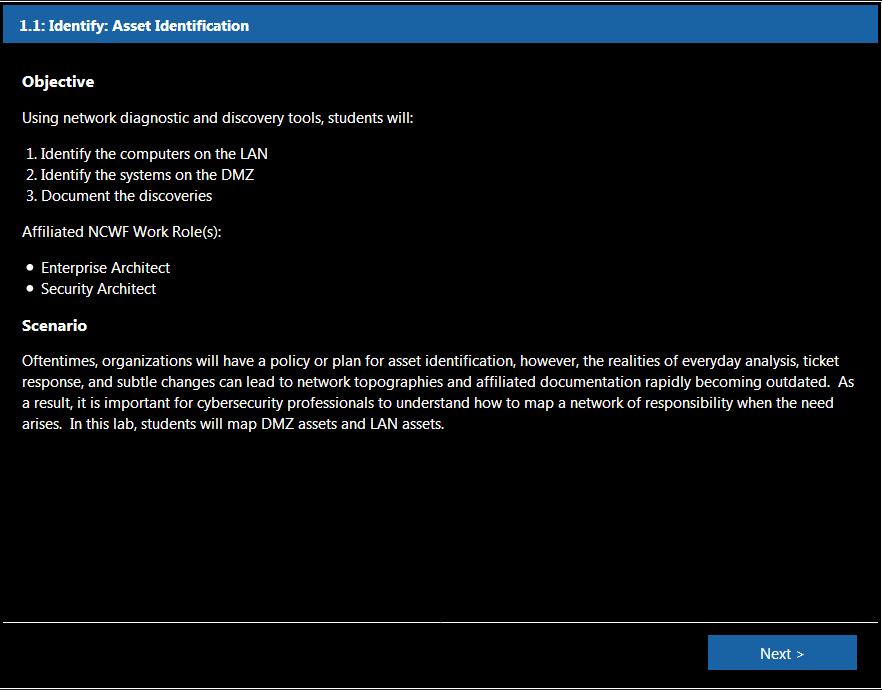
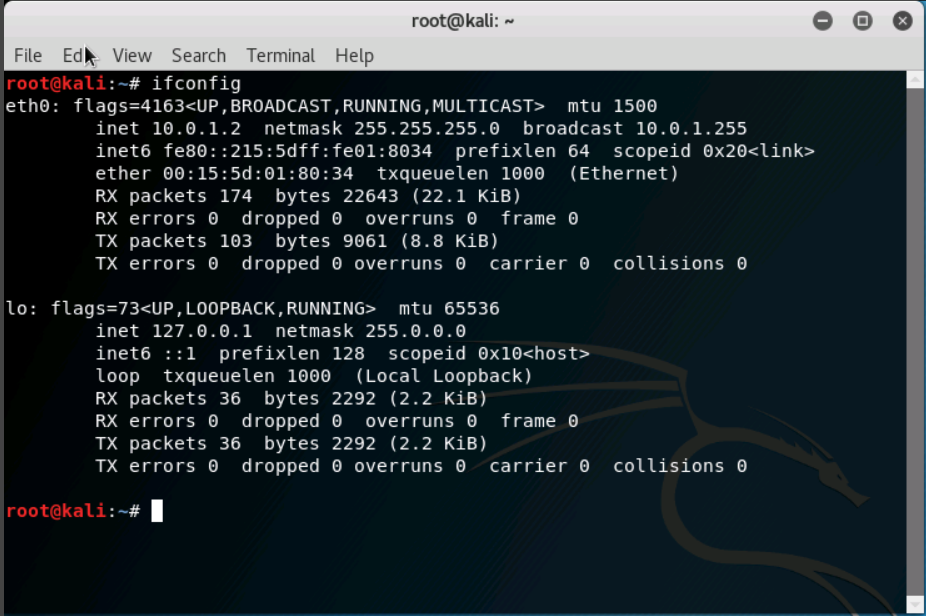


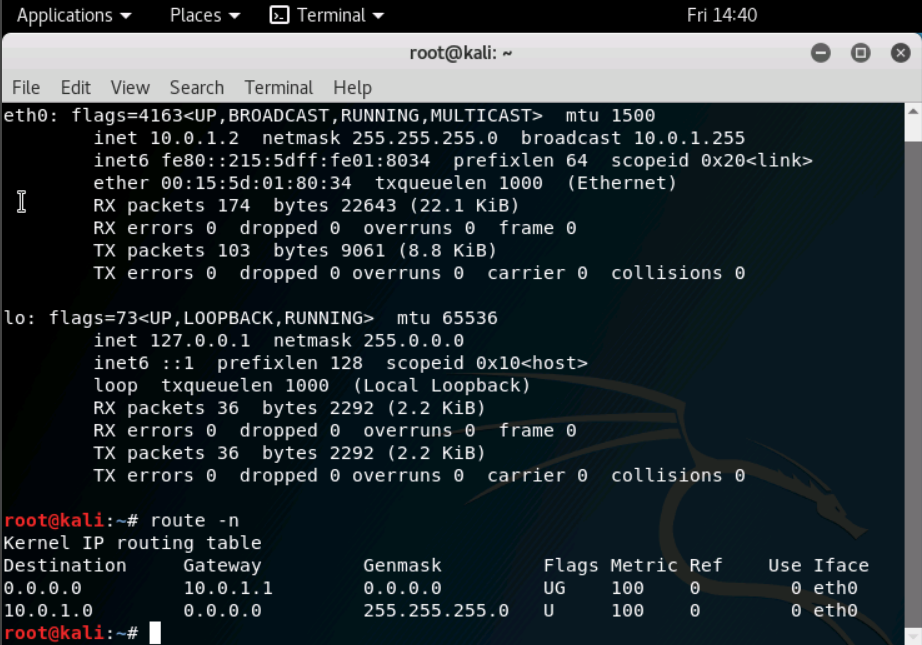
**ASSET IDENTIFICATION**



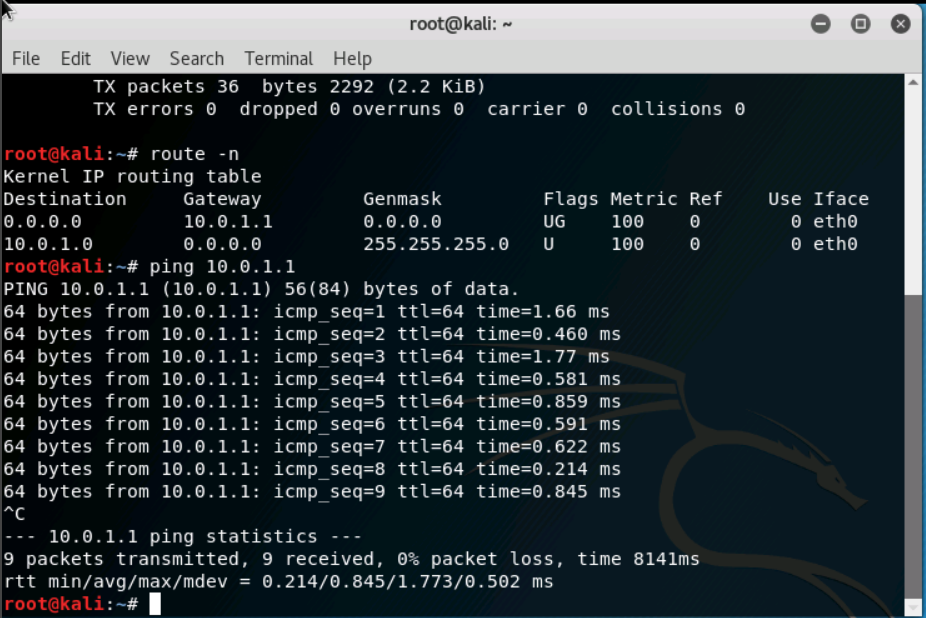
Before we begin our asset identification, type **ifconfig** to take a look at our network interface card (NIC).  
It looks like our **eth0 interface's IP address** is 10.0.1.2 with a **netmask** of 255.255.255.0



