

# **VeriSign Test Bed Project**

## **Pix and Netscreen Configuration**

|                    |                           |
|--------------------|---------------------------|
| Student: Ryan Reed | Faculty Advisor:          |
| J#: J*****         | Date Completed: 7/20/2008 |

# Table of Contents

|                                       |         |
|---------------------------------------|---------|
| I. Organization Description           | Page 3  |
| II. Organizational Chart              | Page 4  |
| III. Central Problem                  | Page 4  |
| IV. Analysis of the Project           | Page 6  |
| V. Research and Sources               | Page 7  |
| VI. Pix Firewall Configuration        | Page 7  |
| I. Interface Configuration            | Page 9  |
| II. Routing                           | Page 10 |
| III. Network Address Translations     | Page 11 |
| IV. Access Lists                      | Page 12 |
| VII. Netscreen Firewall Configuration | Page 13 |
| I. Zone Configurations                | Page 14 |
| II. Routing                           | Page 16 |
| III. Network Address Translations     | Page 16 |
| IV. Service and Address Objects       | Page 17 |
| V. Policies                           | Page 19 |
| VIII. Solution Testing                | Page 20 |
| IX. Conclusion                        | Page 21 |
| X. Appendices                         | Page 22 |

## **I. Organization Description**

VeriSign is one of the leading network security companies in the world. VeriSign's goal is to provide a critical layer of intelligence and security that enables transactions online while protecting data and ensuring sensitive information is kept private. VeriSign offers various services to this extent, from its well known secure transaction services to the lesser known services, such as firewall management and IDS management.

I currently am working in the firewall department at the Providence Security Operations Center for VeriSign. The firewall department has many different responsibilities.

- 1.) Must create tickets sent to us by email from many various Fortune 500 companies
- 2.) Must be able to perform various changes on the firewalls supported by VeriSign, which include Checkpoint, Juniper Netscreen, and Cisco Pix firewalls
- 3.) Must be able to troubleshoot suspected firewall issues with any customer. In most cases, the issues perceived to be firewall issues by the customer tends to be issues somewhere on the clients network.

The Security Operations Center must be open 24 hours a day, 365 days a week. The Security Operations Center does not take a vacation due to the schedules of clients being different from those in America. This means that the SOC currently has 3 shifts to cover the entire day.

## II. Organizational Chart

\*\*Removed from this document\*\*

## III. Central Problem

This project began as a training project intended to get accustomed to using two of the 3 major firewalls that VeriSign currently deals with on a daily basis. The wide range of customers that VeriSign is currently under contract with means that there are many different types of firewalls that we deal with. The Cisco Pix, the Juniper Netscreen, and Checkpoint firewalls are the most commonly employed firewalls that VeriSign deals with.

This training project has been used with many new employees of VeriSign to see how quickly the employee could become familiar with the technologies used on a daily basis. The project also tests the actual knowledge of new employees by attempting to have the employees actually implement what they know. It tests employees' knowledge of subnetting, access-lists, and basic networking knowledge.

### Project Requirements

1. The network should consist of at least 1 workstation node and 1 of firewall.
2. 1 network will be using a Netscreen Firewall and the other shall be using a Cisco Pix Firewall
3. Each Firewall should be able to communicate with any device connected on it's Trusted network with no restrictions
4. The Firewall will need to distribute the IP addresses to each node on its Trusted network.
5. The two firewalls will be connected directly
6. The Netscreen will allow for the following access
  - a. Inbound access
    - i. Source: A node on the Pix Network

- ii. Destination: A server node on the Netscreen Network
    - iii. Services: FTP, HTTP, TCP 1024
  - b. Outbound Access
    - i. Source: A node on the Netscreen Network
    - ii. Destination: A server on the Pix Network
    - iii. Services SSH, TCP 2000, UDP 110
- 7. The Pix will allow for the following access
  - a. Inbound Access
    - i. Source: A node on the Netscreen Network
    - ii. Destination: A server on the Pix Network
    - iii. Services: SSH, DNS, TCP 2000, UDP 110
  - b. Outbound Access
    - i. Source: A node on the Pix Network
    - ii. Destination: A server on the Netscreen Network
    - iii. Services: FTP, HTTP, TCP 1024
- 8. Both firewalls will need the following types of Network Address Translations
  - a. HIDE NAT
    - i. The Netscreen should have two methods of accomplishing this
  - b. NONAT
  - c. 1 to 1 NAT
- 9. Each network should also allow for inbound ping requests from anyone on their Trust network and outbound ping requests from one specific host
- 10. The IP address schemes should make sense and be implemented as Class C networks
- 11. A default route should be implemented on each firewall
  - a. Due to the two firewalls being directly connected, the default route can point to the opposite firewall

## IV. Analysis of the Project

The idea behind the project is relatively simple. This simple network will have 2 different firewalls connected directly with at least 1 real node and 1 hypothetical node behind each firewall. The real problem is working with the firewalls when I have very little actual work with them. Each firewall is different in the way the firewall implements different networking utilities. Unfortunately, although there are standards in the form of protocols, there is no standard among how these protocols are handled once they reach a firewall.

