LAB MANUAL

Course: CSC339 Data Communication and Computer Networks

Department of Computer Science

Learning Procedure

1) Stage J (Journey inside-out the concept)
2) Stage a_1 (Apply the learned)
3) Stage V (Verify the accuracy)
4) Stage a_2 (Assess your work)

COMSATS Institute of Information Technology (CIIT)
Islamabad

Table of Contents

LAB # 01

Statement Purpose:

- Get acquainted with Wireshark
- Make some simple packet captures and observations

Activity outcomes:

- Students will have gained the basic understanding of Wireshark Packet Sniffer to see "protocols in action"
- After this lab, students will have developed basic understanding of digging deep into the network protocols.

Instructor Note:

- In this first Wireshark lab, you'll get acquainted with Wireshark, and make some simple packet captures and observations.

1) Stage J(Journey)

Introduction

One's understanding of network protocols can often be greatly deepened by "seeing protocols in action" and by "playing around with protocols" – observing the sequence of messages exchanged between two protocol entities, delving down into the details of protocol operation, and causing protocols to perform certain actions and then observing these actions and their consequences. This can be done in simulated scenarios or in a "real" network environment such as the Internet. In the Wireshark labs you'll be doing in this course, you'll be running various network applications in different scenarios using your own computer (or you can borrow a friends; let me know if you don't have access to a computer where you can install/run Wireshark). You'll observe the network protocols in your computer "in action," interacting and exchanging messages with protocol entities executing elsewhere in the Internet. Thus, you and your computer will be an integral part of these "live" labs. You'll observe, and you'll learn, by doing

2) Stage a1 (apply)

Lab Activities:

Getting Wireshark

In order to run Wireshark, you will need to have access to a computer that supports both Wireshark and the *libpcap* or *WinPCap* packet capture library. The *libpcap* software will be installed for you, if it is not installed within your operating system, when you install Wireshark. See http://www.wireshark.org/download.html for a list of supported operating systems and download sites

Download and install the Wireshark software:

- Go to http://www.wireshark.org/download.html and download and install the Wireshark binary for your computer.

The Wireshark FAQ has a number of helpful hints and interesting tidbits of information, particularly if you have trouble installing or running Wireshark.

Running Wireshark

When you run the Wireshark program, you'll get a startup screen, as shown below:

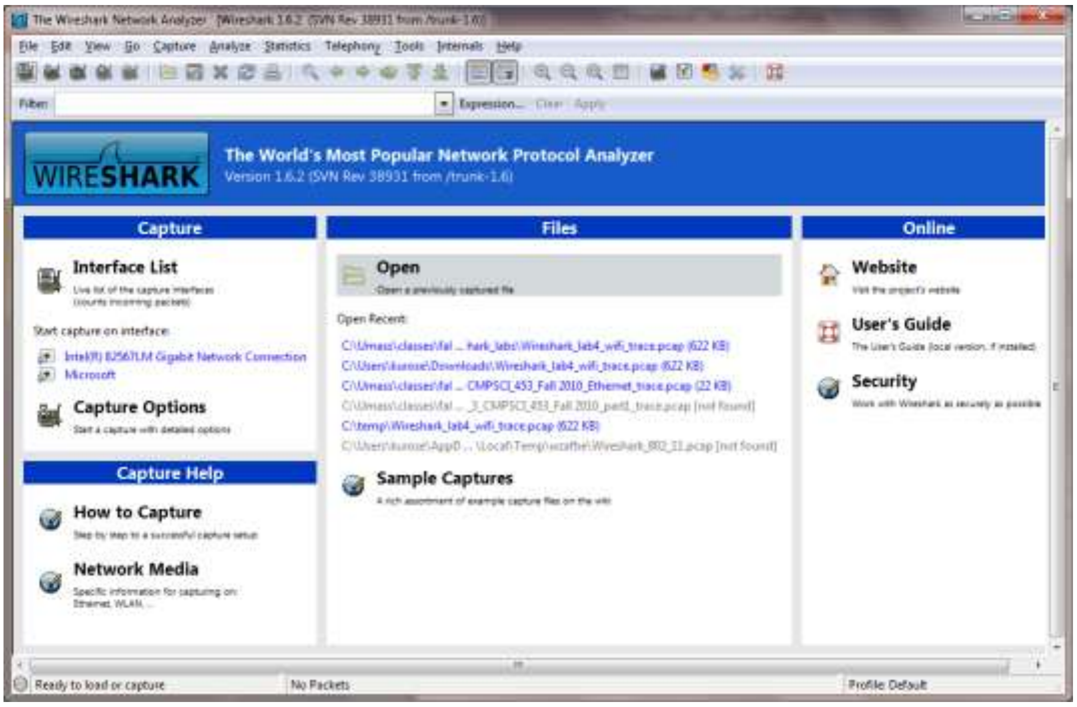
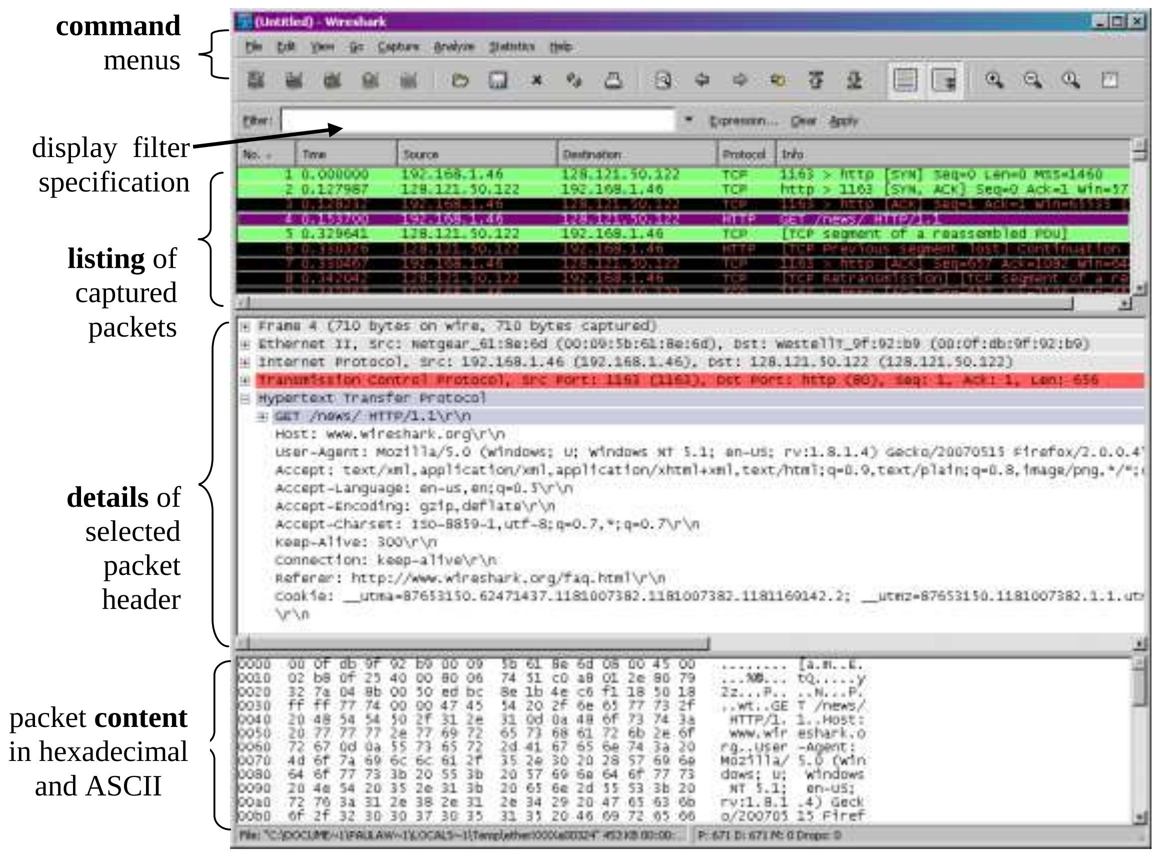

The Wireshark Network Analyzer [Wireshark 1.6.2 (SVN Rev 38931 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Version 1.6.2 (SVN Rev 38931 from /trunk-1.6)
Capture
Interface List
Start capture on interface:
Intel(R) 82567LM Gigabit Network Connection
Microsoft
Capture Options
Capture Help
How to Capture
Network Media
Files
Open
Open Recent:
Sample Captures
Online
Website
User's Guide
Security
Ready to load or capture
No Packets
Profile: Default

Figure 2: Initial Wireshark Screen

Take a look at the upper left hand side of the screen – you'll see an "Interface list". This is the list of network interfaces on your computer. Once you choose an interface, Wireshark will capture all packets on that interface. In the example above, there is an Ethernet interface (Gigabit network Connection) and a wireless interface ("Microsoft").

If you click on one of these interfaces to start packet capture (i.e., for Wireshark to begin capturing all packets being sent to/from that interface), a screen like the one below will be displayed, showing information about the packets being captured. Once you start packet capture, you can stop it by using the Capture pull down menu and selecting Stop.


command menus
display filter specification
listing of captured packets
details of selected packet header
packet **content** in hexadecimal and ASCII
(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply

```
No. .  Time        Source           Destination      Protocol  Info
     1 0.000000    192.168.1.46     128.121.50.122   TCP       1163 > http [SYN] Seq=0 Len=0 MSS=1460
     2 0.127987    128.121.50.122   192.168.1.46     TCP       http > 1163 [SYN, ACK] Seq=0 Ack=1 Win=57
     3 0.128232    192.168.1.46     128.121.50.122   TCP       1163 > http [ACK] Seq=1 Ack=1 Win=65535
     4 0.153700    192.168.1.46     128.121.50.122   HTTP      GET /news/ HTTP/1.1
     5 0.329641    128.121.50.122   192.168.1.46     TCP       [TCP segment of a reassembled PDU]
     6 0.330326    128.121.50.122   192.168.1.46     HTTP      [TCP Previous segment lost] Continuation
     7 0.330467    192.168.1.46     128.121.50.122   TCP       1163 > http [ACK] Seq=657 Ack=1082 Win=64
     8 0.342042    128.121.50.122   192.168.1.46     TCP       [TCP Retransmission] [TCP segment of a re

⊞ Frame 4 (710 bytes on wire, 710 bytes captured)
⊞ Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: WestellT_9f:92:b9 (00:0f:db:9f:92:b9)
⊞ Internet Protocol, Src: 192.168.1.46 (192.168.1.46), Dst: 128.121.50.122 (128.121.50.122)
⊞ Transmission Control Protocol, Src Port: 1163 (1163), Dst Port: http (80), Seq: 1, Ack: 1, Len: 656
⊟ Hypertext Transfer Protocol
   ⊞ GET /news/ HTTP/1.1\r\n
    Host: www.wireshark.org\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4) Gecko/20070515 Firefox/2.0.0.4
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://www.wireshark.org/faq.html\r\n
    Cookie: __utma=87653150.62471437.1181007382.1181007382.1181160142.2; __utmz=87653150.1181007382.1.1.ut
    \r\n

0000  00 0f db 9f 92 b9 00 09  5b 61 8e 6d 08 00 45 00   ........ [a.m..E.
0010  02 b8 0f 25 40 00 80 06  74 51 c0 a8 01 2e 80 79   ...%@... tQ.....y
0020  32 7a 04 8b 00 50 ed bc  8e 1b 4e c6 f1 18 50 18   2z...P.. ..N...P.
0030  ff ff 77 74 00 00 47 45  54 20 2f 6e 65 77 73 2f   ..wt..GE T /news/
0040  20 48 54 54 50 2f 31 2e  31 0d 0a 48 6f 73 74 3a    HTTP/1. 1..Host:
0050  20 77 77 77 2e 77 69 72  65 73 68 61 72 6b 2e 6f    www.wir eshark.o
0060  72 67 0d 0a 55 73 65 72  2d 41 67 65 6e 74 3a 20   rg..User -Agent: 
0070  4d 6f 7a 69 6c 6c 61 2f  35 2e 30 20 28 57 69 6e   Mozilla/ 5.0 (Win
0080  64 6f 77 73 3b 20 55 3b  20 57 69 6e 64 6f 77 73   dows; U;  Windows
0090  20 4e 54 20 35 2e 31 3b  20 65 6e 2d 55 53 3b 20    NT 5.1;  en-US; 
00a0  72 76 3a 31 2e 38 2e 31  2e 34 29 20 47 65 63 6b   rv:1.8.1 .4) Geck
00b0  6f 2f 32 30 30 37 30 35  31 35 20 46 69 72 65 66   o/200705 15 Firef

File: "C:\DOCUME~1\PAULAW~1\LOCALS~1\Temp\ether000a00324" 452 KB 00:00:   P: 671 D: 671 M: 0 Drops: 0
```

Figure 3: Wireshark Graphical User Interface, during packet capture and analysis

The Wireshark interface has five major components:

- The **command menus** are standard pulldown menus located at the top of the window. Of interest to us now are the File and Capture menus. The File menu allows you to save captured packet data or open a file containing previously captured packet data, and exit the Wireshark application. The Capture menu allows you to begin packet capture.
- The **packet-listing window** displays a one-line summary for each packet captured, including the packet number (assigned by Wireshark; this is *not* a packet number contained in any protocol's header), the time at which the packet was captured, the packet's source and destination addresses, the protocol type, and protocol-specific information contained in the packet. The packet listing can be sorted according to any of these categories by clicking on a column name. The protocol type field lists the highest-level protocol that sent or received this packet, i.e., the protocol that is the source or ultimate sink for this packet.
- The **packet-header details window** provides details about the packet selected (highlighted) in the packet-listing window. (To select a packet in the packet-listing window, place the cursor over the packet's one-line summary in the packet-listing window and click with the left mouse button.). These details include information about the Ethernet frame (assuming the packet was sent/received over an Ethernet interface) and IP datagram that contains this packet. The amount of Ethernet and IP-layer detail displayed can be expanded or minimized by clicking on the plus minus boxes to the left of the Ethernet frame or IP datagram line in the packet details window. If the packet has been carried over TCP or UDP, TCP or UDP details will also be displayed, which can similarly be expanded or minimized. Finally, details about the highest-level protocol that sent or received this packet are also provided.
- The **packet-contents window** displays the entire contents of the captured frame, in both ASCII and hexadecimal format.

Towards the top of the Wireshark graphical user interface, is the **packet display filter field,** into which a protocol name or other information can be entered in order to filter the information displayed in the packet-listing window (and hence the packet-header and packet-contents windows). In the example below, we'll use the packet-display filter field to have Wireshark hide (not display) packets except those that correspond to HTTP messages.

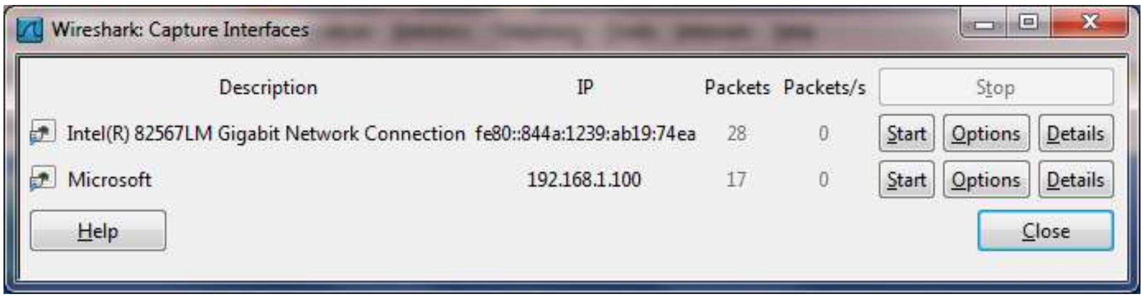
Activity 1:

The best way to learn about any new piece of software is to try it out! We'll assume that your computer is connected to the Internet via a wired Ethernet interface. Indeed, I recommend that you do this first lab on a computer that has a wired Ethernet connection, rather than just a wireless connection. Do the following

1. Start up your favorite web browser, which will display your selected homepage.

2. Start up the Wireshark software. You will initially see a window similar to that shown in Figure 2. Wireshark has not yet begun capturing packets.

3. To begin packet capture, select the Capture pull down menu and select *Interfaces.* This will cause the "Wireshark: Capture Interfaces" window to be displayed, as shown in Figure 4.



Description	IP	Packets	Packets/s
Intel(R) 82567LM Gigabit Network Connection	fe80::844a:1239:ab19:74ea	28	0
Microsoft	192.168.1.100	17	0

Figure 4: Wireshark Capture Interface Window

4. You'll see a list of the interfaces on your computer as well as a count of the packets that have been observed on that interface so far. Click on *Start* for the interface on which you want to begin packet capture (in the case, the Gigabit network Connection). Packet capture will now begin - Wireshark is now capturing all packets being sent/received from/by your computer!

5. Once you begin packet capture, a window similar to that shown in Figure 3 will appear. This window shows the packets being captured. By selecting *Capture* pulldown menu and selecting *Stop*, you can stop packet capture. But don't stop packet capture yet. Let's capture some interesting packets first. To do so, we'll need to generate some network traffic. Let's do so using a web browser, which will use the HTTP protocol that we will study in detail in class to download content from a website.

6. While Wireshark is running, enter the URL: http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html
and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at gaia.cs.umass.edu and exchange HTTP messages with the server in order to download this page, as discussed in section 2.2 of the text. The Ethernet

frames containing these HTTP messages (as well as all other frames passing through your Ethernet adapter) will be captured by Wireshark.

7. After your browser has displayed the INTRO-wireshark-file1.html page (it is a simple one line of congratulations), stop Wireshark packet capture by selecting stop in the Wireshark capture window. The main Wireshark window should now look similar to Figure 3. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! The HTTP message exchanges with the gaia.cs.umass.edu web server should appear somewhere in the listing of packets captured. But there will be many other types of packets displayed as well (see, e.g., the many different protocol types shown in the *Protocol* column in Figure 3). Even though the only action you took was to download a web page, there were evidently many other protocols running on your computer that are unseen by the user. We'll learn much more about these protocols as we progress through the text! For now, you should just be aware that there is often much more going on than "meet's the eye"!

8. Type in "http" (without the quotes, and in lower case – all protocol names are in lower case in Wireshark) into the display filter specification window at the top of the main Wireshark window. Then select *Apply* (to the right of where you entered "http"). This will cause only HTTP message to be displayed in the packet-listing window.

9. Find the HTTP GET message that was sent from your computer to the gaia.cs.umass.edu HTTP server. (Look for an HTTP GET message in the "listing of captured packets" portion of the Wireshark window (see Figure 3) that shows "GET" followed by the gaia.cs.umass.edu URL that you entered. When you select the HTTP GET message, the Ethernet frame, IP datagram, TCP segment, and HTTP message header information will be displayed in the packet-header window[1]. By clicking on '+' and '-' right-pointing and down-pointing arrowheads to the left side of the packet details window, *minimize* the amount of Frame, Ethernet, Internet Protocol, and Transmission Control Protocol information displayed. *Maximize* the amount information displayed about the HTTP protocol. Your Wireshark display should now look roughly (Note, in particular, the minimized amount of protocol information for all protocols except HTTP, and the maximized amount of protocol information for HTTP in the packet-header window).

The goal of this first lab was primarily to introduce you to Wireshark. The following questions will demonstrate that you've been able to get Wireshark up and running, and have explored some of its capabilities. Answer the following questions, based on your Wireshark experimentation:

1. List 3 different protocols that appear in the protocol column in the unfiltered packet-listing window in step 7 above.
2. How long did it take from when the HTTP GET message was sent until the HTTP OK reply was received? (By default, the value of the Time column in the packet-listing window is the amount of time, in seconds, since Wireshark tracing began. To display the Time field in time-of-day format, select the Wireshark *View* pull down menu, then select Time *Display Format*, then select *Time-of-day*.)
3. What is the Internet address of the gaia.cs.umass.edu (also known as www-net.cs.umass.edu)? What is the Internet address of your computer?
4. Print the two HTTP messages (GET and OK) referred to in question 2 above. To do so, select *Print* from the Wireshark *File* command menu, and select the "*Selected Packet Only"* and *"Print as displayed"* radial buttons, and then click OK.

[1] Recall that the HTTP GET message that is sent to the gaia.cs.umass.edu web server is contained within a TCP segment, which is contained (encapsulated) in an IP datagram, which is encapsulated in an Ethernet frame. If this process of encapsulation isn't quite clear yet, review section 1.5 in the text

Solution:

Ans1: The following protocols appeared in the protocol column in the unfiltered packet listing window after downloading a webpage: TCP, UDP, HTTP, DNS.

Ans2: If we look at the frame section of the GET request we see that the time the packet arrived is 11:43:13.422848000

Frame 109(492 bytes on wire, 492 bytes captured)
Arrival Time: sep 17, 2004 11:43:13.42284800
Time delta from previous packet: 6.826032000 seconds
Time since reference or first frame: 9.263432000 seconds
Frame Number: 109
Packet Length: 492 bytes
Capture Length: 492 bytes

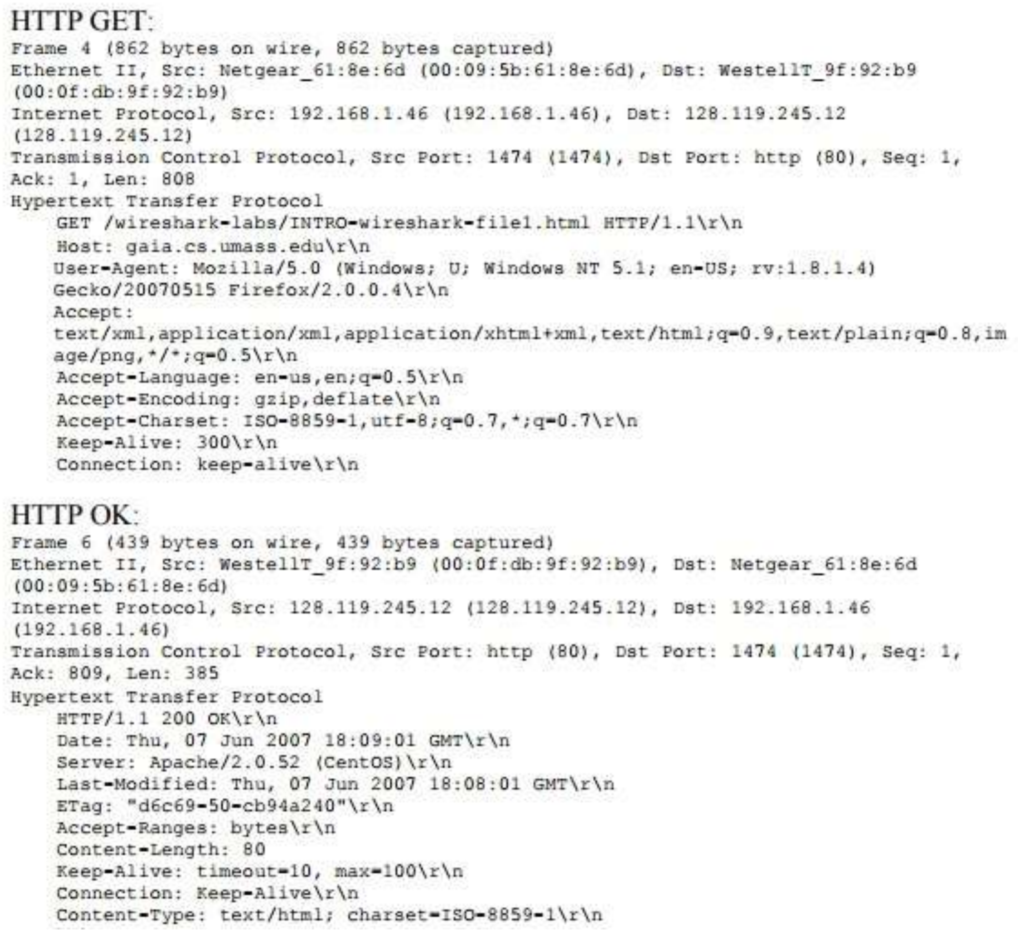
The same section for the HTTP OK shows an arrival time of 11:43:13.43960400

Frame 110(444 bytes on wire, 444 bytes captured)
Arrival Time: sep 17, 2004 11:43:13.439604000
Time delta from previous packet: 0.016756000 seconds
Time since reference or first frame: 9.280188000 seconds
Frame Number: 110
Packet Length: 444 bytes
Capture Length: 444 bytes

The difference of these 2 times gives .43960400 - .426032000 = 0.013572 seconds

Ans3: If we look at the IP section of the GET request, the source and destination
Source: (128.238.244.28 (128.238.244.28)
Destination: 128.119.245.12 (128.119.245.12

The source is the local machine's address and the destination is the web server's public My (local machine's) address = 128.238.244.28
IP address 128.119.245.12 = www-net.cs.umass.edu.


HTTP GET:

```
Frame 4 (862 bytes on wire, 862 bytes captured)
Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: WestellT_9f:92:b9
(00:0f:db:9f:92:b9)
Internet Protocol, Src: 192.168.1.46 (192.168.1.46), Dst: 128.119.245.12
(128.119.245.12)
Transmission Control Protocol, Src Port: 1474 (1474), Dst Port: http (80), Seq: 1,
Ack: 1, Len: 808
Hypertext Transfer Protocol
    GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1\r\n
    Host: gaia.cs.umass.edu\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4)
    Gecko/20070515 Firefox/2.0.0.4\r\n
    Accept:
    text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,im
    age/png,*/*;q=0.5\r\n
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
```

HTTP OK:

```
Frame 6 (439 bytes on wire, 439 bytes captured)
Ethernet II, Src: WestellT_9f:92:b9 (00:0f:db:9f:92:b9), Dst: Netgear_61:8e:6d
(00:09:5b:61:8e:6d)
Internet Protocol, Src: 128.119.245.12 (128.119.245.12), Dst: 192.168.1.46
(192.168.1.46)
Transmission Control Protocol, Src Port: http (80), Dst Port: 1474 (1474), Seq: 1,
Ack: 809, Len: 385
Hypertext Transfer Protocol
    HTTP/1.1 200 OK\r\n
    Date: Thu, 07 Jun 2007 18:09:01 GMT\r\n
    Server: Apache/2.0.52 (CentOS)\r\n
    Last-Modified: Thu, 07 Jun 2007 18:08:01 GMT\r\n
    ETag: "d6c69-50-cb94a240"\r\n
    Accept-Ranges: bytes\r\n
    Content-Length: 80
    Keep-Alive: timeout=10, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html; charset=ISO-8859-1\r\n
```

3) Stage v (verify)

Home Activities:

Activity 1:

Start a new capture, and do some arbitrary web browsing, such as visiting some Wikipedia articles, reading some news, etc. Note down all the protocols used at each layer.

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 02

Statement Purpose:

- Get acquainted with Packet Tracer
- Make some simple Packet Tracer scenarios

Activity outcomes:

- Students will have gained the basic understanding of Packet Tracer to see "protocols in action"
- After this lab, students will have developed basic understanding of digging deep into the network protocols.

Instructor Note:

- In this first Packet Tracer lab, you'll get acquainted with Packet Tracer, and make some simple configurations.

1) StageJ(Journey)

Introduction

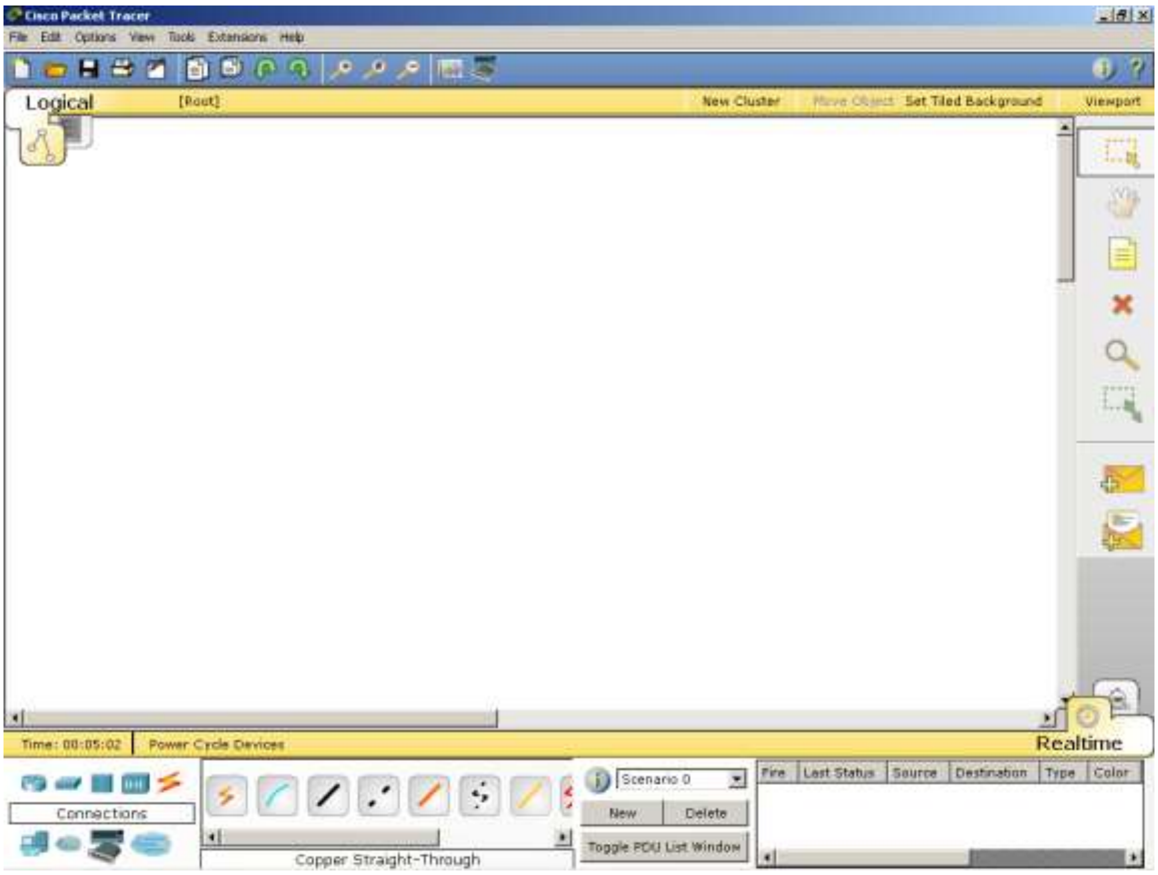
Packet Tracer is a protocol simulator developed by Dennis Frezzo and his team at Cisco Systems. Packet Tracer (PT) is a powerful and dynamic tool that displays the various protocols used in networking, in either Real Time or Simulation mode. The purpose of this lab is to become familiar with the Packet Tracer interface. Learn how to use existing topologies and build your own.
This activity will provide an opportunity to explore the standard lab setup using Packet Tracer simulator. Packet Tracer has two file formats it can create: .pkt files (network simulation model files) and .pka files (activity files for practice). When you create your own networks in Packet Tracer, or modify existing files from your instructor or your peers, you will often use the .pkt file format. When you launched this activity from the curriculum, these instructions appeared. They are the result of the .pka, Packet Tracer activity file format. At the bottom of these instructions are two buttons: Check Results (which gives you feedback on how much of the activity you have completed) and Reset Activity (which starts the activity over, if you want to clear your work or gain more practice).

2) Stage a1 (apply)

Lab Activities:

Introduction to the Packet Tracer Interface using a Hub Topology

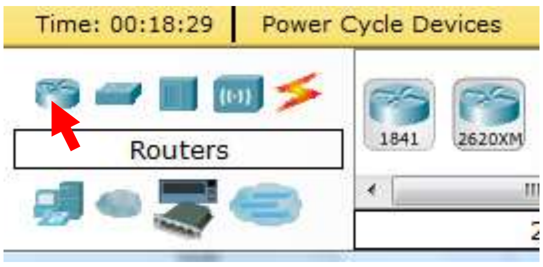
Step 1: Start Packet Tracer

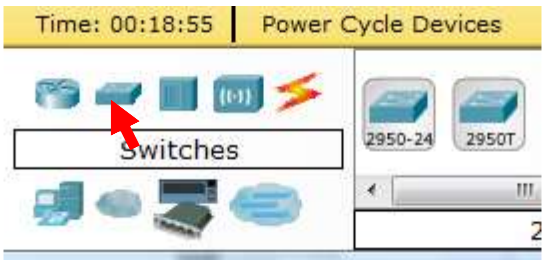


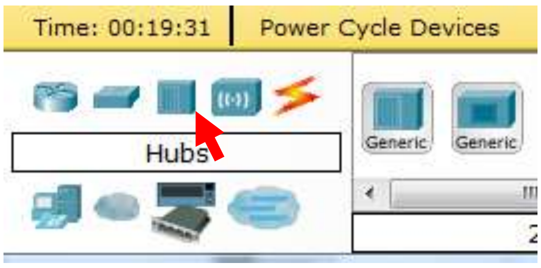
Step 2: Choosing Devices and Connections

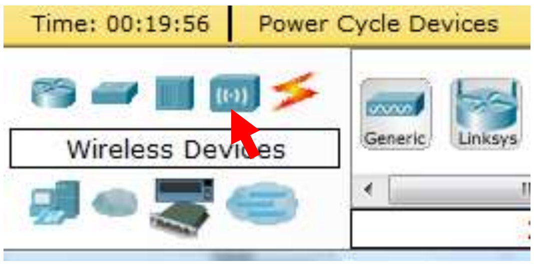
We will begin building our network topology by selecting devices and the media in which to connect them. Several types of devices and network connections can be used. For this lab we will keep it simple by using End Devices, Switches, Hubs, and Connections.

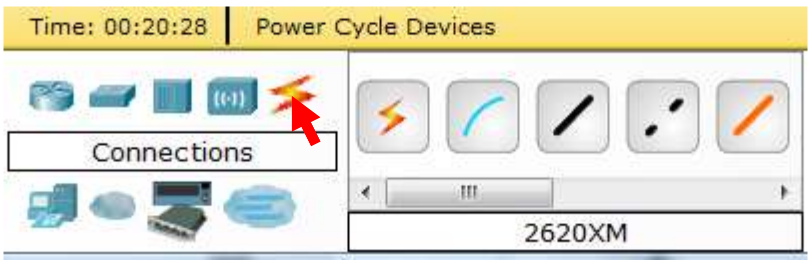
Single click on each group of devices and connections to display the various choices. The devices you see may differ slightly.

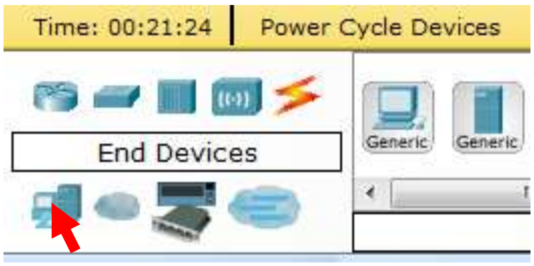


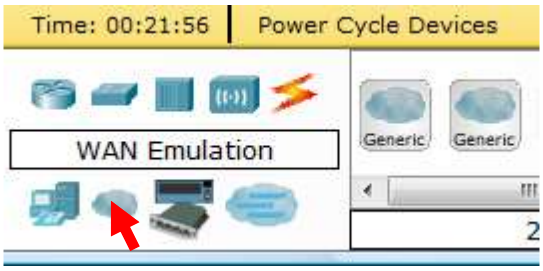


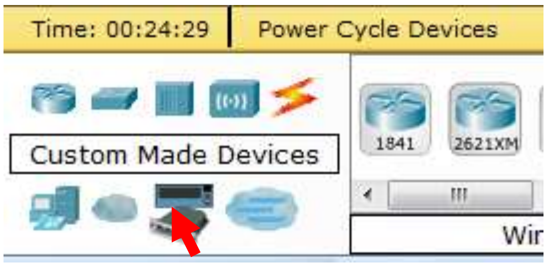


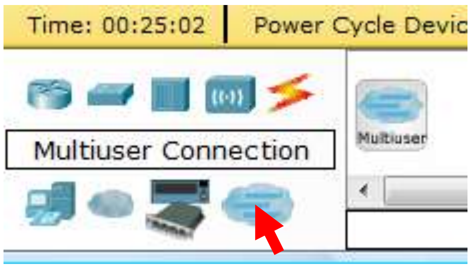












1. Other than generic routers, name 3 router models available on the simulation software.

   ____

2. What are the two types of serial cables available for WAN connectivity?

   ____

3. What are the two types of copper cable connectors?

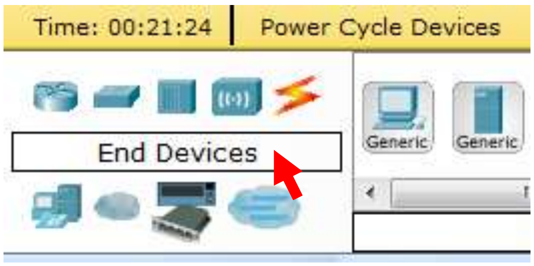
   ____

4. Other than generic end devices, enumerate four end devices available.

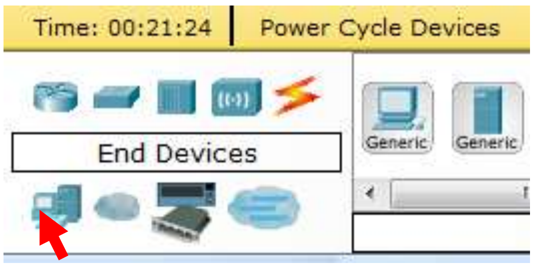
   ____

Step 3: Building the Topology – Adding Hosts

Single click on the End Devices.