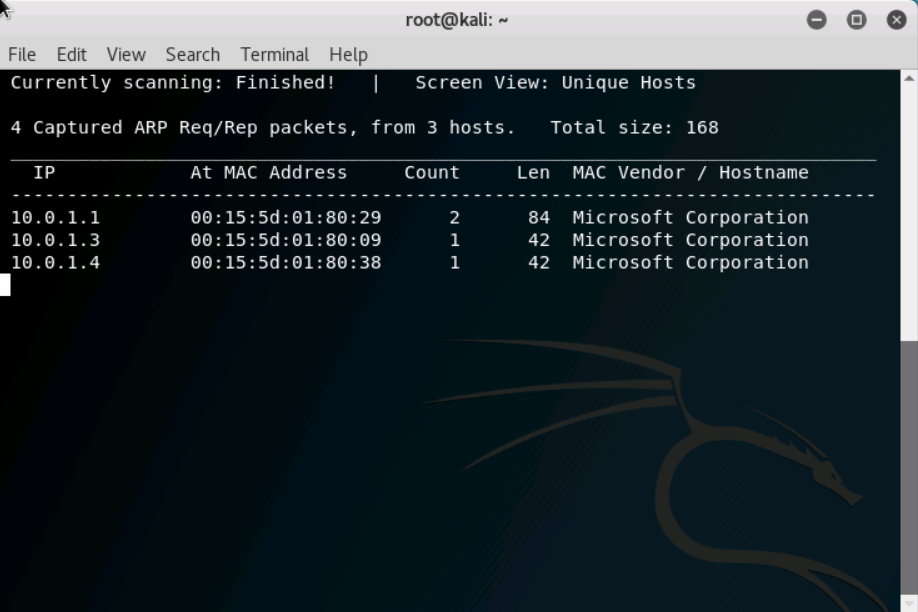
Now, let's figure out our gateway.  
Type **route -n** to see routing information. It looks like our gateway is 10.0.1.1 with a netmask of 255.255.255.0.



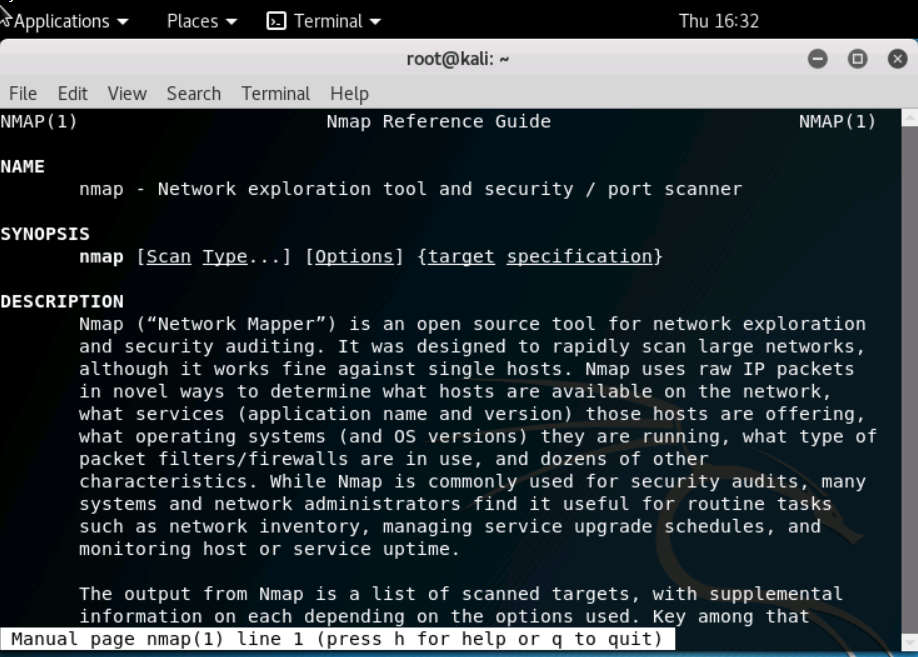
Before we identify our assets on the LAN, let's send ICMP echo requests to the internal router to see if it is up.  
Type **ping 10.0.1.1.** We can see that the router is communicating back to this Kali machine.  Type **CTRL C** to stop the requests.



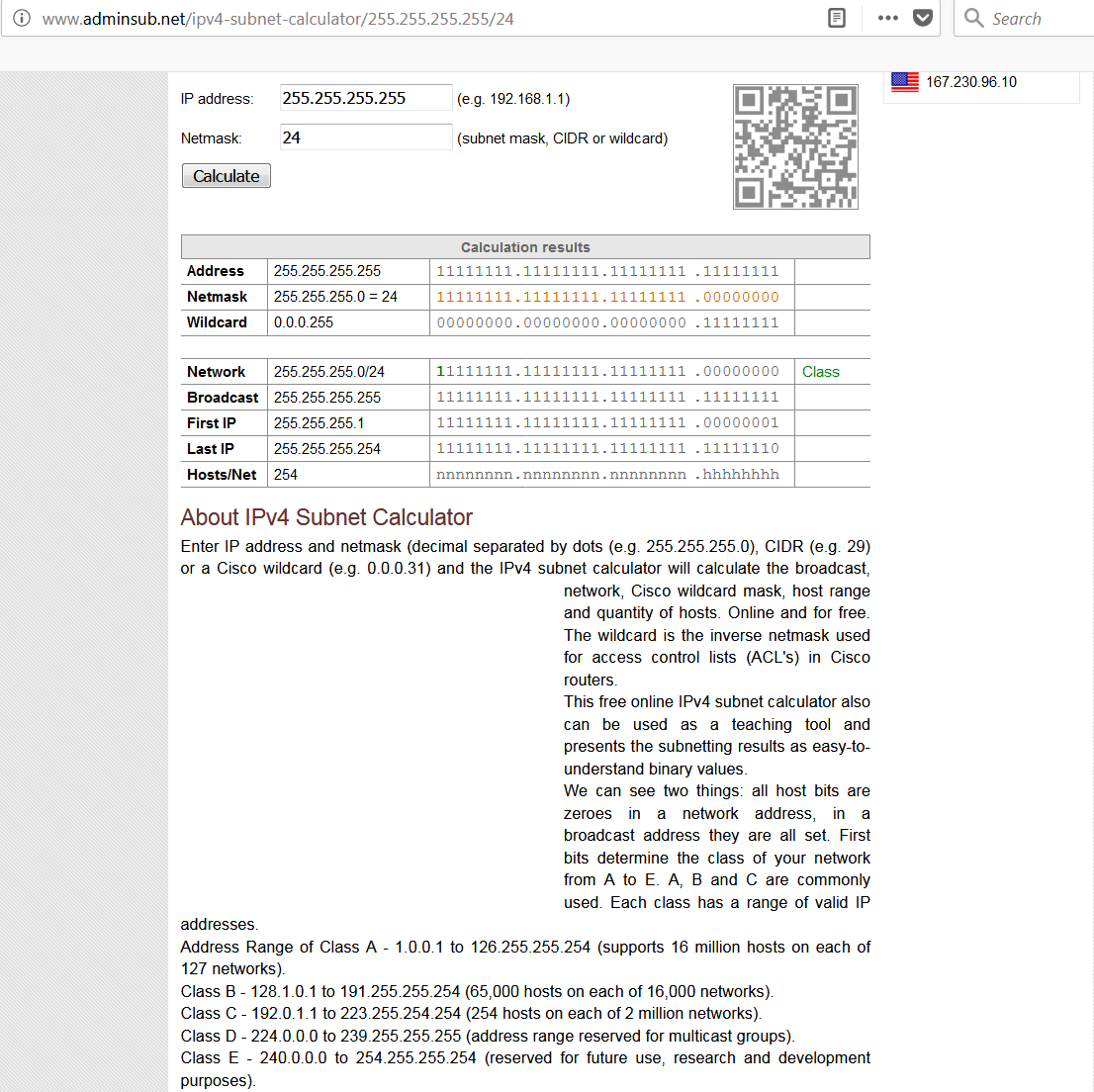
Type **netdiscover -r 10.0.1.1/24** to run the tool.  
Netdiscover has located not only the router, but also two other machines on the LAN as well.  Type **CTRL C** to go back to the original prompt.