Because there are not enough workstations in the lab, it means that only 1 real node will be actually being able to test the rules put in place. Rules created for the hypothetical node will not be able to be tested properly but we can make reasonable assumptions based on the rules for the functioning nodes. The testing of the rules for working node will help to determine if the rules for the hypothetical nodes would work if those nodes actually existed.

I will have to come up with a basic IP address scheme to cover the networks. The first scheme will cover from the network behind the Netscreen, the second scheme will cover the network behind the Pix firewall, and the last scheme will have to address the direct connection of one firewall to another. I believe the best way to simulate this is to use the IP scheme shown in figure 4.1. Behind the Netscreen firewall, we will be using a 192.168.1.0/24 IP address scheme. Behind the Pix firewall, an IP address scheme of 172.16.3.0/25 will be used. These two schemes will be used as the IP addresses in use are non-routable, private IPs, like you would see in the real world. The addressing of 4.2.2.0/24 between the firewall would simulate a scheme seen on the internet (for instance, 4.2.2.2 is the address of a public DNS server). These address schemes will allow for us to communicate, pending NATs and access-lists, across the firewalls without issue.

## V. Research and Sources

Cisco Pix. 13 July 2008 <[http://en.wikipedia.org/wiki/Cisco\\_PIX](http://en.wikipedia.org/wiki/Cisco_PIX)>.

Cisco Systems, Inc. "Cisco Pix Firewall Command Reference, Version 6.3." 2004. Cisco Systems. 8 July 2008 <[http://www.cisco.com/en/US/docs/security/pix/pix63/command/reference/63\\_cmd.pdf](http://www.cisco.com/en/US/docs/security/pix/pix63/command/reference/63_cmd.pdf)>.

Juniper Networks. Customer Support Center: Using a DIP Pool (KB ID: DB4748). 31 Jan 2008. 14 July 2008 <<http://kb.juniper.net/CUSTOMERSERVICE/KB4748>>.

Juniper Networks. Netscreen-25/Netscreen-50 - Enterprise VPN Solution. 12 July 2008 <[http://www.juniper.net/products\\_and\\_services/firewall\\_slash\\_ipsec\\_vpn/Netscreen\\_25\\_slash\\_Netscreen\\_50/](http://www.juniper.net/products_and_services/firewall_slash_ipsec_vpn/Netscreen_25_slash_Netscreen_50/)>.

SecManager. Basic Netscreen Commands. 10 July 2008 <[http://www.secmanager.com/Basic\\_Netscreen\\_Commands](http://www.secmanager.com/Basic_Netscreen_Commands)>.

Wikipedia. Juniper Networks. 10 July 2008 <[http://en.wikipedia.org/wiki/Juniper\\_Networks](http://en.wikipedia.org/wiki/Juniper_Networks)>.

## VI. Pix Firewall Configuration

Cisco created the Pix firewall “was one of the first products in this [network security] market segment” (Cisco Pix). The Pix firewall has been one of the leading firewall brands since its first development in 1994, when the problem with IP address shortages was first realized. The device was designed to help with hiding private networks behind the IP address of the firewall, thereby helping to take multiple private IPs and hide them behind just one register IP address.

Along with the strong ability network address technology used by the Pix, the firewall also restricts traffic as to protect the private network behind it. The access list is used in this case. The access list accepts traffic based on the rules in place on the firewall. The default method of traffic restriction is to deny any traffic except those the traffic that matches any of the rules in the policy. Why would the Pix use a default traffic drop

when inspecting traffic? This is due to the idea that a proactive solution should be utilized instead of a reactive solution.

In a reactive environment, the firewall would have a default “allow any” that would allow any traffic to pass the firewall except those matched in the access list. This type of firewall would leave the network open for attack until a user decided that it was time to block the user. This would create a huge security hole that depends solely on the firewall administrator to close. If, for instance, a firewall administrator was unable to get to a computer for a length of time and a hacker was on the network, that hacker would have unlimited access to the network to capture whatever information he could until the administrator was able to put in the access to block this activity.

The second option is a proactive solution. This type of firewall blocks all traffic except that which exists in the access lists. This means that, by default, all traffic is blocked and a malicious user could not access the network due to the rules having to be in place in the first place to allow this user through. The problem with this type of solution is that specific access would have to be opened for every workstation and user on the network. This solution is much more elegant and secure. That is why most firewalls work on the proactive method of network security rather than the reactive method.

Now to the actual configuration of the device. This is done through the command line interface. There is a graphical user interface for configuring the firewall but it is not being utilized on this firewall. We will be connecting to the firewall through a serial management cable as none of the interfaces have been configured yet so the firewall will not accept our connection yet.