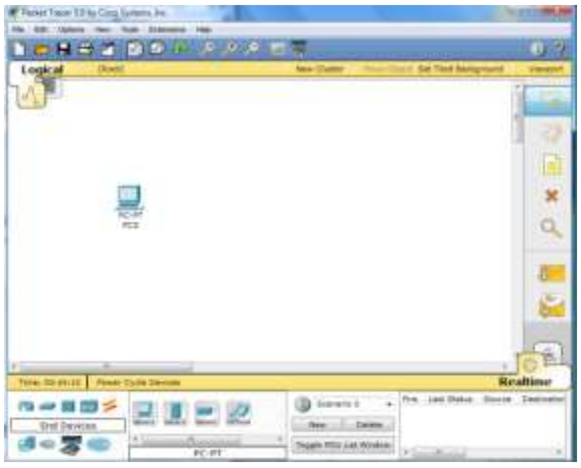
Single click on the Generic host.



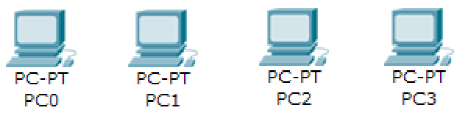
Move the cursor into topology area. You will notice it turns into a plus "+" sign.

+

Single click in the topology area and it copies the device.



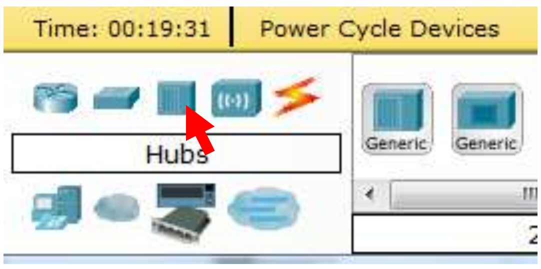
Add three more hosts.

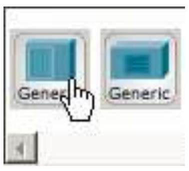


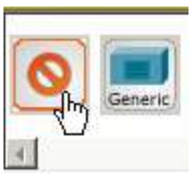
Step 4: Building the Topology – Connecting the Hosts to Hubs and Switches

Adding a Hub

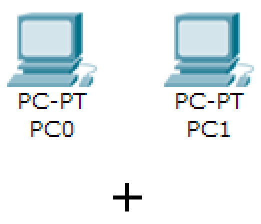
Select a hub, by clicking once on Hubs and once on a Generic hub.

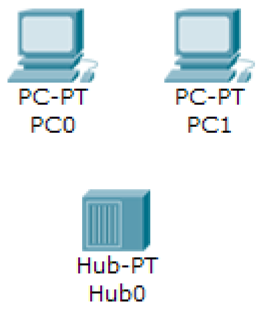




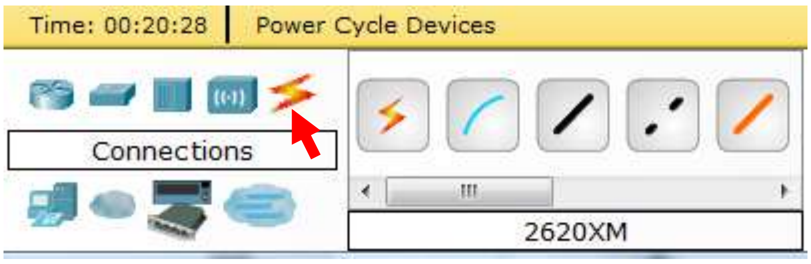


Add the hub by moving the plus sign "+" below PC0 and PC1 and click once.

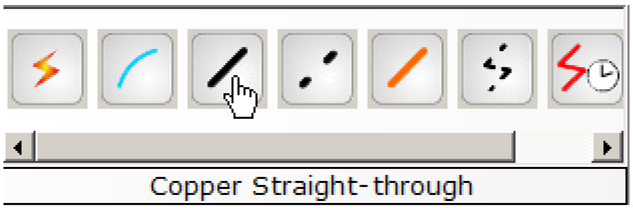




Connect PC0 to Hub0 by first choosing Connections.

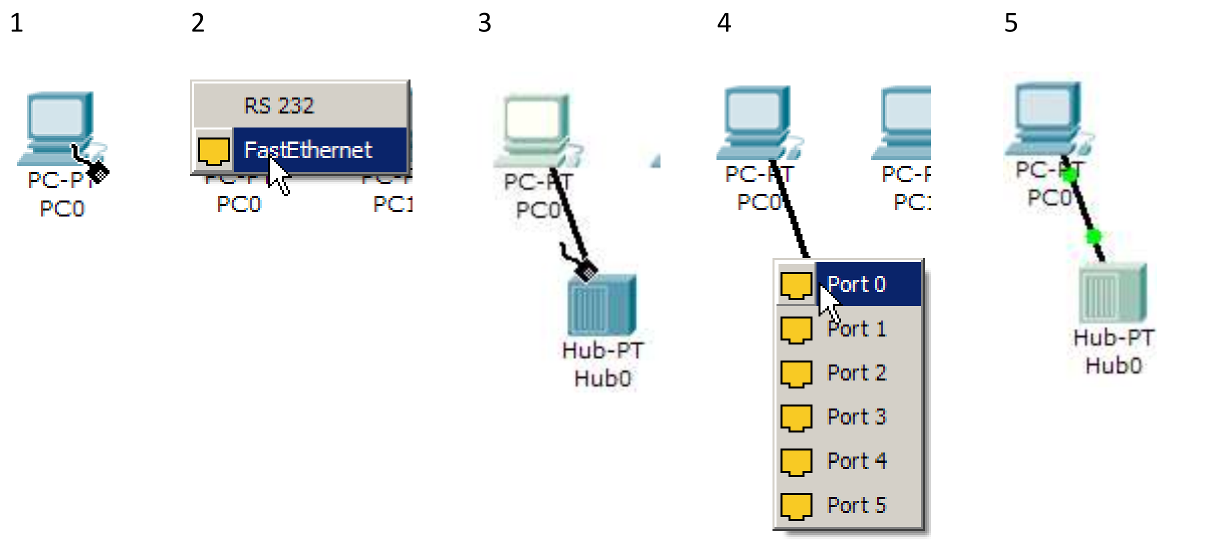


Click once on the Copper Straight-through cable.

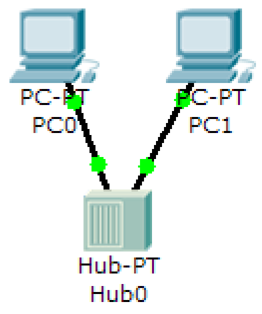


Perform the following steps to connect PC0 to Hub0:

1. Click once on PC0
2. Choose FastEthernet
3. Drag the cursor to Hub0
4. Click once on Hub0 and choose Port 0
5. Notice the green link lights on both the PC0 Ethernet NIC and the Hub0 Port 0 showing that the link is active.

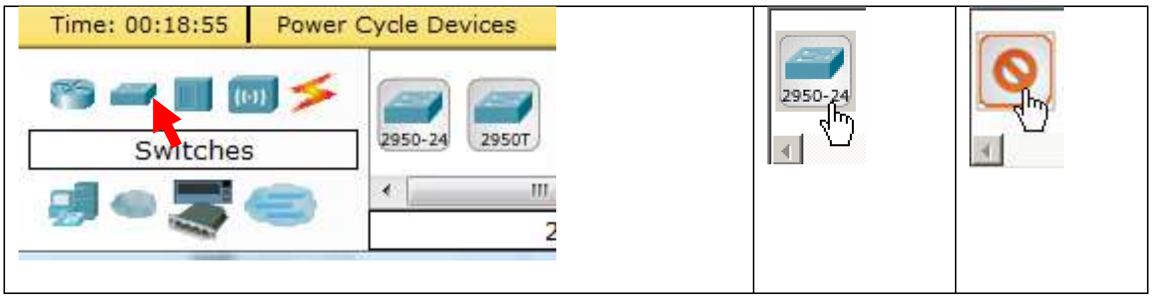


Repeat the steps above for PC1 connecting it to Port 1 on Hub0. (The actual hub port you choose does not matter.)

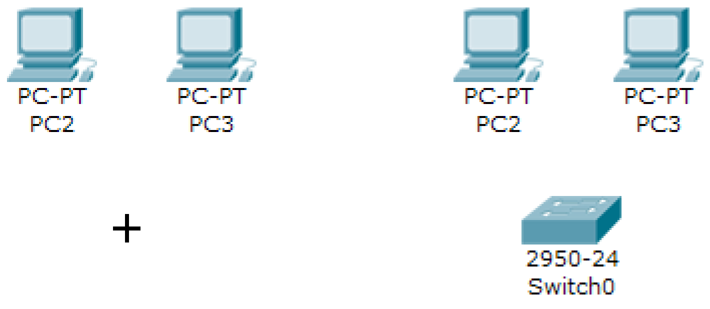


Adding a Switch

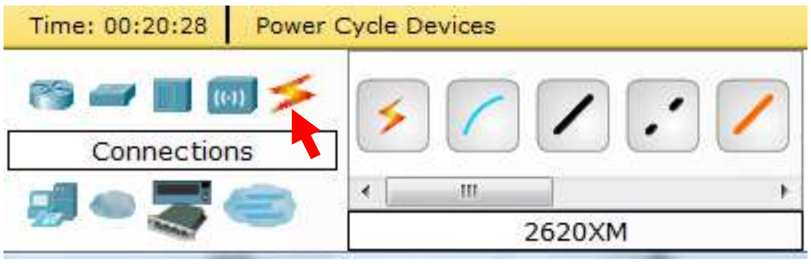
Select a switch, by clicking once on Switches and once on a 2950-24 switch.



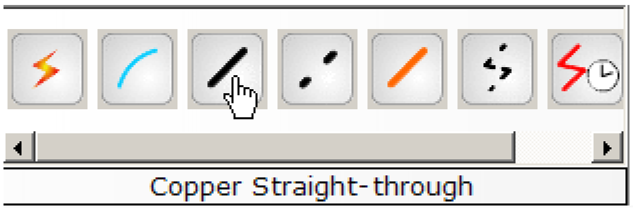
Add the switch by moving the plus sign "+" below PC2 and PC3 and click once.



Connect PC2 to Hub0 by first choosing Connections.

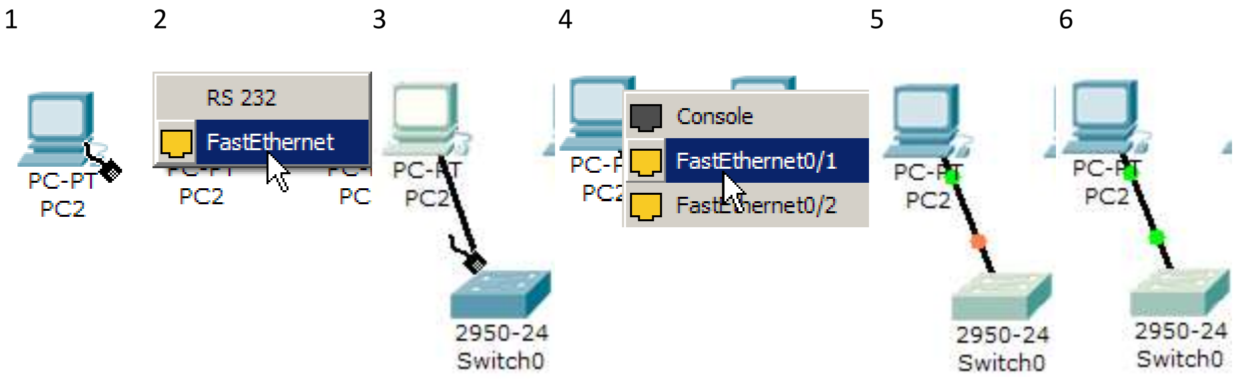


Click once on the Copper Straight-through cable.

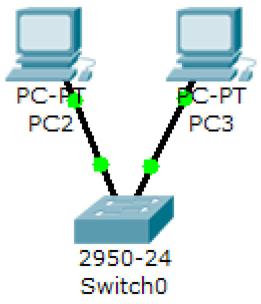


Perform the following steps to connect PC2 to Switch0:

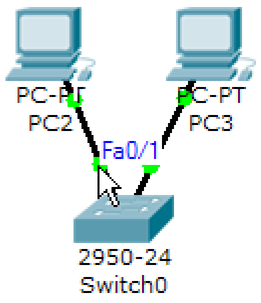
1. Click once on PC2
2. Choose FastEthernet
3. Drag the cursor to Switch0
4. Click once on Switch0 and choose FastEthernet0/1
5. Notice the green link lights on PC2 Ethernet NIC and amber light Switch0 FastEthernet0/1 port. The switch port is temporarily not forwarding frames, while it goes through the stages for the Spanning Tree Protocol (STP) process.
6. After a about 30 seconds the amber light will change to green indicating that the port has entered the forwarding stage. Frames can now forwarded out the switch port.



Repeat the steps above for PC3 connecting it to Port 3 on Switch0 on port FastEtherent0/2. (The actual switch port you choose does not matter.)

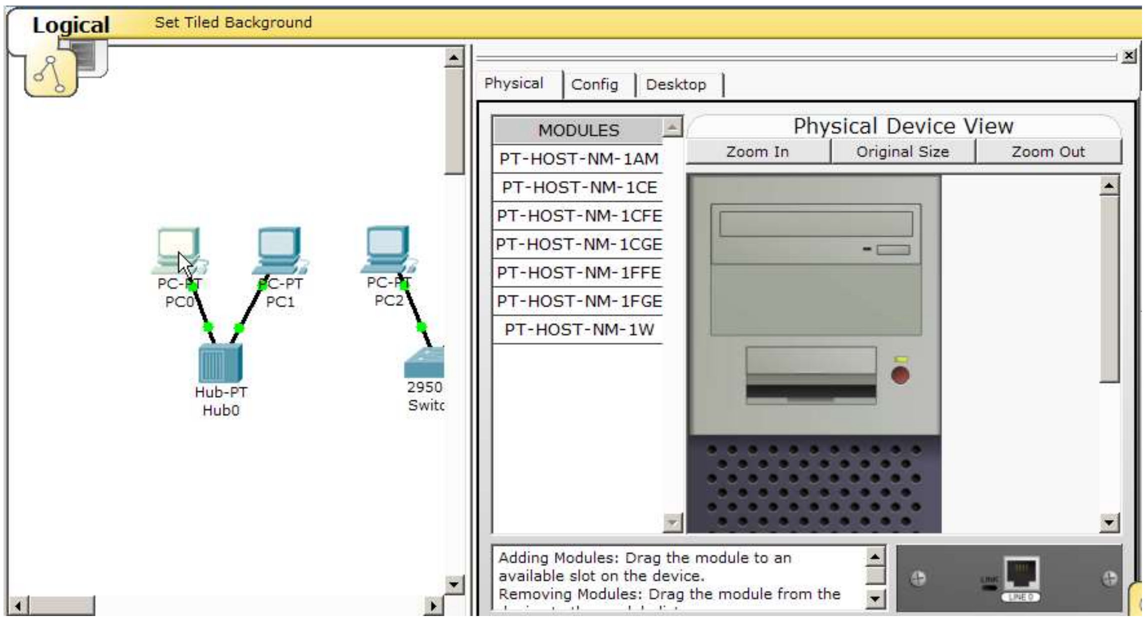


Move the cursor over the link light to view the port number. Fa means FastEthernet, 100 Mbps Ethernet.



Step 5: Configuring IP Addresses and Subnet Masks on the Hosts
Before we can communicate between the hosts we need to configure IP Addresses and Subnet Masks on the devices.

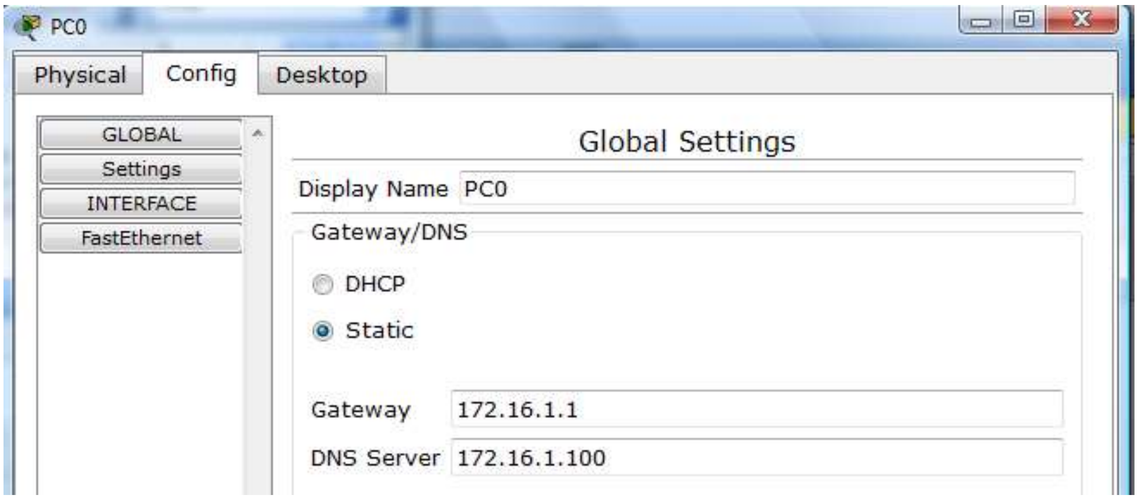
Click once on PC0.



Adding Modules: Drag the module to an available slot on the device.
Removing Modules: Drag the module from the

Choose the Config tab and click on Settings. It is here that you can change the name of PC0. It is also here where you would enter a Gateway IP Address, also known as the default gateway and the

DNS Server IP Address. We will discuss this later, but this would be the IP address of the local router. If you want, you can enter the Gateway IP Address 172.16.1.1 and DNS Server IP Address 172.16.1.100, although it will not be used in this lab.



Global Settings

Display Name: PC0

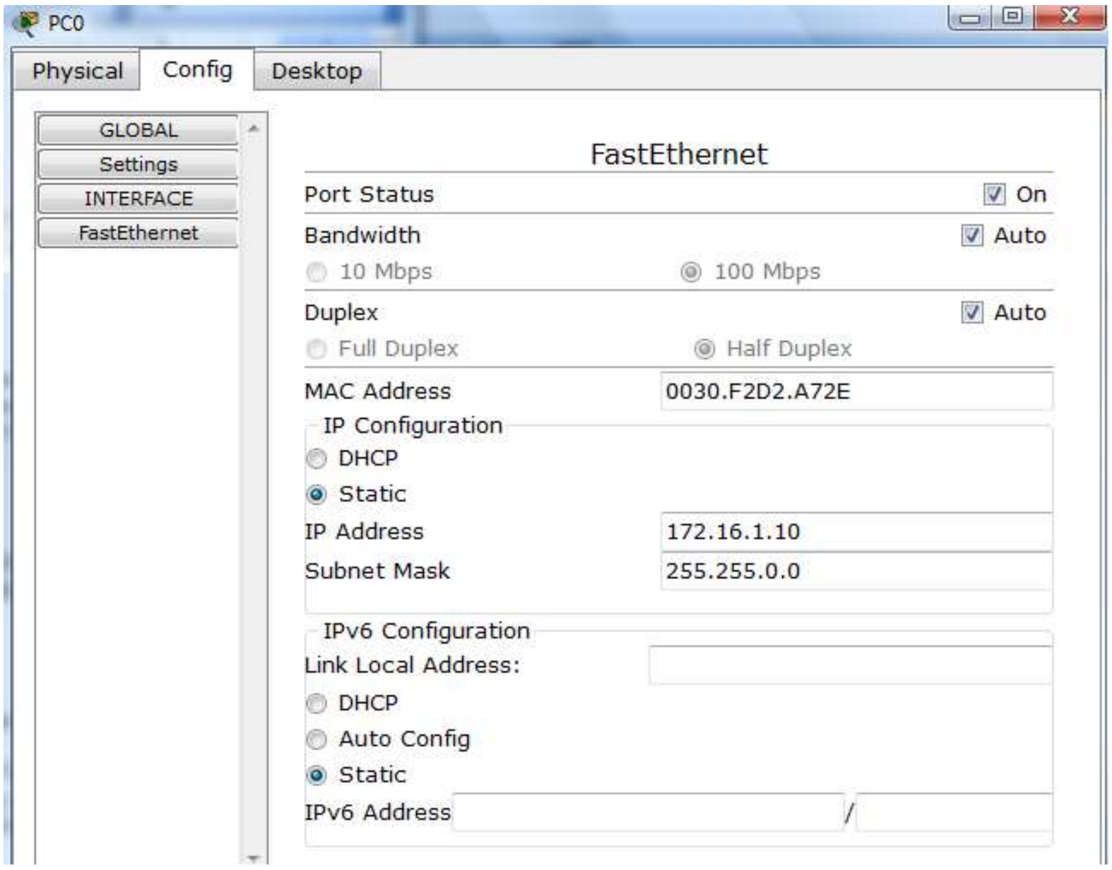
Gateway/DNS

DHCP [✓ Static]

Gateway: 172.16.1.1

DNS Server: 172.16.1.100

Click on Interface and then FastEthernet. Although we have not yet discussed IP Addresses, add the IP Address to 172.16.1.10. Click once in the Subnet Mask field to enter the default Subnet Mask. You can leave this at 255.255.0.0.



FastEthernet

Port Status: ✓ On

Bandwidth: ✓ Auto

10 Mbps [✓ 100 Mbps]

Duplex: ✓ Auto

Full Duplex [✓ Half Duplex]

MAC Address: 0030.F2D2.A72E

IP Configuration

DHCP [✓ Static]

IP Address: 172.16.1.10

Subnet Mask: 255.255.0.0

IPv6 Configuration

Link Local Address:

DHCP Auto Config [✓ Static]

IPv6 Address: /

Also, notice this is where you can change the Bandwidth (speed) and Duplex of the Ethernet NIC (Network Interface Card). The default is Auto (autonegotiation), which means the NIC will negotiate with the hub or switch. The bandwidth and/or duplex can be manually set by removing the check from the Auto box and choosing the specific option.

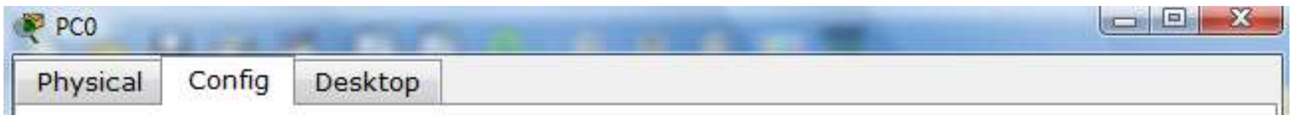
Bandwidth - Auto

If the host is connected to a hub or switch port which can do 100 Mbps, then the Ethernet NIC on the host will choose 100 Mbps (Fast Ethernet). Otherwise, if the hub or switch port can only do 10 Mbps, then the Ethernet NIC on the host will choose 10 Mbps (Ethernet).

Duplex - Auto

Hub: If the host is connected to a hub, then the Ethernet NIC on the host will choose Half Duplex.

Switch: If the host is connected to a switch, and the switch port is configured as Full Duplex (or Autonegotiation), then the Ethernet NIC on the host will choose Full Duplex. If the switch port is configured as Half Duplex, then the Ethernet NIC on the host will choose Half Duplex. (Full Duplex is a much more efficient option.)
The information is automatically saved when entered.
To close this dialog box, click the “X” in the upper right.

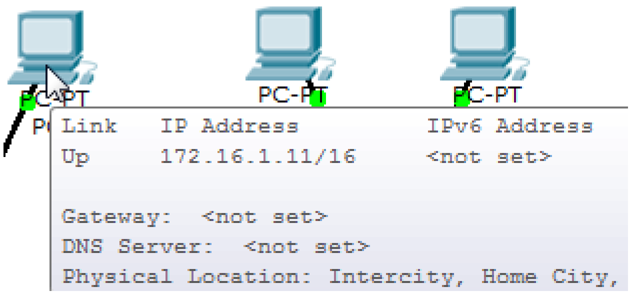


Repeat these steps for the other hosts. Use the information below for IP Addresses and Subnet Masks.

Host	IP Address	Subnet Mask
PC0	172.16.1.10	255.255.0.0
PC1	172.16.1.11	255.255.0.0
PC2	172.16.1.12	255.255.0.0
PC3	172.16.1.13	255.255.0.0

Verify the information

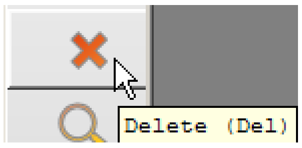
To verify the information that you entered, move the Select tool (arrow) over each host.



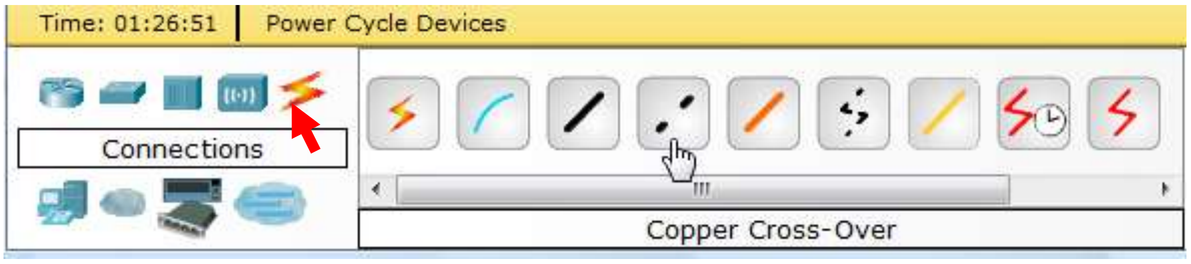
```
Link  IP Address       IPv6 Address
Up    172.16.1.11/16   <not set>

Gateway:  <not set>
DNS Server:  <not set>
Physical Location: Intercity, Home City,
```

Deleting a Device or Link

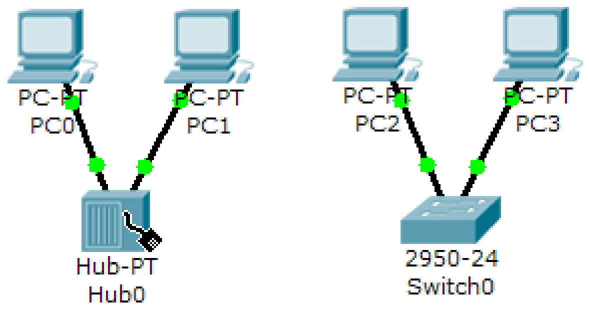
To delete a device or link, choose the Delete tool and click on the item you wish to delete.



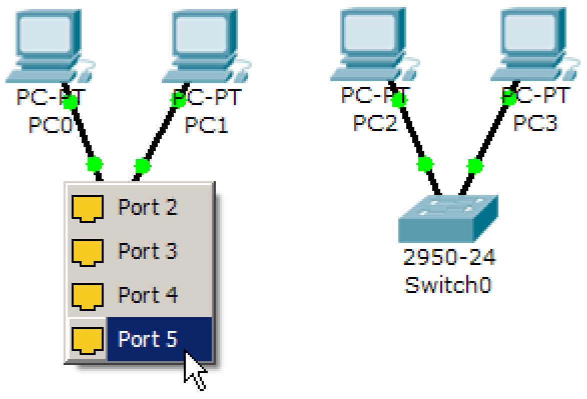
Step 6: Connecting Hub0 to Switch0
To connect like-devices, like a Hub and a Switch, we will use a Cross-over cable. Click once the Cross-over Cable from the Connections options.



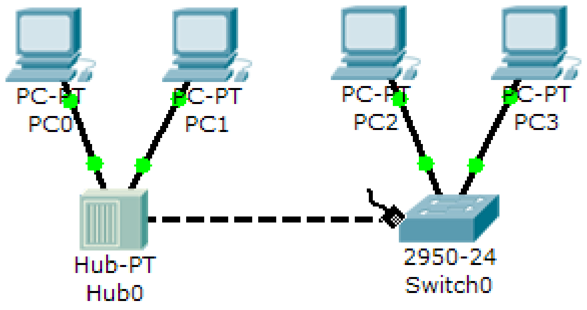
Move the Connections cursor over Hub0 and click once.



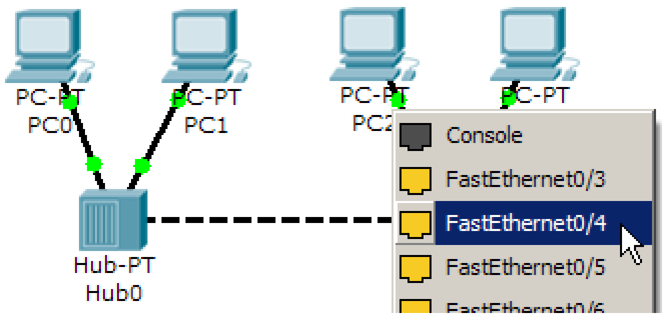
Select Port 5 (actual port does not matter).



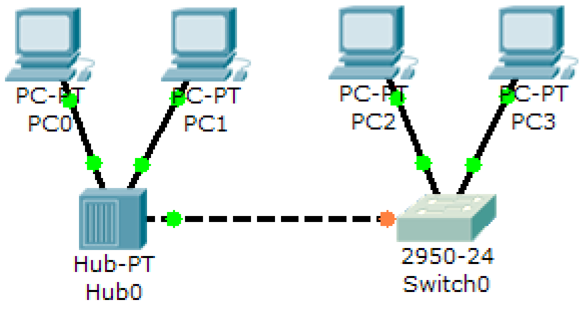
Move the Connections cursor to Switch0.

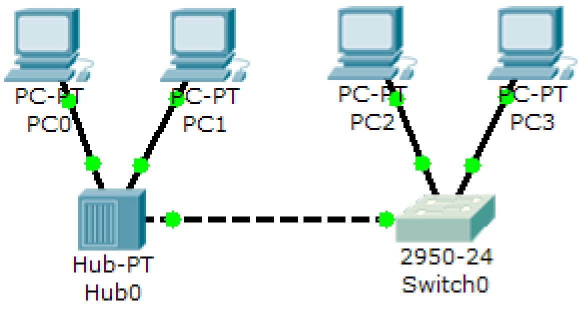


Click once on Switch0 and choose FastEthernet0/4 (actual port does not matter).



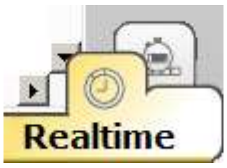
The link light for switch port FastEthernet0/4 will begin as amber and eventually change to green as the Spanning Tree Protocol transitions the port to forwarding.



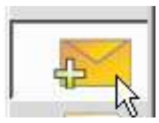


Step 7: Verifying Connectivity in Realtime Mode

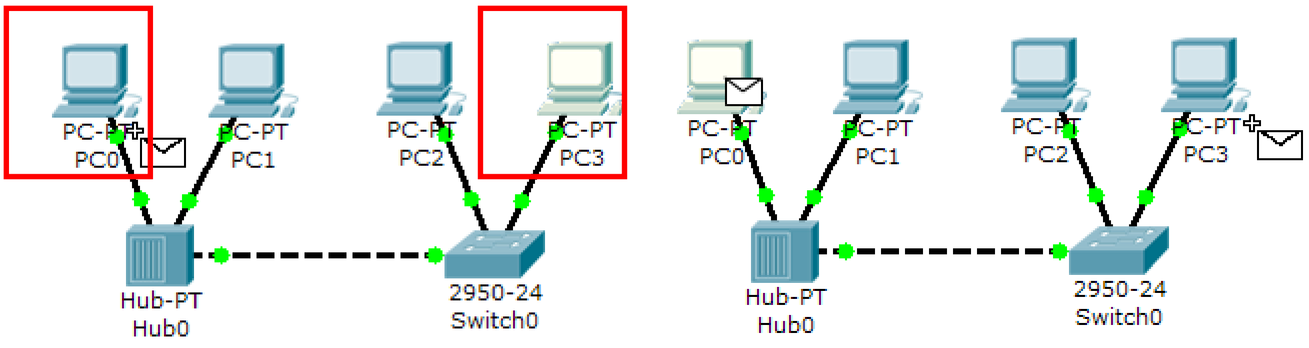
Be sure you are in Realtime mode.



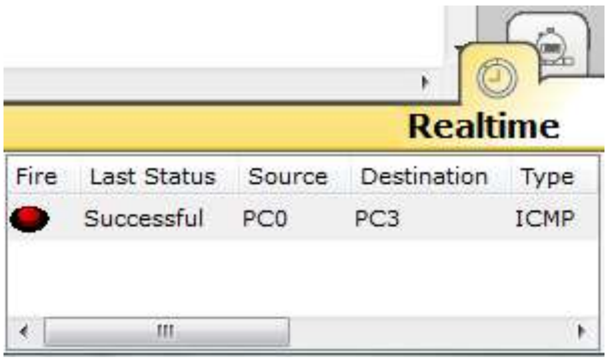
Select the Add Simple PDU tool used to ping devices.



Click once on PC0, then once on PC3.



The PDU Last Status should show as Successful.



Fire	Last Status	Source	Destination	Type
●	Successful	PC0	PC3	ICMP

Change the IP address of PC3 to 172.16.2.13. Perform a ping from PC0 to PC3. What is the ping result?

____

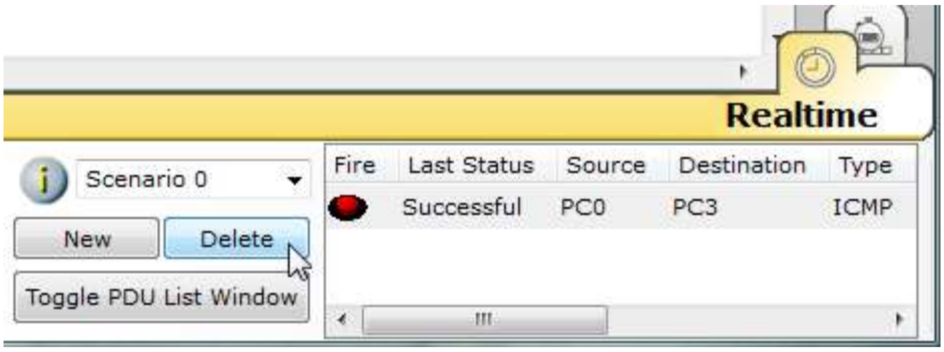
Return the IP address of PC3 to 172.16.1.13. Change the IP address of PC2 to 172.17.1.12. Perform a ping from PC0 to PC2. What is the ping result?

____

Resetting the Network

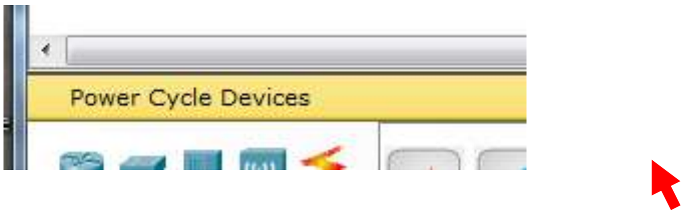
At this point we will want to reset the network, whenever you want to reset the network and begin the simulation again, perform the following tasks:

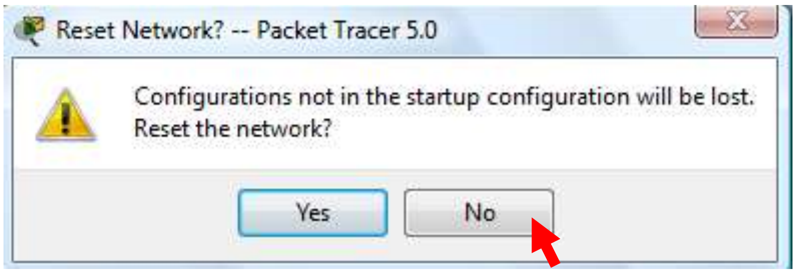
Click Delete in the PDU area.