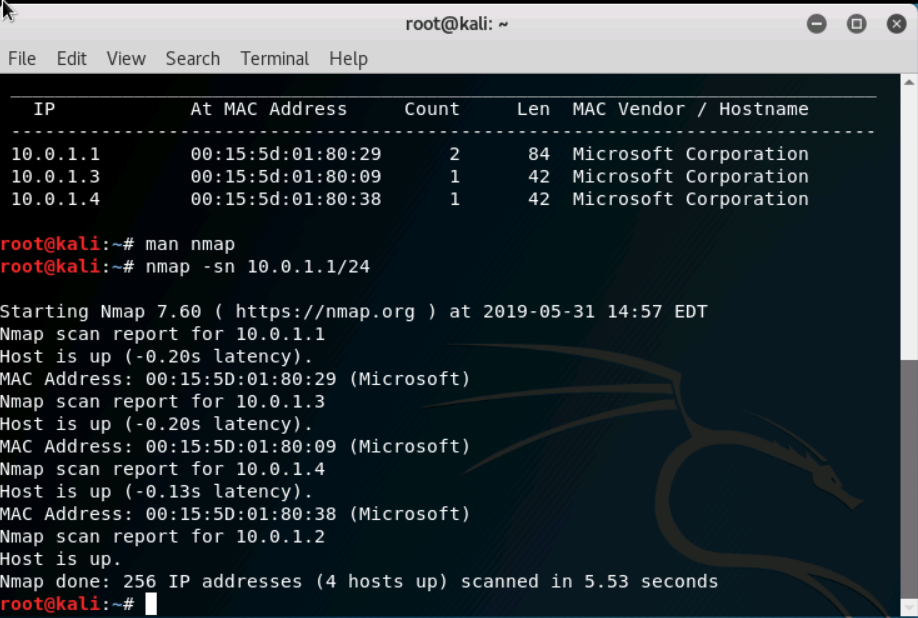
In the next few steps, we will be using nmap, which is short for network mapper.  Nmap is not only very powerful, but it is also very popular within the cybersecurity community.  
Type **man nmap** to learn more about the Command Line Interface (CLI) tool.  Type **q** when you are finished learning about nmap and its associated options.

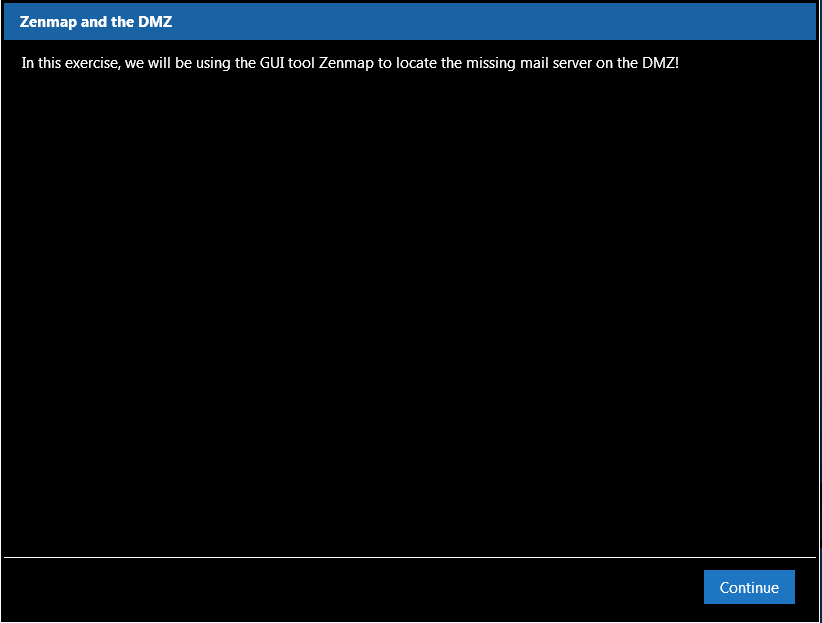


Let's conduct a ping scan, also known as a ping sweep, on the LAN.  Type **nmap -sn 10.0.1.1/24**.  The /24 is referring to the netmask of 255.255.255.0 that the subnet has.

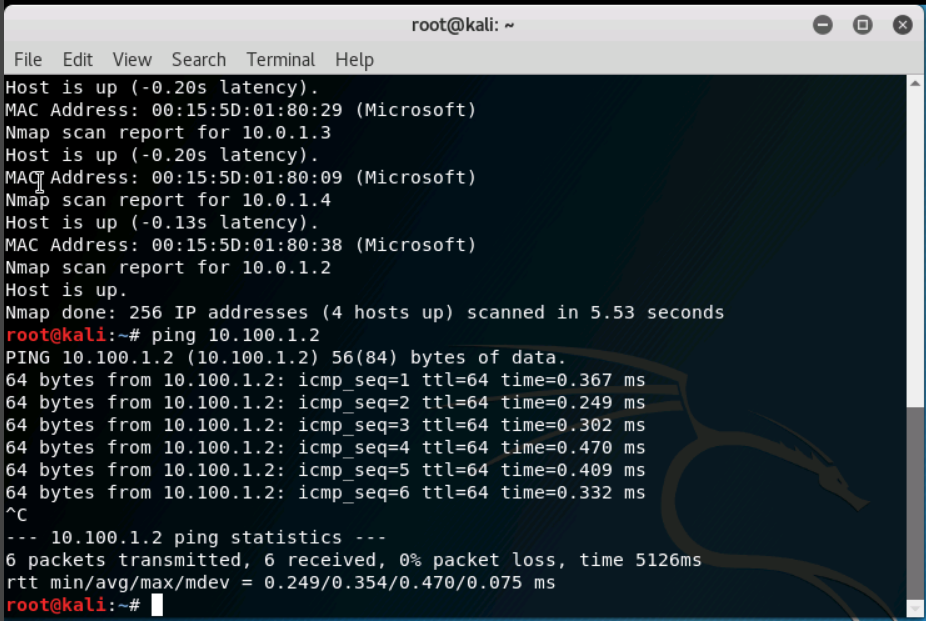


nmaLet's conduct a ping scan, also known as a ping sweep, on the LAN.  Type **nmap -sn 10.0.1.1/24**.  The /24 is referring to the netmask of 255.255.255.0 that the subnet has. This nmap scan confirms our findings from the netdiscover scan to be true.  Both scans show that the router, as well as two other IP addresses, are up and running.  The third host's address is the eth0 IP of our Kali machine.

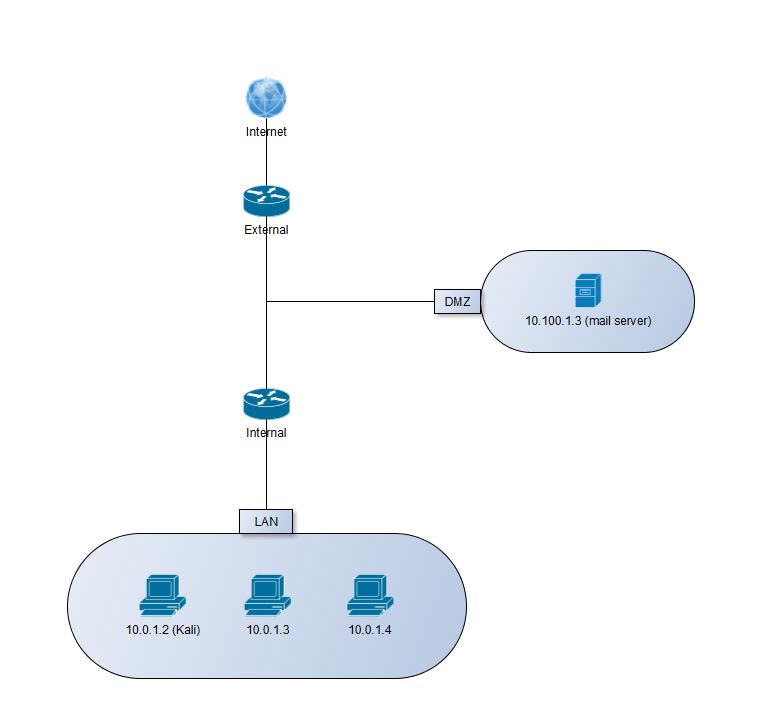




Let's move on from the LAN to the DMZ. Type **ping 10.100.1.2** to send ICMP echo requests to the same router, but to the DMZ-facing interface. We can see that the router is communicating back to this Kali machine. Type **CTRL C** to stop the requests.