## VI. I. Interface Configuration

The Pix firewall works on the idea that each interface must be configured and subsequently an access list should be tied directly to the interface. What does this mean for traffic inspection? This means that traffic inspection is based upon the interface that the traffic is first entering regardless of what interface the traffic might be leaving. Due to the environment of this lab, there are only two interfaces being utilized that will need to be configured, an “outside” interface and an “inside” interface. The outside interface will be the interface that is facing the “internet” of insecure networks and the inside interface will be connected to the trusted networks or internal network. In most Pix firewall setups, there are more interfaces in use for protecting DMZ networks, etc but the network we have setup will only need two interfaces. First, the outside interface must be configured.

```
conf t  
int eth0  
nameif outside  
ip address 4.2.2.254 255.255.255.0  
no shutdown  
exit
```

The first line of code is the “conf t.” This is short for configure terminal. Configure terminal allows for us to actually start entering our commands that will alter the configuration of the firewall. “int eth0” stands for Interface Ethernet0 which will bring us into the interface configuration. This allows us to change settings specifically related to this interface. “Nameif outside” allows us to name this interface. Outside is a name widely used when describing the interface facing the untrusted networks. The “ip address 4.2.2.254 255.255.255.0” gives the interface an ip address of 4.2.2.254, which is the IP address we decided to use earlier for this interface. The “no shutdown” command will ensure the interface is up and not down. Lastly, we need to exit out of this interface’s configuration so that we can configure the second interface.

```
int eth1  
nameif inside  
ip address 172.16.3.1 255.255.255.0  
no shutdown  
exit
```

The configuration of this interface is extremely similar to that of the outside interface. The “conf t” command is not necessary here as we should still be in the configuration terminal from the eth0 configuration. Entering into the eth1 interface configuration, we name the interface inside with the “nameif inside” command. We also must give the interface an ip address that will also be utilized by the internal network. The IP address 172.16.4.1 will be used by this interface. We have to make sure the interface is brought up and then exit the interface’s configuration. The configuring of the interfaces is now complete.

## **VI. II. Routing**

Routing allows for traffic to be moved through the firewall. A route must say what traffic will be routed, based on destination, through which interface the traffic will leave, and to what gateway the traffic will be passed. In our case, we will only need one route command to route the traffic.

```
route outside 0.0.0.0 0.0.0.0 4.2.2.1
```

This command says that any traffic that is not directly connected should route through the outside interface and to the 4.2.2.1 gateway. If we had multiple interfaces or had

another gateway on the 4.2.2.0 network, we could route traffic destined to 69.69.69.0, for instance, to the outside interface via a gateway such as 4.2.2.25. Because our internal network is directly connected, we have no need for another route besides the default route.

### VI. III. Network Address Translations

Network address translation, or a NAT, allows us to hide many IPs behind one IP or various other methods of translating IPs to another. For instance, the HIDE NAT hides all traffic from certain IPs behind one other IP. This is a many-to-one translation. A 1-to-1 NAT will translate traffic from IP to another IP address. The final NAT is a NONAT which does not NAT traffic.

```
nat (inside) 1 172.16.3.0 255.255.255.0
global (outside) 1 interface

static (inside,outside) 4.2.2.200 172.16.3.11 netmask 255.255.255.255
static (inside,outside) 172.16.3.5 172.16.3.5 netmask 255.255.255.255
```

First we need to create the HIDE NAT for hiding any traffic that does not match the other NATs. We create a NAT with “nat (inside) 1 172.16.3.0 255.255.255.0.” This creates a NAT that has an id number of 1 and says any traffic on the inside interface with an IP address on the 172.16.3.0 network will be translated. Next we assign this NAT to a global. This global “says” traffic leaving this interface, matching NAT id 1, will hide behind the interface IP address, which happens to be 4.2.2.254. Any IP address could have been used instead of “interface” but it’s easier to just use the interface IP.

The third line should be read as “traffic with the IP address of 4.2.2.200 from the outside interface will be translated to 172.16.3.11 through the inside interface.” Statics are

confusing to read and can be difficult to understand. So any traffic from 4.2.2.200 will be translated to the address 172.16.3.11 and will leave the firewall through the inside interface. This creates a 1-to-1 NAT. The last static create a NONAT so that the traffic to and from 172.16.3.5 will not be translated. This is to override the global HIDE NAT created earlier.

#### VI. IV. Access Lists

```
access-list external extended permit tcp host 192.168.1.5 host 172.16.3.10 eq ssh  
access-list external extended permit tcp host 192.168.1.5 host 172.16.3.10 eq domain  
access-list external extended permit tcp host 192.168.1.5 host 172.16.3.10 eq 2000  
access-list external extended permit udp host 192.168.1.5 host 172.16.3.10 eq 110  
access-list external extended permit icmp host 192.168.1.5 host 172.16.3.5  
access-list external extended permit icmp host 4.2.2.1 host 172.16.3.5
```

```
access-list internal extended permit tcp host 172.16.3.5 host 192.168.3.10 eq ftp  
access-list internal extended permit tcp host 172.16.3.5 host 192.168.3.10 eq 1024  
access-list internal extended permit tcp host 172.16.3.5 host 192.168.3.10 eq www  
access-list internal extended permit icmp host 172.16.3.5 any
```

```
access-group external in interface outside  
access-group internal in interface inside
```

Here is where will determine exactly what traffic will be allowed to pass this firewall. The above is relatively easy to understand just by reading the command. Reading the first command will help us to understand exactly what is happening. This rule will be applied to the “access-list external.” We could have used any name instead of external but this makes it easy to differentiate the difference between the internal network and external network. The external access-list will use extended permissions to permit tcp from the 192.168.1.5 host to the 172.16.3.10 using ssh (port 23).