Fire	Last Status	Source	Destination	Type
●	Successful	PC0	PC3	ICMP

Now, Power Cycle Devices and confirm the action.

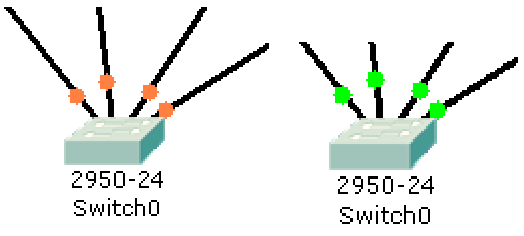



Reset Network? -- Packet Tracer 5.0

Configurations not in the startup configuration will be lost. Reset the network?

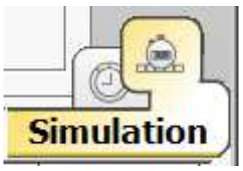
Waiting for Spanning Tree Protocol (STP)

Note: Because Packet Tracer also simulates the Spanning Tree Protocol, at times the switch may show amber lights on its interfaces. You will need to wait for the lights to turn green on the switches before they will forward any Ethernet frames.

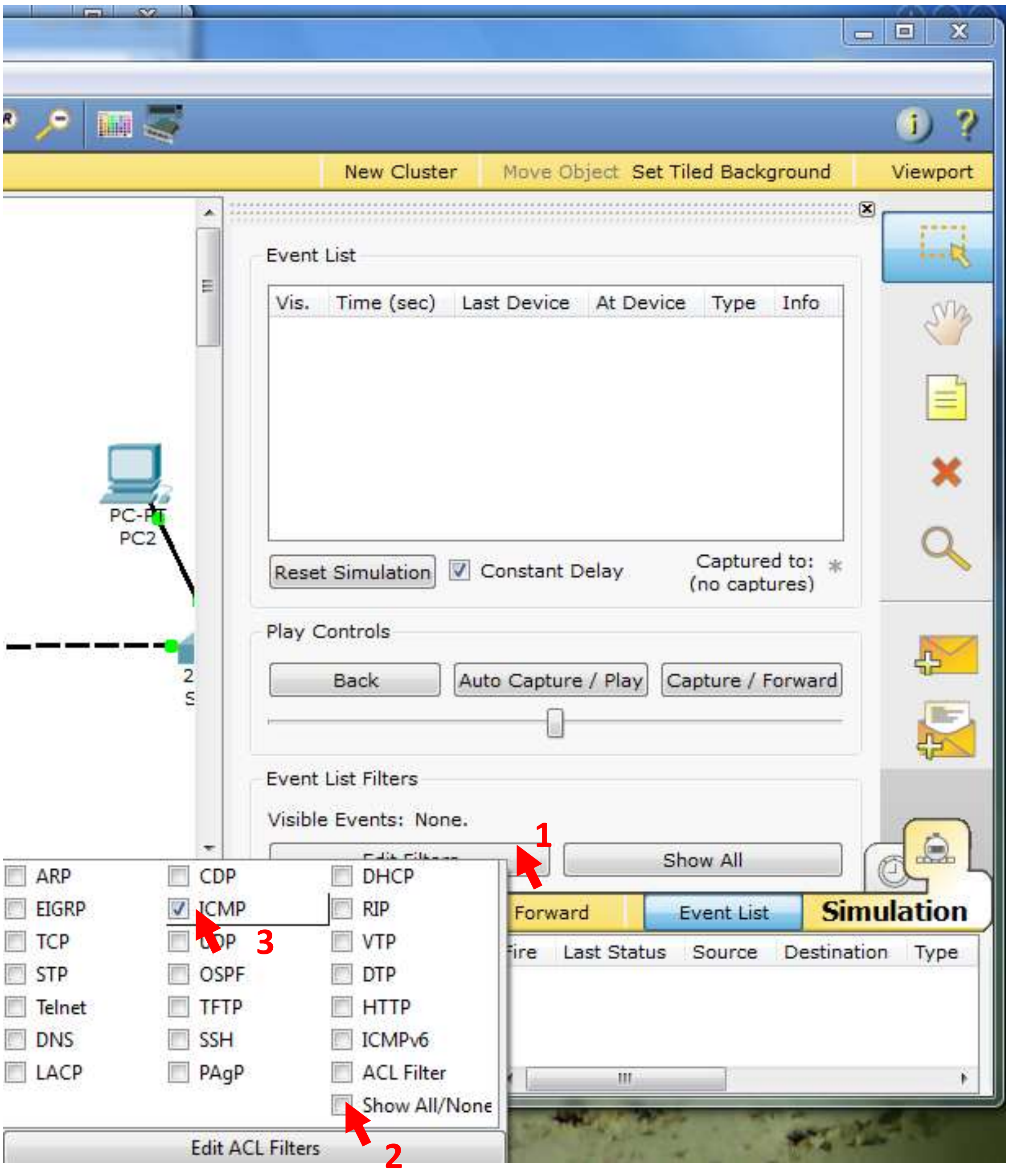


Step 8: Verifying Connectivity in Simulation Mode

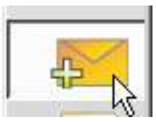
Be sure you are in Simulation mode.



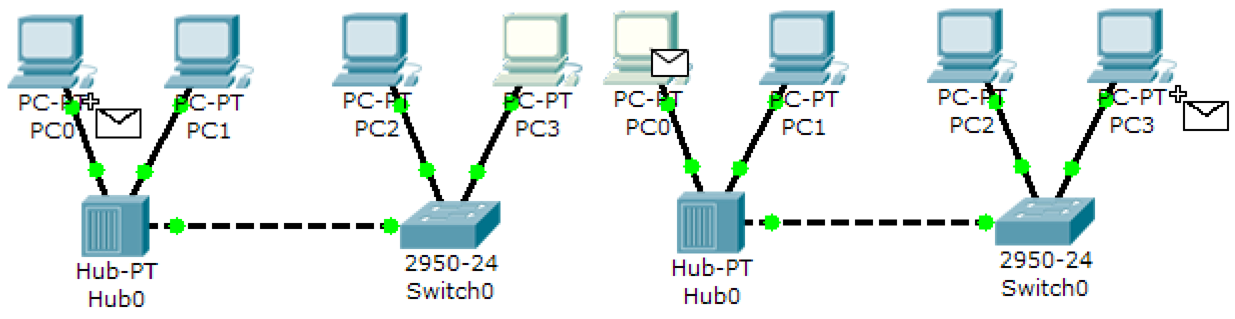
Deselect all filters (All/None) and select only ICMP.



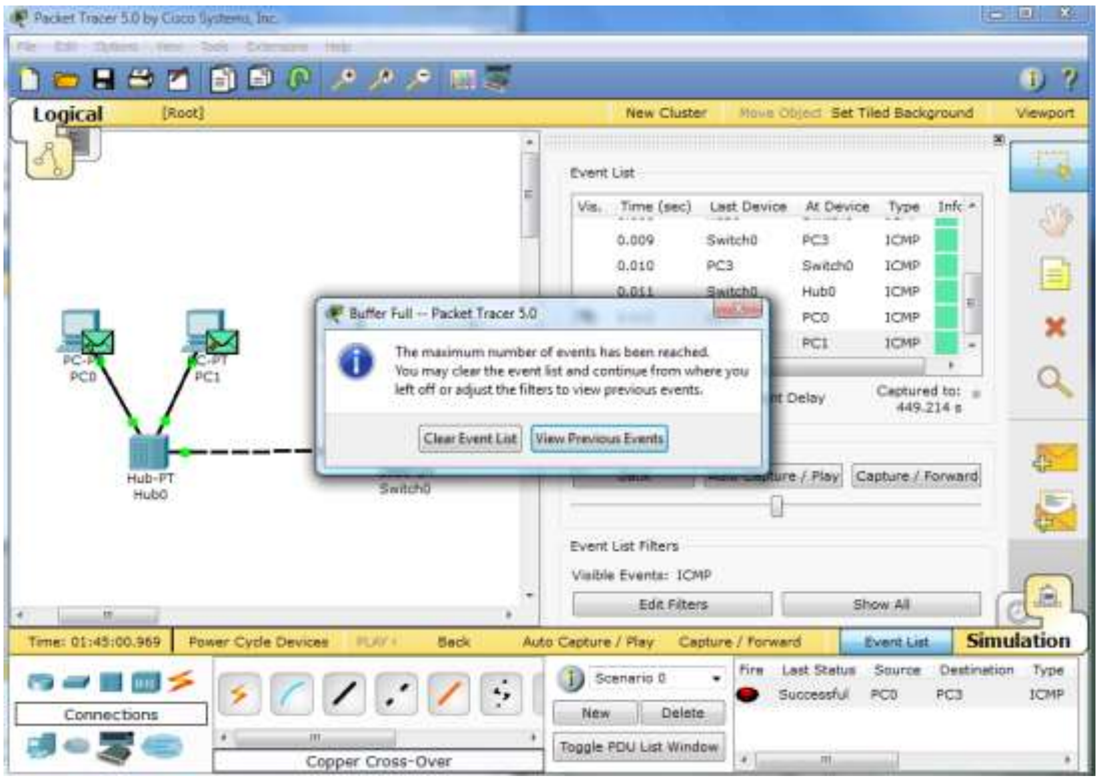
Select the Add Simple PDU tool used to ping devices..



Click once on PC0, then once on PC3.



Continue clicking Capture/Forward button until the ICMP ping is completed. You should see the ICMP messages move between the hosts, hub and switch. The PDU Last Status should show as Successful. Click on Clear Event List if you do not want to look at the events or click Preview Previous Events if you do. For this exercise it does not matter.



The maximum number of events has been reached. You may clear the event list and continue from where you left off or adjust the filters to view previous events.

Vis.	Time (sec)	Last Device	At Device	Type	Info
	0.009	Switch0	PC3	ICMP	🟩
	0.010	PC3	Switch0	ICMP	🟩
	0.011	Switch0	Hub0	ICMP	🟩
			PC0	ICMP	🟩
			PC1	ICMP	🟩

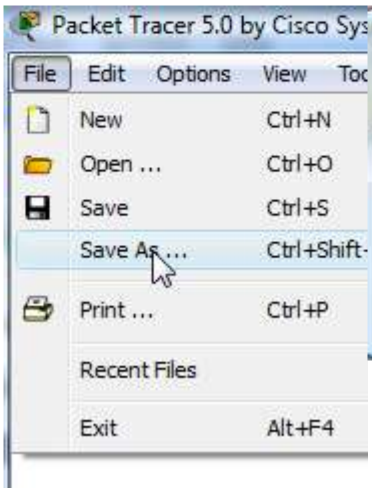
Captured to: 449.214 s

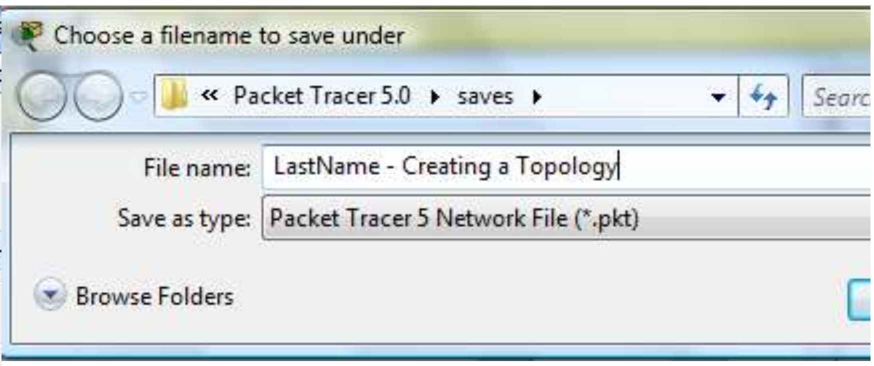
Visible Events: ICMP

Fire	Last Status	Source	Destination	Type
●	Successful	PC0	PC3	ICMP

Step 9: Saving the Topology

Perform the following steps to save the topology (uses .pkt file extension).

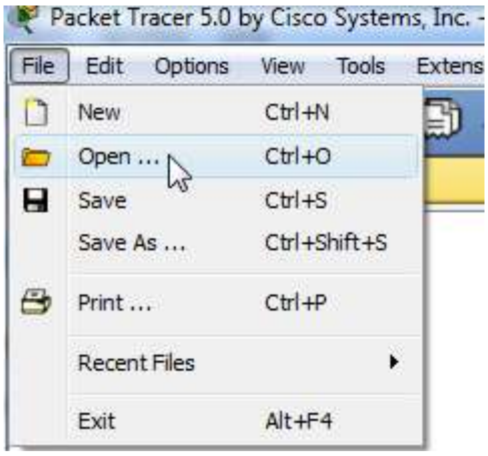


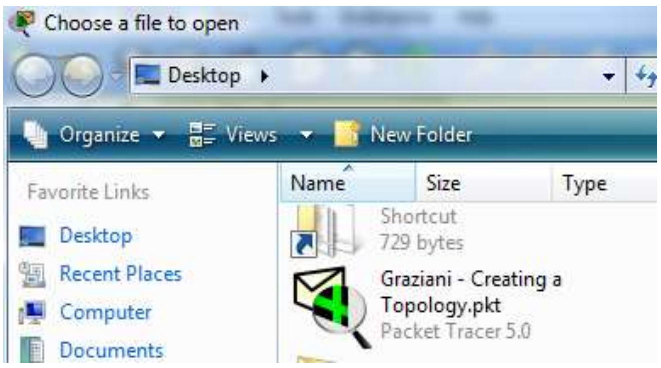

Choose a filename to save under
« Packet Tracer 5.0 ▸ saves ▸
Searc
Browse Folders

File name: LastName - Creating a Topology

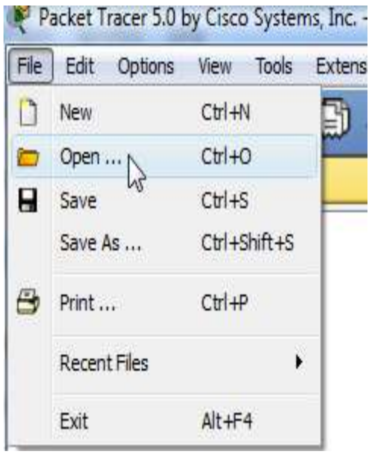
Save as type: Packet Tracer 5 Network File (*.pkt)

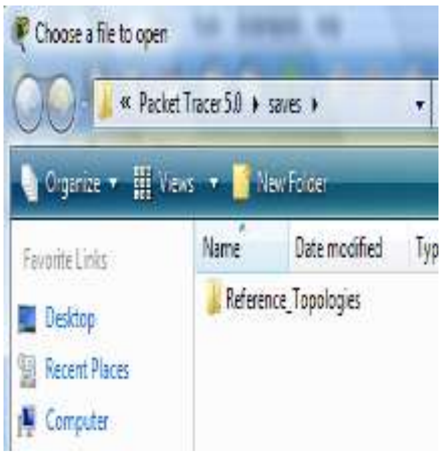
Opening Existing Topologies

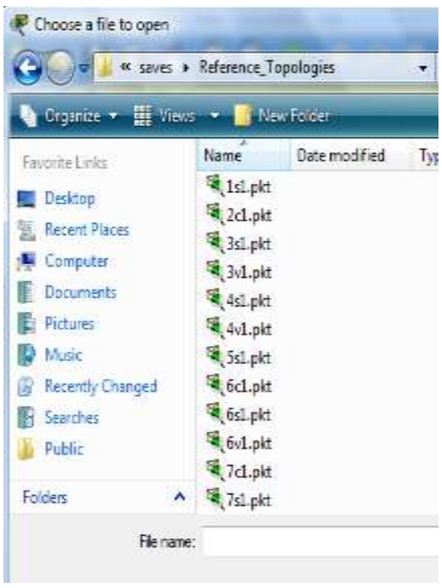




Opening Existing PT Topologies







3) Stage v (verify)

Home Activities:

Activity 1:

Make topologies in packet tracer and provide connectivity:

1) Point-to-point
2) Bus
   a. Linear bus
   b. Distributed bus
3) Star
   a. Extended star
   b. Distributed Star
4) Ring
5) Mesh
   a. Fully connected network
   b. Partially connected network
6) Hybrid

4) Stage a2 (assess)

Submit the home activity before next lab

Statement Purpose:

- Explore several aspects of the HTTP protocol: the basic GET/response interaction, HTTP message formats, retrieving large HTML files, retrieving HTML files with embedded objects, and HTTP authentication and security.

Activity outcomes:

- Students will gain better understanding of the HTTP protocol.

Instructor Note:

- In this first Wireshark lab, you'll captures some http packets using wire shark and make some observations on them.

1) Stage J(Journey)

<u>Introduction</u>

- Having gotten our feet wet with the Wireshark packet sniffer in the introductory lab, we're now ready to use Wireshark to investigate protocols in operation. In this lab, we'll explore several aspects of the HTTP protocol: the basic GET/response interaction, HTTP message formats, retrieving large HTML files, retrieving HTML files with embedded objects, and HTTP authentication and security.

2) Stage a1 (apply)

<u>Lab Activities:</u>

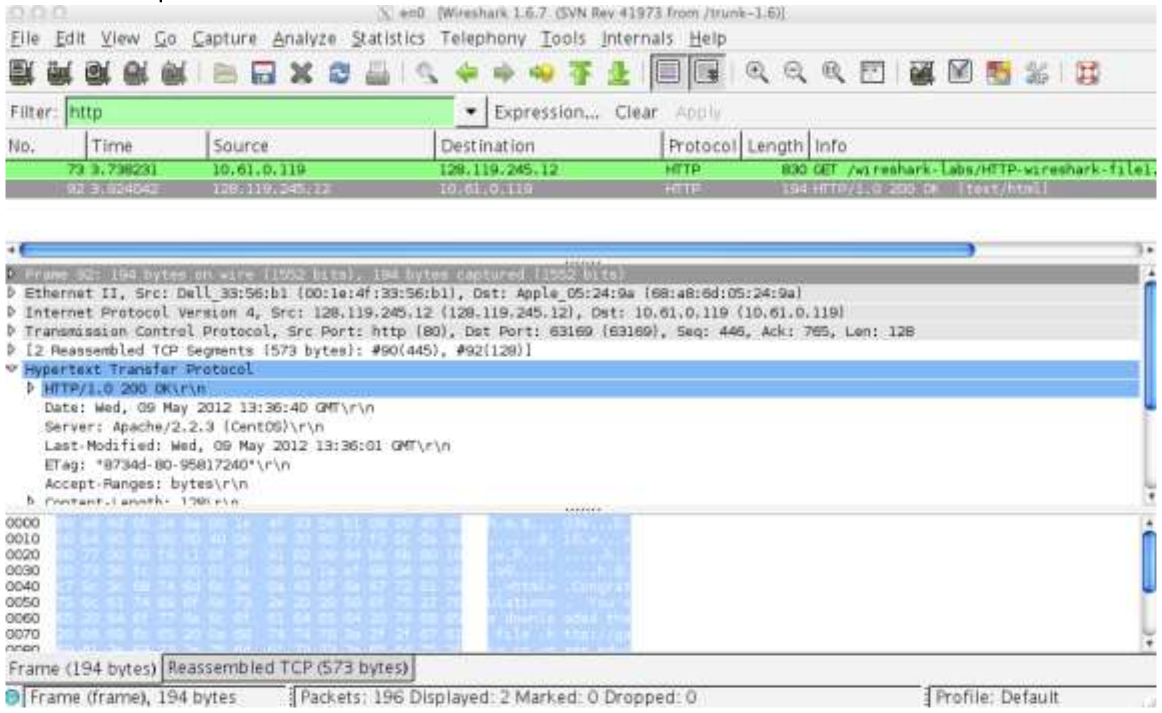
Activity 1:

The Basic HTTP GET/response interaction

Let's begin our exploration of HTTP by downloading a very simple HTML file - one that is very short, and contains no embedded objects. Do the following:

1. Start up your web browser.
2. Start up the Wireshark packet sniffer, as described in the Introductory lab (but don't yet begin packet capture). Enter "http" (just the letters, not the quotation marks) in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packets).
3. Wait a bit more than one minute (we'll see why shortly), and then begin Wireshark packet capture.
4. Enter the following to your browser
   <u>http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html</u>
   Your browser should display the very simple, one-line HTML file.
5. Stop Wireshark packet capture.

Your Wireshark window should look similar to the window shown in Figure 1. If you are unable to run Wireshark on a live network connection, you can download a packet trace that was created when the steps above were followed.[2]


en0 [Wireshark 1.6.7 (SVN Rev 41973 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: http
Expression... Clear Apply
Frame (194 bytes) Reassembled TCP (573 bytes)

```
No.  Time      Source          Destination     Protocol Length Info
  73 3.738231  10.61.0.119     128.119.245.12  HTTP     830 GET /wireshark-labs/HTTP-wireshark-file1.
  92 3.824042  128.119.245.12  10.61.0.119     HTTP     194 HTTP/1.0 200 OK  (text/html)

▷ Frame 92: 194 bytes on wire (1552 bits), 194 bytes captured (1552 bits)
▷ Ethernet II, Src: Dell_33:56:b1 (00:1e:4f:33:56:b1), Dst: Apple_05:24:9a (68:a8:6d:05:24:9a)
▷ Internet Protocol Version 4, Src: 128.119.245.12 (128.119.245.12), Dst: 10.61.0.119 (10.61.0.119)
▷ Transmission Control Protocol, Src Port: http (80), Dst Port: 63169 (63169), Seq: 446, Ack: 765, Len: 128
▷ [2 Reassembled TCP Segments (573 bytes): #90(445), #92(128)]
▽ Hypertext Transfer Protocol
  ▷ HTTP/1.0 200 OK\r\n
    Date: Wed, 09 May 2012 13:36:40 GMT\r\n
    Server: Apache/2.2.3 (CentOS)\r\n
    Last-Modified: Wed, 09 May 2012 13:36:01 GMT\r\n
    ETag: "8734d-80-95817240"\r\n
    Accept-Ranges: bytes\r\n
  ▷ Content-Length: 128\r\n

0000
0010
0020
0030
0040
0050
0060
0070

Frame (frame), 194 bytes   Packets: 196 Displayed: 2 Marked: 0 Dropped: 0   Profile: Default
```

Figure 1: Wireshark Display after http://gaia.cs.umass.edu/wireshark-labs/ HTTP-wireshark-file1.html has been retrieved by your browser

The example in Figure 1 shows in the packet-listing window that two HTTP messages were captured: the GET message (from your browser to the gaia.cs.umass.edu web server) and the response message from the server to your browser. The packet-contents window shows details of the selected message (in this case the HTTP OK message, which is highlighted in the packet-listing window). Recall that since the HTTP message was carried inside a TCP segment, which was carried inside an IP datagram, which was carried within an Ethernet frame, Wireshark displays the Frame, Ethernet, IP, and TCP packet information as well. We want to minimize the amount of non-HTTP data displayed (we're interested in HTTP here, and will be investigating these other protocols is later labs), so make sure the boxes at the far left of the Frame, Ethernet, IP and TCP information have a plus sign or a right-pointing triangle (which means there is hidden, undisplayed information), and the HTTP line has a minus sign or a down-pointing triangle (which means that all information about the HTTP message is displayed).

(*Note:* You should ignore any HTTP GET and response for favicon.ico. If you see a reference to this file, it is your browser automatically asking the server if it (the server) has a small icon file that should

[2] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file http-ethereal-trace-1. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the http-ethereal-trace-1 trace file. The resulting display should look similar to Figure 1. (The Wireshark user interface displays just a bit differently on different operating systems, and in different versions of Wireshark).

be displayed next to the displayed URL in your browser. We'll ignore references to this pesky file in this lab.).

By looking at the information in the HTTP GET and response messages, answer the following questions. When answering the following questions, you should print out the GET and response messages (see the introductory Wireshark lab for an explanation of how to do this) and indicate where in the message you've found the information that answers the following questions. When you hand in your assignment, annotate the output so that it's clear where in the output you're getting the information for your answer (e.g., for our classes, we ask that students markup paper copies with a pen, or annotate electronic copies with text in a colored font).

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?
2. What languages (if any) does your browser indicate that it can accept to the server?
3. What is the IP address of your computer? Of the gaia.cs.umass.edu server?
4. What is the status code returned from the server to your browser?
5. When was the HTML file that you are retrieving last modified at the server?
6. How many bytes of content are being returned to your browser?
7. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

In your answer to question 5 above, you might have been surprised to find that the document you just retrieved was last modified within a minute before you downloaded the document. That's because (for this particular file), the gaia.cs.umass.edu server is setting the file's last-modified time to be the current time, and is doing so once per minute. Thus, if you wait a minute between accesses, the file will appear to have been recently modified, and hence your browser will download a "new" copy of the document.

Solution:

Ans1: Both are running HTTP 1.1

Ans2: Accept-Language: en-us, en

Ans3: My IP address is 192.168.1.46 and the server's is 128.119.245.12

Ans4: HTTP/1.1 200 OK (text/html)

Ans5: Last-Modified: Thu, 07 Jun 2007 22:09:01 GMT

Ans6: Content-Length: 126

Ans7: No all of the headers can be found in the raw data.

Activity 2:

The HTTP CONDITIONAL GET/response interaction

Recall from Section 2.2.6 of the text, that most web browsers perform object caching and thus perform a conditional GET when retrieving an HTTP object. Before performing the steps below, make sure your browser's cache is empty. (To do this under Firefox, select *Tools->Clear Recent History* and check the Cache box, or for Internet Explorer, select *Tools->Internet Options->Delete File;* these actions will remove cached files from your browser's cache.) Now do the following:

- Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browser
  http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html
  Your browser should display a very simple five-line HTML file.
- Quickly enter the same URL into your browser again (or simply select the refresh button on your browser)
- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
- (*Note:* If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-2 packet trace to answer the questions below; see footnote 1. This trace file was gathered while performing the steps above on one of the author's computers.)

Answer the following questions:

8. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?
9. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?
10. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?
11. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

Solution:

Ans8: No.

Ans9: Yes because we can see the contents in the Line-based text data field
Ans10: Yes. The information following is: Thu, 07 Jun 2007 16:29:01 GMT which is the date of the last modification of the file from the previous get request.
Ans11: The status code and phrase returned from the server is HTTP/1.1 304 Not Modified. The server didn't return the contents of the file since the browser loaded it from its cache.

Activity 3:

Retrieving Long Documents

In our examples thus far, the documents retrieved have been simple and short HTML files. Let's next see what happens when we download a long HTML file. Do the following:

- Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browserhttp://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html
  Your browser should display the rather lengthy US Bill of Rights.
- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed.

- (*Note:* If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-3 packet trace to answer the questions below; see footnote 1. This trace file was gathered while performing the steps above on one of the author's computers.)

In the packet-listing window, you should see your HTTP GET message, followed by a multiple-packet TCP response to your HTTP GET request. This multiple-packet response deserves a bit of explanation. Recall from Section 2.2 (see Figure 2.9 in the text) that the HTTP response message consists of a status line, followed by header lines, followed by a blank line, followed by the entity body. In the case of our HTTP GET, the entity body in the response is the *entire* requested HTML file. In our case here, the HTML file is rather long, and at 4500 bytes is too large to fit in one TCP packet. The single HTTP response message is thus broken into several pieces by TCP, with each piece being contained within a separate TCP segment (see Figure 1.24 in the text). In recent versions of Wireshark, Wireshark indicates each TCP segment as a separate packet, and the fact that the single HTTP response was fragmented across multiple TCP packets is indicated by the "TCP segment of a reassembled PDU" in the Info column of the Wireshark display. Earlier versions of Wireshark used the "Continuation" phrase to indicated that the entire content of an HTTP message was broken across multiple TCP segments.. We stress here that there is no "Continuation" message in HTTP!
Answer the following questions:

12. How many HTTP GET request messages did your browser send? Which packet number in the trace contains the GET message for the Bill or Rights?
13. Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?
14. What is the status code and phrase in the response?

How many data-containing TCP segments were needed to carry the single HTTP response and the text of the Bill of Rights?

Solution:

Ans12: There was 1 HTTP GET request message sent by my browser

Ans13: There were 5 data containing TCP segments containing 309 ,1452 ,1452, 1452 and 144 bytes respectively for a total of 4500 bytes.

Ans14: 200 OK

Activity 4:

HTML Documents with Embedded Objects

Now that we've seen how Wireshark displays the captured packet traffic for large HTML files, we can look at what happens when your browser downloads a file with embedded objects, i.e., a file that includes other objects (in the example below, image files) that are stored on another server(s).

Do the following:

- Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browserhttp://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html
  Your browser should display a short HTML file with two images. These two images are referenced in the base HTML file. That is, the images themselves are not contained in the HTML; instead the URLs for the images are contained in the downloaded HTML file. As discussed in the textbook, your browser will have to retrieve these logos from the indicated web sites. Our publisher's logo is retrieved from the www.aw-bc.com web site. The image

of the cover for our 5th edition (one of our favorite covers) is stored at the manic.cs.umass.edu server.

- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed.
- (*Note:* If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-4 packet trace to answer the questions below; see footnote 1. This trace file was gathered while performing the steps above on one of the author's computers.)

Answer the following questions:

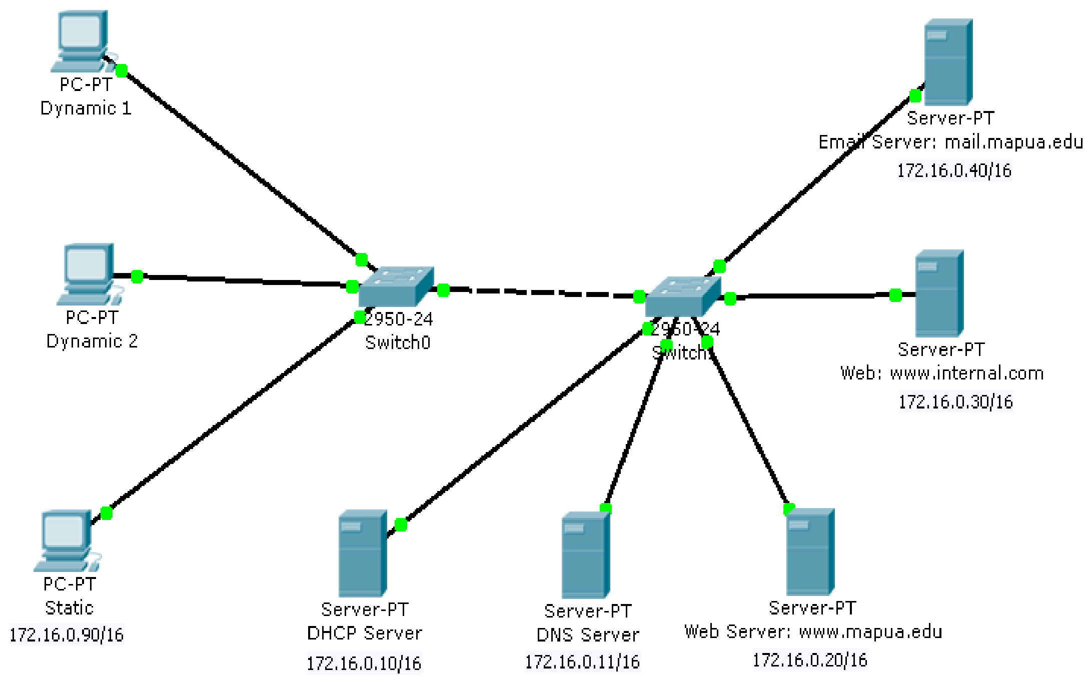
15. How many HTTP GET request messages did your browser send? To which Internet addresses were these GET requests sent?
16. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two web sites in parallel? Explain.

Solution:

Ans15: There were 3 HTTP GET requests sent to the following Internet addresses: a. 128.119.245.12 b. 128.119.240.90 c. 165.193.123.218

Ans16: By checking the TCP ports we can see if our files were downloaded serially or in parallel. In this case the 2 images were transmitted over 2 TCP connections therefore they were downloaded serially.

Activity 5 (Packet Tracer Part)



Instructions:

1. Start Packet Tracer using **Realtime** mode.

- Options -> Preferences
  - Enable "Show Link Lights"
  - Disable "Hide Device Label"

2. Configuring the www.mapua.edu Web Server

Add a server.

Global Settings:

- Change the Display Name to “**Web Server: www.mapua.edu**”
- Set the Gateway to **172.16.0.1**

FastEthernet:

- Set the IP address to **172.16.0.20**
- Set the Subnet Mask to **255.255.0.0**

DHCP:

- Set the Service to **Off**

DNS:

- Set the Service to **Off**

HTTP

- Set the both the HTTP and HTTPS Service to **On**
- Change the sentence, “<hr> Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.” to “<hr> Welcome to MapuaInstitute’s of Technology’s public web page!” You may add other information as well.

Email:

- Set the SMTP Service and POP3 Service to **Off**

3. Configuring the www.internal.com Web Server

Add a server.

Global Settings:

- Change the Display Name to “**Web Server: www.internal.com**”
- Set the Gateway to **172.16.0.1**

FastEthernet:

- Set the IP address to **172.16.0.30**
- Set the Subnet Mask to **255.255.0.0**

DHCP:

- Set the Service to **Off**

DNS:

- Set the Service to **Off**

HTTP:

- Change the sentence, “<hr> Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.” to “<hr>This is the corporate internal network!” You may add other information as well.

3) Stage v (verify)

Home Activities:

Activity 1:

HTTP Authentication

Finally, let's try visiting a web site that is password-protected and examine the sequence of HTTP message exchanged for such a site. The URL http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html is password protected. The username is "wireshark-students" (without the quotes), and the password is "network" (again, without the quotes). So let's access this "secure" password-protected site. Do the following:

- Make sure your browser's cache is cleared, as discussed above, and close down your browser. Then, start up your browser
- Start up the Wireshark packet sniffer
- Enter the following URL into your browserhttp://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html
  Type the requested user name and password into the pop up box.
- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
- (*Note:* If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-5 packet trace to answer the questions below; see footnote 2. This trace file was gathered while performing the steps above on one of the author's computers.)

Now let's examine the Wireshark output. You might want to first read up on HTTP authentication by reviewing the easy-to-read material on "HTTP Access Authentication Framework" at http://frontier.userland.com/stories/storyReader$2159

Answer the following questions:

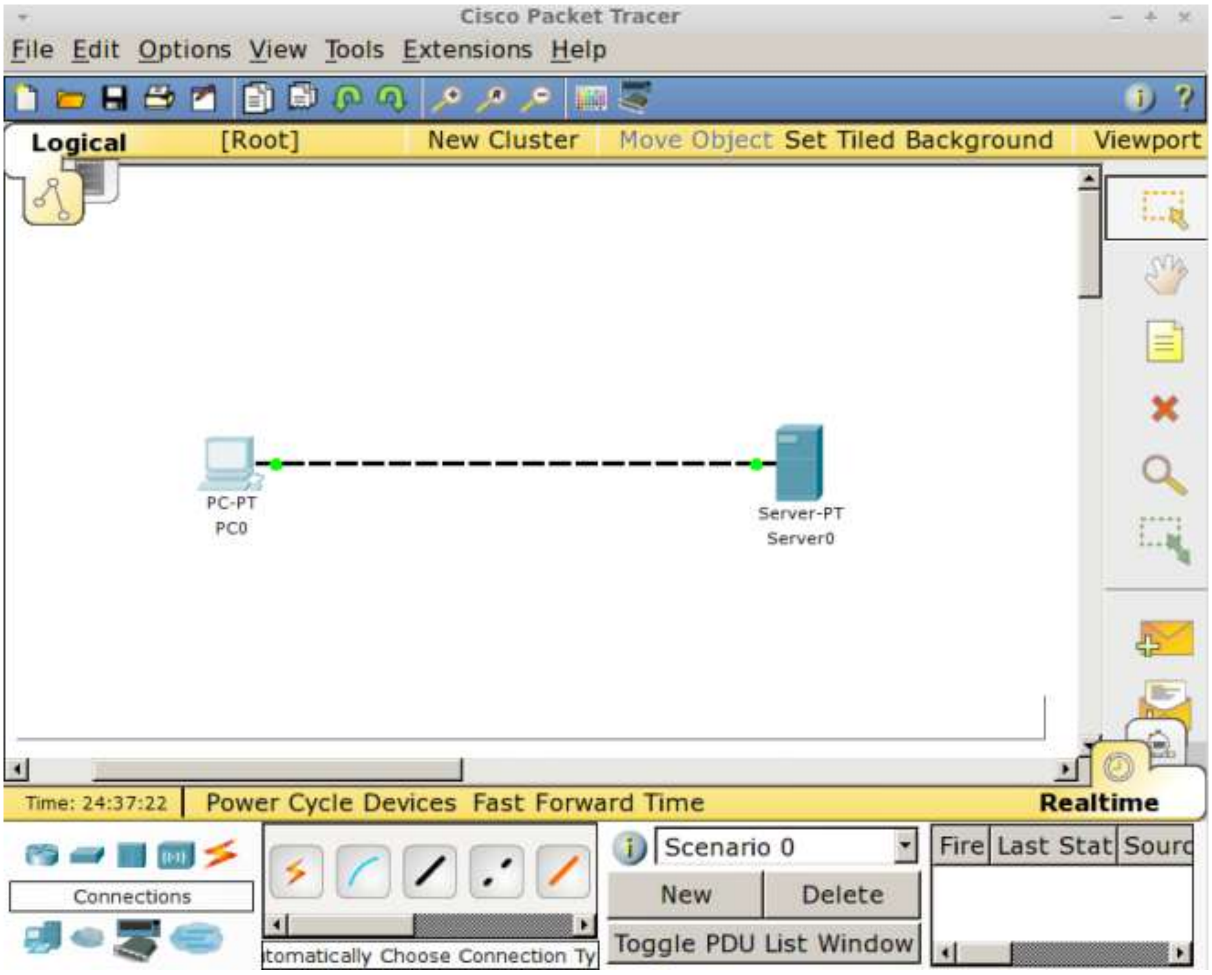
17. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser?
18. When your browser's sends the HTTP GET message for the second time, what new field is included in the HTTP GET message?