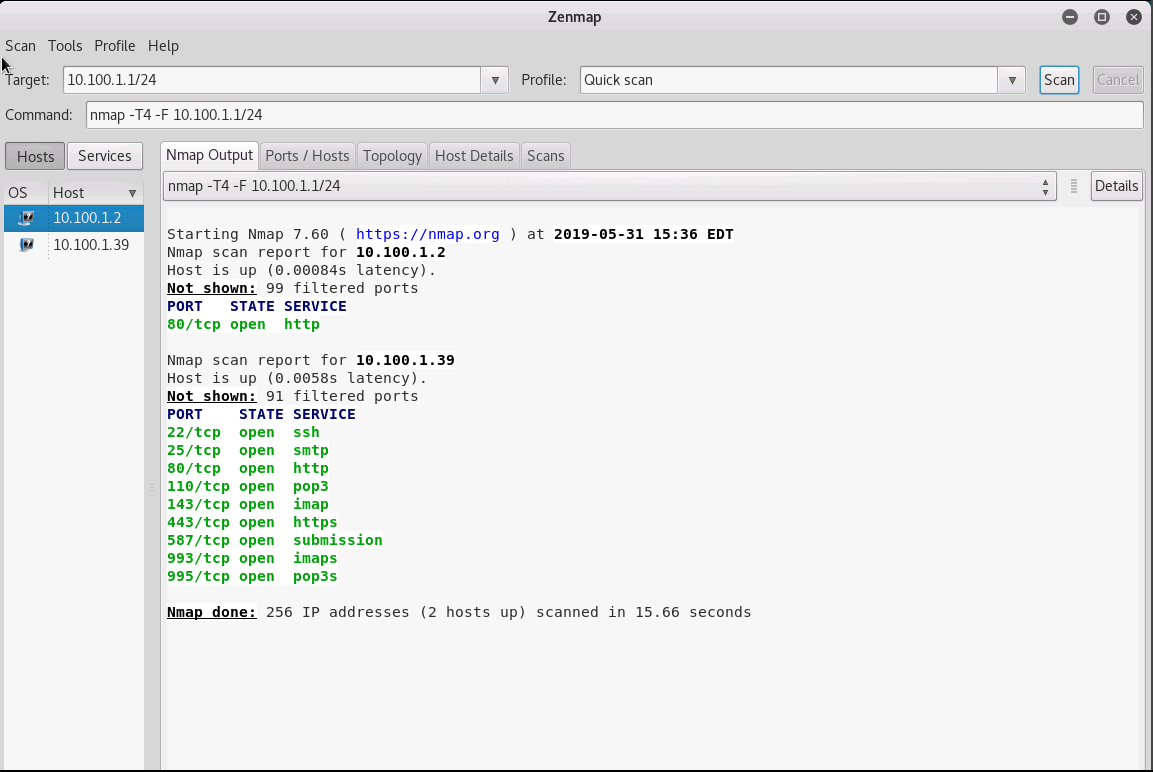
Before we identify our assets on the DMZ, let's see what the network is supposed to look like. Underneath the "Machines" tab on the right side, click on "Network Diagram".  Resize the window to see the whole picture if need be. Click "OK" to go back to your terminal session.

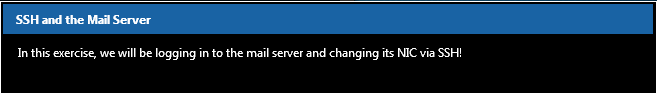


This is what our network should look like.  Note:

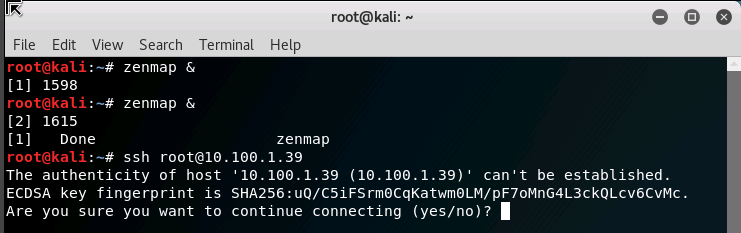
* Our LAN's nmap scan is accurate because it matches the LAN in this diagram
* Additionally, a mail server is in the DMZ with an IP address of 10.100.1.3

In the next couple of steps, we will be using the Graphical User Interface (GUI) version of nmap called Zenmap to locate the mail server! In the terminal, type **zenmap &** to run the process. In the target box, type **10.100.1.1/24** to use the DMZ IP range. In the profile drop-down box, select "Quick Scan". Click "Scan". Eureka!  The scan output shows two IP addresses! **10.100.1.2** is the internal router's DMZ-pointing interface.  However, take a look at **10.100.1.39**.  It has open ports hosting various services! Services like **SMTP, POP3, IMAP, SUBMISSION, IMAPS**, and **POP3S** are protocols specifically designed for sending and receiving emails to and from the Internet. For this scenario, it is safe to assume that we have found the misplaced mail server. Save the scan by selecting "Scan" > "Save Scan". Save the Zenmap scan as dmz.xml on the Desktop. Close your Zenmap window.

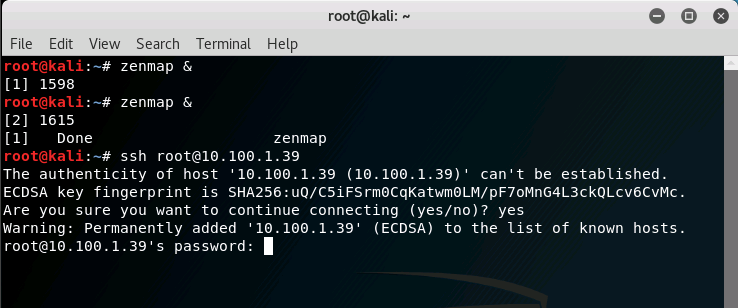




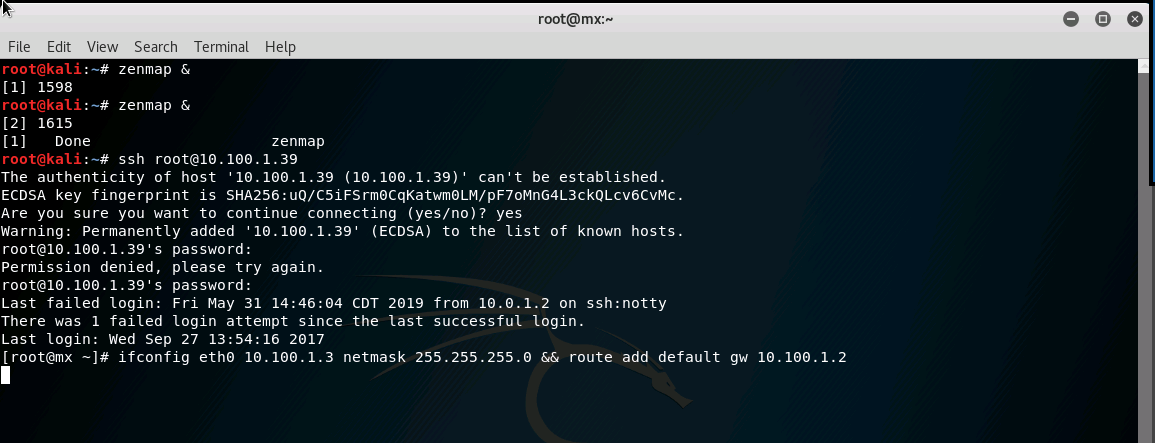
Once we have the XML file saved, let's use SSH to remotely log into the mail server and change its IP address. In your terminal, type **ssh** [root@10.100.1.39](mailto:root@10.100.1.39)**.**

****

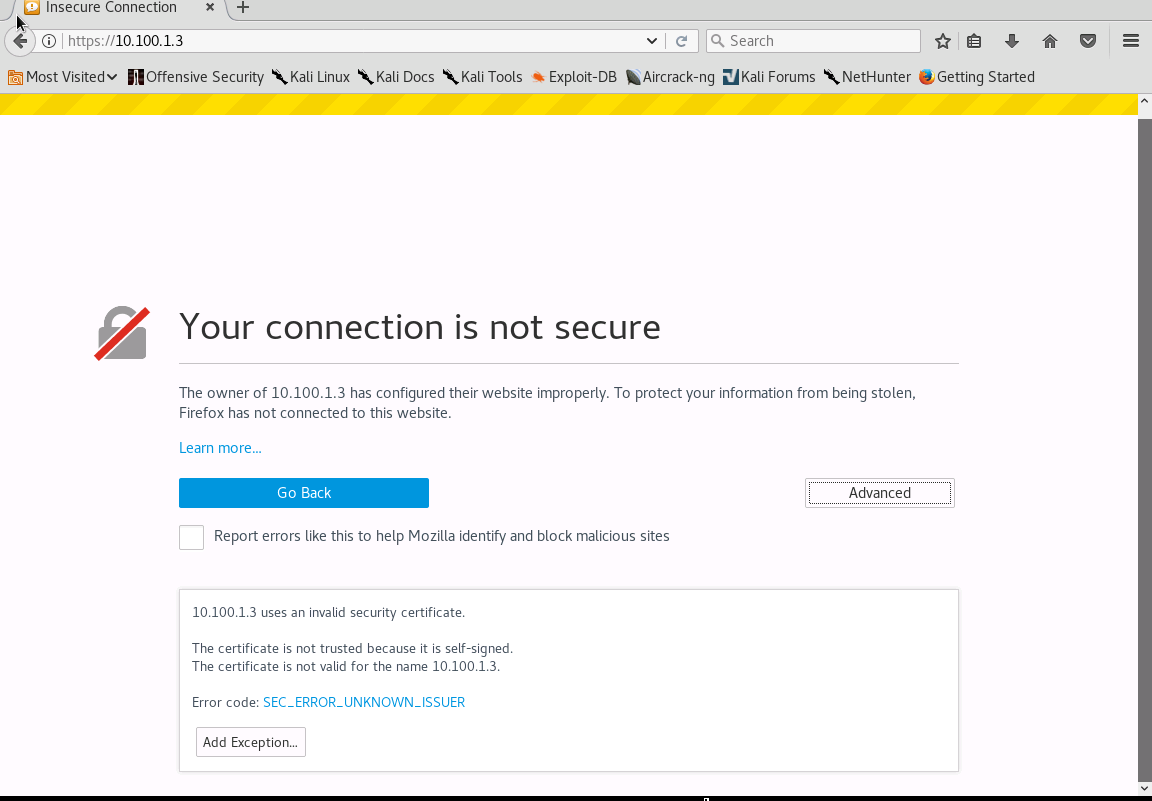
Type **yes** to establish the authenticity.  From our admin records, we know that the mail server password is **packethunters**.