The first 4 lines were put in place to solve part of the requirements requested for this project. The fourth and fifth lines were put in place to allow icmp traffic. The reason for this is that we did not actually have any applications that required those other ports (ssh, dns, 2000, 110, etc) so we could not test our access. To really test our traffic, we needed to allow for PING traffic from the hosts behind the Netscreen to ping a host on our network, in this case 172.16.3.5.

The next four lines of the access list are also easy to understand. The first three are to enable us to solve the rest of the requirements for the Pix firewall. The fourth line is to allow the 172.16.3.5 host to ping any other host. The last two lines are what makes the access-lists work. The access-group command ties an access list to an interface. What the first access-group command says is that the external access-list should be applied to traffic coming in the outside interface. Without the last 2 lines, the access-lists would do absolutely nothing. Finally, the last command we need to do is a “wr mem” which will write all the configuration commands we have performed thus far to the firewall’s memory.

With the access lists in place and assigned to the interfaces, the firewall is now set to accept traffic. Except for the static NATs, most of the commands related to the firewall are extremely easy to understand. The commands are so easy that they can be read and understood right away in most cases. The Netscreen, unfortunately, takes a little more time to understand how the firewall works.

## **VII. Netscreen Firewall Configuration**

The Juniper Netscreen firewall is another extremely common firewall used in the networking industry to perform network address translation and stateful packet inspection. Just like the Pix, the Netscreen can perform many functions from the simple NATing to the more complex virtual private network management. Both, Netscreen

firewalls and Pix firewalls, work on the premise that a default deny is more desirable due to being proactive.

Both firewalls utilize stateful packet inspection. Stateful packet inspection allows for the deep packet inspection of a packet. The firewall will read, not only the header of the packet, but the actual contents of the packet. Why is this important? This is important because if a would be hacker is trying to telnet to the FTP port, 21, of a node, the firewall should be able to see that the packet is not FTP related traffic and drop the packet without ever putting the target node in danger.

So, if both firewalls are similar, how is a Netscreen firewall different from a Pix firewall? Netscreen firewalls do not use interfaces as much as they use what they call "zones." This is different because multiple interfaces can be part of a zone, so the policy does not need to be duplicated for each interface. Netscreen firewalls also do not have policies (what was known as an access list on the Pix firewall) tied to the zones. The policy management is based upon what zone the traffic is originating on and what zone the traffic is destined for. We'll go more into what exactly this means in the policies section.

### **V. I. Zone Configuration**

As mentioned earlier, a Netscreen does not have interfaces. A Netscreen uses zones. In our network environment, there are only two zones. The first is the trust zone in which the private network for the Netscreen is contained. The second zone we will be using is the untrust zone. This zone is for anything not inside the Netscreen's internal network, such as the Pix or Pix network.

```
set interface "ethernet1" zone "Trust"  
set interface "ethernet2" zone "DMZ"  
set interface "ethernet3" zone "Untrust"
```

To begin, we need to create the zones. Our Netscreen is smart enough to be able to configure the zones automatically as there are only two interfaces being utilized on this firewall, but if there were more, the following commands would have been necessary.

This does exactly what it sounds like. The first command sets the interface “ethernet1” as the zone “Trust.” The second line sets the “ethernet2” interface as the zone “DMZ” and third line sets the interface “ethernet3” as the zone “Untrust.” We could have potentially named these zones whatever we wanted but Trust, Untrust, and DMZ are extremely well known networking terms.

```
set interface eth1 ip 192.168.1.1/24  
set interface eth3 ip 4.2.2.1/24
```

Just like the Pix firewall, we need to setup the addressing scheme we will be using for both interfaces. The first line, “set interface eth1 ip 192.168.1.1/24” will set the interfaces IP address to 192.168.1.1 and will allow the network behind ethernet1 to be using the 192.168.1.0 networking scheme. The second line will set interface ethernet3’s ip address to 4.2.2.1 and allows for a 4.2.2.0 networking scheme. This basic setup will have both interfaces up and configured with their zones and IP addresses.

