The part about the NAT that was extremely difficult for me to understand all at once was the part where we had to implement the ACL within NAT to keep the network service from translating certain parts of the network. We defined a pool of usable addresses for the NAT service to choose from. The ACL troubleshooting part helped me learn how to find errors and rectify them to get the network working properly. In that part we achieved a few things: we corrected the wildcard masks, applied ACLs in the correct direction, we reviewed the incorrect ACLs and corrected the order the lists were applied because of “first applied” rule where the first rule takes precedence over the subsequent rules.

# Application and Analysis

These concepts are to be applied in any field with any network size. I would change the redundancy of the commands in a live production environment. This lab was quite different from the previous labs because we were actually looking at the measures you would need to take to manage a network and keep it secure. Last, I think I have somewhat of an understanding between the concepts and the application; proof of this is visible in my procedure.

# Questions

1. NAT was a new topic for me and it was interesting to see how network administrators have to go in and manually create a pool of public IP addresses so that the service could assign public IP addresses dynamically. My discovery for this week’s lab was the wildcard mask. I learned how to troubleshoot ACLs and pay attention to my wildcard masks to keep from permitting the wrong group access to certain parts of the LAN. Another thing I had to really learn this week was how to check if my ACLs were working properly, I had also make sure the order of my ACLS was correct or the ACLs would get applied wrong.

2. I didn’t have a chance to plan this lab; you can say that I planned for this lab by studying the material and participating in the discussing. Afterwards, it was just doing the lab and verifying that I had it done correctly.

3. The most difficult part of this lab was understanding the concepts, not only did we revisit ACLs in a more complicated way, but we also configured NAT on our routers.

4. Um everywhere!!

# Conclusion and Recommendation

This week we covered the foundation of network implementation and how to properly troubleshoot ACLs. Before we were learning about things that you would do to create a network. Now these lessons are to actually bring the network to live production. These skills are useful in securing a network in a live production environment for any company that has network needs.

# Lessons Learned

Securing a network is an important lesson in itself. I am glad to have learned how to troubleshoot such a critical part of network security. Also, I learned is that using “?” after every phrase in a command is not only useful, but also necessary to do when you get to higher level commands because they are really easy to forget. This also prevents the admin from typing the incorrect commands.

# References

Class textbook