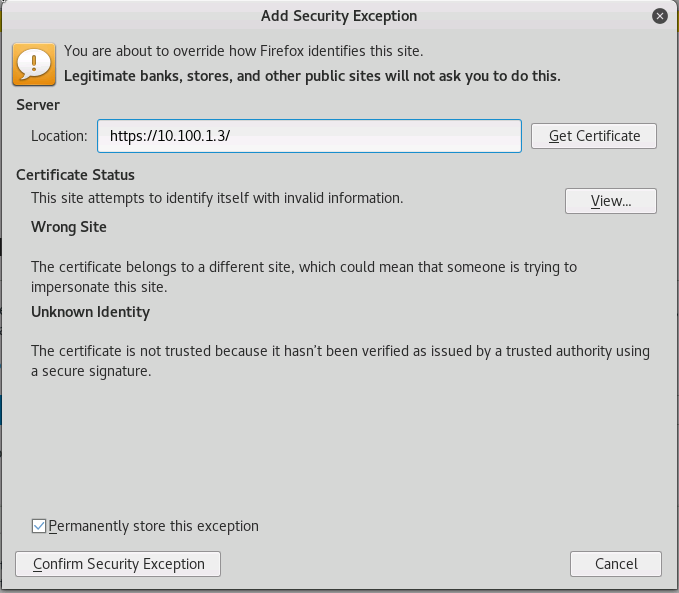


In order to correctly change the IP address, we need to execute two different commands.  We will use "&&" to combine the ifconfig and route commands. Type **ifconfig eth0 10.100.1.3 netmask 255.255.255.0 && route add default gw 10.100.1.2.** After you hit **ENTER**, close out of the terminal window.

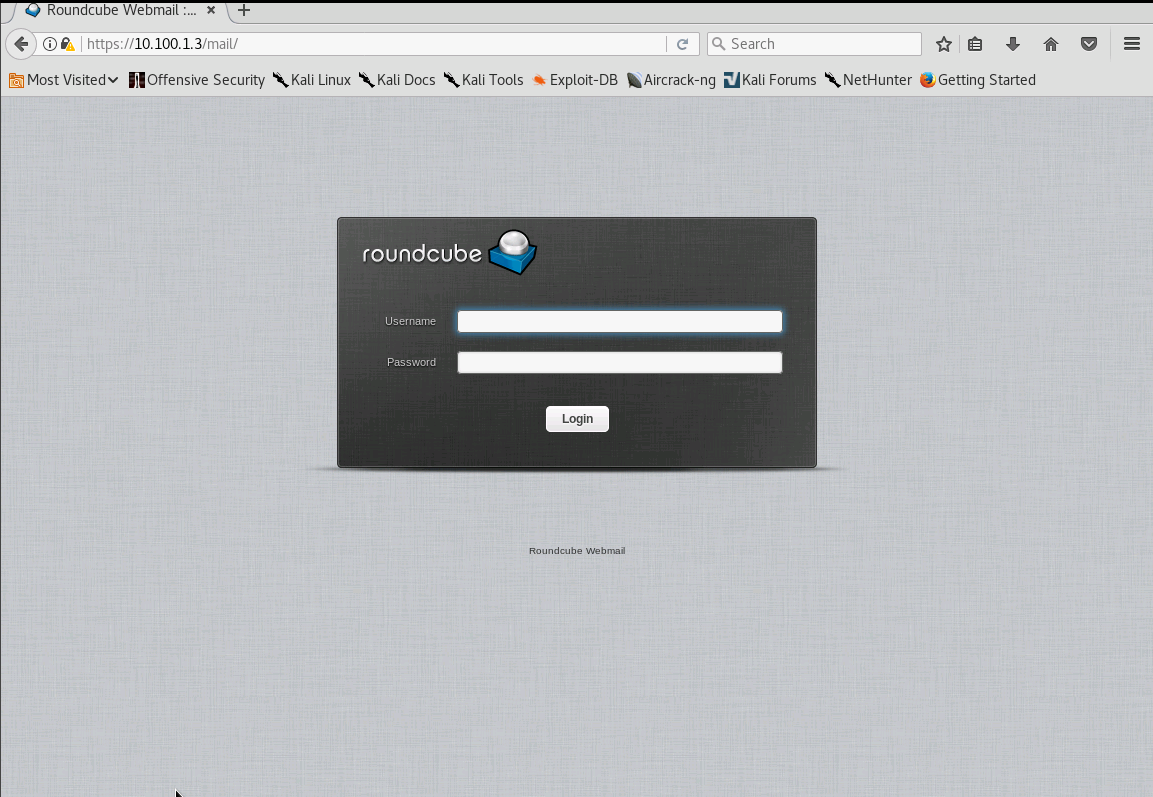


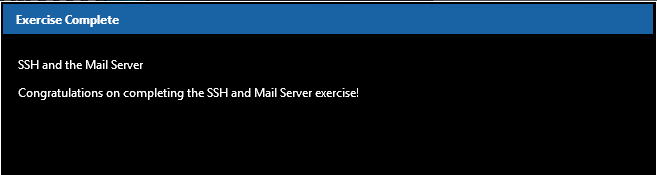
Let's make sure our mail exchange has the correct IP address that we just assigned to it. Open up a Firefox web browser and navigate to <https://10.100.1.3>**.** Click "Advanced", "Add Exception...", and "Confirm Security Exception" to pass the certificate security warning.