## V. II. Routing

Routing is almost exactly like that of a Pix firewall.

```
set route 0.0.0.0/0 interface ethernet3 gateway 4.2.2.254
set interface ethernet3 route
```

This sets the default route for any traffic. This command says to set the route of any traffic that does not match any other routes to be sent to the gateway 4.2.2.254 through interface ethernet3. If we had any other routes, they would take precedence over the default route we just created. In both Netscreen firewalls and Pix firewalls, traffic will take the most specific route in place in the configuration.

The last line basically says to set the interface ethernet3 to route mode, where traffic entering this interface will be routed and not NAT'd. If we wanted to create a hide NAT, we could replace "route" with "nat." This would hide traffic entering this interface to hide behind this interface's IP address. We will be going through a separate method to HIDE NAT our traffic in the next section of this paper.

## V. III. Network Address Translations

There are three types of network address translations that we will be setting up on this firewall. The first is a HIDE NAT, the second is a NONAT, and the third is a 1-to-1 NAT. The NONAT is simply accomplished by putting an interface into route mode, like we did in the above section.

The HIDE NAT can be accomplished in 2 ways. The first is to use set the interface from route to NAT mode. This would mean traffic entering that interface would be NAT'd behind the interface's IP address. A DIP, or Dynamic IP, allows for the creation of a DIP id that can be used in rules later in the policy to use for a NAT. "This DIP application is

very similar to NAT, except that it is done on a policy basis instead of by interface” (Juniper Networks).

The first line is our DIP. What this says is to set a dip, with a dip id of 4, on interface ethernet3. The 2 IP addresses are a range of IP addresses to use as the HIDE NATs. We only want to HIDE behind the one IP so we’ll have a start and end IP address of 4.2.2.100. This NAT will not be used until it has been used in a policy which comes a little later in the paper. This DIP can be used to change the destination NAT but in most cases, a source translation is used.

The second line will create our 1-to-1 NAT using a MIP or Managed IP. This line says set the interface ethernet3’s mip to 4.2.2.101 when coming from the host 192.168.1.101. So, whenever the node with the IP address of 192.168.1.101 initiates traffic to outside the trusted network, the source to 4.2.2.101.

#### **V. IV. Service and Address Objects**

One of the largest differences between Netscreen firewalls and Pix firewalls is in the creation of objects. On a Pix firewall, when you create a rule, you basically give it the destination IP, source IP, and service. On a Netscreen, you have to create host, network, and service objects that will be used in the rules. Here is what the list of objects would look like

```
set address "Trust" "h192.168.1.10" 192.168.1.10 255.255.255.255
set address "Trust" "h192.168.1.5" 192.168.1.5 255.255.255.255
set address "Trust" "n192.168.1.0" 192.168.1.0 255.255.255.0
```

```
set address "Untrust" "h172.16.3.10" 172.16.3.10 255.255.255.255
set address "Untrust" "h172.16.3.5" 172.16.3.5 255.255.255.255
```

```
set service "tcp_1024" protocol tcp dst-port 1024-1024
set service "tcp_2000" protocol tcp dst-port 2000-2000
set service "udp_110" protocol udp dst-port 110-110
```

```
set group service "server"
set group service "server" add "FTP"
set group service "server" add "HTTP"
set group service "server" add "tcp_1024"
exit
```

```
set group service "172server"
set group service "172server" add "SSH"
set group service "172server" add "tcp_2000"
set group service "172server" add "udp_110"
exit
```

When creating a host or network object, you have to create the objects in the zone that the objects would fall in. So, when creating hosts for the 192.168.1.0/24 network, they would have to be in the trust zone. Any host objects from the 172.16.3.0/24 network would have to be in the Untrust zone. The first three lines create the host and network objects in the Trust zone. The next two lines create the hosts objects for the Untrust zone.

The next three lines are for creating the service objects. Service objects do not require a zone when creating the object as they can be used on any zone. When creating the service object, you have to specify the protocol that will be used, tcp or udp, and the destination ports. You can also specify the source ports but in our case, it is unnecessary.

The next line is where we will create a group for the service objects. We do not have to do this but it makes things easier to understand when creating the policies in the future about what is happening. We gave the first group an arbitrary name such as "server." Once we set the group service name, we enter into the group's configuration. The next three lines add FTP, HTTP, and tcp\_1024 to the group. HTTP and FTP are predefined ports that already exist on the firewall. HTTP and FTP also support protocol inspection to ensure that the correct protocol is being used for ports 80 and 21 so creating these services ourselves would create a security conflict. Before we can create the second service group, we have to exit out of the first group's configuration with the "exit" command. Lastly we create the 172server service group and exit out of that configuration too.

