**Objectives**

The goal of this assignment is to enable and configure various services on a DSL server, utilizing common Windows networking tools to exercise these services. During the process, I will analyze the underlying protocols used for requesting and responding to various activities, gaining insight into how they function.

To accomplish this, I will conduct packet captures and observations using a packet sniffer to analyze network traffic. For this lab, I will utilize **Wireshark**, a powerful network protocol analyzer, to capture and examine the data exchanged between devices. Wireshark will allow me to view and interpret messages transmitted by protocols across different layers of the protocol stack. This hands-on approach will enhance my understanding of network communications and protocol behavior, while providing valuable insights into network traffic analysis.

**Getting Wireshark**

To run Wireshark, use a computer that supports both Wireshark and libpcap or WinPCap packet capture library.

**Running Wireshark**

**A screenshot of a computer

Description automatically generated**

**Setting Up the DSL Server for Wireshark Tracing**

Once the IP address of the virtual machine was determined, open the DSL panel which will expose services you want enabled. Below you realize BetaFTP, Monkey Web and SSH servers were activated. It usually turns gray to light grey which symbolizes the activation of the service.

A computer screen shot of a computer screen

Description automatically generated

To make sure all the services are enabled or running by using the process status command, “ps”. To perform this I started up the ATerminal in the virtual machine.

A computer screen shot of a computer screen

Description automatically generated

In the above, the ATerminal shows us that the services are running, I can go and test each service by using Microsoft Windows host browser and point the Universal Resource Locator (URL) to the DSL server that is running.

Use http://<DSL\_IP>

A screenshot of a computer

Description automatically generated

Figure: Windows Chrome Results of DSL Home Page

Since FTP is not considered secure, some of the newer browsers do not support the use of it. I use the FTP utility within the Windows command line by typing the following:

ftp <DSL\_IP>

I will then be prompted for the username and password. Once I get the FTP command prompt, I enter “dir” to obtain the directory listings.

A screenshot of a computer

Description automatically generated

Figure: FTP from the Windows Command Line

Because using FTP protocol from the browser is no longer supported, using FTP utility on command line is the only way I was able to do this.

**Using Wireshark**

I will be performing this operation on a VMware, Host-Only network. A host-only network is an isolated network that does not bridge nor NAT to the internet. As such, only the local Windows host and any virtual system, configured with a VMnet1 interface, can communicate with each other. For this type of network, VMware provides a DHCP service, which allows the DSL server to obtain an IP address.

Using a Host-Only network configuration allows us to really focus on a specific protocol, without having to worry about all the other communication that goes on between a Windows host system and all the network applications (email, web, update services, security software, you name it).

**An overview of the key property columns listed for each packet:**

* No: The index number of the packet in this packet capture file.
* Time: The timestamp of the packet.
* Source: The source IP address.
* Destination: The destination IP address.
* Protocol: The protocol contained in the packet.
* Length: The total length of the packet.
* Info: Some information about the data in the packet (the payload) as interpreted by Wireshark.

**SOLUTION**

A screenshot of a computer

Description automatically generated

To begin packet capture, I select the desired adaptor to trace(VMnet1). Tracing will begin with the selected network.

A screenshot of a computer

Description automatically generated

I am able to verify my capture by making sure the title displays the desired adaptor to capture and that the packets are being captured. And while Wireshark is still running, I enter the following URL <http://128.162.222.128/wireshark-labs/INTRO-wireshark.html>. The below will be displayed on my browser.

A screenshot of a computer

Description automatically generated

In order to display this page, my browser had to establish a virtual connection to my DSL server, sending it Ethernet frames containing HTTP messages (as well as other frames used to aid in the management of the connections).

Once the browser has displayed the INTRO-wireshark.html page, I stop Wireshark packet capture and use the filter to find the HTTP traffic.

A screenshot of a computer

Description automatically generated

In the following screenshot, I will determine whether my browser supports the "favicon" (favorite icon) displayed on the document tab. The **DSL\_IP** refers to the IP address of my DSL server, which I identified by running the “ifconfig” command in the A terminal. During the packet capture process, a large volume of packets was recorded. To streamline the analysis and focus on relevant traffic, I applied an **HTTP filter** in Wireshark to isolate HTTP requests and responses related to the favicon retrieval.A screenshot of a computer

Description automatically generated

**Conclusion**

The goal of this fire Wireshark lab was primarily to strengthen my understanding of virtualization, DSL Linux management and to introduce myself to Wireshark and packet sniffing.