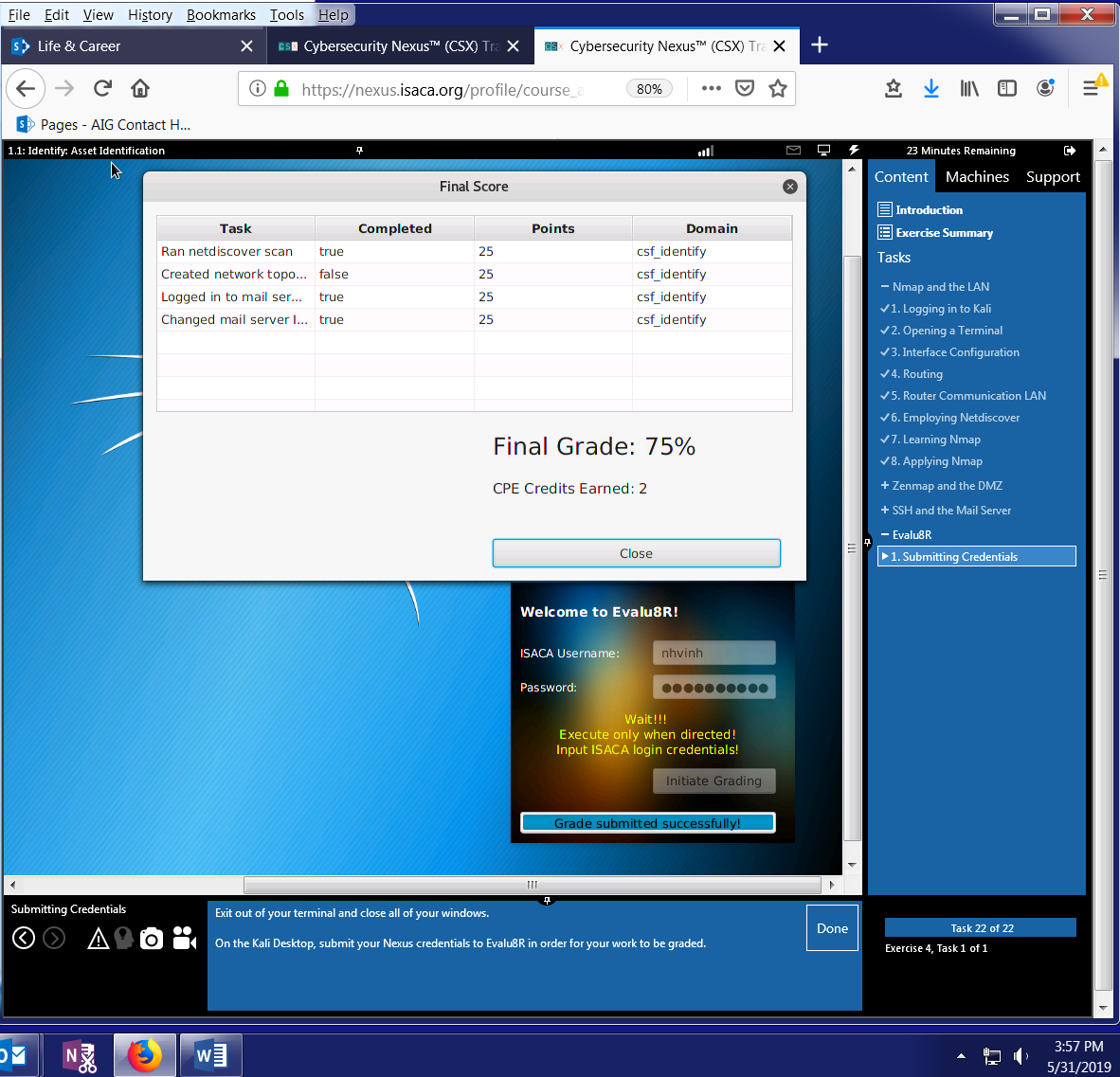


When you get to the webmail login page, close out of Firefox.  We have successfully placed the web server in its correct location within the DMZ!

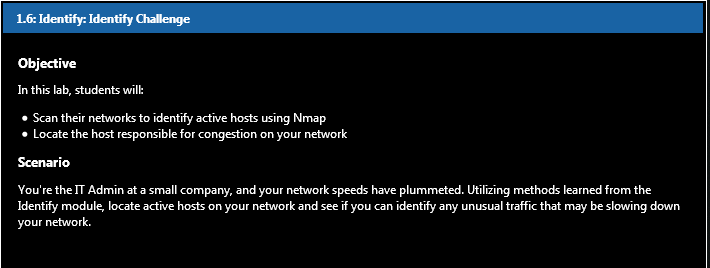




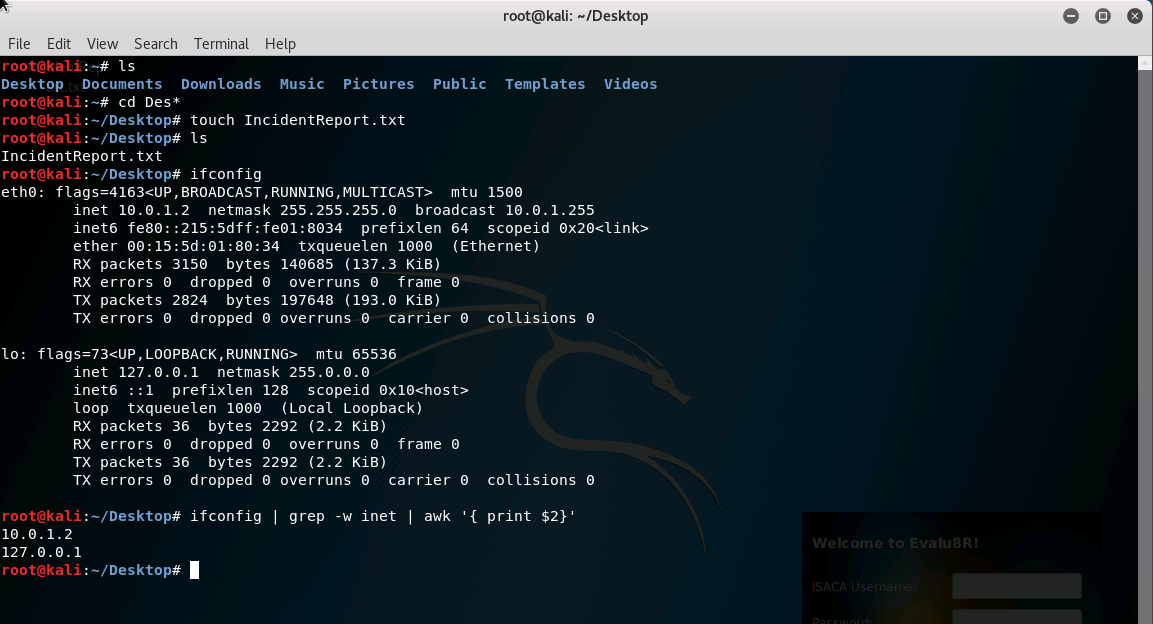
Exit out of your terminal and close all of your windows. On the Kali Desktop, submit your Nexus credentials to Evalu8R in order for your work to be graded.