## V. V. Policies

Policies are to a Netscreen firewall what access lists are to a Pix firewall. The biggest differences between how policies work and how access lists work is that on a Netscreen, rules are applied from and to locations while on a Pix, access-list are only applied on the interfaces the traffic is originating from. Netscreens also have the ability to NAT based on policy rules. This is convenient when you want to NAT traffic from one source to one destination but not a different destination.

```
set policy id 1 from "Untrust" to "Trust" "h172.16.3.5" "h192.168.1.10" "server" permit log count
set policy id 2 from "Untrust" to "Trust" "h172.16.3.5" "n192.168.1.0" ping permit log count
set policy id 3 from "Trust" to "Untrust" "h192.168.1.5" "h172.16.3.10" "172server" permit log count
set policy id 4 from "Trust" to "Untrust" "n192.168.1.0" any ping permit log count
set policy id 5 from "Trust" to "Untrust" "n192.168.1.0" "any" "any" nat src dip-id 4 permit log count
```

Reading this one line at a time may make it easier to understand. An important note is that specifying a policy id is not required. A new policy will be created with the next open policy id. These Policy IDs do not change. For instance, if I removed line 3, Policy ID 4 would not change to Policy ID 3. It would stay as Policy ID 4. I have set the Policy IDs to keep my rules in order.

Reading the first line, the rule is saying to set policy ID as 1 and to permit traffic from the “Untrust” zone to the “Trust” zone from the source 172.16.3.6 to 192.168.1.10 on any service in the “server” service group. This traffic will also be logged and counted for troubleshooting or auditing purposes. The next three lines are very similar in the way the policies are created. We must make sure that the policies are created on the correct zones (to and from) and with the correct service groups.

The last line is the more difficult to understand. It’s similar in structure, setting the policy id from “Trust” to “Untrust” with a source network of 192.168.1.0/24 to any with any service but the section following is a little different. That section of code says to NAT the source with a DIP whose ID is 4. Earlier in this project, we created this DIP so this DIP should be ready for use. DIP ID 4 will translate the source of any traffic from the 192.168.1.0/24 network that did not meet the previous rules to the source of 4.2.2.100. This is our “catch all” HIDE NAT.

## **VIII. Solution Testing**

With the Netscreen policies completed, we should now be prepared to test the solution to see if we can indeed use the access that we have opened. What makes this hard to really test is that there are two hypothetical nodes that we can not entirely test. Also, the requirements also stated that access should be open on specific ports to the non-hypothetical nodes. Unfortunately, we can not entirely test those nodes either as we do not have any applications that can use these ports besides telnet which has been

disabled on these workstations for incoming traffic (system based). Before the policies and access-lists were implemented, I attempted to ping the hosts and see if the traffic would pass.

Figure 8.1 shows an attempt that I made to ping 192.168.1.5 from 172.16.3.5. As you can see, each ping request failed. Figure 8.2 shows a ping request from 192.168.1.5 to 4.2.2.254, which happens to be the Pix firewall outside interface. The ping requests for 4.2.2.254 also timed out due to the packets being dropped by the firewalls.

After applying the policies to both the Netscreen and Pix firewalls, I attempted the same traffic again, pinging first from 172.16.3.5 to 192.168.1.5. Figure 8.3 shows my successful attempt at “pinging” this address. The traffic passed without dropping any packets. Figure 8.4 shows the successful ping attempt from 192.168.1.5 to 4.2.2.254. It appears the rules that were put in place are functioning, as far as can be tested.

## **IX. Conclusion**

Setting up each firewall presents pros and cons for using either firewall. For instance, the Pix firewall tends to be the easiest to understand when setting up. The commands are straight forward, with the exception of the static commands, and easy to understand. Unfortunately, with a Pix firewall, you cannot create NATs based on specific rules, which can mean you have a lot of complicated statics or global NATs.

The Netscreen on the other hand provides a little more security. When you create policies, you don't create tie the policies to just the interface, you set the policy from one zone to another which means, for instance, you can create an any rule from a Trust to Trust2 zone without having the rule also apply to Trust and Untrust zones. Whether protecting a home network or a large business network, either firewall easily has the capabilities to completely lock down a network from suspicious traffic when configured correctly.

## X. Appendices

Figure 2.1 – Organization Layout (High Quality)

