Routing Basics

1 Overview

This exercise explores basic network routing concepts in a Linux environment. These include use of the `route` command to modify Linux routing tables, defining a DNS server in the /etc/resolv.conf file, and an example of using Linux *iptables* to implement Network Address Translation (NAT).

Strictly speaking, this lab explores *packet forwarding*, and not true network routing, which typically involves the use of routing tables that name other network routers. See the *bird-bgp* and *bird-ospf* labs for examples of network routing.

This exercise, (and manual), is not intended to replace instruction or independent reading on the topic of network packet forwarding and routing in Linux systems. The exercise is intended to provide students with an environment with which they can experiment with the mechanics of forwarding network traffic to different network interfaces based on Linux routing tables. The student is only required to view a simple example of using *iptables* for NAT in this lab. See the `iptables2` and the `dmz` labs for a broader look at iptables.

2 Lab Environment

This lab runs in the Labtainer framework, available at http://nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer routing-basics
```

A link to this lab manual will be displayed.

3 Network Configuration

This lab includes a set of networked computers as shown in Figure 1. When the lab starts, you will see several virtual terminals, one connected to each component. Our focus will on the local computers within the gray rectangle.

The gateway is configured to perform packet forwarding between the 2 local LANs, and to forward external addresses to a simulated ISP, e.g., to reach the Internet or the remote gateway. The ws1 and ws2 workstations are pre-configured to forward traffic to the gateway component. The ws3 workstation is not yet configured to forward packets.

The gateway is configured to use NAT to translate sources addresses of traffic from internal IP addresses, e.g., 192.168.1.1, to our external address, i.e., 203.0.113.10.

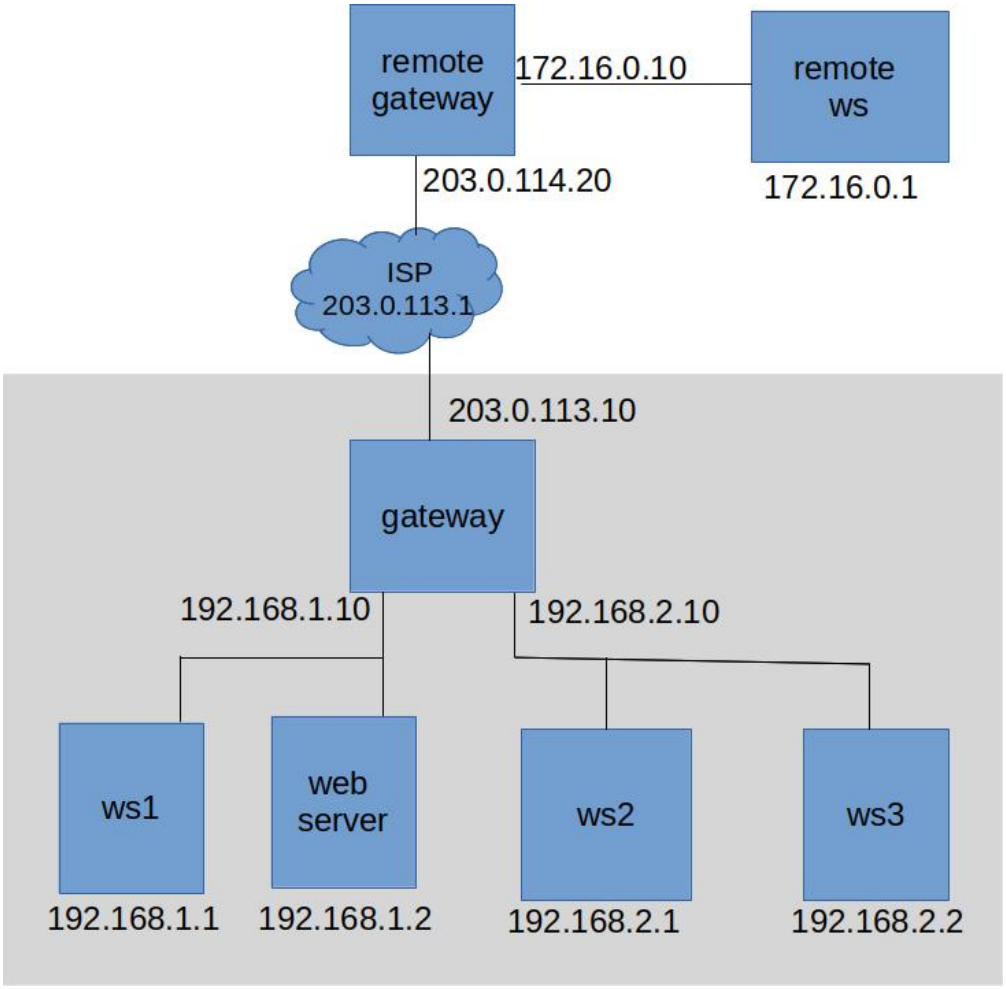

Figure 1: Network topology for routing-basics lab

This lab is designed to avoid use of name servers for the local computers, the intention is to focus on IP addresses and network interfaces.

4 Lab Tasks

4.1 Explore

Use `ifconfig` or `ip addr` on the different computers to familiarize yourself with the subnets associated with each of the computer's network interfaces. Note how ws2 and ws3 are on the same subnet, yet ws1 and the web server are on a different subnet. And note how the gatway has a network interface on the same subnet as ws1, and a different network interface on the same subnet as ws2 and ws3.

4.2 Internal Packet Forwarding

From each of the three workstations, enter the following command:

```
route -n
```

Note how ws1 and ws2 include routing table entries that name the gateway as the *default gateway*. For example, on ws1 the routing table can be read as: "If the destination address is on my eth0 subnet, (192.168.1.0/24), use the eth0 interface and ARP to locate the destination. Otherwise, send the packet to the gateway (192.168.1.10)." This allows ws1 and ws2 to address each other, which can be demonstrated by using `ping` from ws1 to reach ws2 via the gateway. First start `tcpdump` on the gateway to view traffic on the two network interfaces[1]:

```
sudo tcpdump -i eth0 -n -vv
```

Then, on ws1:

```
ping [ws2 IP] -c 3
```

Now consider ws2 and ws3. Since they are both on the same subnet, they can ping each other. Try that for yourself. And look at the tcpdump output on your gateway and note how the only new activity is an ARP request from whichever workstation you issued the ping from.

Next, try to ping ws1 from ws3. That fails because ws3 has no routing table entry defining what to do with traffic that is not destined for a subnetwork of one of the ws3 interfaces.

On ws3, define the gateway component as the *default gateway* using the `route` command, but this time using `sudo` because we are altering the routing:

```
sudo route add default gw [gateway IP]
```

Then try to ping ws1 from ws3. The ws3 routing table now knows what to do with packet addressed to ws1. You should now also be able to reach the web server from ws3, try that:

```
wget 192.168.1.2
```

This is a temporary routing table fix that will revert if the system is restarted, but it can be made permenant with techniques not covered here.

4.3 Routing to the Internet

The gateway component is configured to forward packets to a simulated ISP at 203.0.113.1, which is a hidden component that provides routing to the Internet for this lab. From ws2, try to `wget www.google.com`. Then do the same from ws3. The problem with ws3 is that it has no domain name service (DNS) definition. Note, routing from ws3 to the Internet works fine, which you can confirm by pinging the IP address of www.google.com (as displayed when you pinged from ws2). The ws3 component simply lacks a DNS definition. On ws2, the DNS is defined to be the gateway component, and this is achieved in the `/etc/resolv.conf` file [2]. If you modify that file on ws3 to match that of ws2, that will tell ws3 to use the gateway as its DNS. (While the gateway is not a DNS, it runs a *dnsmasq* service to forward DNS requests to the DNS that it uses.) Modify the `resolv.conf` file and confirm you can resolve the www.google.com address using `wget www.google.com`. Use `sudo vi` or `sudo nano` to modify the file. See the `nix-commands` lab if you are not familiar with text editors.

[1] use the -n option to prevent tcpdump from using reverse DNS to put names to IP address and ports.

[2] Many Linux systems include functions for defining your DNS, and these tools will overwrite the resolv.conf file. So modifying the `resolv.conf` file may be only a temporary fix, which is sufficient for this lab

4.4 Use of Network Address Translation (NAT)

Now, review how the gateway component implements NAT using the `iptables` utility. Consider traffic from ws1 destined for www.google.com. The source IP address on those packets is 192.168.1.1. The ws1 component sends the packets to its default gateway, i.e., our gateway component. The gateway routing table is configured to send external traffic to the simulated ISP at 203.0.113.1. However, before that traffic is sent, we need to translate the source IP address to our exernal 203.0.113.10 address so that google or our ISP knows where to send replies. Use this command on the gatway (use ctrl-c to break out of tcpdump if it is running):

```
sudo iptables -L -v -t nat
```

to view our single NAT rule, having a target of `MASQUERADE`, which will translate source addresses for all traffic destined for our external network interface. (The other 2 NAT rules will be discussed in the following section.)

These iptables rules are defined in the /etc/rc.local file on the gateway component.

Use `wget google.com` from ws1 and observe the source IP address in the tcpdump on the gateway's interface to the subnet containing ws1. Then use ctrl-c to stop tcpdump and restart it on the gateway's network interface on the ISP's subnet:

```
sudo tcpdump -i eth2 -vv -n
```

And then wget again. Note how the source address of the packets has changed.

This illustrates the use of NAT to share a single IP address amongst many computers. Here, the enterprise has but a single externally addressable IP address, yet all of the workstations are able to access services on the Internet.

4.5 Services behind NAT

Another property of NAT is that it "hides" local computers from the internet, thereby providing some ad-hoc security. Go to the remote workstation (`remote_ws`) and try to ping ws1. External to our gateway, there is no way to name the ws1 computer other than via the mapping within the gateway that allows it to forward traffic destined ws1 for sessions *initiated* by ws1. As a result of NAT, nothing on the internet can initiate a connection with the internal workstations.

There are situations in which you want external systems to be able to initiate connections to services that would otherwise be hidden by NAT. NAT rules can be configured to map port numbers of incoming traffic to internal components. For example, the gateway is configured to route web requests on port 80 to the web server. Revisit the output of the

```
iptables -v -n -t nat
```

command. You will notice a *Destination NAT* (DNAT) rule in the PREROUTING chain and a *Source NAT* (SNAT) rule in the POSTROUTING chain. The DNAT rule directs the iptables to replace the destination address with the address of the web server before deciding how to route the packet. The SNAT rule directs iptables to replace the source address with the gateway's address prior to sending the packet to the web server (i.e., after deciding how to route the packet).

Restart your tcpdump on eth0 on the gateway. Then query the web server from the remote workstation:

```
wget 203.0.113.10
```

Observe how the destination address is the web server, and the source address is the gateway.

Note the security implications of these SNAT and DNAT rules. The web server is now accessible to the entire Internet, and this increases the risks to the other computers since they can be addressed by the web server. If the web server becomes compromised, it can be used to attack internal computers. See the `dmz-lab` for an exercise illustrating potential solutions to that problem.

5 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab
```

When you stop the lab, the system will display a path to the zipped lab results on your Linux system. Provide that file to your instructor, e.g., via the Sakai site.

This lab was developed for the Labtainer framework by the Naval Postgraduate School, Center for Cybersecurity and Cyber Operations under National Science Foundation Award No. 1438893. This work is in the public domain, and cannot be copyrighted.