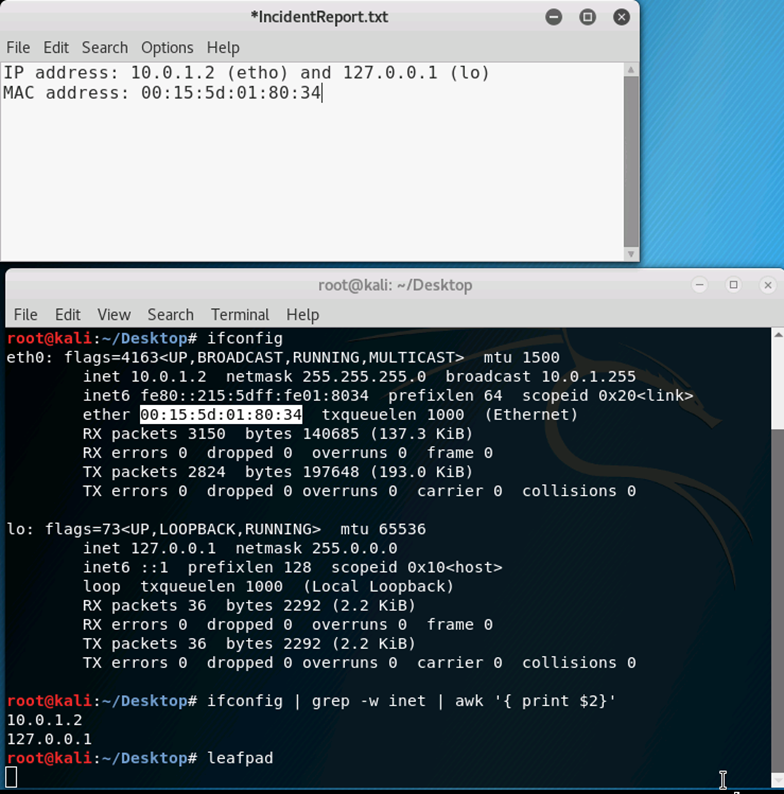
**IDENTITY CHALLENGE**



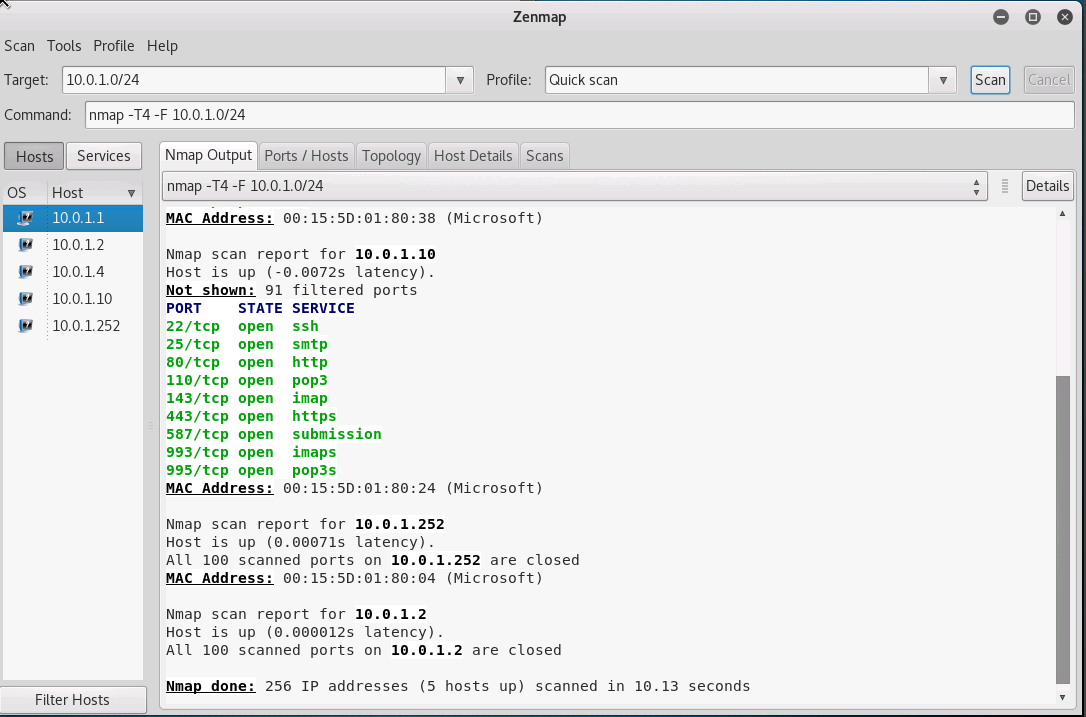
Create a file named IncidentReport.txt on your Desktop. We'll be using this file to document current network information. In the file, record your IP address, MAC address, and any other information that may benefit your investigation.

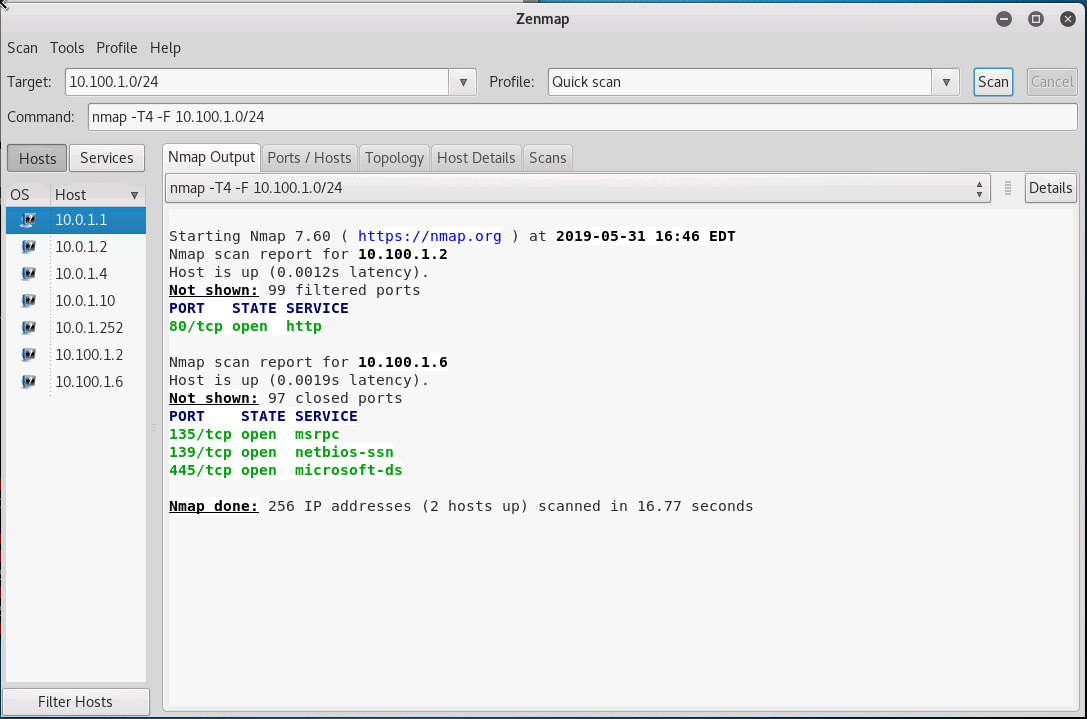


**eth0** (physical interface) and **lo** (special virtual network interface – loopback device for diagnostics and troubleshooting and for connecting to services on local host). A media access control address (MAC address) of a device is a unique identifier assigned to a network interface controller (NIC) for communications at the data link layer of a network segment. MAC addresses are used as a network address for most IEEE 802 network technologies, including Ethernet and Wi-Fi.

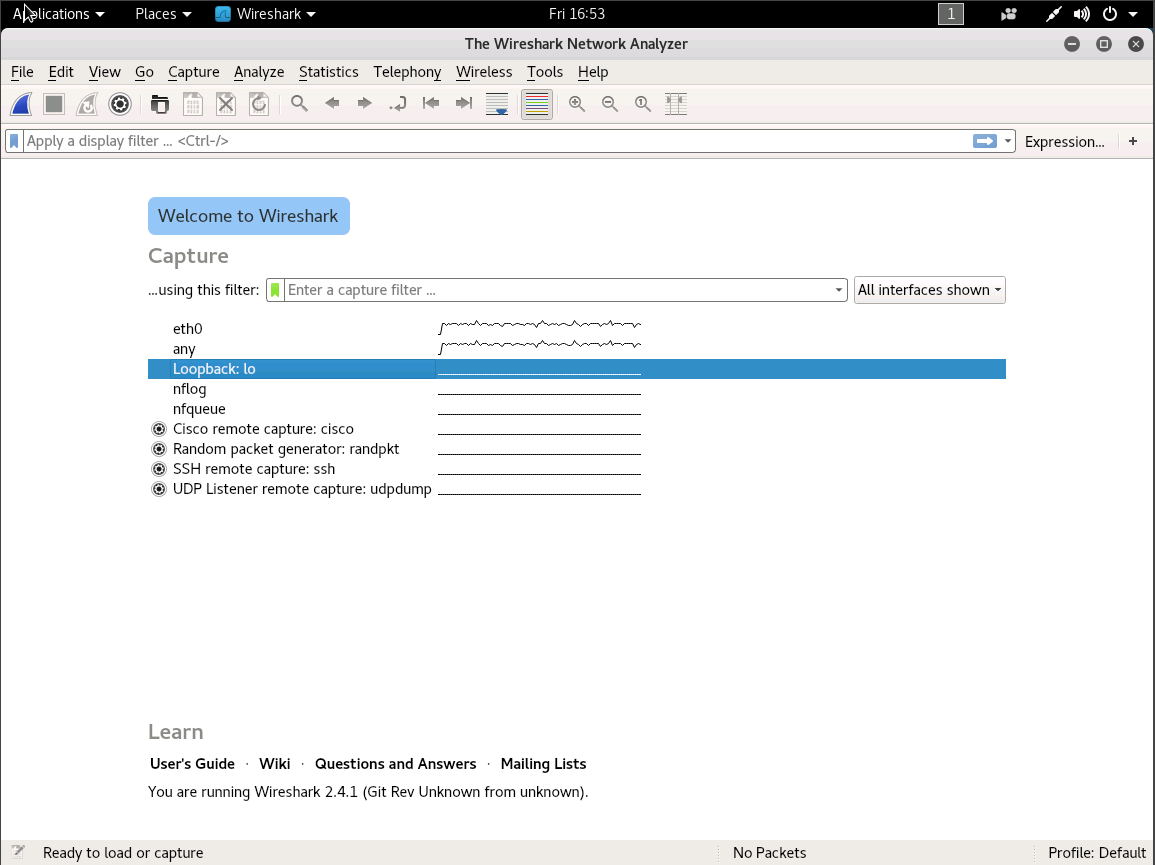


**Scan** the LAN (10.0.1.0/24) and DMZ (10.100.1.0/24) network segments with nmap. In your IncidentReport.txt file, **record** which IP's were discovered on each segment.





Now that you've identified the active hosts on your network, open **Wireshark** and begin capturing on the **eth0** interface. Instantly, you'll notice a large portion of the traffic is from external IP addresses. You check your external firewall logs and find nothing correlating to this traffic. Strange...



Assuming the firewall is working properly, use your Identification skills to locate the true threat inside the network. **Create a new file** on your **Desktop**, **Threat.txt**, and record the **MAC address** and **local IP address** of the device responsible for slowing down the network connection. **HINT:** The command "sudo arp -n" may help in mapping addresses to a device.