\*\* Removed from this document \*\*

Figure 4.1 – Network Layout

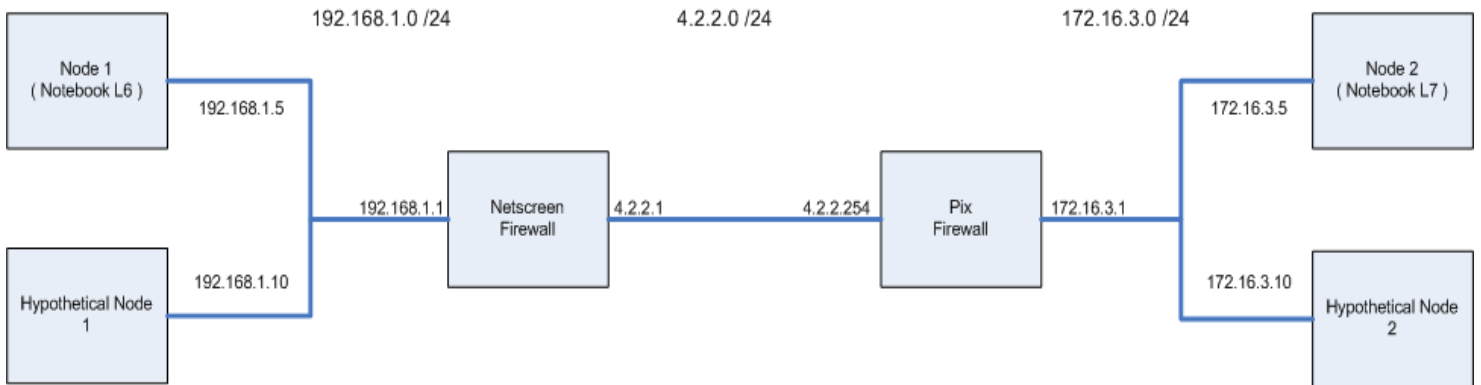
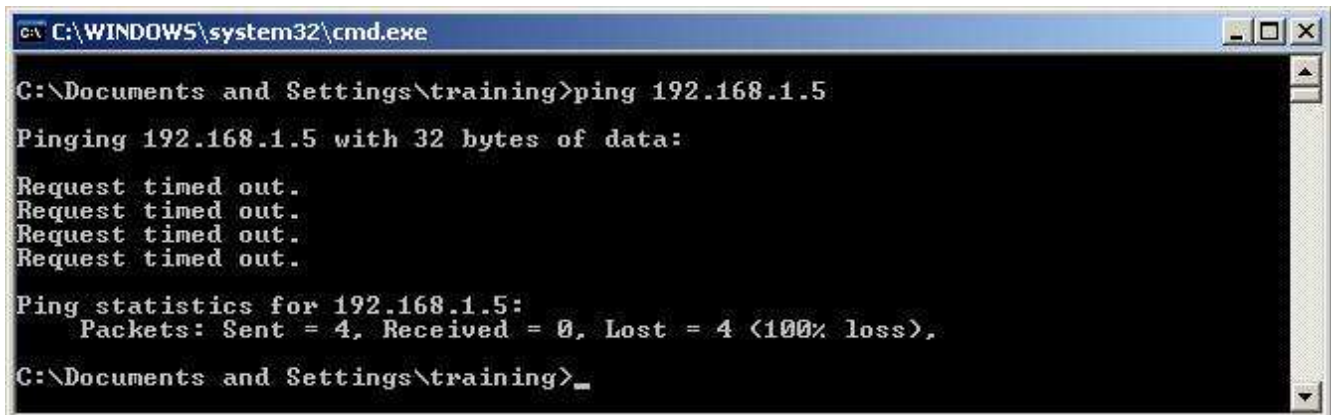


Figure 8.1

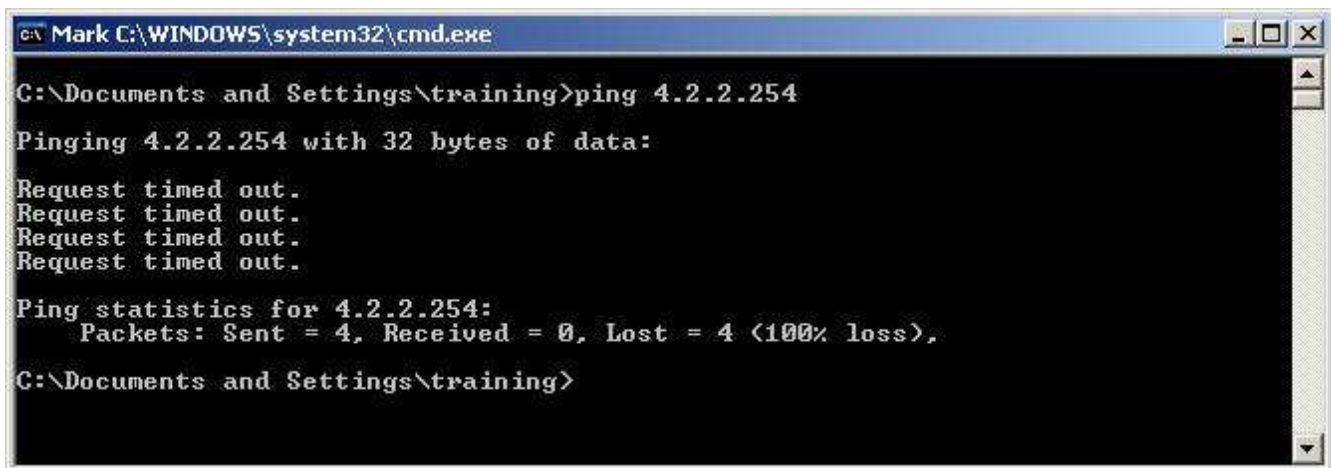


```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\training>ping 192.168.1.5
Pinging 192.168.1.5 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.5:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Documents and Settings\training>
```