The username (wireshark-students) and password (network) that you entered are encoded in the string of characters (d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms=) following the "Authorization: Basic" header in the client's HTTP GET message. While it may appear that your username and password are encrypted, they are simply encoded in a format known as Base64 format. The username and password are *not*encrypted! To see this, go to http://www.motobit.com/util/base64-decoder-encoder.asp and enter the base64-encoded string d2lyZXNoYXJrLXN0dWRlbnRz and decode. *Voila!* You have translated from Base64 encoding to ASCII encoding, and thus should see your username! To view the password, enter the remainder of the string Om5ldHdvcms= and press decode. Since anyone can download a tool like Wireshark and sniff packets (not just their own) passing by their network adaptor, and anyone can translate from Base64 to ASCII (you just did it!), it should be clear to you that simple passwords on WWW sites are not secure unless additional measures are taken.

Activity 2:

Provide web services in the said topology using this information:

1) Set the server ip into “192.168.78.1”
2) Set the DNS “Name” into whatever you wish, mine is “http://www.hesemeleh.com“


Cisco Packet Tracer
File Edit Options View Tools Extensions Help
Logical [Root] New Cluster Move Object Set Tiled Background Viewport
PC-PT
PC0
Server-PT
Server0
Time: 24:37:22 Power Cycle Devices Fast Forward Time Realtime
Connections
Scenario 0
New Delete
Toggle PDU List Window
Fire Last Stat Sourc
tomatically Choose Connection Ty

4) Stage a2 (assess)

Submit the home activity before next lab

Statement Purpose:

- To take a closer look at the client side of DNS. While execution of a DNS query, much can go on “under the covers,” invisible to the DNS clients, as the hierarchical DNS servers communicate with each other to either recursively or iteratively resolve the client’s DNS query.

Activity outcomes:

- Students will be able to better understand the working of DNS

Instructor Note:

- In this first Wireshark lab, you’ll captures some DNS packets using wireshark and make some observations on them.

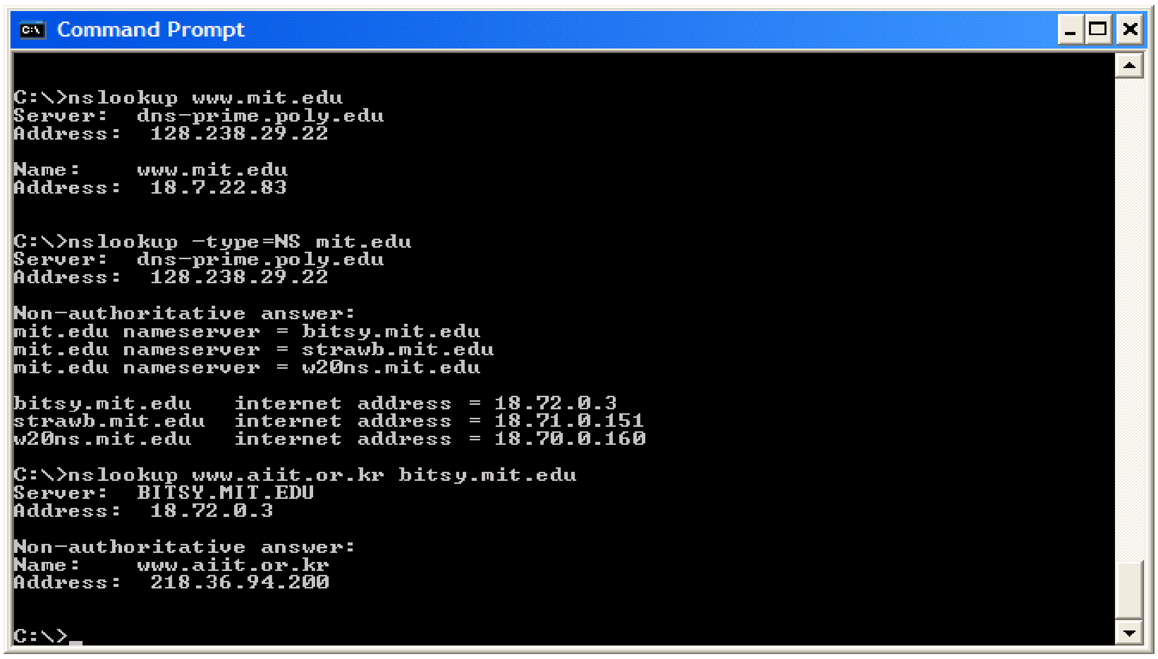
1) StageJ(Journey)

Introduction

nslookup

In this lab, we’ll make extensive use of the *nslookup* tool, which is available in most Linux/Unix and Microsoft platforms today. To run *nslookup* in Linux/Unix, you just type the *nslookup* command on the command line. To run it in Windows, open the Command Prompt and run *nslookup* on the command line.

In it is most basic operation, *nslookup* tool allows the host running the tool to query any specified DNS server for a DNS record. The queried DNS server can be a root DNS server, a top-level-domain DNS server, an authoritative DNS server, or an intermediate DNS server (see the textbook for definitions of these terms). To accomplish this task, *nslookup* sends a DNS query to the specified DNS server, receives a DNS reply from that same DNS server, and displays the result.



```
C:\>nslookup www.mit.edu
Server:  dns-prime.poly.edu
Address:  128.238.29.22

Name:    www.mit.edu
Address:  18.7.22.83


C:\>nslookup -type=NS mit.edu
Server:  dns-prime.poly.edu
Address:  128.238.29.22

Non-authoritative answer:
mit.edu nameserver = bitsy.mit.edu
mit.edu nameserver = strawb.mit.edu
mit.edu nameserver = w20ns.mit.edu

bitsy.mit.edu   internet address = 18.72.0.3
strawb.mit.edu  internet address = 18.71.0.151
w20ns.mit.edu   internet address = 18.70.0.160

C:\>nslookup www.aiit.or.kr bitsy.mit.edu
Server:  BITSY.MIT.EDU
Address:  18.72.0.3

Non-authoritative answer:
Name:    www.aiit.or.kr
Address:  218.36.94.200


C:\>_
```

The above screenshot shows the results of three independent *nslookup* commands (displayed in the Windows Command Prompt). In this example, the client host is located on the campus of Polytechnic University in Brooklyn, where the default local DNS server is dns-prime.poly.edu. When running *nslookup*, if no DNS server is specified, then *nslookup* sends the query to the default DNS server, which in this case is dns-prime.poly.edu. Consider the first command:

```
nslookup www.mit.edu
```

In words, this command is saying “please send me the IP address for the host www.mit.edu”. As shown in the screenshot, the response from this command provides two pieces of information: (1) the name and IP address of the DNS server that provides the answer; and (2) the answer itself, which is the host name and IP address of www.mit.edu. Although the response came from the local DNS server at Polytechnic University, it is quite possible that this local DNS server iteratively contacted several other DNS servers to get the answer, as described in Section 2.5 of the textbook.
Now consider the second command:

```
nslookup –type=NS mit.edu
```

In this example, we have provided the option “-type=NS” and the domain “mit.edu”. This causes *nslookup* to send a query for a type-NS record to the default local DNS server. In words, the query is saying, “please send me the host names of the authoritative DNS for mit.edu”. (When the –type option is not used, *nslookup* uses the default, which is to query for type A records.) The answer, displayed in the above screenshot, first indicates the DNS server that is providing the answer (which is the default local DNS server) along with three MIT nameservers. Each of these servers is indeed an authoritative DNS server for the hosts on the MIT campus. However, *nslookup* also indicates that the answer is “non-authoritative,” meaning that this answer came from the cache of some server rather than from an authoritative MIT DNS server. Finally, the answer also includes the IP addresses of the authoritative DNS servers at MIT. (Even though the type-NS query generated by *nslookup* did not explicitly ask for the IP addresses, the local DNS server returned these “for free” and *nslookup* displays the result.)
Now finally consider the third command:

```
nslookup www.aiit.or.kr bitsy.mit.edu
```

In this example, we indicate that we want to the query sent to the DNS server bitsy.mit.edu rather than to the default DNS server (dns-prime.poly.edu). Thus, the query and reply transaction takes place directly between our querying host and bitsy.mit.edu. In this example, the DNS server bitsy.mit.edu provides the IP address of the host www.aiit.or.kr, which is a web server at the Advanced Institute of Information Technology (in Korea).
Now that we have gone through a few illustrative examples, you are perhaps wondering about the general syntax of *nslookup* commands. The syntax is:

```
nslookup –option1 –option2 host-to-find dns-server
```

In general, *nslookup* can be run with zero, one, two or more options. And as we have seen in the above examples, the dns-server is optional as well; if it is not supplied, the query is sent to the default local DNS server.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

Now that we have provided an overview of *nslookup*, it is time for you to test drive it yourself. Do the following (and write down the results):

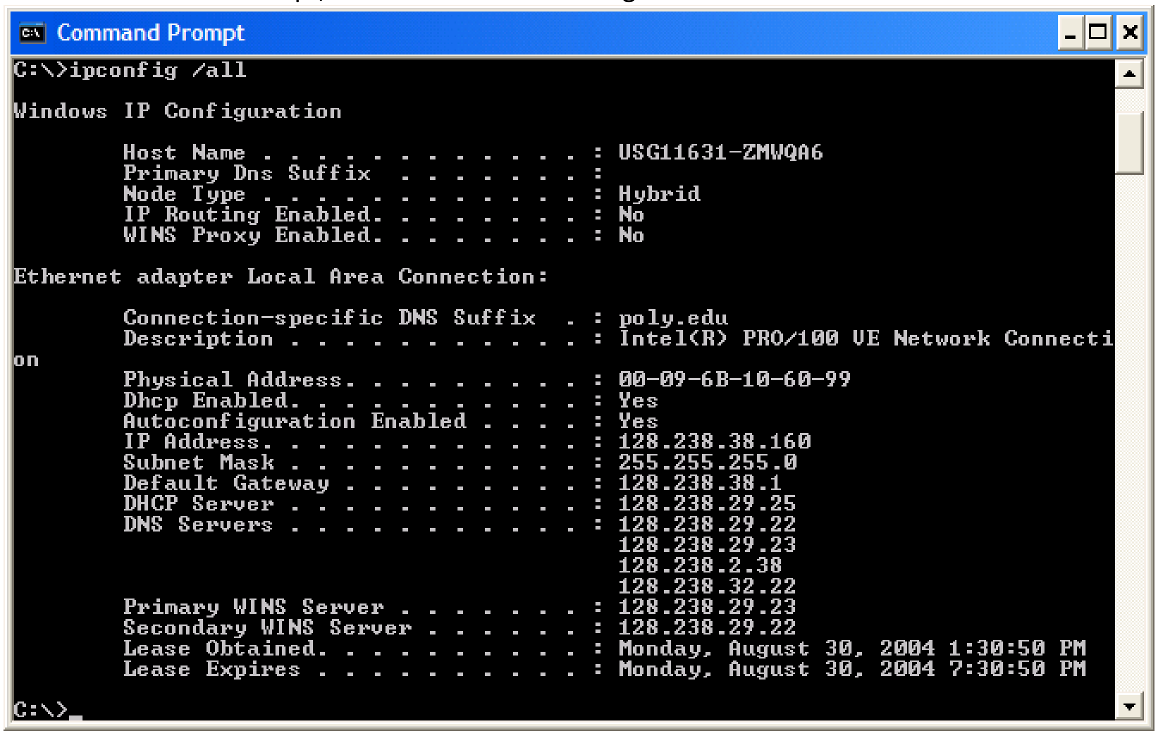
6. Run *nslookup* to obtain the IP address of a Web server in Asia. What is the IP address of that server?
7. Run *nslookup* to determine the authoritative DNS servers for a university in Europe.
8. Run *nslookup* so that one of the DNS servers obtained in Question 2 is queried for the mail servers for Yahoo! mail. What is its IP address?

Solution:

Ans1: I performed nslookup for www.rediff.com. Its IP address is 208.184.138.70
Ans2: I performed nslookup for a European University in Ioannina Greece. Its IP address is 128.238.29.22
Ans3: the IP address of the mail server(s) is 18.72.0.3.

ipconfig

ipconfig (for Windows) and *ifconfig* (for Linux/Unix) are among the most useful little utilities in your host, especially for debugging network issues. Here we'll only describe *ipconfig*, although the Linux/Unix *ifconfig* is very similar. *ipconfig* can be used to show your current TCP/IP information, including your address, DNS server addresses, adapter type and so on. For example, if you all this information about your host simply by entering
ipconfig \all
into the Command Prompt, as shown in the following screenshot.



```
C:\>ipconfig /all

Windows IP Configuration

        Host Name . . . . . . . . . . . . : USG11631-ZMWQA6
        Primary Dns Suffix  . . . . . . . :
        Node Type . . . . . . . . . . . . : Hybrid
        IP Routing Enabled. . . . . . . . : No
        WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : poly.edu
        Description . . . . . . . . . . . : Intel(R) PRO/100 VE Network Connecti
on
        Physical Address. . . . . . . . . : 00-09-6B-10-60-99
        Dhcp Enabled. . . . . . . . . . . : Yes
        Autoconfiguration Enabled . . . . : Yes
        IP Address. . . . . . . . . . . . : 128.238.38.160
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 128.238.38.1
        DHCP Server . . . . . . . . . . . : 128.238.29.25
        DNS Servers . . . . . . . . . . . : 128.238.29.22
                                            128.238.29.23
                                            128.238.2.38
                                            128.238.32.22
        Primary WINS Server . . . . . . . : 128.238.29.23
        Secondary WINS Server . . . . . . : 128.238.29.22
        Lease Obtained. . . . . . . . . . : Monday, August 30, 2004 1:30:50 PM
        Lease Expires . . . . . . . . . . : Monday, August 30, 2004 7:30:50 PM

C:\>_
```

ipconfig is also very useful for managing the DNS information stored in your host. In Section 2.5 we learned that a host can cache DNS records it recently obtained. To see these cached records, after the prompt C:\> provide the following command:
ipconfig /displaydns
Each entry shows the remaining Time to Live (TTL) in seconds. To clear the cache, enter
ipconfig /flushdns
Flushing the DNS cache clears all entries and reloads the entries from the hosts file

Activity 2:

Now that we are familiar with *nslookup* and *ipconfig*, we're ready to get down to some serious business. Let's first capture the DNS packets that are generated by ordinary Web-surfing activity.

- Use *ipconfig* to empty the DNS cache in your host.
- Open your browser and empty your browser cache. (With Internet Explorer, go to Tools menu and select Internet Options; then in the General tab select Delete Files.)
- Open Wireshark and enter "ip.addr == your_IP_address" into the filter, where you obtain your_IP_address with ipconfig. This filter removes all packets that neither originate nor are destined to your host.
- Start packet capture in Wireshark.
- With your browser, visit the Web page: http://www.ietf.org
- Stop packet capture.

If you are unable to run Wireshark on a live network connection, you can download a packet trace file that was captured while following the steps above on one of the author's computers[3]. Answer the following questions. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[4] to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

1. Locate the DNS query and response messages. Are then sent over UDP or TCP?
2. What is the destination port for the DNS query message? What is the source port of DNS response message?
3. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?
4. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
5. Examine the DNS response message. How many "answers" are provided? What do each of these answers contain?
6. Consider the subsequent TCP SYN packet sent by your host. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?
7. This web page contains images. Before retrieving each image, does your host issue new DNS queries?

Solution:

Ans1: They are sent over UDP
Ans2: The destination port for the DNS query is 53 and the source port of the DNS response is 53.
Ans3: It's sent to 192.168.1.1, which is the IP address of one of my local DNS servers.

.

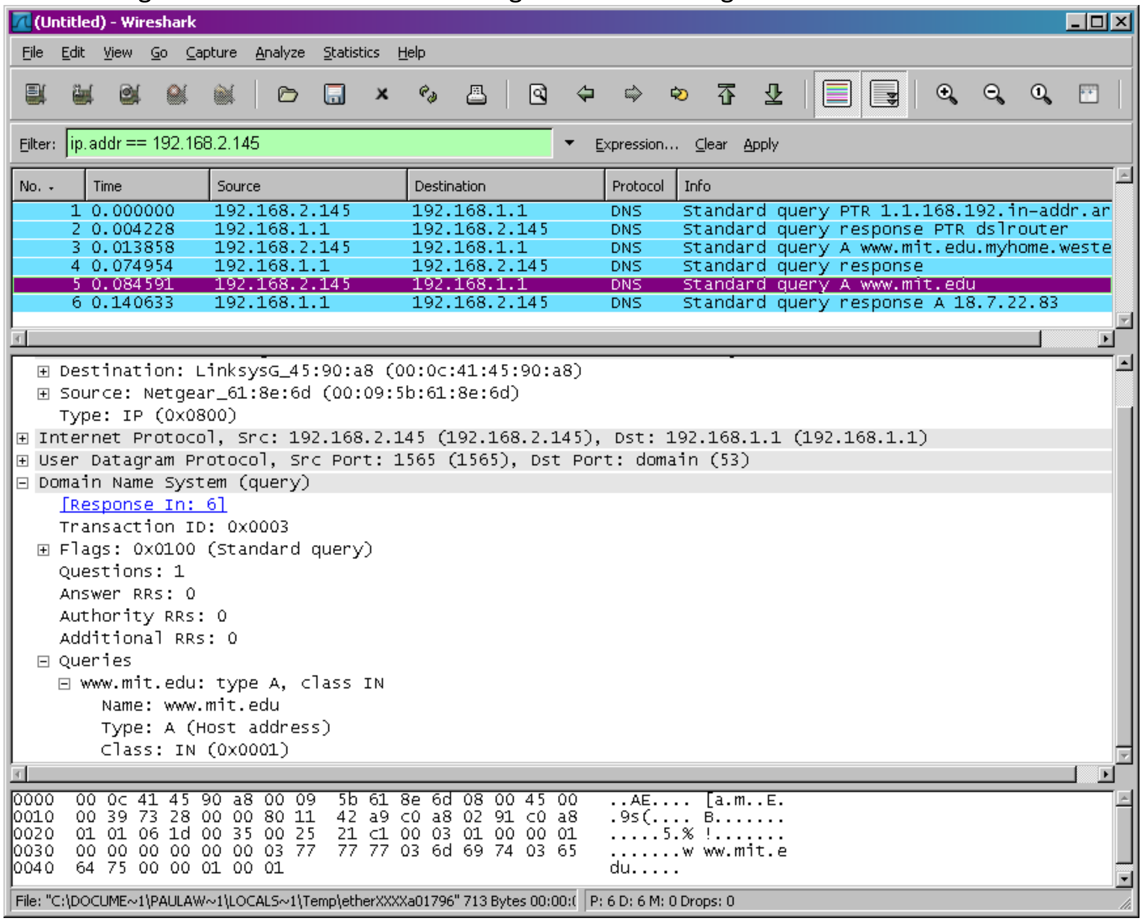
Ans4: It's a type A Standard Query and it doesn't contain any answers.
Ans5: : There were 2 answers containing information about the name of the host, the type of address, class, the TTL, the data length and the IP address.
Answers www.ietf.org: type A, class IN, addr 209.173.57.180
Name:
www.ietf.org
Type: A (Host address)
Class: IN (0x0001)
Time to live: 30 minutes
Data length: 4
Addr: 209.173.57.180
www.ietf.org: type A, class IN, addr 209.173.53.180
Name:
www.ietf.org
Type: A (Host address) Class: IN (0x0001)
Time to live: 30 minutes
Data length: 4
Addr: 209.173.53.180
Ans6: The first SYN packet was sent to 209.173.57.180 which corresponds to the first IP address provided in the DNS response message.
Ans7: No

Activity 3:

Now let's play with *nslookup*[5].

- Start packet capture.
- Do an *nslookup* on www.mit.edu
- Stop packet capture.

You should get a trace that looks something like the following:


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: ip.addr == 192.168.2.145
Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.2.145	192.168.1.1	DNS	Standard query PTR 1.1.168.192.in-addr.ar
2	0.004228	192.168.1.1	192.168.2.145	DNS	Standard query response PTR dslrouter
3	0.013858	192.168.2.145	192.168.1.1	DNS	Standard query A www.mit.edu.myhome.weste
4	0.074954	192.168.1.1	192.168.2.145	DNS	Standard query response
5	0.084591	192.168.2.145	192.168.1.1	DNS	Standard query A www.mit.edu
6	0.140633	192.168.1.1	192.168.2.145	DNS	Standard query response A 18.7.22.83

```
  ⊞ Destination: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
  ⊞ Source: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
    Type: IP (0x0800)
⊞ Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 192.168.1.1 (192.168.1.1)
⊞ User Datagram Protocol, Src Port: 1565 (1565), Dst Port: domain (53)
⊟ Domain Name System (query)
    [Response In: 6]
    Transaction ID: 0x0003
  ⊞ Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊟ www.mit.edu: type A, class IN
        Name: www.mit.edu
        Type: A (Host address)
        Class: IN (0x0001)
```

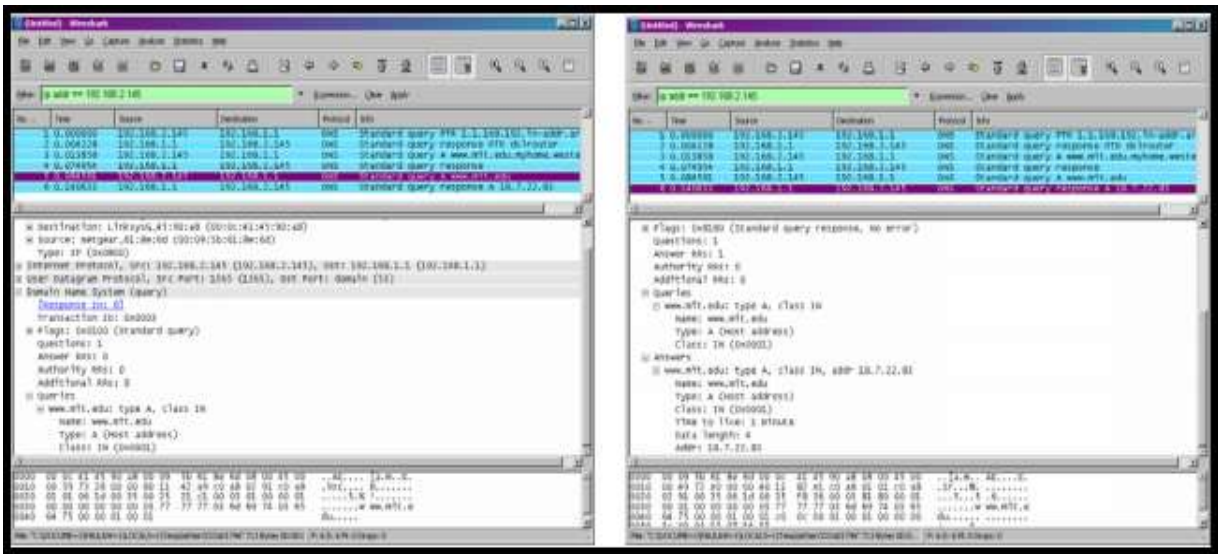
```
0000  00 0c 41 45 90 a8 00 09  5b 61 8e 6d 08 00 45 00   ..AE.... [a.m..E.
0010  00 39 73 28 00 00 80 11  42 a9 c0 a8 02 91 c0 a8   .9s(.... B.......
0020  01 01 06 1d 00 35 00 25  21 c1 00 03 01 00 00 01   .....5.% !.......
0030  00 00 00 00 00 00 03 77  77 77 03 6d 69 74 03 65   .......w ww.mit.e
0040  64 75 00 00 01 00 01                               du.....
```

File: "C:\DOCUME~1\PAULAW~1\LOCALS~1\Temp\etherXXXXa01796" 713 Bytes 00:00:(P: 6 D: 6 M: 0 Drops: 0

We see from the above screenshot that *nslookup* actually sent three DNS queries and received three DNS responses. For the purpose of this assignment, in answering the following questions, ignore the first two sets of queries/responses, as they are specific to *nslookup* and are not normally generated by standard Internet applications. You should instead focus on the last query and response messages.

1. What is the destination port for the DNS query message? What is the source port of DNS response message?
2. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?
3. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?
4. Examine the DNS response message. How many “answers” are provided? What do each of these answers contain?
5. Provide a screenshot.

Solution:



```
Filter: ip.addr == 192.168.2.145
No. Time      Source         Destination    Protocol Info
  1 0.000000  192.168.2.145  192.168.1.1    DNS      Standard query PTR 1.1.168.192.in-addr.arpa
  2 0.004228  192.168.1.1    192.168.2.145  DNS      Standard query response PTR dslrouter
  3 0.013858  192.168.2.145  192.168.1.1    DNS      Standard query A www.mit.edu.myhome.westell
  4 0.074954  192.168.1.1    192.168.2.145  DNS      Standard query response
  5 [illegible] 192.168.2.145 192.168.1.1   DNS      Standard query A www.mit.edu
  6 0.180833  192.168.1.1    192.168.2.145  DNS      Standard query response A 18.7.22.83

  Destination: LinksysG_41:92:a8 (00:0c:41:45:92:a8)
  Source: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
  Type: IP (0x0800)
Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 1165 (1165), Dst Port: domain (53)
Domain Name System (query)
   [Response In: 6]
   Transaction ID: 0x0003
  Flags: 0x0100 (Standard query)
   Questions: 1
   Answer RRs: 0
   Authority RRs: 0
   Additional RRs: 0
  Queries
   www.mit.edu: type A, class IN
      Name: www.mit.edu
      Type: A (Host address)
      Class: IN (0x0001)
```

```
Filter: ip.addr == 192.168.2.145
No. Time      Source         Destination    Protocol Info
  1 0.000000  192.168.2.145  192.168.1.1    DNS      Standard query PTR 1.1.168.192.in-addr.arpa
  2 0.004228  192.168.1.1    192.168.2.145  DNS      Standard query response PTR dslrouter
  3 0.013858  192.168.2.145  192.168.1.1    DNS      Standard query A www.mit.edu.myhome.westell
  4 0.074954  192.168.1.1    192.168.2.145  DNS      Standard query response
  5 0.088581  192.168.2.145  192.168.1.1    DNS      Standard query A www.mit.edu
  6 [illegible] 192.168.1.1  192.168.2.145  DNS      Standard query response A 18.7.22.83

  Flags: 0x8180 (Standard query response, No error)
   Questions: 1
   Answer RRs: 1
   Authority RRs: 0
   Additional RRs: 0
  Queries
   www.mit.edu: type A, class IN
      Name: www.mit.edu
      Type: A (Host address)
      Class: IN (0x0001)
  Answers
   www.mit.edu: type A, class IN, addr 18.7.22.83
      Name: www.mit.edu
      Type: A (Host address)
      Class: IN (0x0001)
      Time to live: 1 minute
      Data length: 4
      Addr: 18.7.22.83
```

Ans1: The destination port of the DNS query is 53 and the source port of the DNS response is 53.
Ans2: It's sent to 192.168.1.1 which as we can see from the ipconfig –all screenshot, is the default local DNS server.
Ans3: The query is of type A and it doesn't contain any answers.
Ans4: The response DNS message contains one answer containing the name of the host, the type of address, the class, and the IP address. Answers

www.mit.edu:
type A,
class IN, addr 18.7.22.83
Name:
www.mit.edu Type: A (Host address) Class: IN (0x0001) Time to live: 1 minute Data length: 4 Addr: 18.7.22.83

Ans5: Screen shots provided

Activity 4:

Now repeat the previous experiment, but instead issue the command:
nslookup –type=NS mit.edu
Answer the following questions[6] :

1. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?
2. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
3. Examine the DNS response message. What MIT nameservers does the response message provide? Does this response message also provide the IP addresses of the MIT namesers?
4. Provide a screenshot.

Solution:

Ans1: It was sent to 128.238.29.22 which is my default DNS server.
Ans2: It's a type NS DNS query that doesn't contain any answers.

Ans3: The nameservers are bitsy, strawb and w20ns. We can find their IP addresses if we expand the Additional records field in Wireshark as seen below. Answers mit.edu: type NS, class inet, ns bitsy.mit.edu mit.edu: type NS, class inet, ns strawb.mit.edu mit.edu: type NS, class inet, ns w20ns.mit.edu Additional records bitsy.mit.edu: type A, class inet, addr 18.72.0.3 strawb.mit.edu: type A, class inet, addr 18.71.0.151 w20ns.mit.edu: type A, class inet, addr 18.70.0.160

3) Stage v (verify)

Home Activities:

Activity 1:

Repeat the previous experiment, but instead issue the command:
nslookup www.aiit.or.kr bitsy.mit.edu
Answer the following questions[7]:

- To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server? If not, what does the IP address correspond to?
- Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
- Examine the DNS response message. How many "answers" are provided? What does each of these answers contain?
- Provide a screenshot.

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 05

Statement Purpose:

- Investigate the behavior of the celebrated TCP protocol in detail
- Analyze a trace of the TCP segments sent and received in transferring a 150KB file from your computer to a remote server.
- Study TCP's use of sequence and acknowledgement numbers for providing reliable data transfer
- Study TCP's congestion control algorithm – slow start and congestion avoidance – in action; and we'll look at TCP's receiver-advertised flow control mechanism.
- Study TCP connection setup and investigate the performance (throughput and round-trip time) of the TCP connection between student's computer and the server.

Activity outcomes:

- Students will gain better understanding of the TCP protocol

Instructor Note:

- In this first Wireshark lab, you'll captures some TCP packets using wireshark and make some observations on them.

1) StageJ(Journey)

Introduction

Capturing a bulk TCP transfer from your computer to a remote server Before beginning our exploration of TCP, we'll need to use Wireshark to obtain a packet trace of the TCP transfer of a file from your computer to a remote server. You'll do so by accessing a Web page that will allow you to enter the name of a file stored on your computer (which contains the ASCII text of *Alice in Wonderland*), and then transfer the file to a Web server using the HTTP POST method (see section 2.2.3 in the text). We're using the POST method rather than the GET method as we'd like to transfer a large amount of data *from* your computer to another computer. Of course, we'll be running Wireshark during this time to obtain the trace of the TCP segments sent and received from your computer.

2) Stage a1 (apply)

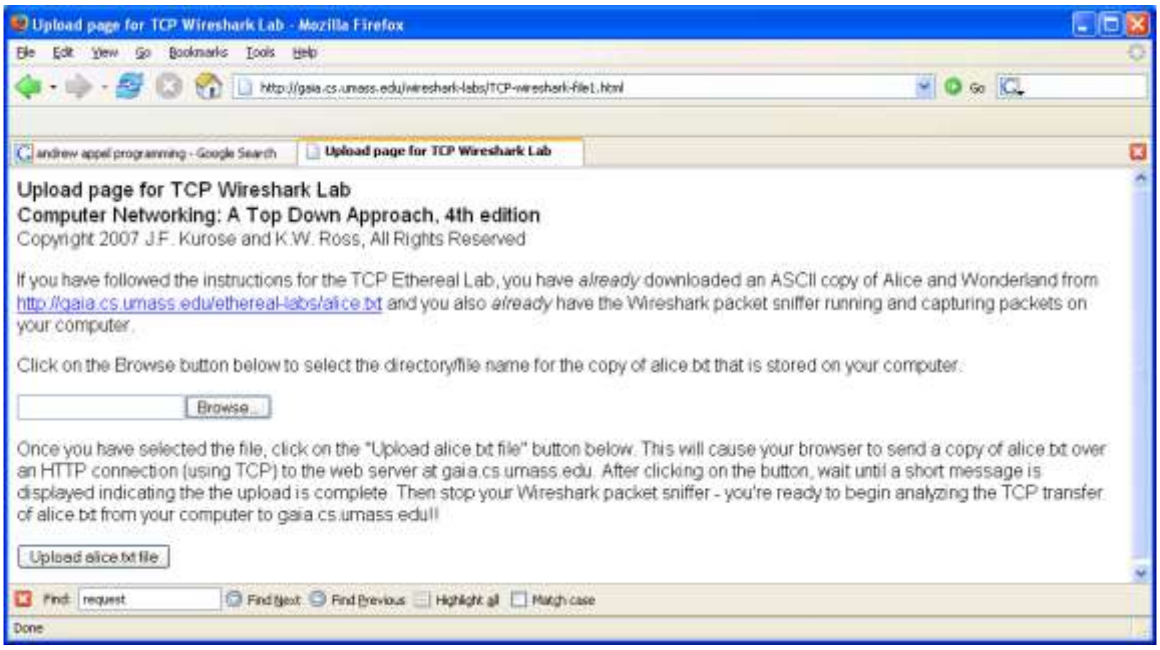
Lab Activities:

Activity 1:

1. **Examples**

Do the following:

- Start up your web browser. Go the http://gaia.cs.umass.edu/wireshark-labs/alice.txt and retrieve an ASCII copy of *Alice in Wonderland.* Store this file somewhere on your computer.
- Next go to http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html.
- You should see a screen that looks like:


Upload page for TCP Wireshark Lab - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html
andrew appel programming - Google Search
Upload page for TCP Wireshark Lab
Browse...
Upload alice.txt file
Find: request
Find Next
Find Previous
Highlight all
Match case
Done

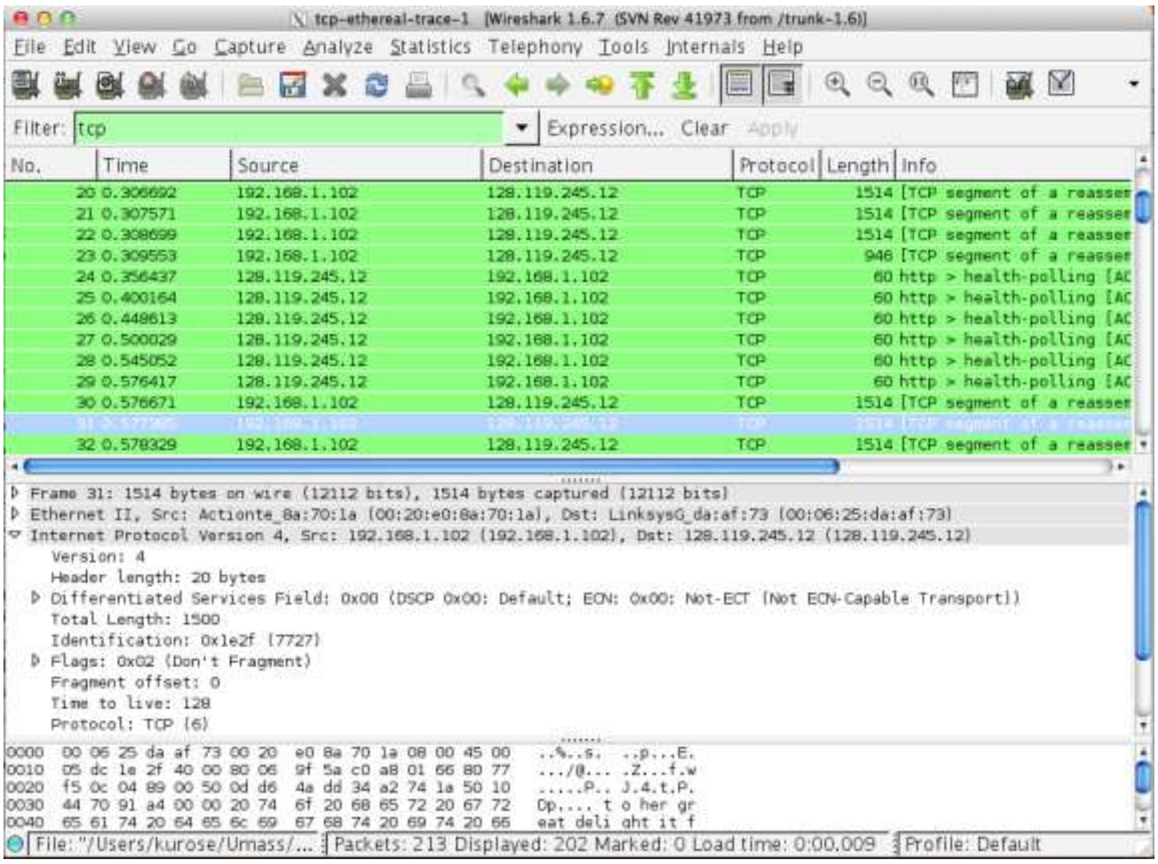
Upload page for TCP Wireshark Lab
Computer Networking: A Top Down Approach, 4th edition
Copyright 2007 J.F. Kurose and K.W. Ross, All Rights Reserved

If you have followed the instructions for the TCP Ethereal Lab, you have *already* downloaded an ASCII copy of Alice and Wonderland from http://gaia.cs.umass.edu/ethereal-labs/alice.txt and you also *already* have the Wireshark packet sniffer running and capturing packets on your computer.