**How to check Local and External IP address on Kali Linux**

Lubos Rendek

Kali Linux

08 January 2017

Contents

* + [1. Objective](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-objective)
    - [1.1. External IP Address](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-1-external-ip-address)
      * [1.1.1. Using WEB Browser](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-1-1-using-web-browser)
      * [1.1.2. Using Command Line](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-1-2-using-command-line)
    - [1.2. Internal IP Address](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-2-internal-ip-address)
      * [1.2.1. Checking GUI Network Settings](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-2-1-checking-gui-network-settings)
      * [1.2.2. Using ip or ifconfig commands](https://linuxconfig.org/how-to-check-my-local-and-external-ip-address-on-kali-linux#h1-2-2-using-ip-or-ifconfig-commands)

**Objective**

The following article will illustrate some of the common ways on how to determine a local and public IP address on Kali Linux.

**External IP Address**

**Using WEB Browser**

Perhaps the simplest way on how to determine your Local and Public IP address is by using your web browser. Follow this link to [view your Local and Public IP address](https://linuxconfig.org/check-your-local-and-public-ip-address) instantly on your web browser. This page was specifically created for the purpose to promptly help our readers with this kind information.

**Using Command Line**

wget and curl commands can be also used to discover your external IP address. Open up a terminal by right-clicking on your desktop and selecting the Open Terminal menu. Execute the following [linux command](https://linuxconfig.org/linux-commands)s to retrieve your external WAN IP address:

# echo $(wget -qO - https://api.ipify.org)

OR

# echo $(curl -s https://api.ipify.org)

**Internal IP Address**

**Checking GUI Network Settings**

The first method on how to find your local IP address on Kali Linux is by reviewing your network settings using a Graphical User Interface. The process on how you navigate to network settings is largely dependent on your desktop's GUI configuration.  
  
In case you are using a default Kali Linux desktop configuration, simply left-click on the top right network icon or power button. From there, click on tools button which will open up a settings window. On the All Settings window find and double click on the "network" icon.   
  
This will display your internal IP address allocated to your network card along side with DNS and gateway configuration.



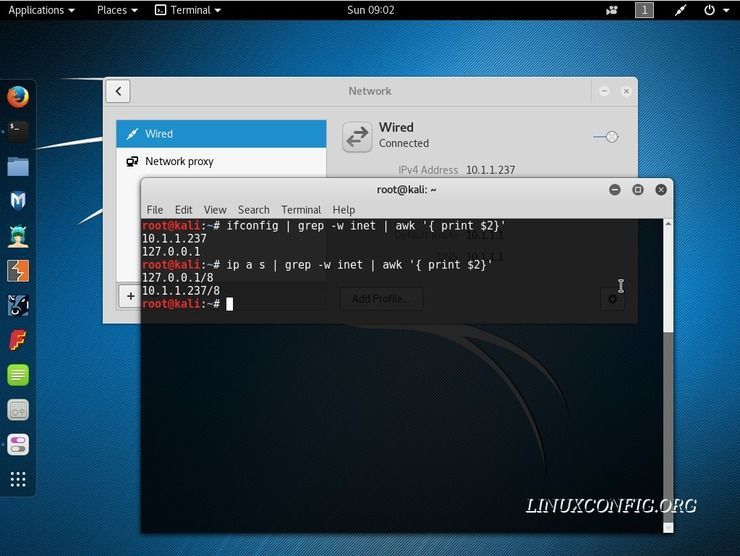
**Using ip or ifconfig commands**

Probably to most common and perhaps even recommended way on how to list your internal IP address is by use of ip and ifconfig commands. Right-click on your desktop and select an "Open Terminal" menu to initiate a new terminal session. The following two commands will list all your internal IP addresses:

# ifconfig | grep -w inet | awk '{ print $2}'

OR

# ip a s | grep -w inet | awk '{ print $2}'



**Demystifying ifconfig and network interfaces in Linux**

This post explains *ifconfig* output of common developer’s box, paying special attention to parameters poorly explained in official documentation. It also slightly touches Linux network interfaces.

**ifconfig**

*ifconfig* is a command line tool for UNIX-like systems that allows for diagnosing and configuring network interfaces. At boot time, it sets up network interfaces such as Loopback and Ethernet. Most of the time, however, ifconfig is used for network diagnostics.

Before diving into details of its output, let’s first make clear what is an interface.

**network interface**

A *network interface* is a software interface to networking hardware. Linux kernel distinguishes between two types of network interfaces: *physical* and *virtual*.

Physical network interface represents an actual network hardware device such as network interface controller (NIC). In practice, you’ll often find eth0 interface, which represents Ethernet network card.

Virtual network interface doesn’t represent any hardware device and is usually linked to one. There are different kinds of virtual interfaces: Loopback, bridges, VLANs, tunnel interfaces and so on. With proliferation of *software defined networks*, virtual interfaces become wildly used.

**Demystifying ifconfig output**

Let’s have a look at ifconfig output for a developer’s box with installed Ubuntu and Docker.

$ ifconfig

docker0 Link encap:Ethernet HWaddr 02:42:2d:66:fc:f1

inet addr:172.17.0.1 Bcast:0.0.0.0 Mask:255.255.0.0

inet6 addr: fe80::42:2dff:fe66:fcf1/64 Scope:Link

UP BROADCAST MULTICAST MTU:1500 Metric:1

RX packets:2 errors:0 dropped:0 overruns:0 frame:0

TX packets:3 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:152 (152.0 B) TX bytes:258 (258.0 B)

eth0 Link encap:Ethernet HWaddr 08:00:27:31:65:b5

inet addr:10.0.2.15 Bcast:10.0.2.255 Mask:255.255.255.0

inet6 addr: fe80::3db9:eaaa:e0ae:6e09/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:1089467 errors:0 dropped:0 overruns:0 frame:0

TX packets:508121 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:903808796 (903.8 MB) TX bytes:31099448 (31.0 MB)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:65536 Metric:1

RX packets:9643 errors:0 dropped:0 overruns:0 frame:0

TX packets:9643 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:719527 (719.5 KB) TX bytes:719527 (719.5 KB)

**interfaces**

There are 3 network interfaces on the box:

eth0 is a physical interface representing Ethernet network card. It’s used for communication with other computers on the network and on the Internet.

lo is a special virtual network interface called *loopback device*. Loopback is used mainly for diagnostics and troubleshooting, and to connect to services running on local host.

docker0 is a virtual bridge interface created by *Docker*. This bridge creates a separate network for docker containers and allows them to communicate with each other.

**interface details**

Let’s look closely at details of *ifconfig* output:

Link encap shows how packets are encapsulated for transmission. Most interfaces wrap packets in *Ethernet* frames.

HWaddr is hardware address of the ethernet interface (also known as MAC address).

inet addr is IPv4 address assigned to the interface.

Bcast is broadcast address for the interface.

Mask is network mask for the interface.

inet6 addr is IPv6 address assigned to the interface.

Scope is scope of IPv6 address. It can be *link-local* or *global*. Link-local address is used in local area network and is not routable. Global address is routable.

UP indicates that kernel modules related to the interface have been loaded and interface is activated.

BROADCAST indicates that interface is configured to handle broadcast packets, which is required for obtaining IP address via DHCP.

RUNNING indicates that interface is ready to accept data.

MULTICAST indicates that interface supports multicasting.

MTU is maximum transmission unit. IP datagrams larger than MTU bytes will be fragmented into multiple Ethernet frames.

Metric determines the cost of using the interface. Interfaces with lower cost have higher priority.

**interface stats**

RX packets is a total number of packets received.

RX errors shows a total number of packets received with error. This includes too-long-frames errors, ring-buffer overflow errors, CRC errors, frame alignment errors, fifo overruns, and missed packets.

RX dropped is a number of dropped packets due to unintended VLAN tags or receiving IPv6 frames when interface is not configured for IPv6.

RX overruns is a number of received packets that experienced fifo overruns, caused by rate at which a buffer gets full and kernel isn’t able to empty it.

RX frame is a number of misaligned frames, i.e. frames with length not divisible by 8.

TX packets is total number of packets transmitted.

TX errors, TX dropped and TX overruns are similar to RX equivalents.

TX carriers is a number of packets that experienced loss of carriers. This usually happens when link is flapping.

TX collisions is a number of transmitted packets that experienced Ethernet collisions.

TX txqueuelen is length of transmission queue.

RX bytes is a total number of bytes received over interface.

TX bytes is a total number of bytes transmitted over interface.

**Summary**

Despite being superseded by *ip* command, *ifconfig* is still commonly used and provides lots of useful details about network interfaces, both physical and virtual.