Figure 8.2

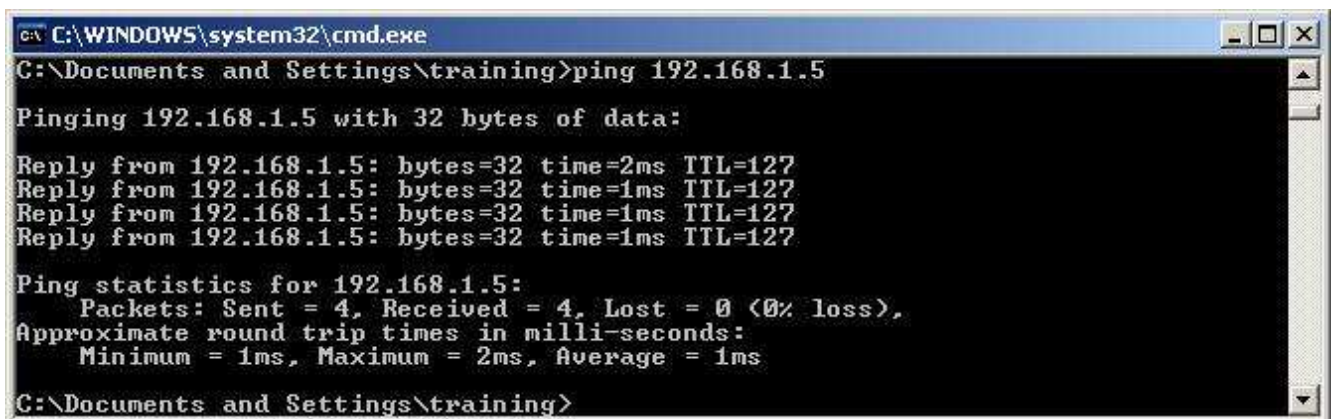


```
Mark C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\training>ping 4.2.2.254
Pinging 4.2.2.254 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 4.2.2.254:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Documents and Settings\training>
```

Figure 8.3

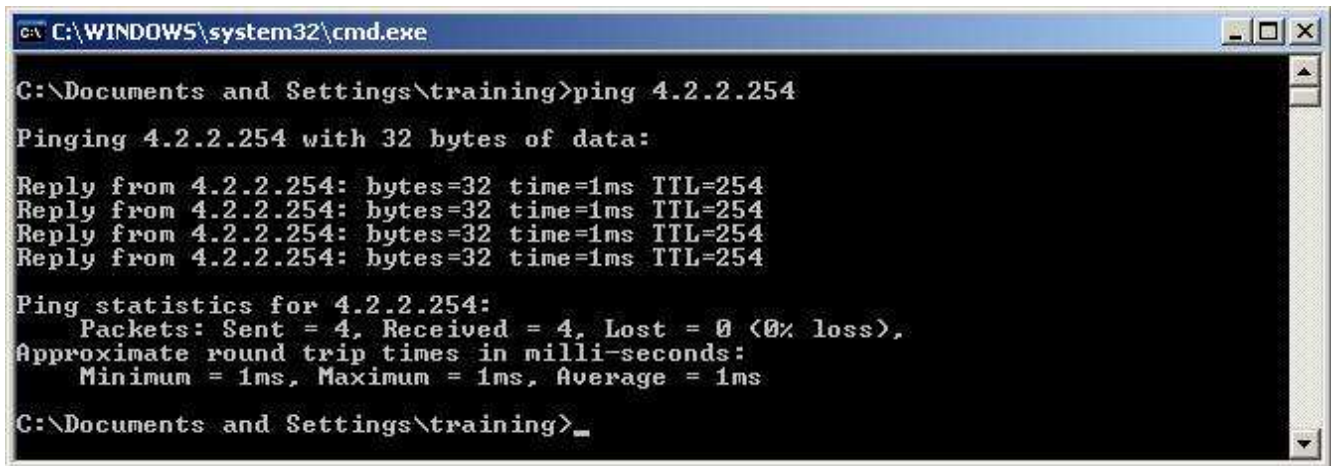


```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\training>ping 192.168.1.5
Pinging 192.168.1.5 with 32 bytes of data:
Reply from 192.168.1.5: bytes=32 time=2ms TTL=127
Reply from 192.168.1.5: bytes=32 time=1ms TTL=127
Reply from 192.168.1.5: bytes=32 time=1ms TTL=127
Reply from 192.168.1.5: bytes=32 time=1ms TTL=127

Ping statistics for 192.168.1.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 2ms, Average = 1ms

C:\Documents and Settings\training>
```

Figure 8.4



```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\training>ping 4.2.2.254
Pinging 4.2.2.254 with 32 bytes of data:
Reply from 4.2.2.254: bytes=32 time=1ms TTL=254
Reply from 4.2.2.254: bytes=32 time=1ms TTL=254
Reply from 4.2.2.254: bytes=32 time=1ms TTL=254
Reply from 4.2.2.254: bytes=32 time=1ms TTL=254
Ping statistics for 4.2.2.254:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms
C:\Documents and Settings\training>_
```