Click on the Browse button below to select the directory/file name for the copy of alice.txt that is stored on your computer.

Once you have selected the file, click on the "Upload alice.txt file" button below. This will cause your browser to send a copy of alice.txt over an HTTP connection (using TCP) to the web server at gaia.cs.umass.edu. After clicking on the button, wait until a short message is displayed indicating the the upload is complete. Then stop your Wireshark packet sniffer - you're ready to begin analyzing the TCP transfer of alice.txt from your computer to gaia.cs.umass.edu!!

- Use the *Browse* button in this form to enter the name of the file (full path name) on your computer containing *Alice in Wonderland* (or do so manually).Don’t yet press the “*Upload alice.txt file*” button.
- Now start up Wireshark and begin packet capture *(Capture->Start)* and then press *OK* on the Wireshark Packet Capture Options screen (we’ll not need to select any options here).
- Returning to your browser, press the “*Upload alice.txt file*” button to upload the file to the gaia.cs.umass.edu server. Once the file has been uploaded, a short congratulations message will be displayed in your browser window.
- Stop Wireshark packet capture. Your Wireshark window should look similar to the window shown below.


tcp-ethereal-trace-1 [Wireshark 1.6.7 (SVN Rev 41973 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: tcp
Expression... Clear Apply

```
No.   Time       Source           Destination      Protocol Length Info
   20 0.306692   192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
   21 0.307571   192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
   22 0.308699   192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
   23 0.309553   192.168.1.102    128.119.245.12   TCP       946 [TCP segment of a reasser
   24 0.356437   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   25 0.400164   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   26 0.448613   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   27 0.500029   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   28 0.545052   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   29 0.576417   128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   30 0.576671   192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
   31 0.577[illegible] 192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
   32 0.578329   192.168.1.102    128.119.245.12   TCP      1514 [TCP segment of a reasser
```

```
▷ Frame 31: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits)
▷ Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
▽ Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102), Dst: 128.119.245.12 (128.119.245.12)
    Version: 4
    Header length: 20 bytes
  ▷ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 1500
    Identification: 0x1e2f (7727)
  ▷ Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (6)
```

```
0000  00 06 25 da af 73 00 20  e0 8a 70 1a 08 00 45 00   ..%..s.  ..p...E.
0010  05 dc 1e 2f 40 00 80 06  9f 5a c0 a8 01 66 80 77   .../@... .Z...f.w
0020  f5 0c 04 89 00 50 0d d6  4a dd 34 a2 74 1a 50 10   .....P.. J.4.t.P.
0030  44 70 91 a4 00 00 20 74  6f 20 68 65 72 20 67 72   Dp.... t o her gr
0040  65 61 74 20 64 65 6c 69  67 68 74 20 69 74 20 66   eat deli ght it f
```

```
File: "/Users/kurose/Umass/... | Packets: 213 Displayed: 202 Marked: 0 Load time: 0:00.009 | Profile: Default
```

If you are unable to run Wireshark on a live network connection, you can download a packet trace file that was captured while following the steps above on one of the author's computers[8]. You may well find it valuable to download this trace even if you've captured your own trace and use it, as well as your own trace, when you explore the questions below.

A first look at the captured trace

Before analyzing the behavior of the TCP connection in detail, let's take a high level view of the trace.

- First, filter the packets displayed in the Wireshark window by entering "tcp" (lowercase, no quotes, and don't forget to press return after entering!) into the display filter specification window towards the top of the Wireshark window.

What you should see is series of TCP and HTTP messages between your computer and gaia.cs.umass.edu. You should see the initial three-way handshake containing a SYN message. You should see an HTTP POST message. Depending on the version of Wireshark you are using, you might see a series of "HTTP Continuation" messages being sent from your computer to gaia.cs.umass.edu. Recall from our discussion in the earlier HTTP Wireshark lab, that is no such thing as an HTTP Continuation message – this is Wireshark's way of indicating that there are multiple TCP segments being used to carry a single HTTP message. In more recent versions of Wireshark, you'll see "[TCP

[8] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file tcp-ethereal-trace-1. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the tcp-ethereal-trace-1 trace file.

segment of a reassembled PDU]" in the Info column of the Wireshark display to indicate that this TCP segment contained data that belonged to an upper layer protocol message (in our case here, HTTP). You should also see TCP ACK segments being returned from gaia.cs.umass.edu to your computer.

Answer the following questions, by opening the Wireshark captured packet file *tcp-ethereal-trace-1* in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip (that is download the trace and open that trace in Wireshark; see footnote 2). Whenever possible, when answering a question you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[9] to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

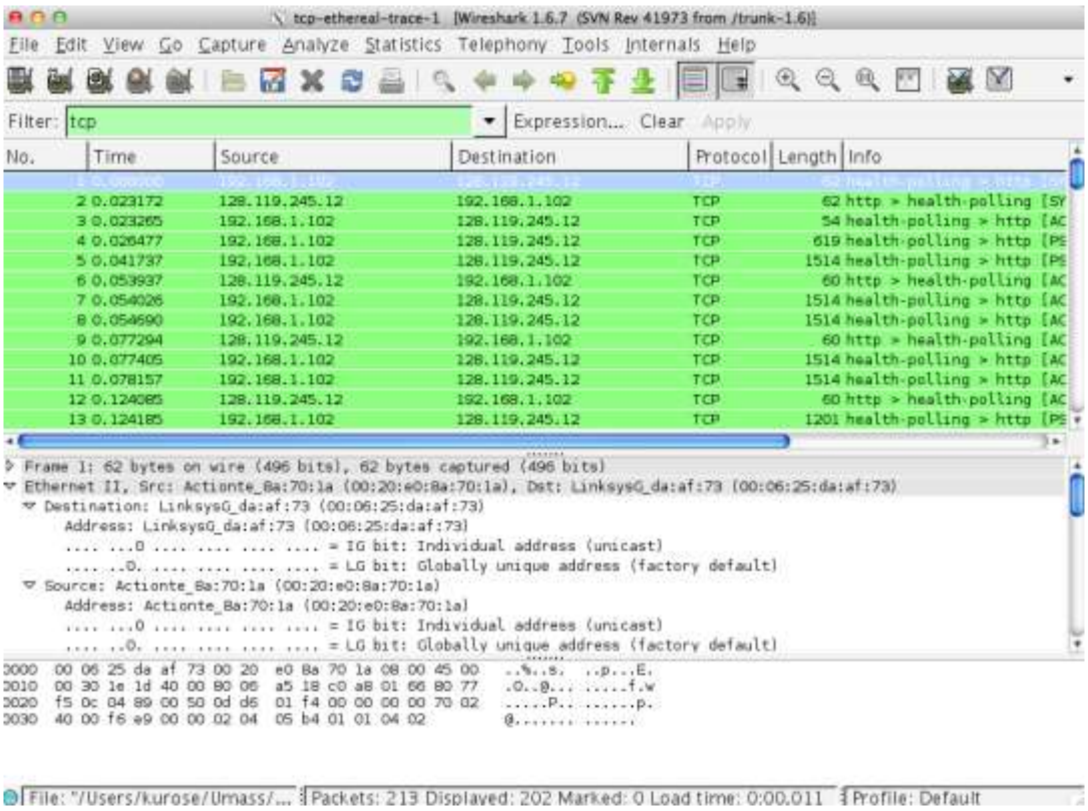
1. What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window" (refer to Figure 2 in the "Getting Started with Wireshark" Lab if you're uncertain about the Wireshark windows.
2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

If you have been able to create your own trace, answer the following question:

3. What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?
4.

Since this lab is about TCP rather than HTTP, let's change Wireshark's "listing of captured packets" window so that it shows information about the TCP segments containing the HTTP messages, rather than about the HTTP messages. To have Wireshark do this, select *Analyze->Enabled Protocols.* Then uncheck the HTTP box and select *OK*. You should now see a Wireshark window that looks like:

[9] What do we mean by "annotate"? If you hand in a paper copy, please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate.



```
tcp-ethereal-trace-1  [Wireshark 1.6.7 (SVN Rev 41973 from /trunk-1.6)]
Filter: tcp
No.  Time        Source           Destination      Protocol Length Info
   1 0.000000    192.168.1.102    128.119.245.12   TCP        62 health-polling > http [SY
   2 0.023172    128.119.245.12   192.168.1.102    TCP        62 http > health-polling [SY
   3 0.023265    192.168.1.102    128.119.245.12   TCP        54 health-polling > http [AC
   4 0.026477    192.168.1.102    128.119.245.12   TCP       619 health-polling > http [PS
   5 0.041737    192.168.1.102    128.119.245.12   TCP      1514 health-polling > http [PS
   6 0.053937    128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
   7 0.054026    192.168.1.102    128.119.245.12   TCP      1514 health-polling > http [AC
   8 0.054690    192.168.1.102    128.119.245.12   TCP      1514 health-polling > http [AC
   9 0.077294    128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
  10 0.077405    192.168.1.102    128.119.245.12   TCP      1514 health-polling > http [AC
  11 0.078157    192.168.1.102    128.119.245.12   TCP      1514 health-polling > http [AC
  12 0.124085    128.119.245.12   192.168.1.102    TCP        60 http > health-polling [AC
  13 0.124185    192.168.1.102    128.119.245.12   TCP      1201 health-polling > http [PS

▷ Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
▽ Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
  ▽ Destination: LinksysG_da:af:73 (00:06:25:da:af:73)
      Address: LinksysG_da:af:73 (00:06:25:da:af:73)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
  ▽ Source: Actionte_8a:70:1a (00:20:e0:8a:70:1a)
      Address: Actionte_8a:70:1a (00:20:e0:8a:70:1a)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)

0000  00 06 25 da af 73 00 20  e0 8a 70 1a 08 00 45 00   ..%..s.  ..p...E.
0010  00 30 1e 1d 40 00 80 06  a5 18 c0 a8 01 66 80 77   .0..@... .....f.w
0020  f5 0c 04 89 00 50 0d d6  01 f4 00 00 00 00 70 02   .....P.. ......p.
0030  40 00 f6 e9 00 00 02 04  05 b4 01 01 04 02         @....... ......

File: "/Users/kurose/Umass/... | Packets: 213 Displayed: 202 Marked: 0 Load time: 0:00.011 | Profile: Default
```

This is what we're looking for - a series of TCP segments sent between your computer and gaia.cs.umass.edu. We will use the packet trace that you have captured (and/or the packet trace *tcp-ethereal-trace-1* in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip; see earlier footnote) to study TCP behavior in the rest of this lab.

Solution:

Ans1: Client computer (source) IP address: 192.168.1.102 TCP port number: 1161
Ans2: Destination computer: gaia.cs.umass.edu IP address: 128.119.245.12 TCP port number: 80
Ans3: If you did this problem on your own computer, you'll have your own solution

Activity 2:

TCP Basics

Answer the following questions for the TCP segments:

1. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?
2. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?
3. What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.
4. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each

TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the EstimatedRTT value (see Section 3.5.3, page 239 in text) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 239 for all subsequent segments.

   Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the gaia.cs.umass.edu server. Then select: *Statistics->TCP Stream Graph->Round Trip Time Graph.*

5. What is the length of each of the first six TCP segments?[10]
6. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?
7. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?
8. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

Solution:

Ans1: Sequence number of the TCP SYN segment is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu. The value is 0 in this trace. The SYN flag is set to 1 and it indicates that this segment is a SYN segment.

Ans2: Sequence number of the SYNACK segment from gaia.cs.umass.edu to the client computer in reply to the SYN has the value of 0 in this trace. The value of the ACKnowledgement field in the SYNACK segment is 1. The value of the ACKnowledgement field in the SYNACK segment is determined by gaia.cs.umass.edu by adding 1 to the initial sequence number of SYN segment from the client computer (i.e. the sequence number of the SYN segment initiated by the client computer is 0.). The SYN flag and Acknowledgement flag in the segment are set to 1 and they indicate that this segment is a SYNACK segment

Ans3: No. 4 segment is the TCP segment containing the HTTP POST command. The sequence number of this segment has the value of 1.

Ans4: The HTTP POST segment is considered as the first segment. Segments 1 – 6 are No. 4, 5, 7, 8, 10, and 11 in this trace respectively. The ACKs of segments 1 – 6 are No. 6, 9, 12, 14, 15, and 16 in this trace.

Segment 1 sequence number: 1
Segment 2 sequence number: 566
Segment 3 sequence number: 2026
Segment 4 sequence number: 3486
Segment 5 sequence number: 4946
Segment 6 sequence number: 6406
The sending time and the received time of ACKs are tabulated in the following table
EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT
EstimatedRTT after the receipt of the ACK of segment 1
EstimatedRTT = RTT for Segment 1 = 0.02746 second
EstimatedRTT after the receipt of the ACK of segment 2:
EstimatedRTT = 0.875 * 0.02746 + 0.125 * 0.035557 = 0.0285
EstimatedRTT after the receipt of the ACK of segment 3:
EstimatedRTT = 0.875 * 0.0285 + 0.125 * 0.070059 = 0.0337

EstimatedRTT after the receipt of the ACK of segment 4:
EstimatedRTT = 0.875 * 0.0337+ 0.125 * 0.11443 = 0.0438
EstimatedRTT after the receipt of the ACK of segment 5:
EstimatedRTT = 0.875 * 0.0438 + 0.125 * 0.13989 = 0.0558
EstimatedRTT after the receipt of the ACK of segment 6:
EstimatedRTT = 0.875 * 0.0558 + 0.125 * 0.18964 = 0.0725

Ans5: Length of the first TCP segment (containing the HTTP POST): 565 bytes Length of each of the other five TCP segments: 1460 bytes (MSS)
Ans6: The minimum amount of buffer space (receiver window) advertised at gaia.cs.umass.edu for the entire trace is 5840 bytes, which shows in the first acknowledgement from the server. This receiver window grows steadily until a maximum receiver buffer size of 62780 bytes. The sender is never throttled due to lacking of receiver buffer space by inspecting this trace.
Ans7: There are no retransmitted segments in the trace file. We can verify this by checking the sequence numbers of the TCP segments in the trace file. In the TimeSequence-Graph (Stevens) of this trace, all sequence numbers from the source (192.168.1.102) to the destination (128.119.245.12) are increasing monotonically with respect to time. If there is a retransmitted segment, the sequence number of this retransmitted segment should be smaller than those of its neighboring segments.
Ans8: The computation of TCP throughput largely depends on the selection of averaging time period. As a common throughput computation, in this question, we select the average time period as the whole connection time. Then, the average throughput for this TCP connection is computed as the ratio between the total amount data and the total transmission time. The total amount data transmitted can be computed by the difference between the sequence number of the first TCP segment (i.e. 1 byte for No. 4 segment) and the acknowledged sequence number of the last ACK (164091 bytes for No. 202 segment). Therefore, the total data are 164091 - 1 = 164090 bytes. The whole transmission time is the difference of the time instant of the first TCP segment (i.e., 0.026477 second for No.4 segment) and the time instant of the last ACK (i.e., 5.455830 second for No. 202 segment). Therefore, the total transmission time is 5.455830 - 0.026477 = 5.4294 seconds. Hence, the throughput for the TCP connection is computed as 164090/5.4294 = 30.222 KByte/sec.

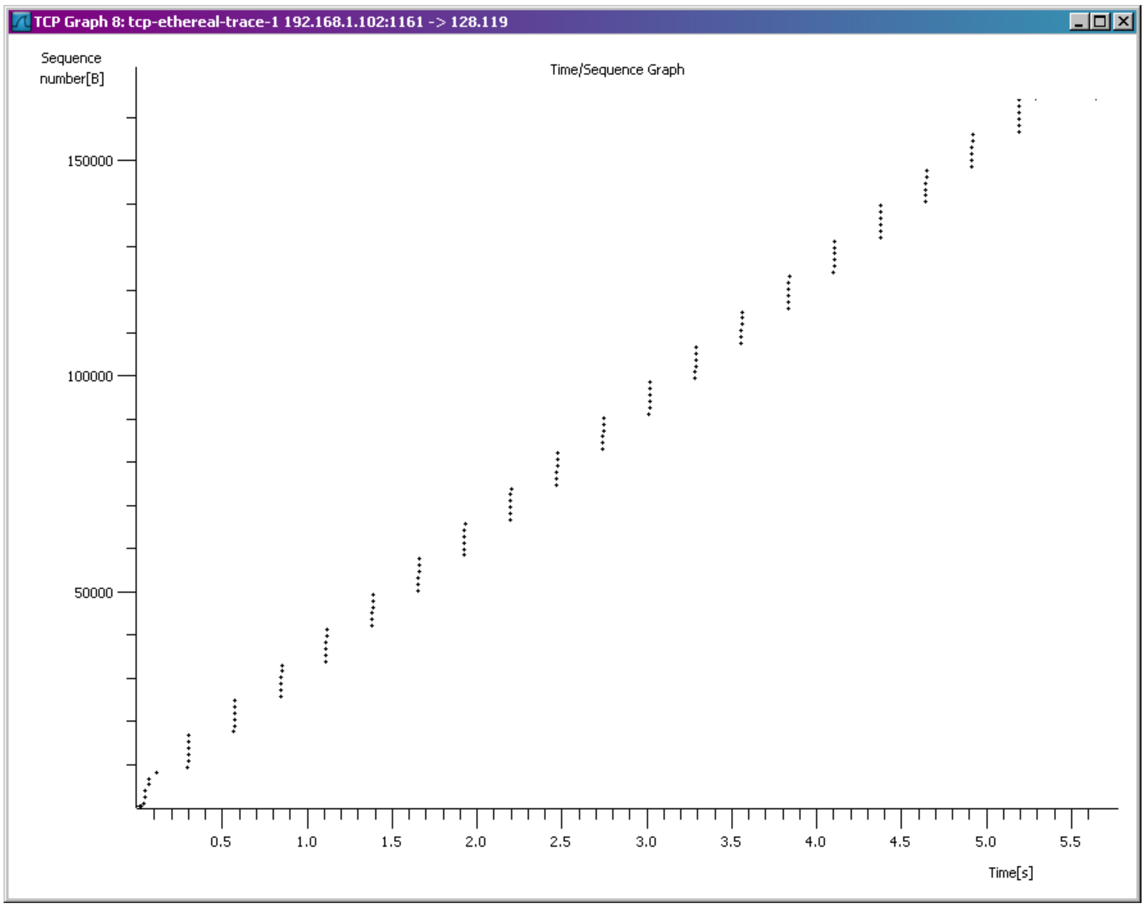
3) Stage v (verify)

Home Activities:

TCP congestion control in action

Let’s now examine the amount of data sent per unit time from the client to the server. Rather than (tediously!) calculating this from the raw data in the Wireshark window, we’ll use one of Wireshark’s TCP graphing utilities - *Time-Sequence-Graph(Stevens)* - to plot out data.

- Select a TCP segment in the Wireshark’s “listing of captured-packets” window. Then select the menu :*Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens)*. You should see a plot that looks similar to the following plot, which was created from the captured packets in the packet trace *tcp-ethereal-trace-1* in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip (see earlier footnote):



Here, each dot represents a TCP segment sent, plotting the sequence number of the segment versus the time at which it was sent. Note that a set of dots stacked above each other represents a series of packets that were sent back-to-back by the sender.

Answer the following questions for the TCP segments the packet trace *tcp-ethereal-trace-1* in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip

5. Use the *Time-Sequence-Graph(Stevens*) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 06

Statement Purpose:

- To familiarize the students with the socket programming.

Activity outcomes:

- After this lab the students will have basic knowledge of socket programming and they should be able to write simple network applications, such as client-server application for chat.

Instructor Note:

- In this lab, you’ll learn how to make socket in different scenarios.

1) Stage J(Journey)

Introduction

3.1.The BSD Socket API

Sockets are like Unix pipes in that they use file descriptors to define the endpoints of a communication channel. When you make the system call to create a pipe, you pass in the pointer to an integer array with two elements. The successful completion of the pipe call returns 0 and initializes the array with two file descriptors, one to which you can write and one from which you can read. The socket follows a more complex set up process which we describe below.

Creating a socket

```
#include <sys/types.h>
#include <sys/socket.h>
```

int socket(int domain, int type, int protocol);The socket call creates one single endpoint for communications, unlike pipe whichreturns two. If the value returned is less than 0, the call was unsuccessful and the errorcan be discovered through errno and perror. If the value returned is greater than or equalto 0, the call was successful and the returned value is a File-descriptor. Althouth theAF_UNIX domain adds some interesting and flexible options for IPC within the same host, we will leave it alone for now. We are more interested in domains that allow two hosts connected to the Internet to communicate and will focus exclusively on the AF_INET for now.The next parameter, type, can assume one of three possible values as described below:

•SOCK_DGRAM: provides datagram communication semantics, that is, this is a connectionless protocol that will do its best to deliver the data without making any promises that the service is reliable. There are no guarantees that the datagrams are delivered in the order they are sent.

•SOCK_STREAM: provides a bidirectional virtual circuit that is reliable and orderpreserving (FIFO).

•SOCK_RAW: provides the ability to send packets directly to a network device driver, enabling user space applications to provide networking protocols that are not implemented in the kernel.The last parameter, protocol, can be left unspecified (value 0) so that the default protocolthat supports the protocol family and socket type is used. In case, multiple protocols existfor the selected tuple <domain, type>, then a specific protocol must be specified. After the successful completion of the socket call, what you have is a connection endpoint that is not attached anywhere. Before any communication can happen, the socket must be associated to "something". The process of connecting the socket is an asymmetric task, that is, it is performed differently at each endpoint of the socket. Server applications (which run on "infinite loops"), create the socket, get it ready to be hooked up to something, and then wait for someone to request a connection to the socket. Client processes, on the other hand, create a socket, tell the system to which address they want to connect it, and attempt establishing the connection to the server. The server process then accepts the connection request and the socket is finally ready for communication.

Binding an address to a socket

Before we proceed, you need to understand what an Internet address and how it is represented in Unix. Header file <netinet/in.h> defines a 32-bit address for an Internet host. It actually identifies a specific network interface on a specific system on the Internet using the data structure below:

```
structin_addr {
__u32 s_addr;
};
```

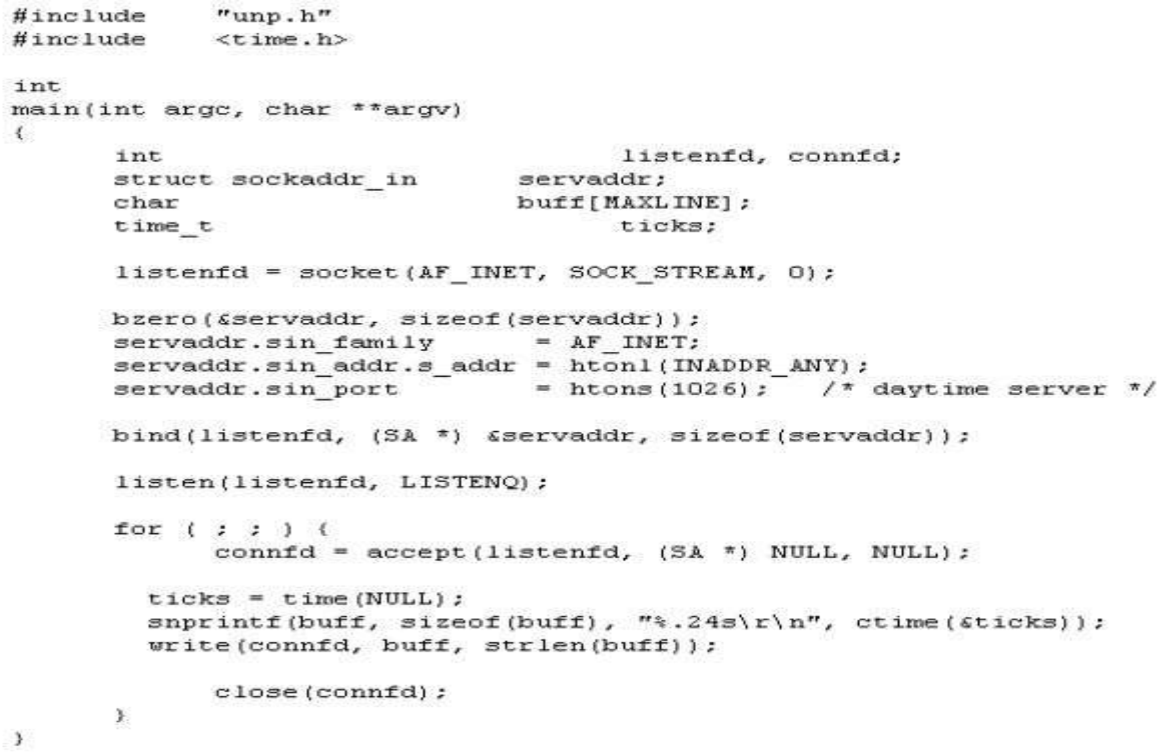
2) Stage a1 (apply)

Lab Activities:

Activity 1:

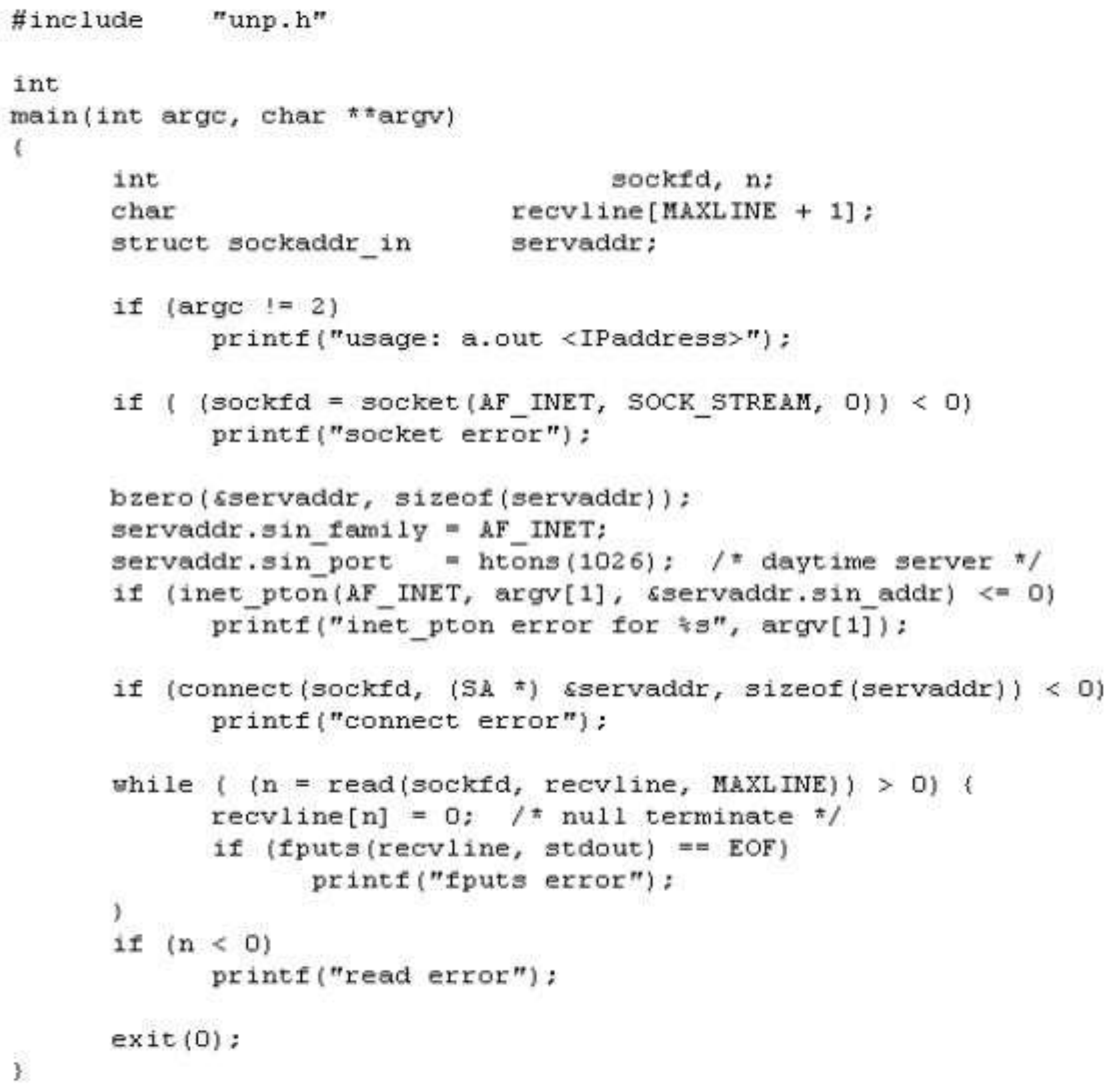
Examples

A simple Day time server



```
#include     "unp.h"
#include     <time.h>

int
main(int argc, char **argv)
{
        int                               listenfd, connfd;
        struct sockaddr_in        servaddr;
        char                          buff[MAXLINE];
        time_t                            ticks;

        listenfd = socket(AF_INET, SOCK_STREAM, 0);

        bzero(&servaddr, sizeof(servaddr));
        servaddr.sin_family        = AF_INET;
        servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
        servaddr.sin_port          = htons(1026);    /* daytime server */

        bind(listenfd, (SA *) &servaddr, sizeof(servaddr));

        listen(listenfd, LISTENQ);

        for ( ; ; ) {
                connfd = accept(listenfd, (SA *) NULL, NULL);

          ticks = time(NULL);
          snprintf(buff, sizeof(buff), "%.24s\r\n", ctime(&ticks));
          write(connfd, buff, strlen(buff));

                close(connfd);
        }
}
```

A simple Day time client



```
#include    "unp.h"

int
main(int argc, char **argv)
{
        int                             sockfd, n;
        char                        recvline[MAXLINE + 1];
        struct sockaddr_in          servaddr;

        if (argc != 2)
              printf("usage: a.out <IPaddress>");

        if ( (sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
              printf("socket error");

        bzero(&servaddr, sizeof(servaddr));
        servaddr.sin_family = AF_INET;
        servaddr.sin_port   = htons(1026);  /* daytime server */
        if (inet_pton(AF_INET, argv[1], &servaddr.sin_addr) <= 0)
              printf("inet_pton error for %s", argv[1]);

        if (connect(sockfd, (SA *) &servaddr, sizeof(servaddr)) < 0)
              printf("connect error");

        while ( (n = read(sockfd, recvline, MAXLINE)) > 0) {
              recvline[n] = 0;  /* null terminate */
              if (fputs(recvline, stdout) == EOF)
                    printf("fputs error");
        }
        if (n < 0)
              printf("read error");

        exit(0);
}
```

Date and Time

Lets create a server that continuously runs and sends the date and time as soon as a client connects to it.

Socket Server Example

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <time.h>

int main(intargc, char *argv[])
{
```

```
intlistenfd = 0, connfd = 0;
structsockaddr_inserv_addr;

charsendBuff[1025];
time_t ticks;

listenfd = socket(AF_INET, SOCK_STREAM, 0);
memset(&serv_addr, '0', sizeof(serv_addr));
memset(sendBuff, '0', sizeof(sendBuff));

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
serv_addr.sin_port = htons(5000);

bind(listenfd, (structsockaddr*)&serv_addr, sizeof(serv_addr));

listen(listenfd, 10);

while(1)
   {
connfd = accept(listenfd, (structsockaddr*)NULL, NULL);

ticks = time(NULL);
snprintf(sendBuff, sizeof(sendBuff), "%.24s\r\n", ctime(&ticks));
write(connfd, sendBuff, strlen(sendBuff));

close(connfd);
sleep(1);
    }
}
```

Socket Client Example

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <arpa/inet.h>

int main(intargc, char *argv[])
{
intsockfd = 0, n = 0;
charrecvBuff[1024];
structsockaddr_inserv_addr;

if(argc != 2)
```

```
   {
printf("\n Usage: %s <ip of server> \n",argv[0]);
return 1;
   }

memset(recvBuff, '0',sizeof(recvBuff));
if((sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
   {
printf("\n Error : Could not create socket \n");
return 1;
   }

memset(&serv_addr, '0', sizeof(serv_addr));

serv_addr.sin_family = AF_INET;
serv_addr.sin_port = htons(5000);

if(inet_pton(AF_INET, argv[1], &serv_addr.sin_addr)<=0)
   {
printf("\n inet_pton error occured\n");
return 1;
   }

if( connect(sockfd, (structsockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
   {
printf("\n Error : Connect Failed \n");
return 1;
   }

while ( (n = read(sockfd, recvBuff, sizeof(recvBuff)-1)) > 0)
   {
recvBuff[n] = 0;
if(fputs(recvBuff, stdout) == EOF)
      {
printf("\n Error : Fputs error\n");
      }
   }

if(n < 0)
   {
printf("\n Read error \n");
   }

return 0;
}
```

3) Stage v (verify)

Home Activities:

Activity 1:

- This client/server pair runs a simple TCP socket program as an Echo Server that only allows one client to connect to the server.
- This client/server pair improves on the previous Echo Server that allows multiple clients to connect to the server.
- This server improves on the previous Echo Server since it allows the "connection thread" of the server to stop executing via a command sent from the client.
- This server improves on the previous Echo Server since it uses a time out on the server's connection TCP socket to continuously check if it should stop the "connection thread" of the server from executing.
- This example shows how to send an object across a TCP socket by serializing a class.
- This is an example of an extremely simple web server. The client is a web browser.

The following are some simple UDP socket examples.

- This client/server pair runs a simple UDP socket program as an Echo/Uppercase Server that only allows the client to send one message to the server.
- This client sends multiple messages to the Echo/Uppercase Server to check if packets get dropped.
- This program scans the UDP ports on the local machine to see if they are being used.

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 07

Statement Purpose:

- Explore several aspects of UDP protocol.

Activity outcomes:

- After this lab, the students will get better understanding of the UDP protocol.

Instructor Note:

- In this first Wireshark lab, you'll captures some UDP packets using wireshark and make some observations on them.

1) Stage J(Journey)

Introduction

In this lab, we'll take a quick look at the UDP transport protocol. As we saw in Chapter 3 of the text1 , UDP is a streamlined, no-frills protocol. You may want to re-read section 3,3 in the text before doing this lab. Because UDP is simple and sweet, we'll be able to cover it pretty quickly in this lab. So if you've another appointment to run off to in 30 minutes, no need to worry, as you should be able to finish this lab with ample time to spare.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

Start capturing packets in Wireshark and then do something that will cause your host to send and receive several UDP packets. It's also likely that just by doing nothing (except capturing packets via Wireshark) that some UDP packets sent by others will appear in your trace. In particular, the Simple Network Management Protocol (SNMP - chapter 9 in the text) sends SNMP messages inside of UDP, so it's likely that you'll find some SNMP messages (and therefore UDP packets) in your trace.

After stopping packet capture, set your packet filter so that Wireshark only displays the UDP packets sent and received at your host. Pick one of these UDP packets and expand the UDP fields in the details window. If you are unable to find UDP packets or are unable to run Wireshark on a live network connection, you can download a packet trace containing some UDP packets.[11]

Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[12] to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet*

[11] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file http-ethereal-trace-5, which contains some UDP packets carrying SNMP messages. The traces in this zip file were collected by Wireshark running on one of the author's computers. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the http-ethereal-trace-5 trace file.

summary line, and select the minimum amount of packet detail that you need to answer the question.

1. Select *one* UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. (You shouldn't look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields.
2. By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.
3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.
4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)
5. What is the largest possible source port number? (Hint: see the hint in 4.)
6. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. To answer this question, you'll need to look into the Protocol field of the IP datagram containing this UDP segment
7. Examine a pair of UDP packets in which your host sends the first UDP packet and the second UDP packet is a reply to this first UDP packet. (Hint: for a second packet to be sent in response to a first packet, the sender of the first packet should be the destination of the second packet). Describe the relationship between the port numbers in the two packets.

Solution:

1. Select one packet. From this packet, determine how many fields there are in the UDP header. (Do not look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields
Ans: The UDP header contains 4 fields: source port, destination port, length, and checksum.
2. From the packet content field, determine the length (in bytes) of each of the UDP header fields.
Ans: Each of the UDP header fields is 2 bytes long.
3. The value in the Length field is the length of what? Verify your claim with your captured UDP packet.
Ans: The value in the length field is the sum of the 8 header bytes, plus the 42 encapsulated data bytes.
4. What is the maximum number of bytes that can be included in a UDP payload.
Ans: The maximum number of bytes that can be included in a UDP payload is 216 – 1 less the header bytes. This gives 65535 – 8 = 65527 bytes.
5. What is the largest possible source port number?
Ans: The largest possible source port number is 216 – 1 = 65535.
6. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. (To answer this question, you'll need to look into the IP header.)
Ans: The IP protocol number for UDP is 0x11 hex, which is 17 in decimal value
7. Examine a pair of UDP packets in which the first packet is sent by your host and the second packet is a reply to the first packet. Describe the relationship between the port numbers in the two packets.
Ans: The source port of the UDP packet sent by the host is the same as the destination port of the reply packet, and conversely the destination port of the UDP packet sent by the host is the same as the source port of the reply packet.

3) Stage v (verify)

Home Activities:

Activity 1:

Capture a small UDP packet. Manually verify the checksum in this packet. Show all work and explain all steps.

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 08

Statement Purpose:

1.1. we'll investigate the IP protocol, focusing on the IP datagram
1.2. analyze a trace of IP datagrams sent and received by an execution of the traceroute program
1.3. Investigate the various fields in the IP datagram, and study IP fragmentation in detail.

Activity Outcomes:

1.1. Students will be able to gain better understanding of the IP protocol

Instructor Note:

1) Stage J(Journey)

Introduction

In order to generate a trace of IP datagrams for this lab, we'll use the traceroute program to send datagrams of different sizes towards some destination, *X*. Recall that traceroute operates by first sending one or more datagrams with the time-to-live (TTL) field in the IP header set to 1; it then sends a series of one or more datagrams towards the same destination with a TTL value of 2; it then sends a series of datagrams towards the same destination with a TTL value of 3; and so on. Recall that a router must decrement the TTL in each received datagram by 1 (actually, RFC 791 says that the router must decrement the TTL by *at least* one). If the TTL reaches 0, the router returns an ICMP message (type 11 – TTL-exceeded) to the sending host. As a result of this behavior, a datagram with a TTL of 1 (sent by the host executing traceroute) will cause the router one hop away from the sender to send an ICMP TTL-exceeded message back to the sender; the datagram sent with a TTL of 2 will cause the router two hops away to send an ICMP message back to the sender; the datagram sent with a TTL of 3 will cause the router three hops away to send an ICMP message back to the sender; and so on. In this manner, the host executing traceroute can learn the identities of the routers between itself and destination *X* by looking at the source IP addresses in the datagrams containing the ICMP TTL-exceeded messages.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

We'll want to run traceroute and have it send datagrams of various lengths.

Solution:

- **Windows.** The tracert program (used for our ICMP Wireshark lab) provided with Windows does not allow one to change the size of the ICMP echo request (ping) message sent by the tracert program. A nicer Windows traceroute program is *pingplotter*, available both in free version and shareware versions at http://www.pingplotter.com. Download and install *pingplotter*, and test it out by performing a few traceroutes to your favorite sites. The size of the ICMP echo request message can be explicitly set in *pingplotter* by selecting the menu item *Edit-> Options->Packet Options* and then filling in the *Packet Size* field. The default packet size is 56 bytes. Once *pingplotter* has sent a series of packets with the increasing TTL values, it restarts the sending process again with a TTL of 1, after waiting *Trace Interval* amount of time. The value of *Trace Interval* and the number of intervals can be explicitly set in *pingplotter*.

- **Linux/Unix/MacOS.** With the Unix/MacOStraceroute command, the size of the UDP datagram sent towards the destination can be explicitly set by indicating the number of bytes in the datagram; this value is entered in the traceroute command line immediately after the name or address of the destination. For example, to send traceroute datagrams of 2000 bytes towards gaia.cs.umass.edu, the command would be:%traceroute gaia.cs.umass.edu 2000

Do the following:

- Start up Wireshark and begin packet capture *(Capture->Start)* and then press *OK* on the Wireshark Packet Capture Options screen (we'll not need to select any options here).
- If you are using a Windows platform, start up*pingplotter* and enter the name of a target destination in the "Address to Trace Window." Enter 3 in the "# of times to Trace" field, so you don't gather too much data. Select the menu item *Edit->Advanced Options->Packet Options* and enter a value of 56 in the *Packet Size* field and then press OK. Then press the Trace button. You should see a *pingplotter* window that looks something like this:
- Next, send a set of datagrams with a longer length, by selecting *Edit->Advanced Options->Packet Options* and enter a value of 2000 in the *Packet Size* field and then press OK. Then press the Resume button.
-
- Finally, send a set of datagrams with a longer length, by selecting *Edit->Advanced Options->Packet Options* and enter a value of 3500 in the *Packet Size* field and then press OK. Then press the Resume button.
- Stop Wireshark tracing.
- If you are using a Unix or Mac platform, enter three traceroute commands, one with a length of 56 bytes, one with a length of 2000 bytes, and one with a length of 3500 bytes.
- Stop Wireshark tracing.

If you are unable to run Wireshark on a live network connection, you can download a packet trace file that was captured while following the steps above on one of the author's Windows computers[13]. You may well find it valuable to download this trace even if you've captured your own trace and use it, as well as your own trace, when you explore the questions below.

Activity 2:

In your trace, you should be able to see the series of ICMP Echo Request (in the case of Windows machine) or the UDP segment (in the case of Unix) sent by your computer and the ICMP TTL-exceeded messages returned to your computer by the intermediate routers. In the questions below, we'll assume you are using a Windows machine; the corresponding questions for the case of a Unixmachine should be clear. Whenever possible, when answering a question below you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. When you hand in your assignment, annotate the output so that it's clear where in the output you're getting the information for your answer (e.g., for our classes, we ask that students markup paper copies with a pen, or annotate electronic copies with text in a colored font).To print a packet, use

[13] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file *ip-ethereal-trace-1*. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the ip-ethereal-trace-1 trace file.

File->Print, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.
Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol part of the packet in the packet details window.

2. Within the IP packet header, what is the value in the upper layer protocol field?
3. How many bytes are in the IP header? How many bytes are in the payload *of the IP datagram*? Explain how you determined the number of payload bytes.
4. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.

Next, sort the traced packets according to IP source address by clicking on the *Source* column header; a small downward pointing arrow should appear next to the word *Source*. If the arrow points up, click on the *Source* column header again. Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol portion in the "details of selected packet header" window. In the "listing of captured packets" window, you should see all of the subsequent ICMP messages (perhaps with additional interspersed packets sent by other protocols running on your computer) below this first ICMP. Use the down arrow to move through the ICMP messages sent by your computer.

5. Which fields in the IP datagram *always* change from one datagram to the next within this series of ICMP messages sent by your computer?
6. Which fields stay constant? Which of the fields *must* stay constant? Which fields must change? Why?
7. Describe the pattern you see in the values in the Identification field of the IP datagram

Next (with the packets still sorted by source address) find the series of ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router.
8. What is the value in the Identification field and the TTL field?
9. Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why?

3) Stage v (verify)

<u>Home Activities:</u>

Activity 1:

Sort the packet listing according to time again by clicking on the *Time* column.

1. Find the first ICMP Echo Request message that was sent by your computer after you changed the *Packet Size* in *pingplotter* to be 2000. Has that message been fragmented across more than one IP datagram? [Note: if you find your packet has not been fragmented, you should download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the *ip-ethereal-trace-1*packet trace. If your computer has an Ethernet interface, a packet size of 2000 *should* cause fragmentation.[14]]

[14] The packets in the *ip-ethereal-trace-1* trace file in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip are all less that 1500 bytes. This is because the computer on which the trace was gathered has an Ethernet card that limits the length of the maximum IP packet to 1500 bytes (40 bytes of TCP/IP header data and 1460 bytes of upper-layer protocol payload). This 1500 byte value is the standard maximum length allowed by Ethernet. If your trace indicates a datagram longer 1500 bytes, and your computer is using an Ethernet connection, then Wireshark is reporting the wrong IP datagram length; it will likely also show only one large IP datagram rather than multiple smaller datagrams.. This inconsistency in reported lengths is due to the

2. Print out the first fragment of the fragmented IP datagram. What information in the IP header indicates that the datagram been fragmented? What information in the IP header indicates whether this is the first fragment versus a latter fragment? How long is this IP datagram?
3. Print out the second fragment of the fragmented IP datagram. What information in the IP header indicates that this is not the first datagram fragment? Are the more fragments? How can you tell?
4. What fields change in the IP header between the first and second fragment?

Now find the first ICMP Echo Request message that was sent by your computer after you changed the *Packet Size* in *pingplotter* to be 3500.

5. How many fragments were created from the original datagram?
6. What fields change in the IP header among the fragments?

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

interaction between the Ethernet driver and the Wireshark software. We recommend that if you have this inconsistency, that you perform this lab using the *ip-ethereal-trace-1* trace file.

LAB # 09

Statement Purpose:

Investigate the behavior of the NAT protocol

Activity Outcomes:

The students will gain better understanding of the NAT

Instructor Note:

1) StageJ(Journey)

Introduction

In this lab, we'll capture packets from a simple web request from a client PC in a home network to a www.google.com server. Within the home network, the home network router provides a NAT service, as discussed in Chapter 4. Figure 1 shows our Wireshark trace-collection scenario. As in our other Wireshark labs, we collect a Wireshark trace on the client PC in our home network. This file is called NAT_home_side[15]. Because we are also interested in the packets being sent by the NAT router into the ISP, we'll collect a second trace file at a PC (not shown) tapping into the link from the home router into the ISP network, as shown in Figure 1. (The hub device shown on the ISP side of the router is used to tap into the link between the NAT router and the first hop router in the ISP). Client-to-server packets captured by Wireshark at this point will have undergone NAT translation. The Wireshark trace file captured on the ISP side of the home router is called NAT_ISP_side.

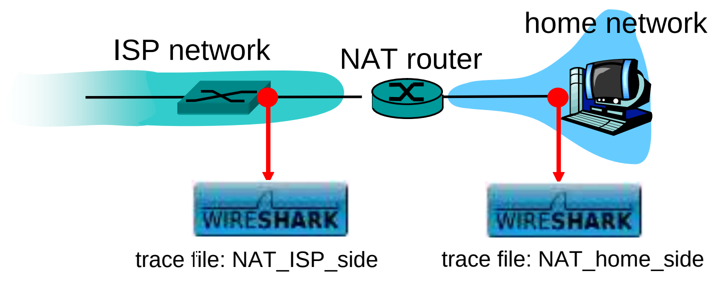


Figure 1: NAT trace collection scenario

2) Stage a1 (apply)

Lab Activities:

Activity 1:

Open the NAT_home_side file and answer the following questions. You might find it useful to use a Wireshark filter so that only frames containing HTTP messages are displayed from the trace file.

[15] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the files need for this lab.

Solution: When answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[16] to explain your answer. To print a packet, use *File->Print*, choose *selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question

1. What is the IP address of the client?
2. The client actually communicates with several different Google servers in order to implement "safe browsing." (See extra credit section at the end of this lab). The main Google server that will serve up the main Google web page has IP address 64.233.169.104. In order to display only those frames containing HTTP messages that are sent to/from this Google, server, enter the expression "http &&ip.addr == 64.233.169.104" (without quotes) into the Filter: field in Wireshark.
3. Consider now the HTTP GET sent from the client to the Google server (whose IP address is IP address 64.233.169.104) at time 7.109267. What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP GET?
4. At what time[17] is the corresponding 200 OK HTTP message received from the Google server? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP 200 OK message?
5. Recall that before a GET command can be sent to an HTTP server, TCP must first set up a connection using the three-way SYN/ACK handshake. At what time is the client-to-server TCP SYN segment sent that sets up the connection used by the GET sent at time 7.109267? What are the source and destination IP addresses and source and destination ports for the TCP SYN segment? What are the source and destination IP addresses and source and destination ports of the ACK sent in response to the SYN. At what time is this ACK received at the client? (Note: to find these segments you will need to clear the Filter expression you entered above in step 2. If you enter the filter "tcp", only TCP segments will be displayed by Wireshark).

In the following we'll focus on the two HTTP messages (GET and 200 OK) and the TCP SYN and ACK segments identified above. Our goal below will be to locate these two HTTP messages and two TCP segments in the trace file (NAT_ISP_side) captured on the link between the router and the ISP. Because these captured frames will have already been forwarded through the NAT router, some of the IP address and port numbers will have been changed as a result of NAT translation.

Activity 2:

Open the NAT_ISP_side:

Solution:

Note that the time stamps in this file and in NAT_home_side are not synchronized since the packet captures at the two locations shown in Figure 1 were not started simultaneously. (Indeed, you should discover that the timestamps of a packet captured at the ISP link is actually less that the timestamp of the packet captured at the client PC).

1. In the NAT_ISP_side trace file, find the HTTP GET message was sent from the client to the Google server at time 7.109267 (where t=7.109267 is time at which this was sent as recorded in the NAT_home_side trace file). At what time does this message appear in the

[16] What do we mean by "annotate"? If you hand in a paper copy, please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate.

[17] Specify time using the time since the beginning of the trace (rather than absolute, wall-clock time).

NAT_ISP_side trace file? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP GET (as recording in the NAT_ISP_side trace file)? Which of these fields are the same, and which are different, than in your answer to question 3 (activity one) above?

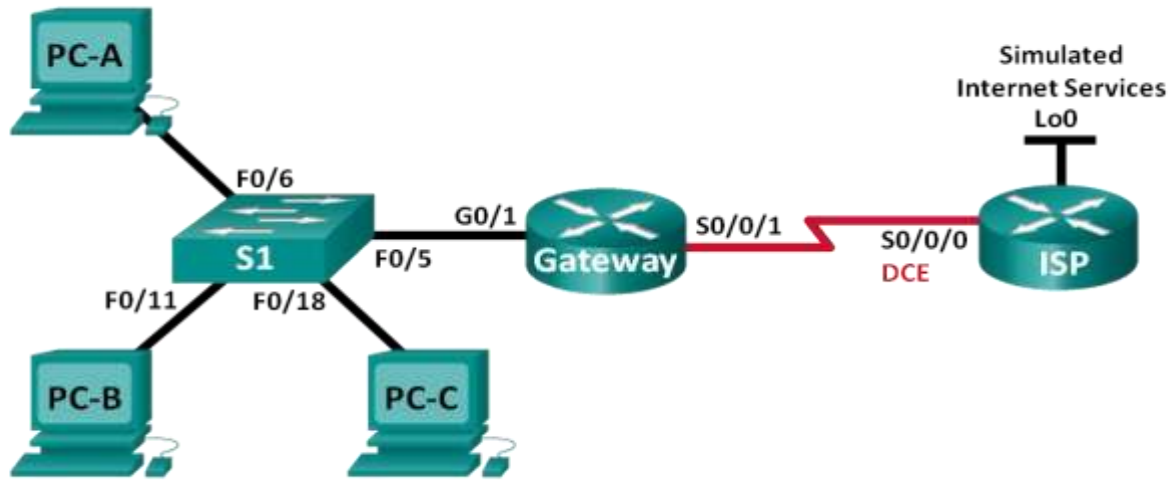
2. Are any fields in the HTTP GET message changed? Which of the following fields in the IP datagram carrying the HTTP GET are changed: Version, Header Length, Flags, Checksum. If any of these fields have changed, give a reason (in one sentence) stating why this field needed to change.
3. In the NAT_ISP_side trace file, at what time is the first 200 OK HTTP message received from the Google server? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP 200 OK message? Which of these fields are the same, and which are different than your answer to question 4(activity one) above?
4. In the NAT_ISP_side trace file, at what time were the client-to-server TCP SYN segment and the server-to-client TCP ACK segment corresponding to the segments in question 5 above captured? What are the source and destination IP addresses and source and destination ports for these two segments? Which of these fields are the same, and which are different than your answer to question 5(activity one) above?

Using your answers to activity one and two above, fill in the NAT translation table entries for HTTP connection considered in questions of activity one and two.

Activity 3:

Packet tracer part

Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway
Gateway	G0/1	192.168.1.1	255.255.255.0	N/A
	S0/0/1	209.165.201.18	255.255.255.252	N/A
ISP	S0/0/0 (DCE)	209.165.201.17	255.255.255.252	N/A
	Lo0	192.31.7.1	255.255.255.255	N/A
PC-A	NIC	192.168.1.20	255.255.255.0	192.168.1.1
PC-B	NIC	192.168.1.21	255.255.255.0	192.168.1.1
PC-C	NIC	192.168.1.22	255.255.255.0	192.168.1.1

Objectives

Part 1: Build the Network and Verify Connectivity

Part 2: Configure and Verify NAT Pool Overload

Part 3: Configure and Verify PAT

Background / Scenario

In the first part of the lab, your company is allocated the public IP address range of 209.165.200.224/29 by the ISP. This provides the company with six public IP addresses. Dynamic NAT pool overload uses a pool of IP addresses in a many-to-many relationship. The router uses the first IP address in the pool and assigns connections using the IP address plus a unique port number. After the maximum number of translations for a single IP address have been reached on the router (platform and hardware specific), it uses the next IP address in the pool.

In Part 2, the ISP has allocated a single IP address, 209.165.201.18, to your company for use on the Internet connection from the company Gateway router to the ISP. You will use the Port Address Translation (PAT) to convert multiple internal addresses into the one usable public address. You will test, view, and verify that the translations are taking place, and you will interpret the NAT/PAT statistics to monitor the process.

Required Resources

- 2 Routers (Cisco 1941)
- 1 Switch (Cisco 2960)
- 3 PCs (Windows 7, Vista, or XP with terminal emulation program, such as Tera Term)
- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet and serial cables as shown in the topology

Part 1: Build the Network and Verify Connectivity

In Part 1, you will set up the network topology and configure basic settings, such as the interface IP addresses, static routing, device access, and passwords.

Step 1: Cable the network as shown in the topology.

Step 2: Configure PC hosts.

Step 3: Initialize and reload the routers and switches.

Step 4: Configure basic settings for each router.

a. Disable DNS lookup.

b. Configure IP addresses for the routers as listed in the Addressing Table.

c. Set the clock rate to **128000** for DCE serial interface.

d. Configure device name as shown in the topology.

e. Assign **cisco** as the console and vty passwords.

f. Assign **class** as the encrypted privileged EXEC mode password.

g. Configure **logging synchronous** to prevent console messages from interrupting the command entry.

Step 5: Configure static routing.

a. Create a static route from the ISP router to the Gateway router.

```
ISP(config)# ip route 209.165.200.224 255.255.255.248 209.165.201.18
```

b. Create a default route from the Gateway router to the ISP router.

```
Gateway(config)# ip route 0.0.0.0 0.0.0.0
209.165.201.17
```

Step 6: Verify network connectivity.

a. From the PC hosts, ping the G0/1 interface on the Gateway router. Troubleshoot if the pings are unsuccessful.

b. Verify that the static routes are configured correctly on both routers.

Part 2: Configure and Verify NAT Pool Overload

In Part 2, you will configure the Gateway router to translate the IP addresses from the 192.168.1.0/24 network to one of the six usable addresses in the 209.165.200.224/29 range.

Step 1: Define an access control list that matches the LAN private IP addresses.

ACL 1 is used to allow the 192.168.1.0/24 network to be translated.

```
Gateway(config)# access-list 1 permit 192.168.1.0
0.0.0.255
```

Step 2: Define the pool of usable public IP addresses.

```
Gateway(config)# ip nat pool public_access 209.165.200.225
209.165.200.230 netmask 255.255.255.248
```

Step 3: Define the NAT from the inside source list to the outside pool.

```
Gateway(config)# ip nat inside source list 1 pool public_access
overload
```

Step 4: Specify the interfaces.

Issue the **ip nat inside** and **ip nat outside** commands to the interfaces.

```
Gateway(config)# interface g0/1
Gateway(config-if)# ip nat inside
Gateway(config-if)# interface s0/0/1
Gateway(config-if)# ip nat outside
```

Step 5: Verify the NAT pool overload configuration.

a. From each PC host, ping the 192.31.7.1 address on the ISP router.

b. Display NAT statistics on the Gateway router.

```
Gateway# show ip nat statistics
Total active translations: 3 (0 static, 3 dynamic; 3
extended)
Peak translations: 3, occurred 00:00:25
ago Outside interfaces:
  Serial0/0/1
Inside interfaces:
  GigabitEthernet0/1
Hits: 24  Misses: 0
CEF Translated packets: 24, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list 1 pool public_accessrefcount3
poolpublic_access: netmask 255.255.255.248start
209.165.200.225 end 209.165.200.230type generic, total
addresses 6, allocated 1 (16%), misses 0
Total doors: 0
Appl
doors: 0
Normal
doors: 0
Queued Packets: 0
```

c. Display NATs on the Gateway router.

```
Gateway# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global icmp 209.165.200.225:0 192.168.1.20:1      192.31.7.1:1
192.31.7.1:0 icmp 209.165.200.225:1 192.168.1.21:1     192.31.7.1:1
192.31.7.1:1 icmp 209.165.200.225:2 192.168.1.22:1     192.31.7.1:1
192.31.7.1:2
```

Note: Depending on how much time has elapsed since you performed the pings from each PC, you may not see all three translations. ICMP translations have a short timeout value.

How many Inside local IP addresses are listed in the sample output above? __________ 3

How many Inside global IP addresses are listed? __________ 1

How many port numbers are used paired with the Inside global addresses _________ 3

What would be the result of pinging the Inside local address of PC-A from the ISP router? Why?

____

____

The ping would fail because the router knows the location of the Inside global address in its routing table but the Inside local address is not advertised.

Part 3: Configure and Verify PAT

In Part 3, you will configure PAT by using an interface instead of a pool of addresses to define the outside address. Not all of the commands in Part 2 will be reused in Part 3.

Step 1: Clear NATs and statistics on the Gateway router.

Step 2: Verify the configuration for NAT.

a. Verify that statistics have been cleared.

b. Verify that the outside and inside interfaces are configured for NATs.

c. Verify that the ACL is still configured for NATs.

What command did you use to confirm the results from steps a to c?

____

```
Gateway# show ip nat statistics
```

Step 3: Remove the pool of useable public IP addresses.

```
Gateway(config)# no ip nat pool public_access 209.165.200.225
209.165.200.230 netmask 255.255.255.248
```

Step 4: Remove the NAT translation from inside source list to outside pool.

```
Gateway(config)# no ip nat inside source list 1 pool public_access
```

`overload` **Step 5: Associate the source list with the outside interface.**

```
Gateway(config)# ip nat inside source list 1 interface serial 0/0/1
```

`overload` **Step 6: Test the PAT configuration.**

a. From each PC, ping the 192.31.7.1 address on the ISP router.

b. Display NAT statistics on the Gateway router.

```
Gateway# show ip nat statistics
Total active translations: 3 (0 static, 3 dynamic; 3
extended)
Peak translations: 3, occurred 00:00:19
ago Outside interfaces:
  Serial0/0/1
Inside interfaces:
  GigabitEthernet0/1
Hits: 24  Misses: 0
CEF Translated packets: 24, CEF Punted packets: 0
Expired translations:
0 Dynamic mappings:
-- Inside Source
[Id: 2] access-list 1 interface Serial0/0/1 refcount 3

Total doors: 0
Appl doors: 0
Normal doors: 0
Queued Packets: 0
```

c. Display NAT translations on Gateway.

```
Gateway# show ip nat translations
Pro Inside global       Inside local        Outside local       Outside
global icmp209.165.201.18:3  192.168.1.20:1      192.31.7.1:1
192.31.7.1:3 icmp 209.165.201.18:1  192.168.1.21:1      192.31.7.1:1
192.31.7.1:1 icmp 209.165.201.18:4  192.168.1.22:1      192.31.7.1:1
192.31.7.1:4
```

3) Stage v (verify)

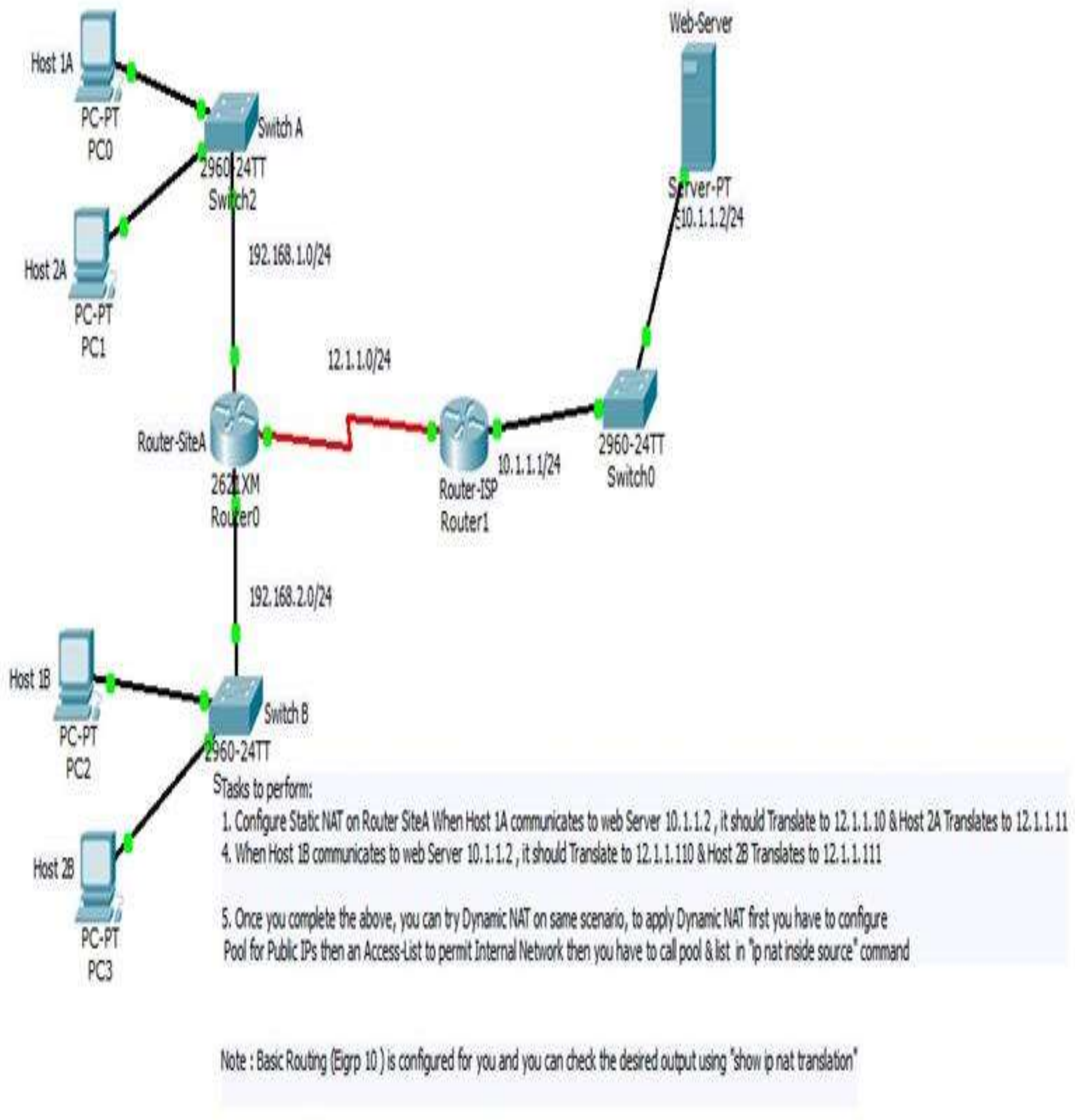
Home Activities:

Activity 1:

The trace files investigated above have additional connections to Google servers above and beyond the HTTP GET, 200 OK request/response studied above. For example, in the NAT_home_side trace file, consider the client-to-server GET at time 1.572315, and the GET at time 7.573305. Research the use of these two HTTP messages and write a half page explanation of the purpose of each of these messages.

Activity 2: (packet tracer)

Configure static NAT on Router side A host 1A communicate to web server 10.1.1.2, it should translate to 12.1.1.10 & host 2A translate to 12.1.1.114. When host 1B communicate to web server 10.1.1.2, it should translate to 12.1.1.10 & host 2B translate to 12.1.1.111.



STasks to perform:

1. Configure Static NAT on Router SiteA When Host 1A communicates to web Server 10.1.1.2 , it should Translate to 12.1.1.10 & Host 2A Translates to 12.1.1.11
4. When Host 1B communicates to web Server 10.1.1.2 , it should Translate to 12.1.1.110 & Host 2B Translates to 12.1.1.111

5. Once you complete the above, you can try Dynamic NAT on same scenario, to apply Dynamic NAT first you have to configure Pool for Public IPs then an Access-List to permit Internal Network then you have to call pool & list in "ip nat inside source" command

Note : Basic Routing (Eigrp 10) is configured for you and you can check the desired output using "show ip nat translation"

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

Answers will vary, but should include that PAT minimizes the number of public addresses needed to provide
Internet access, and that PAT, like NAT, serves to "hide" private addresses from outside networks.

LAB # 10

Statement Purpose:

1. we'll explore several aspects of the ICMP protocol:
   1.1. ICMP messages generating by the Ping program;
   1.2. ICMP messages generated by the Traceroute program;
   1.3. The format and contents of an ICMP message.

Activity Outcomes:

The students will gain better understanding of the ICMP

Instructor Note:

1) Stage J(Journey)

Introduction

In this lab, we'll explore several aspects of the ICMP protocol: (1) ICMP messages generating by the Ping program; (2) ICMP messages generated by the Traceroute program; (3) the format and contents of an ICMP message. Before attacking this lab, you're encouraged to review the ICMP material in section 4.4.3 of the text1 . We present this lab in the context of the Microsoft Windows operating system. However, it is straightforward to translate the lab to a Unix or Linux environment.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

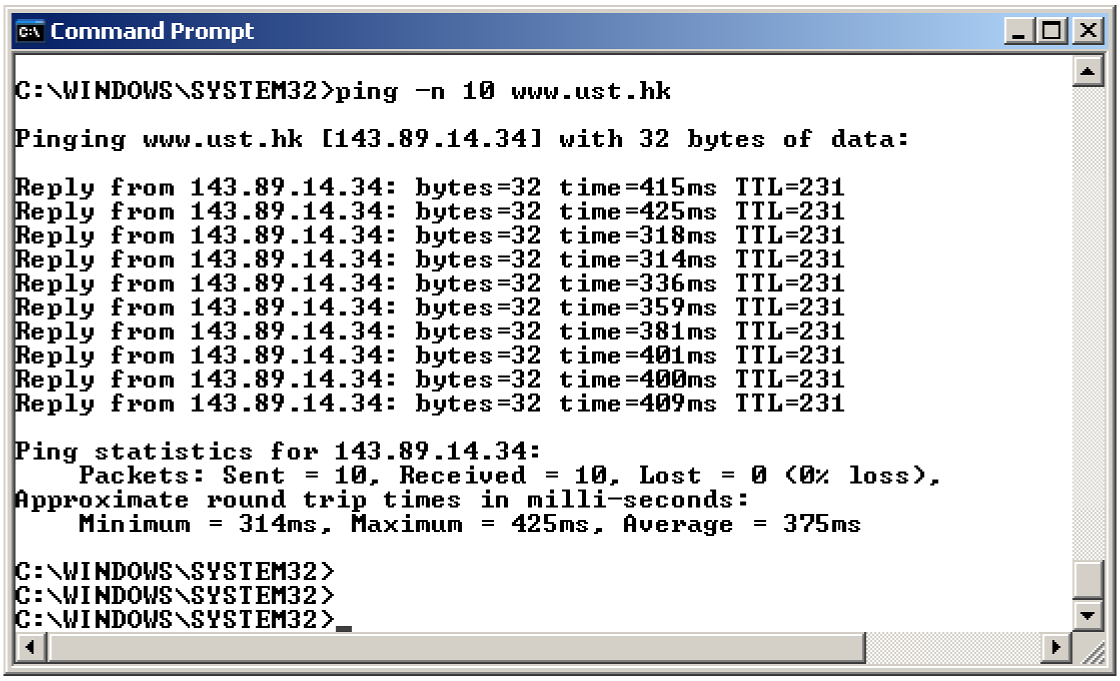
Let's begin our ICMP adventure by capturing the packets generated by the Ping program. You may recall that the Ping program is simple tool that allows anyone (for example, a network administrator) to verify if a host is live or not. The Ping program in the source host sends a packet to the target IP address; if the target is live, the Ping program in the target host responds by sending a packet back to the source host. As you might have guessed (given that this lab is about ICMP), both of these Ping packets are ICMP packets.

Solution:

- Let's begin this adventure by opening the Windows Command Prompt application (which can be found in your Accessories folder).
- Start up the Wireshark packet sniffer, and begin Wireshark packet capture.

- The *ping* command is in c:\windows\system32, so type either "*ping –n 10 hostname*" or "*c:\windows\system32\ping –n 10 hostname*" in the MS-DOS command line (without quotation marks), where hostname is a host on another continent. If you're outside of Asia, you may want to enter www.ust.hk for the Web server at Hong Kong University of Science and Technology. The argument *"-n 10"* indicates that 10 ping messages should be sent. Then run the Ping program by typing return.
- When the Ping program terminates, stop the packet capture in Wireshark.

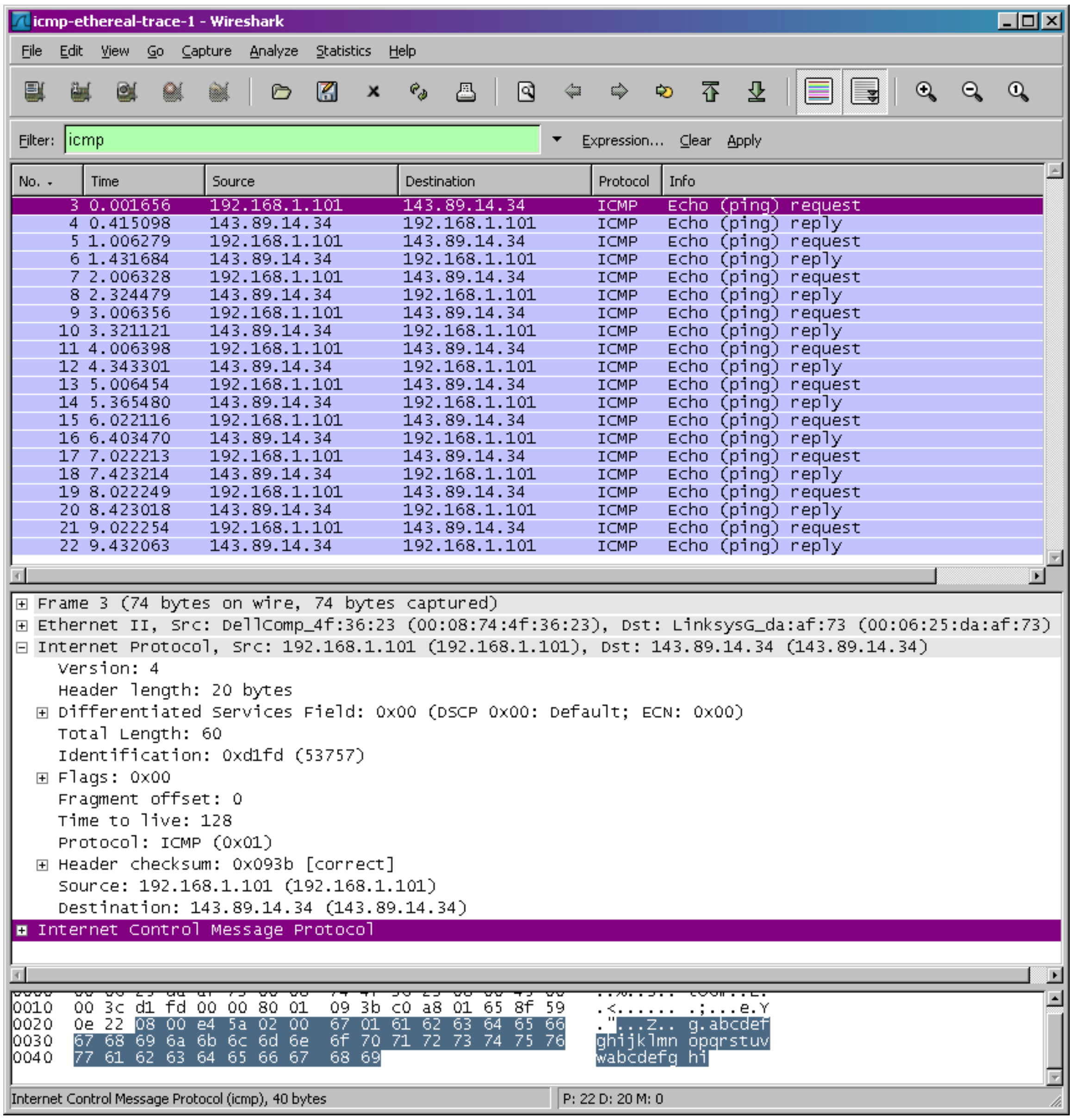
At the end of the experiment, your Command Prompt Window should look something like Figure 1. In this example, the source ping program is in Massachusetts and the destination Ping program is in Hong Kong. From this window we see that the source ping program sent 10 query packets and received 10 responses. Note also that for each response, the source calculates the round-trip time (RTT), which for the 10 packets is on average 375 msec.



```
C:\WINDOWS\SYSTEM32>ping -n 10 www.ust.hk

Pinging www.ust.hk [143.89.14.34] with 32 bytes of data:

Reply from 143.89.14.34: bytes=32 time=415ms TTL=231
Reply from 143.89.14.34: bytes=32 time=425ms TTL=231
Reply from 143.89.14.34: bytes=32 time=318ms TTL=231
Reply from 143.89.14.34: bytes=32 time=314ms TTL=231
Reply from 143.89.14.34: bytes=32 time=336ms TTL=231
Reply from 143.89.14.34: bytes=32 time=359ms TTL=231
Reply from 143.89.14.34: bytes=32 time=381ms TTL=231
Reply from 143.89.14.34: bytes=32 time=401ms TTL=231
Reply from 143.89.14.34: bytes=32 time=400ms TTL=231
Reply from 143.89.14.34: bytes=32 time=409ms TTL=231

Ping statistics for 143.89.14.34:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 314ms, Maximum = 425ms, Average = 375ms

C:\WINDOWS\SYSTEM32>
C:\WINDOWS\SYSTEM32>
C:\WINDOWS\SYSTEM32>_
```

Figure 1 Command Prompt window after entering Ping command.

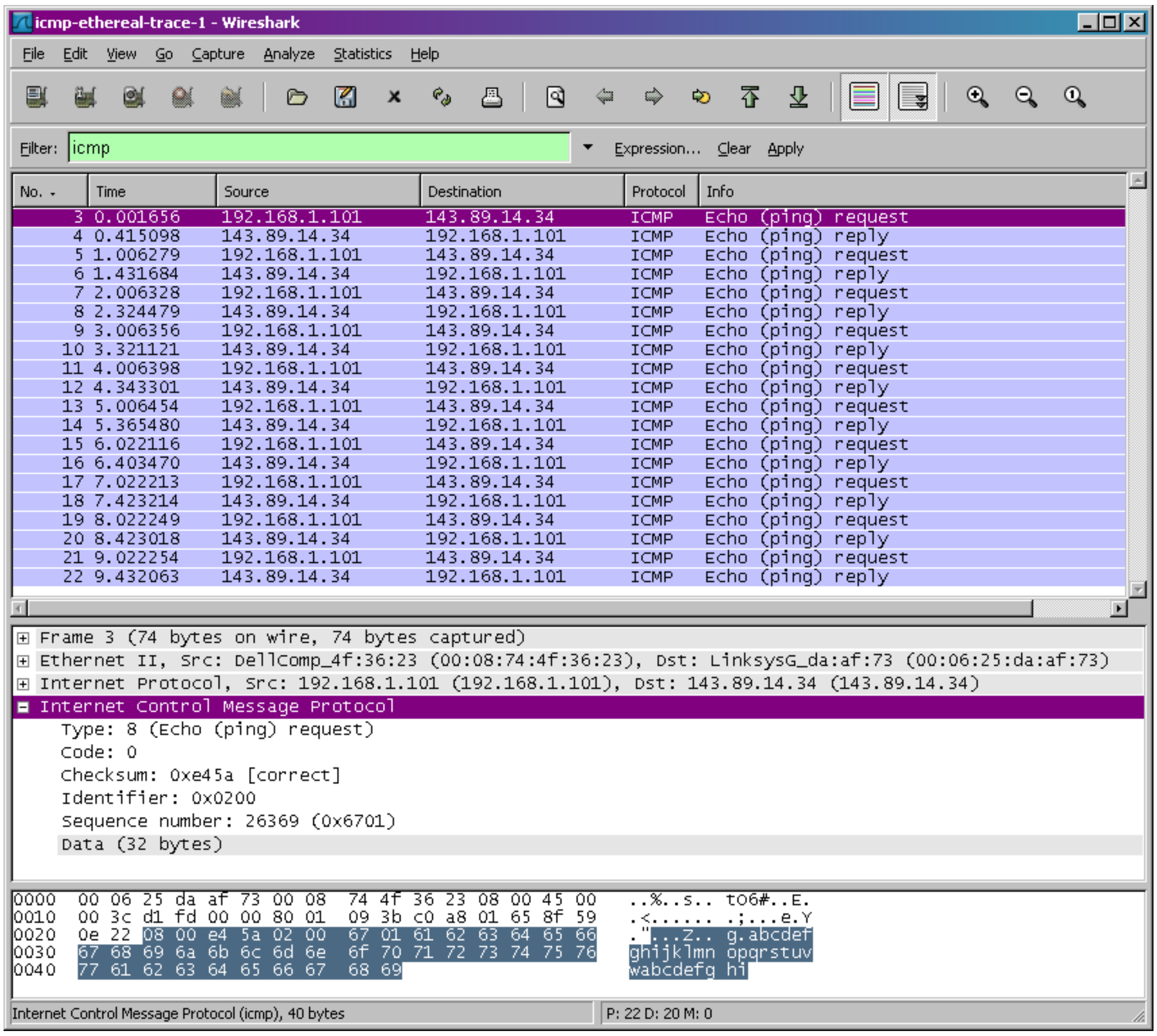
Figure 2 provides a screenshot of the Wireshark output, after "icmp" has been entered into the filter display window. Note that the packet listing shows 20 packets: the 10 Ping queries sent by the source and the 10 Ping responses received by the source. Also note that the source's IP address is a private address (behind a NAT) of the form 192.168/12; the destination's IP address is that of the Web server at HKUST. Now let's zoom in on the first packet (sent by the client); in the figure below, the packet contents area provides information about this packet. We see that the IP datagram within this packet has protocol number 01, which is the protocol number for ICMP. This means that the payload of the IP datagram is an ICMP packet.



```
icmp-ethereal-trace-1 - Wireshark
Filter: icmp

No.  Time      Source         Destination    Protocol Info
  3 0.001656  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  4 0.415098  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  5 1.006279  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  6 1.431684  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  7 2.006328  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  8 2.324479  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  9 3.006356  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 10 3.321121  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 11 4.006398  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 12 4.343301  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 13 5.006454  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 14 5.365480  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 15 6.022116  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 16 6.403470  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 17 7.022213  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 18 7.423214  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 19 8.022249  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 20 8.423018  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 21 9.022254  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 22 9.432063  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply

⊞ Frame 3 (74 bytes on wire, 74 bytes captured)
⊞ Ethernet II, Src: DellComp_4f:36:23 (00:08:74:4f:36:23), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
⊟ Internet Protocol, Src: 192.168.1.101 (192.168.1.101), Dst: 143.89.14.34 (143.89.14.34)
    Version: 4
    Header length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 60
    Identification: 0xd1fd (53757)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 128
    Protocol: ICMP (0x01)
  ⊞ Header checksum: 0x093b [correct]
    Source: 192.168.1.101 (192.168.1.101)
    Destination: 143.89.14.34 (143.89.14.34)
⊞ Internet Control Message Protocol

0010  00 3c d1 fd 00 00 80 01  09 3b c0 a8 01 65 8f 59   .<...... .;...e.Y
0020  0e 22 08 00 e4 5a 02 00  67 01 61 62 63 64 65 66   ."...Z.. g.abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi

Internet Control Message Protocol (icmp), 40 bytes    P: 22 D: 20 M: 0
```

Figure 2 Wireshark output for Ping program with Internet Protocol expanded.

Figure 3 focuses on the same ICMP but has expanded the ICMP protocol information in the packet contents window. Observe that this ICMP packet is of Type 8 and Code 0 - a so-called ICMP “echo request” packet. (See Figure 4.23 of text.) Also note that this ICMP packet contains a checksum, an identifier, and a sequence number.


icmp-ethereal-trace-1 - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: icmp
Expression... Clear Apply

```
No.  Time      Source         Destination    Protocol Info
  3 0.001656  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  4 0.415098  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  5 1.006279  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  6 1.431684  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  7 2.006328  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
  8 2.324479  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
  9 3.006356  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 10 3.321121  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 11 4.006398  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 12 4.343301  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 13 5.006454  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 14 5.365480  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 15 6.022116  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 16 6.403470  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 17 7.022213  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 18 7.423214  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 19 8.022249  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 20 8.423018  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply
 21 9.022254  192.168.1.101  143.89.14.34   ICMP     Echo (ping) request
 22 9.432063  143.89.14.34   192.168.1.101  ICMP     Echo (ping) reply

⊞ Frame 3 (74 bytes on wire, 74 bytes captured)
⊞ Ethernet II, Src: DellComp_4f:36:23 (00:08:74:4f:36:23), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
⊞ Internet Protocol, Src: 192.168.1.101 (192.168.1.101), Dst: 143.89.14.34 (143.89.14.34)
⊟ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xe45a [correct]
    Identifier: 0x0200
    Sequence number: 26369 (0x6701)
    Data (32 bytes)

0000  00 06 25 da af 73 00 08  74 4f 36 23 08 00 45 00   ..%..s.. tO6#..E.
0010  00 3c d1 fd 00 00 80 01  09 3b c0 a8 01 65 8f 59   .<...... .;...e.Y
0020  0e 22 08 00 e4 5a 02 00  67 01 61 62 63 64 65 66   ."...Z.. g.abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi

Internet Control Message Protocol (icmp), 40 bytes    P: 22 D: 20 M: 0
```

Figure 3 Wireshark capture of ping packet with ICMP packet expanded.

Activity 2:

Let's now continue our ICMP adventure by capturing the packets generated by the Traceroute program. You may recall that the Traceroute program can be used to figure out the path a packet takes from source to destination. Traceroute is discussed in Section 1.4 and in Section 4.4 of the text. Traceroute is implemented in different ways in Unix/Linux/MacOS and in Windows. In Unix/Linux, the source sends a series of UDP packets to the target destination using an unlikely destination port number; in Windows, the source sends a series of ICMP packets to the target destination. For both operating systems, the program sends the first packet with TTL=1, the second packet with TTL=2, and so on. Recall that a router will decrement a packet's TTL value as the packet passes through the router. When a packet arrives at a router with TTL=1, the router sends an ICMP error packet back to the source. In the following, we'll use the native Windows *tracert* program. A shareware version of a much nice Windows Traceroute program is *pingplotter* (www.pingplotter.com). We'll use *pingplotter* in our Wireshark IP lab since it provides additional functionality that we'll need there.

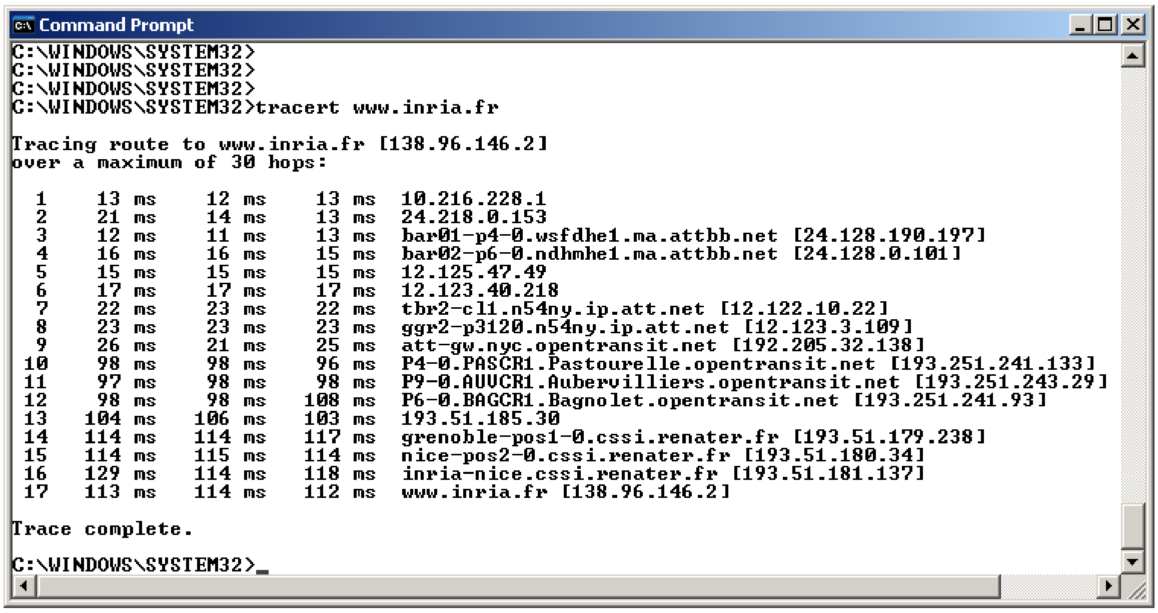
Solution:

1. Let's begin by opening the Windows Command Prompt application (which can be found in your Accessories folder).
2. Start up the Wireshark packet sniffer, and begin Wireshark packet capture.

]]>

3. The *tracert* command is in c:\windows\system32, so type either "*tracert hostname*" or "*c:\windows\system32\tracert hostname*" in the MS-DOS command line (without quotation marks), where hostname is a host on another continent. (Note that on a Windows machine, the command is "*tracert*" and not "*traceroute*".) If you're outside of Europe, you may want to enter www.inria.fr for the Web server at INRIA, a computer science research institute in France. Then run the Traceroute program by typing return.
4. When the Traceroute program terminates, stop packet capture in Wireshark.

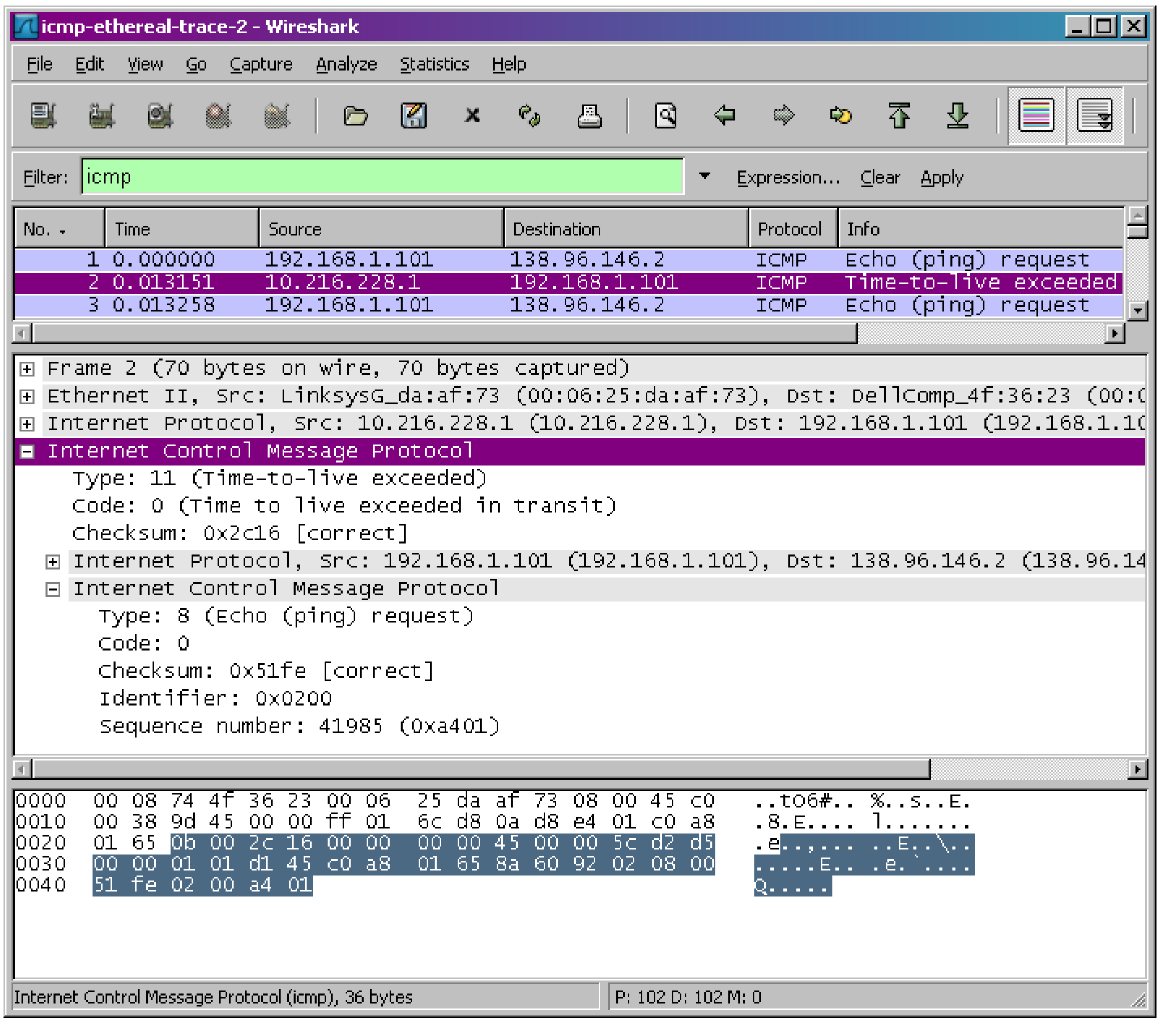
At the end of the experiment, your Command Prompt Window should look something like Figure 4. In this figure, the client Traceroute program is in Massachusetts and the target destination is in France. From this figure we see that for each TTL value, the source program sends three probe packets. Traceroute displays the RTTs for each of the probe packets, as well as the IP address (and possibly the name) of the router that returned the ICMP TTL-exceeded message.



```
C:\WINDOWS\SYSTEM32>
C:\WINDOWS\SYSTEM32>
C:\WINDOWS\SYSTEM32>
C:\WINDOWS\SYSTEM32>tracert www.inria.fr

Tracing route to www.inria.fr [138.96.146.2]
over a maximum of 30 hops:

  1    13 ms    12 ms    13 ms  10.216.228.1
  2    21 ms    14 ms    13 ms  24.218.0.153
  3    12 ms    11 ms    13 ms  bar01-p4-0.wsfdhe1.ma.attbb.net [24.128.190.197]
  4    16 ms    16 ms    15 ms  bar02-p6-0.ndhmhe1.ma.attbb.net [24.128.0.101]
  5    15 ms    15 ms    15 ms  12.125.47.49
  6    17 ms    17 ms    17 ms  12.123.40.218
  7    22 ms    23 ms    22 ms  tbr2-cl1.n54ny.ip.att.net [12.122.10.22]
  8    23 ms    23 ms    23 ms  ggr2-p3120.n54ny.ip.att.net [12.123.3.109]
  9    26 ms    21 ms    25 ms  att-gw.nyc.opentransit.net [192.205.32.138]
 10    98 ms    98 ms    96 ms  P4-0.PASCR1.Pastourelle.opentransit.net [193.251.241.133]
 11    97 ms    98 ms    98 ms  P9-0.AUVCR1.Aubervilliers.opentransit.net [193.251.243.29]
 12    98 ms    98 ms   108 ms  P6-0.BAGCR1.Bagnolet.opentransit.net [193.251.241.93]
 13   104 ms   106 ms   103 ms  193.51.185.30
 14   114 ms   114 ms   117 ms  grenoble-pos1-0.cssi.renater.fr [193.51.179.238]
 15   114 ms   115 ms   114 ms  nice-pos2-0.cssi.renater.fr [193.51.180.34]
 16   129 ms   114 ms   118 ms  inria-nice.cssi.renater.fr [193.51.181.137]
 17   113 ms   114 ms   112 ms  www.inria.fr [138.96.146.2]

Trace complete.

C:\WINDOWS\SYSTEM32>_
```

Figure 4 Command Prompt window displays the results of the Traceroute program.

Figure 5 displays the Wireshark window for an ICMP packet returned by a router. Note that this ICMP error packet contains many more fields than the Ping ICMP messages.


icmp-ethereal-trace-2 - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: icmp Expression... Clear Apply

```
No.   Time      Source          Destination     Protocol  Info
  1 0.000000    192.168.1.101   138.96.146.2    ICMP      Echo (ping) request
  2 0.013151    10.216.228.1    192.168.1.101   ICMP      Time-to-live exceeded
  3 0.013258    192.168.1.101   138.96.146.2    ICMP      Echo (ping) request
```

```
⊞ Frame 2 (70 bytes on wire, 70 bytes captured)
⊞ Ethernet II, Src: LinksysG_da:af:73 (00:06:25:da:af:73), Dst: DellComp_4f:36:23 (00:0
⊞ Internet Protocol, Src: 10.216.228.1 (10.216.228.1), Dst: 192.168.1.101 (192.168.1.10
⊟ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0x2c16 [correct]
  ⊞ Internet Protocol, Src: 192.168.1.101 (192.168.1.101), Dst: 138.96.146.2 (138.96.14
  ⊟ Internet Control Message Protocol
      Type: 8 (Echo (ping) request)
      Code: 0
      Checksum: 0x51fe [correct]
      Identifier: 0x0200
      Sequence number: 41985 (0xa401)
```

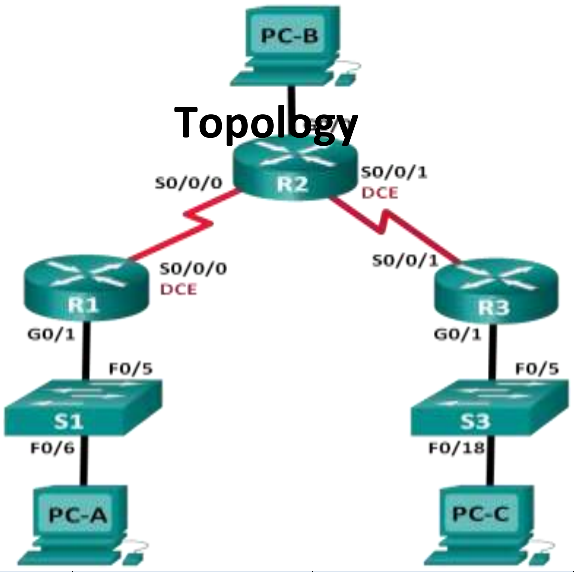
```
0000  00 08 74 4f 36 23 00 06  25 da af 73 08 00 45 c0   ..tO6#.. %..s..E.
0010  00 38 9d 45 00 00 ff 01  6c d8 0a d8 e4 01 c0 a8   .8.E.... l.......
0020  01 65 0b 00 2c 16 00 00  00 00 45 00 00 5c d2 d5   .e..,... ..E..\..
0030  00 00 01 01 d1 45 c0 a8  01 65 8a 60 92 02 08 00   .....E.. .e.`....
0040  51 fe 02 00 a4 01                                  Q.....
```

Internet Control Message Protocol (icmp), 36 bytes   P: 102 D: 102 M: 0

Figure 5 Wireshark window of ICMP fields expanded for one ICMP error packet.

Activity 3
Lab – Dynamic Routing using RIP

Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway

R1	G0/1	172.30.10.1	255.255.255.0	N/A
	S0/0/0 (DCE)	10.1.1.1	255.255.255.252	N/A
R2	G0/0	209.165.201.1	255.255.255.0	N/A
	S0/0/0	10.1.1.2	255.255.255.252	N/A
	S0/0/1 (DCE)	10.2.2.2	255.255.255.252	N/A
R3	G0/1	172.30.30.1	255.255.255.0	N/A
	S0/0/1	10.2.2.1	255.255.255.252	N/A
S1	N/A	VLAN 1	N/A	N/A
S3	N/A	VLAN 1	N/A	N/A
PC-A	NIC	172.30.10.3	255.255.255.0	172.30.10.1
PC-B	NIC	209.165.201.2	255.255.255.0	209.165.201.1
PC-C	NIC	172.30.30.3	255.255.255.0	172.30.30.1

Objectives

Part 1: Build the Network and Configure Basic Device Settings

Part 2: Configure and Verify RIPv2 Routing

- Configure and verify RIPv2 is running on routers.
- Configure a passive interface.
- Examine routing tables.
- Disable automatic summarization.
- Configure a default route.
- Verify end-to-end connectivity.

Background / Scenario

- RIP version 2 (RIPv2) is used for routing of IPv4 addresses in small networks. RIPv2 is a classless, distancevector routing protocol, as defined by RFC 1723. Because RIPv2 is a classless routing protocol, subnet masks are included in the routing updates. By default, RIPv2 automatically summarizes networks at major network boundaries. When automatic summarization has been disabled, RIPv2 no longer summarizes networks to their classful address at boundary routers.
- RIPng (RIP Next Generation) is a distance-vector routing protocol for routing IPv6 addresses, as defined by RFC 2080. RIPng is based on RIPv2 and has the same administrative distance and 15-hop limitation.
- In this lab, you will configure the network topology with RIPv2 routing, disable automatic summarization, propagate a default route, and use CLI commands to display and verify RIP routing information.

Required Resources

- 3 Routers (Cisco 1941)
- 2 Switches (Cisco 2960)
- 3 PCs (Windows 7, Vista, or XP with terminal emulation program, such as Tera Term)

- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet and Serial cables as shown in the topology

Part 1: Build the Network and Configure Basic Device Settings

- In Part 1, you will set up the network topology and configure basic settings.

Step 1: Cable the network as shown in the topology.

Step 2: Initialize and reload the router and switch.

Step 3: Configure basic settings for each router and switch.

- Disable DNS lookup.
- Configure device names as shown in the topology.
- Configure password encryption.
- Assign **class** as the privileged EXEC password.
- Assign **cisco** as the console and vty passwords.
- Configure a MOTD banner to warn users that unauthorized access is prohibited.
- Configure **logging synchronous** for the console line.
- Configure the IP address listed in the Addressing Table for all interfaces.
- Configure a description to each interface with an IP address.
- Configure the clock rate if applicable to the DCE serial interface.
- Copy the running-configuration to the startup-configuration.

Step 4: Configure PC hosts.

Refer to the Addressing Table for PC host address information.

Step 5: Test connectivity.

At this point, the PCs are unable to ping each other.

a. Each workstation should be able to ping the attached router. Verify and troubleshoot if necessary.

b. The routers should be able to ping one another. Verify and troubleshoot if necessary.

Part 2: Configure and Verify RIPv2 Routing

You will configure RIPv2 routing on all routers in the network and then verify that routing tables are updated correctly. After RIPv2 has been verified, you will disable automatic summarization, configure a default route, and verify end-to-end connectivity.

Step 1: Configure RIPv2 routing.

a. On R1, configure RIPv2 as the routing protocol and advertise the appropriate networks.

```
R1# config t
R1(config)# router rip
R1(config-router)#version 2
R1(config-router)#network 172.30.0.0
R1(config-router)#network 10.0.0.0
```

b. Configure RIPv2 on R3 and use the **network** statement to add appropriate networks and prevent routing updates on the LAN interface.

c. Configure RIPv2 on R2. Do not advertise the 209.165.201.0 network.

Step 2: Examine current state of network.

a. The status of the two serial links can quickly be verified using the **show ip interface brief** command on R2.

```
R2# show ip interface brief
Interface                IP-Address      OK? Method Status Protocol
Embedded-Service-Engine0/0 unassigned      YES unset administratively down down
GigabitEthernet0/0       209.165.201.1   YES manual up                 up
GigabitEthernet0/1       unassigned      YES unset  administratively down down
Serial0/0/0              10.1.1.2        YES manual up                 up
Serial0/0/1              10.2.2.2        YES manual up                 up
```

b. Check connectivity between PCs.

From PC-A, is it possible to ping PC-B? _________ Why?

No, R2 is not advertising the route to PC-B.

From PC-A, is it possible to ping PC-C? _________ Why?

No, R1 and R3 do not have routes to the specific subnets on the remote router.

From PC-C, is it possible to ping PC-B? _________ Why?

No, R2 is not advertising the route to PC-B.

From PC-C, is it possible to ping PC-A? _________ Why?

No, R1 and R3 do not have routes to the specific subnets on the remote router.

c. Verify that RIPv2 is running on the routers.

You can use the **debug ip rip, show ip protocols**, and **show run** commands to confirm that RIPv2 is running. The **show ip protocols** command output for R1 is shown below.

```
R1# show ip protocols
Routing Protocol is "rip"
Outgoing update filter list for all interfaces is not set
Incoming update filter list for all interfaces is not set
Sending updates every 30 seconds, next due in 7 seconds
Invalid after 180 seconds, hold down 180, flushed after 240
Redistributing: rip
Default version control: send version 2, receive 2
  Interface             Send  Recv  Triggered RIP  Key-chain
  Serial0/0/0           2     2
Automatic network summarization is in effect
Maximum path: 4
Routing for Networks:
  10.0.0.0
  172.30.0.0
Passive Interface(s):
    GigabitEthernet0/1
Routing Information Sources:
  Gateway         Distance      Last Update
  10.1.1.2             120
Distance: (default is 120)
```

When issuing the **debug ip rip** command on R2, what information is provided that confirms RIPv2 is running?

____

RIP: sending v2 updates to 224.0.0.9 via Serial 0/0/0 (10.1.1.2).

When you are finished observing the debugging outputs, issue the **undebug all** command at the privileged EXEC prompt.

When issuing the **show run** command on R3, what information is provided that confirms RIPv2 is running?

____

```
router
rip
version 2
```

d. Examine the automatic summarization of routes.

The LANs connected to R1 and R3 are composed of discontiguous networks. R2 displays two equal-cost paths to the 172.30.0.0/16 network in the routing table. R2 displays only the major classful network address of 172.30.0.0 and does not display any of the subnets for this network.

```
R2# show ip route
<Output omitted>
      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.1.0/30 is directly connected, Serial0/0/0
L        10.1.1.2/32 is directly connected, Serial0/0/0
C        10.2.2.0/30 is directly connected, Serial0/0/1
L        10.2.2.2/32 is directly connected, Serial0/0/1
R     172.30.0.0/16 [120/1] via 10.2.2.1, 00:00:23, Serial0/0/1
                    [120/1] via 10.1.1.1, 00:00:09, Serial0/0/0
      209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C        209.165.201.0/24 is directly connected, GigabitEthernet0/0
L        209.165.201.1/32 is directly connected, GigabitEthernet0/0
```

R1 displays only its own subnets for the 172.30.0.0 network. R1 does not have any routes for the 172.30.0.0 subnets on R3.

```
R1# show ip route
<Output omitted>
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.1.0/30 is directly connected, Serial0/0/0
L        10.1.1.1/32 is directly connected, Serial0/0/0
R        10.2.2.0/30 [120/1] via 10.1.1.2, 00:00:21, Serial0/0/0
      172.30.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.30.10.0/24 is directly connected, GigabitEthernet0/1
L        172.30.10.1/32 is directly connected, GigabitEthernet0/1
```

R3 only displays its own subnets for the 172.30.0.0 network. R3 does not have any routes for the 172.30.0.0 subnets on R1.

```
R3# show ip route
<Output omitted>
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
```

```
C        10.2.2.0/30 is directly connected, Serial0/0/1
L        10.2.2.1/32 is directly connected, Serial0/0/1
R        10.1.1.0/30 [120/1] via 10.2.2.2, 00:00:23, Serial0/0/1
      172.30.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.30.30.0/24 is directly connected, GigabitEthernet0/1
L        172.30.30.1/32 is directly connected, GigabitEthernet0/1
```

Use the **debug ip rip** command on R2 to determine the routes received in the RIP updates from R3 and list them here.

____172.30.0.0/16

R3 is not sending any of the 172.30.0.0 subnets, only the summarized route of 172.30.0.0/16, including the subnet mask. Therefore, the routing tables on R1 and R2 do not display the 172.30.0.0 subnets on R3.

Step 3: Disable automatic summarization.

The **no auto-summary** command is used to turn off automatic summarization in RIPv2. Disable auto summarization on all routers. The routers will no longer summarize routes at major classful network boundaries. R1 is shown here as an example.

R1(config)# router rip

R1(config-router)#no auto-summary

a. Issue the**clear ip route ***command to clear the routing table.

R1(config-router)#**end**

```
R1# clear ip route *
```

c. Examine the routing tables. Remember will it take some time to converge the routing tables after clearing them.

The LAN subnets connected to R1 and R3 should now be included in all three routing tables.

```
R2# show ip route
<Output omitted>
Gateway of last resort is not set
      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.1.0/30 is directly connected, Serial0/0/0
L        10.1.1.2/32 is directly connected, Serial0/0/0
C        10.2.2.0/30 is directly connected, Serial0/0/1
L        10.2.2.2/32 is directly connected, Serial0/0/1
172.30.0.0/16 is variably subnetted, 3 subnets, 2 masks
R        172.30.0.0/16 [120/1] via 10.2.2.1, 00:01:01, Serial0/0/1
                       [120/1] via 10.1.1.1, 00:01:15, Serial0/0/0
R        172.30.10.0/24 [120/1] via 10.1.1.1, 00:00:21, Serial0/0/0
R        172.30.30.0/24 [120/1] via 10.2.2.1, 00:00:04, Serial0/0/1
209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C        209.165.201.0/24 is directly connected, GigabitEthernet0/0
L        209.165.201.1/32 is directly connected, GigabitEthernet0/0
R1# show ip route
<Output omitted>
Gateway of last resort is not set
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.1.0/30 is directly connected, Serial0/0/0
L        10.1.1.1/32 is directly connected, Serial0/0/0
```

```
R        10.2.2.0/30 [120/1] via 10.1.1.2, 00:00:12, Serial0/0/0
172.30.0.0/16 is variably subnetted, 3 subnets, 2 masks
C        172.30.10.0/24 is directly connected, GigabitEthernet0/1
L        172.30.10.1/32 is directly connected, GigabitEthernet0/1
R        172.30.30.0/24 [120/2] via 10.1.1.2, 00:00:12, Serial0/0/0
```

```
R3# show ip route
<Output omitted>
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.2.2.0/30 is directly connected, Serial0/0/1
L        10.2.2.1/32 is directly connected, Serial0/0/1
R        10.1.1.0/30 [120/1] via 10.2.2.2, 00:00:23, Serial0/0/1
      172.30.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.30.30.0/24 is directly connected, GigabitEthernet0/1
L        172.30.30.1/32 is directly connected, GigabitEthernet0/1
R        172.30.10.0 [120/2] via 10.2.2.2, 00:00:16, Serial0/0/1
```

d. Use the**debug ip rip**command on R2 to exam the RIP updates.

```
R2# debug ip rip
```

After 60 seconds, issue the **no debug ip rip** command.

What routes are in the RIP updates that are received from R3?

____

_______ 172.30.30.0/24

Are the subnet masks now included in the routing updates? __________ yes

Step 4: Configure and redistribute a default route for Internet access.

a. From R2, create a static route to network 0.0.0.0 0.0.0.0, using the **ip route** command. This forwards any unknown destination address traffic to the R2 G0/0 toward PC-B, simulating the Internet by setting a Gateway of Last Resort on the R2 router.

```
R2(config)# ip route 0.0.0.0 0.0.0.0 209.165.201.2
```

b. R2 will advertise a route to the other routers if the**default-information originate**command is added to its RIP configuration.

```
R2(config)# router rip
R2(config-router)#default-information originate
```

Step 5: Verify the routing configuration.

a. View the routing table on R1.

```
R1# show ip route
<Output omitted>
Gateway of last resort is 10.1.1.2 to network 0.0.0.0

R*    0.0.0.0/0 [120/1] via 10.1.1.2, 00:00:13, Serial0/0/0
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.1.0/30 is directly connected, Serial0/0/0
L        10.1.1.1/32 is directly connected, Serial0/0/0
R        10.2.2.0/30 [120/1] via 10.1.1.2, 00:00:13, Serial0/0/0
      172.30.0.0/16 is variably subnetted, 3 subnets, 2 masks
```

```
C        172.30.10.0/24 is directly connected, GigabitEthernet0/1
L        172.30.10.1/32 is directly connected, GigabitEthernet0/1
R        172.30.30.0/24 [120/2] via 10.1.1.2, 00:00:13, Serial0/0/0
```

b. How can you tell from the routing table that the subnetted network shared by R1 and R3 has a pathway for Internet traffic?

____

There is a Gateway of Last Resort, and the default route shows up in the table as being learned via RIP.

c. View the routing table on R2.

d. How is the pathway for Internet traffic provided in its routing table?

____

R2 has a static default route to 0.0.0.0 via 209.165.201.2, which is directly connected to G0/0.

Step 6: Verify connectivity.

e. Simulate sending traffic to the Internet by pinging from PC-A and PC-C to 209.165.201.2.

f. Were the pings successful? ______ Yes

g. Verify that hosts within the subnetted network can reach each other by pinging between PC-A and PC-C.

h. Were the pings successful? ______ Yes

Note: It may be necessary to disable the PCs firewall.

Reflection

1. Why would you turn off automatic summarization for RIPv2?

____

So the routers will no longer summarize routes at major classful network boundaries.

2. In both scenarios, how did R1 and R3 learn the pathway to the Internet?

____

From RIP routing updatesreceived from the router where the default route was configured (R2).

3. How are configuring RIPv2 and RIPng different?

____ Answers may vary. RIPv2 is configured through network statements where RIPng is configured on interfaces.

3) Stage v (verify)

Home Activities:

Activity 1:

You should hand in a screen shot of the Command Prompt window similar to Figure 1 above. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[18] to explain your answer. To print a packet, use *File->Print*, choose *selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

You should answer the following questions:

1. What is the IP address of your host? What is the IP address of the destination host?
2. Why is it that an ICMP packet does not have source and destination port numbers?
3. Examine one of the ping request packets sent by your host. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?
4. Examine the corresponding ping reply packet. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?

What to Hand In:

For this part of the lab, you should hand in a screen shot of the Command Prompt window. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout to explain your answer. To print a packet, use *File->Print*, choose *selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

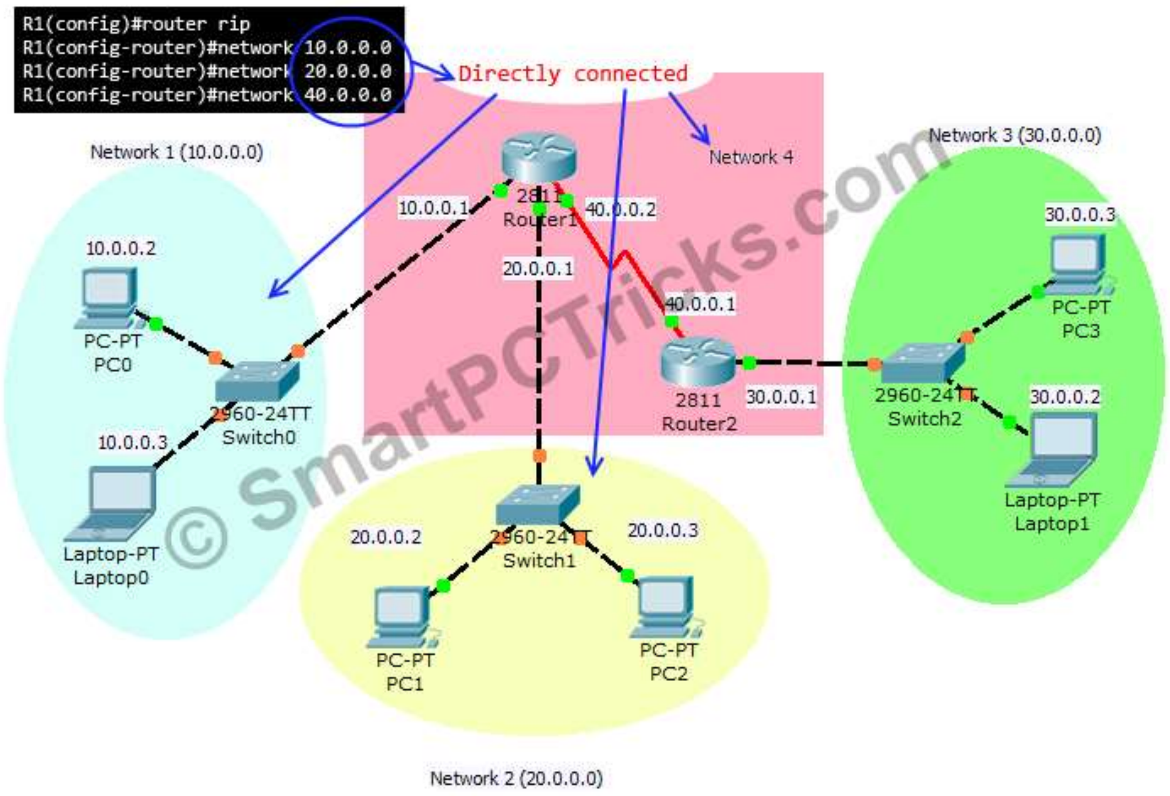
Answer the following questions:

5. What is the IP address of your host? What is the IP address of the target destination host?
6. If ICMP sent UDP packets instead (as in Unix/Linux), would the IP protocol number still be 01 for the probe packets? If not, what would it be?
7. Examine the ICMP echo packet in your screenshot. Is this different from the ICMP ping query packets in the first half of this lab? If yes, how so?
8. Examine the ICMP error packet in your screenshot. It has more fields than the ICMP echo packet. What is included in those fields?
9. Examine the last three ICMP packets received by the source host. How are these packets different from the ICMP error packets? Why are they different?
10. Within the tracert measurements, is there a link whose delay is significantly longer than others? Refer to the screenshot in Figure 4, is there a link whose delay is significantly longer than others? On the basis of the router names, can you guess the location of the two routers on the end of this link?

Activity 2:

Configure the network with dynamic routing with mention networks address.

[18] What do we mean by "annotate"? If you hand in a paper copy, please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate.



```
R1(config)#router rip
R1(config-router)#network 10.0.0.0
R1(config-router)#network 20.0.0.0
R1(config-router)#network 40.0.0.0
```

Router1: 10.0.0.1 255.0.0.0

Router1: 20.0.0.1 255.0.0.0

Router1: 40.0.0.2 255.0.0.0

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

LAB # 11

Statement Purpose:

DHCP is used extensively in corporate, university and home-network wired and wireless LANs to dynamically assign IP addresses to hosts (as well as to configure other network configuration information). In this lab, students will dig deep into the working of DHCP

Activity Outcomes:

Students will gain better understanding of DHCP

Instructor Note:

1) StageJ(Journey)

Introduction

Recall that DHCP is normally used to assign a computer its IP address, as well as other parameters such as the address of the local router. Your computer, the client, uses the DHCP protocol to communicate with a DHCP server on the local network. Other computers on the local network also interact with the DHCP server. In deployments, there are several variations. For example, the local agent may be a DHCP relay that relays messages between local computers and a remote DHCP server. Or the DHCP server may be replicated for reliability, so that there are two or more local DHCP servers. For our purposes, it is sufficient to think about a single DHCP server.

The complete DHCP exchange involves four types of packets: Discover, for your computer to locate the DHCP server; Offer, for the server to offer an IP address; Request, for your computer to ask for an offered address; and Ack, for the server to grant the address lease. However, when a computer is reestablishing its IP address on a network that it has previously used, it may perform a short exchange involving only two types of DHCP packets: Request, to ask for the same IP address as from the same server as was used before; and ACK for the server to grant the address lease.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

In this section you are expected to set up your own DHCP server.

Solution:

In order to observe DHCP in action, we'll perform several DHCP-related commands and capture the DHCP messages exchanged as a result of executing these commands. Do the following19:

[19] If you are unable to run Wireshark live on a computer, you can download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file *dhcp-ethereal-trace-1*. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the dhcp-ethereal-trace-1 trace file. You can then use this trace file to answer the questions below.

Begin by opening the Windows Command Prompt application (which can be found in your Accessories folder). As shown in Figure 1, enter “ipconfig /release”. The executable for ipconfig is in C:\windows\system32. This command releases your current IP address, so that your host’s IP address becomes 0.0.0.0.

Start up the Wireshark packet sniffer, as described in the introductory Wireshark lab and begin Wireshark packet capture.

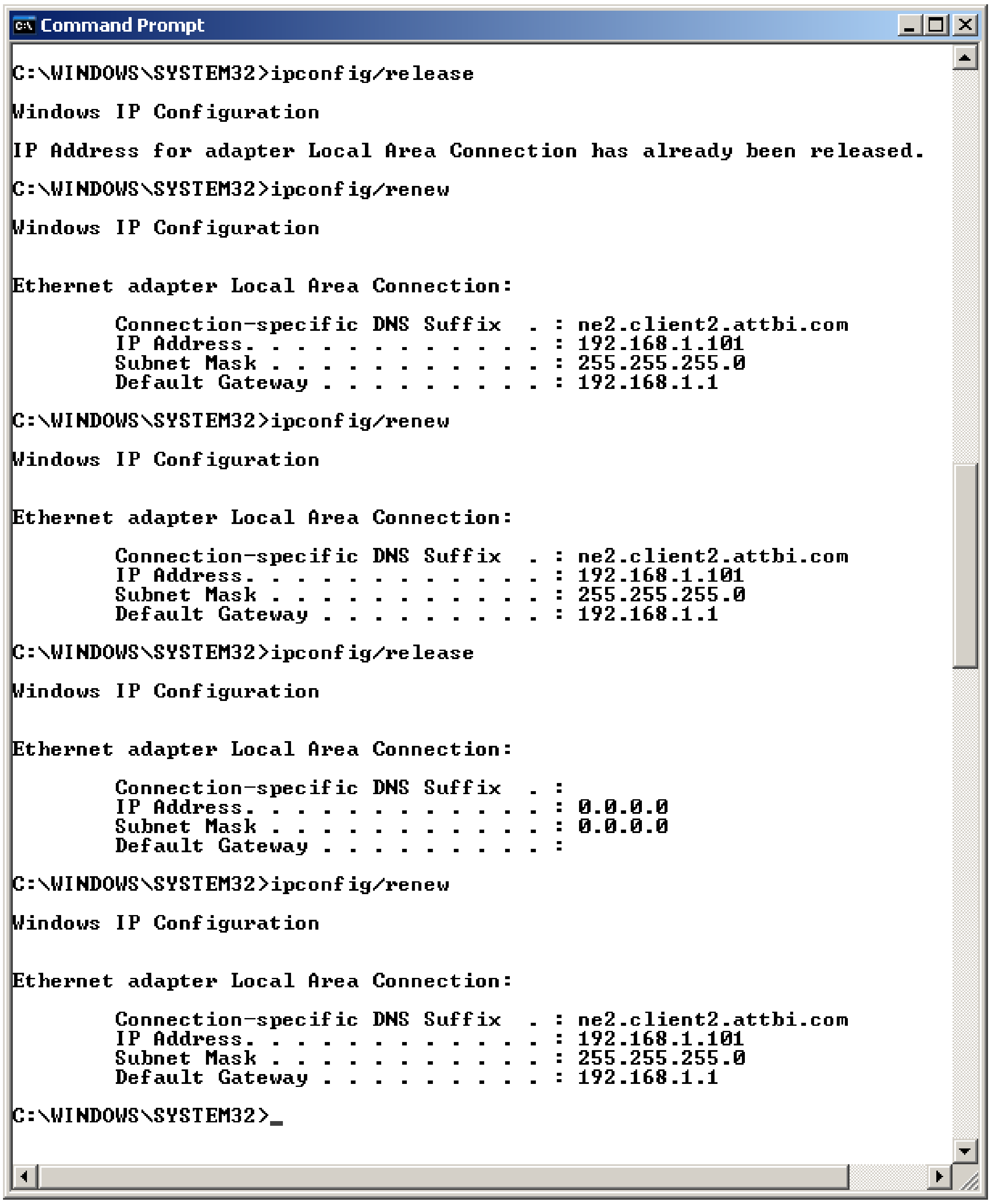
Now go back to the Windows Command Prompt and enter “ipconfig /renew”. This instructs your host to obtain a network configuration, including a new IP address. In Figure 1, the host obtains the IP address 192.168.1.108

Wait until the “ipconfig /renew” has terminated. Then enter the same command “ipconfig /renew” again.

When the second “ipconfig /renew” terminates, enter the command “ipconfig/release” to release the previously-allocated IP address to your computer.

Finally, enter “ipconfig /renew” to again be allocated an IP address for your computer.

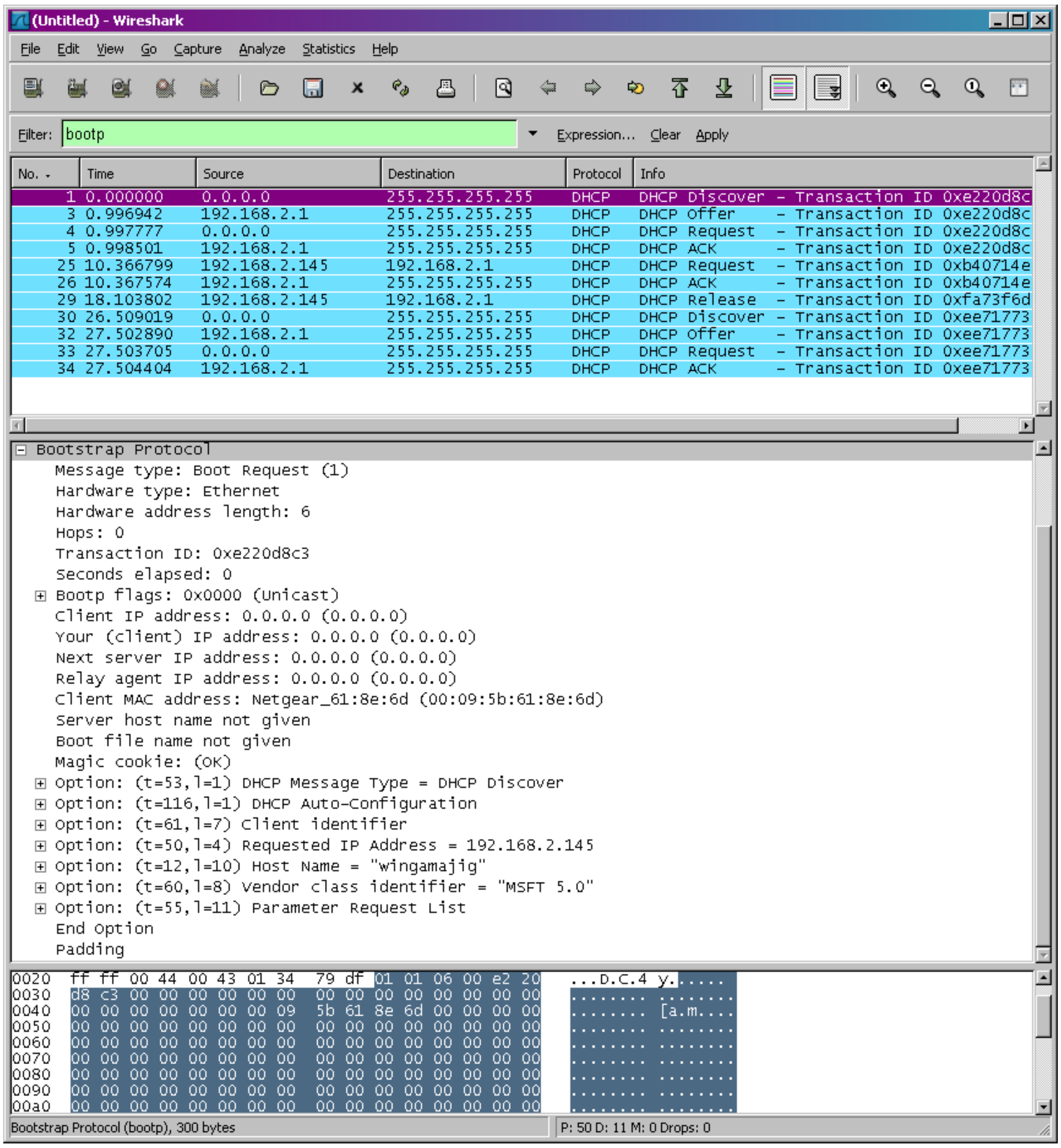
Stop Wireshark packet capture.



```
C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration

IP Address for adapter Local Area Connection has already been released.

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 0.0.0.0
        Subnet Mask . . . . . . . . . . . : 0.0.0.0
        Default Gateway . . . . . . . . . :

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>_
```

Figure 1 Command Prompt window showing sequence of *ipconfig* commands that you should enter.

Now let's take a look at the resulting Wireshark window. To see only the DHCP packets, enter into the filter field "bootp". (DHCP derives from an older protocol called BOOTP. Both BOOTP and DHCP use the same port numbers, 67 and 68. To see DHCP packets in the current version of Wireshark, you need to enter "bootp" and not "dhcp" in the filter.)
We see from Figure 2 that the first *ipconfig* renew command caused four DHCP packets to be generated: a DHCP Discover packet, a DHCP Offer packet, a DHCP Request packet, and a DHCP ACK packet.


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: bootp
Expression... Clear Apply

```
No. .  Time        Source          Destination      Protocol  Info
   1 0.000000    0.0.0.0         255.255.255.255  DHCP      DHCP Discover  - Transaction ID 0xe220d8c
   3 0.996942    192.168.2.1     255.255.255.255  DHCP      DHCP Offer     - Transaction ID 0xe220d8c
   4 0.997777    0.0.0.0         255.255.255.255  DHCP      DHCP Request   - Transaction ID 0xe220d8c
   5 0.998501    192.168.2.1     255.255.255.255  DHCP      DHCP ACK       - Transaction ID 0xe220d8c
  25 10.366799   192.168.2.145   192.168.2.1      DHCP      DHCP Request   - Transaction ID 0xb40714e
  26 10.367574   192.168.2.1     255.255.255.255  DHCP      DHCP ACK       - Transaction ID 0xb40714e
  29 18.103802   192.168.2.145   192.168.2.1      DHCP      DHCP Release   - Transaction ID 0xfa73f6d
  30 26.509019   0.0.0.0         255.255.255.255  DHCP      DHCP Discover  - Transaction ID 0xee71773
  32 27.502890   192.168.2.1     255.255.255.255  DHCP      DHCP Offer     - Transaction ID 0xee71773
  33 27.503705   0.0.0.0         255.255.255.255  DHCP      DHCP Request   - Transaction ID 0xee71773
  34 27.504404   192.168.2.1     255.255.255.255  DHCP      DHCP ACK       - Transaction ID 0xee71773
```

```
⊟ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xe220d8c3
    Seconds elapsed: 0
  ⊞ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
    Server host name not given
    Boot file name not given
    Magic cookie: (OK)
  ⊞ Option: (t=53,l=1) DHCP Message Type = DHCP Discover
  ⊞ Option: (t=116,l=1) DHCP Auto-Configuration
  ⊞ Option: (t=61,l=7) Client identifier
  ⊞ Option: (t=50,l=4) Requested IP Address = 192.168.2.145
  ⊞ Option: (t=12,l=10) Host Name = "wingamajig"
  ⊞ Option: (t=60,l=8) Vendor class identifier = "MSFT 5.0"
  ⊞ Option: (t=55,l=11) Parameter Request List
    End Option
    Padding
```

```
0020  ff ff 00 44 00 43 01 34  79 df 01 01 06 00 e2 20   ...D.C.4 y.......
0030  d8 c3 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 00 09  5b 61 8e 6d 00 00 00 00   ........ [a.m....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

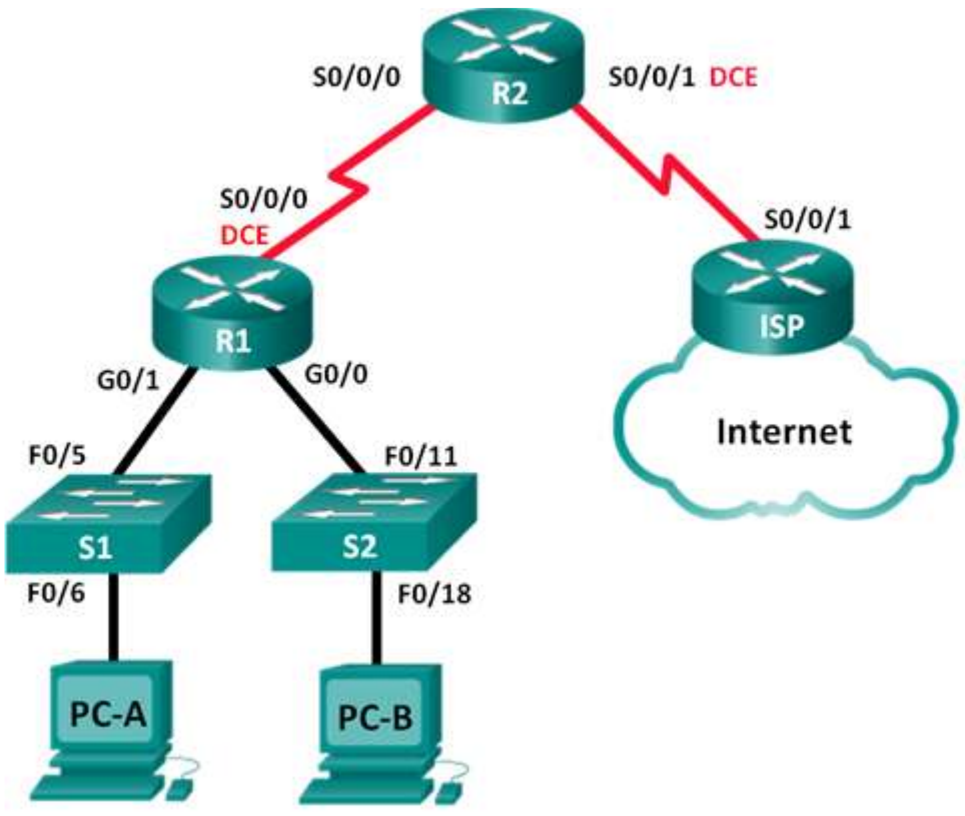
```
Bootstrap Protocol (bootp), 300 bytes        P: 50 D: 11 M: 0 Drops: 0
```

Figure 2 Wireshark window with first DHCP packet – the DHCP Discover packet – expanded.

Activity 2 (Packet Tracer part)

Configuring Basic DHCPv4 on a Router

Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway
R1	G0/0	192.168.0.1	255.255.255.0	N/A
	G0/1	192.168.1.1	255.255.255.0	N/A
	S0/0/0 (DCE)	192.168.2.253	255.255.255.252	N/A
R2	S0/0/0	192.168.2.254	255.255.255.252	N/A
PC-A	NIC	DHCP	DHCP	DHCP
PC-B	NIC	DHCP	DHCP	DHCP

Objectives

Part 1: Build the Network and Configure Basic Device Settings

Part 2: Configure a DHCPv4 Server and a DHCP Relay Agent

Background / Scenario

The Dynamic Host Configuration Protocol (DHCP) is a network protocol that lets network administrators manage and automate the assignment of IP addresses. Without DHCP, the administrator must manually assign and configure IP addresses, preferred DNS servers, and default gateways. As the network grows in size, this becomes an administrative problem when devices are moved from one internal network to another.

In this scenario, the company has grown in size, and the network administrators can no longer assign IP addresses to devices manually. Your job is to configure the R2 router to assign IPv4 addresses on two different subnets connected to router R1..

Required Resources

- 3 Routers (Cisco 1941)
- 2 Switches (Cisco 2960)
- 2 PCs (Windows 7, Vista, or XP with terminal emulation program, such as Tera Term)
- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet and serial cables as shown in the topology

Part 1: Build the Network and Configure Basic Device Settings

In Part 1, you will set up the network topology and configure the routers and switches with basic settings, such as passwords and IP addresses. You will also configure the IP settings for the PCs in the topology.

Step 1: Cable the network as shown in the topology.

Step 2: Initialize and reload the routers and switches.

Step 3: Configure basic settings for each router.

a. Disable DNS lookup

b. Configure the device name as shown in the topology.

c. Assign **class** as the encrypted privileged EXEC mode password.

d. Assign **cisco** as the console and vty passwords.

e. Configure **logging synchronous** to prevent console messages from interrupting command entry.

f. Configure the IP addresses for all the router interfaces according to the Addressing Table.

g. Configure the serial DCE interface on R1 with a clock rate of 128000.

h. Copy the running configuration to the startup configuration.

Part 2: Configure a DHCPv4 Server and a DHCP Relay Agent

To automatically assign address information on the network, you will configure R2 as a DHCPv4 server and R1 as a DHCP relay agent.

Step 1: Configure DHCPv4 server settings on router R2.

On R2, you will configure a DHCP address pool for each of the R1 LANs. Use the pool name **R1G0** for the G0/0 LAN and **R1G1** for the G0/1 LAN. You will also configure the addresses to be excluded from the address pools. Best practice dictates that excluded addresses be configured first, to guarantee that they are not accidentally leased to other devices.

Exclude the first 9 addresses in each R1 LAN starting with .1. All other addresses should be available in the DHCP address pool. Make sure that each DHCP address pool includes a default gateway, the domain **ccnalab.com**, a DNS server (209.165.200.225), and a lease time of 2 days. On the lines below, write the commands necessary for configuring DHCP services on router R2, including the DHCP-excluded addresses and the DHCP address pools.

____

______

```
R2(config)# ipdhcp excluded-address 192.168.0.1 192.168.0.9
R2(config)# ipdhcp excluded-address 192.168.1.1 192.168.1.9
R2(config)# ipdhcp pool R1G1
R2(dhcp-config)#network 192.168.1.0 255.255.255.0
R2(dhcp-config)#default-router 192.168.1.1
R2(dhcp-config)#dns-server 209.165.200.225
R2(dhcp-config)#domain-name ccna-lab.com
R2(dhcp-config)#lease 2
R2(dhcp-config)#exit
R2(config)# ipdhcp pool R1G0
R2(dhcp-config)#network 192.168.0.0 255.255.255.0
R2(dhcp-config)#default-router 192.168.0.1
R2(dhcp-config)#dns-server 209.165.200.225
R2(dhcp-config)# domain-name ccna-lab.com
R2(dhcp-config)#lease 2
```

On PC-A or PC-B, open a command prompt and enter the **ipconfig /all** command. Did either of the host PCs receive an IP address from the DHCP server? Why?

____

______

The host computers will not have received IP addresses from the DHCP server at R2 until R1 is configured as a DHCP relay agent

Step 2: Configure R1 as a DHCP relay agent.

Configure IP helper addresses on R1 to forward all DHCP requests to the R2 DHCP server.
On the lines below, write the commands necessary to configure R1 as a DHCP relay agent for the R1 LANs.

____

____

```
R1(config)# interface g0/0
R1(config-if)#ip helper-address 192.168.2.254
R1(config-if)#exit
R1(config)# interface g0/1
R1(config-if)#ip helper-address

192.168.2.254
```

Step 3: Record IP settings for PC-A and PC-B.

On PC-A and PC-B, issue the **ipconfig /all** command to verify that the PCs have received IP address information from the DHCP server on R2. Record the IP and MAC address for each PC.

____

_________ Answers may vary.

Based on the DHCP pool that was configured on R2, what are the first available IP addresses that PC-A and PC-B can lease?

____

_________ PC-B: 192.168.0.10, and PC-A: 192.168.1.10

Step 4: Verify DHCP services and address leases on R2.

a. On R2, enter the **show ipdhcp binding** command to view DHCP address leases.

```
R2# show ipdhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
192.168.0.10        011c.c1de.91c3.5d       Mar 13 2013 02:07 AM    Automatic
192.168.1.10        0100.2170.0c05.0c       Mar 13 2013 02:09 AM    Automatic
```

Along with the IP addresses that were leased, what other piece of useful client identification information is in the output?

____
____

The client hardware addresses identify the specific computers that have joined the network.

b. On R2, enter the **show ipdhcp server statistics** command to view the DHCP pool statistics and message activity.

```
R2# show ipdhcp server statistics
Memory usage           42175
Address pools          2
Database agents        0
Automatic bindings     2
Manual bindings        0
Expired bindings       0
Malformed messages     0
Secure arp entries     0

Message                Received
BOOTREQUEST            0
DHCPDISCOVER           2
DHCPREQUEST            2
DHCPDECLINE            0
DHCPRELEASE            0
DHCPINFORM             2

Message                Sent
BOOTREPLY              0
DHCPOFFER              2
DHCPACK                4
DHCPNAK                0
```

How many types of DHCP messages are listed in the output?

____
__________ Ten different types of DHCP messages are listed.

c. On R2, enter the **show ipdhcp pool** command to view the DHCP pool settings.

```
R2# show ipdhcp pool

Pool R1G1 :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 1
 Pending event
: none  1 subnet is currently
in the pool :
 Current index        IP address range                    Leased addresses
 192.168.1.11         192.168.1.1      - 192.168.1.254     1


Pool R1G0 :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 1
 Pending event
: none  1 subnet is currently
in the pool :
 Current index        IP address range                    Leased addresses
 192.168.0.11         192.168.0.1      - 192.168.0.254     1
```

In the output of the **show ipdhcp pool** command, what does the Current index refer to?

____

____

The next available address for leasing

d. On R2, enter the **show run | section dhcp** command to view the DHCP configuration in the running configuration.

```
R2# show run | section dhcp
ipdhcp excluded-address 192.168.0.1
192.168.0.9
ipdhcp excluded-address 192.168.1.1
192.168.1.9
ipdhcp pool R1G1  network 192.168.1.0
255.255.255.0  default-router
192.168.1.1  domain-name ccna-lab.com
dns-server
209.165.200.225  lease 2

ipdhcp pool R1G0
network 192.168.0.0
255.255.255.0  default-
router 192.168.0.1
domain-name ccna-lab.com
dns-server
209.165.200.225  lease 2
```

e. On R2, enter the **show run interface** command for interfaces G0/0 and G0/1 to view the DHCP relay configuration in the running configuration.

```
R2# show run interface g0/0
Building configuration...


Current configuration :
132 bytes !
interface
GigabitEthernet0/0  ip
```

```
address 192.168.0.1
255.255.255.0  ip helper-
address 192.168.2.254
duplex auto  speed auto end


R2# show run interface g0/1
Building configuration...


Current configuration : 132
bytes ! interface
GigabitEthernet0/1  ip
address 192.168.1.1
255.255.255.0  ip helper-
address 192.168.2.254
duplex auto  speed auto end
```

Reflection

What do you think is the benefit of using DHCP relay agents instead of multiple routers acting as DHCP servers?

____

Having a separate router DHCP server for each subnet would add more complexity and decrease centralized
management for the network. It would also require that each router work harder to manage its own DHCP addressing, in addition to the primary function of routing traffic. One DHCP server (router or computer) that is

3) Stage v (verify)

Home Activities:

Activity 1:

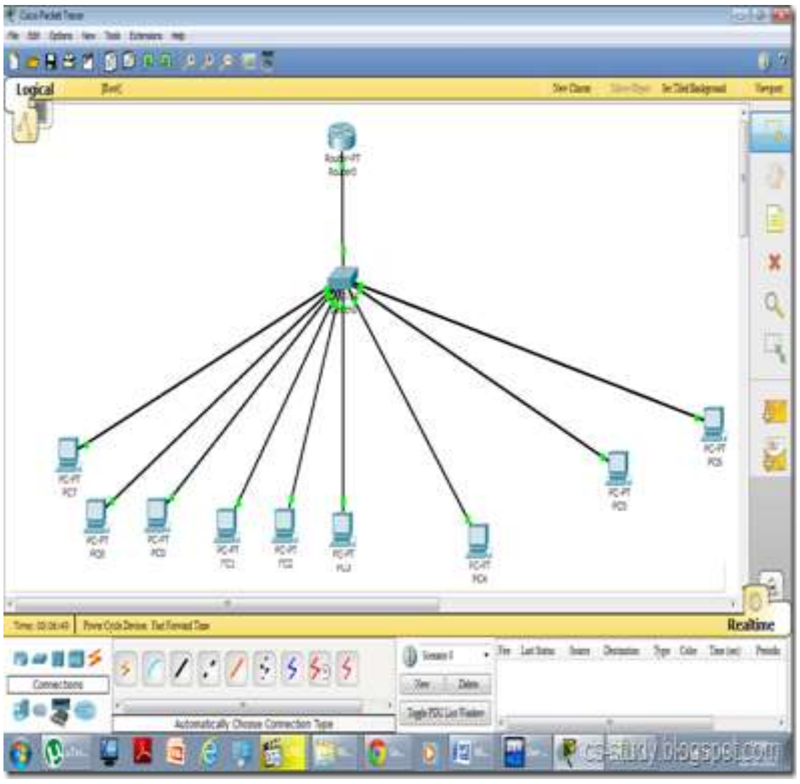
You should hand in a screen shot of the Command Prompt window similar to Figure 1 above. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[20] to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

Answer the following questions:

1. Are DHCP messages sent over UDP or TCP?
2. Draw a timing datagram illustrating the sequence of the first four-packet Discover/Offer/Request/ACK DHCP exchange between the client and server. For each packet, indicated the source and destination port numbers. Are the port numbers the same as in the example given in this lab assignment?
3. What is the link-layer (e.g., Ethernet) address of your host?
4. What values in the DHCP discover message differentiate this message from the DHCP request message?

[20] What do we mean by "annotate"? If you hand in a paper copy, please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate.

5. What is the value of the Transaction-ID in each of the first four (Discover/Offer/Request/ACK) DHCP messages? What are the values of the Transaction-ID in the second set (Request/ACK) set of DHCP messages? What is the purpose of the Transaction-ID field?
6. A host uses DHCP to obtain an IP address, among other things. But a host's IP address is not confirmed until the end of the four-message exchange! If the IP address is not set until the end of the four-message exchange, then what values are used in the IP datagrams in the four-message exchange? For each of the four DHCP messages (Discover/Offer/Request/ACK DHCP), indicate the source and destination IP addresses that are carried in the encapsulating IP datagram.
7. What is the IP address of your DHCP server?
8. What IP address is the DHCP server offering to your host in the DHCP Offer message? Indicate which DHCP message contains the offered DHCP address.
9. In the example screenshot in this assignment, there is no relay agent between the host and the DHCP server. What values in the trace indicate the absence of a relay agent? Is there a relay agent in your experiment? If so what is the IP address of the agent?
10. Explain the purpose of the router and subnet mask lines in the DHCP offer message.
11. In the DHCP trace file noted in footnote 2, the DHCP server offers a specific IP address to the client (see also question 8. above). In the client's response to the first server OFFER message, does the client accept this IP address? Where in the client RESPONSE isis the client's requested address?
12. Explain the purpose of the lease time. How long is the lease time in your experiment?
13. What is the purpose of the DHCP release message? Does the DHCP server issue an acknowledgment of receipt of the client's DHCP request? What would happen if the client's DHCP release message is lost?
14. Clear the *bootp* filter from your Wireshark window. Were any ARP packets sent or received during the DHCP packet-exchange period? If so, explain the purpose of those ARP packets.



Activity 2:

Configure dhcp service on cisco router in mentioned packet tracer topology.

IP addresses are:

Interface 1: 10.0.0.1 255.0.0.0
Interface 2: 20.0.0.1 255.0.0.0

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

LAB # 12

Statement Purpose:

Investigate the Ethernet protocol and the ARP protocol

Activity Outcomes:

The students will gain better understanding of Ethernet and ARP

Instructor Note:

1) StageJ(Journey)

Introduction

In this lab, we'll investigate the Ethernet protocol and the ARP protocol. Before beginning this lab, you'll probably want to review sections 5.4.1 (link-layer addressing and ARP) and 5.4.2 (Ethernet) in the text. RFC 826 (ftp://ftp.rfc-editor.org/innotes/std/std37.txt) contains the gory details of the ARP protocol, which is used by an IP device to determine the IP address of a remote interface whose Ethernet address is known.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

Let's begin by capturing a set of Ethernet frames to study.

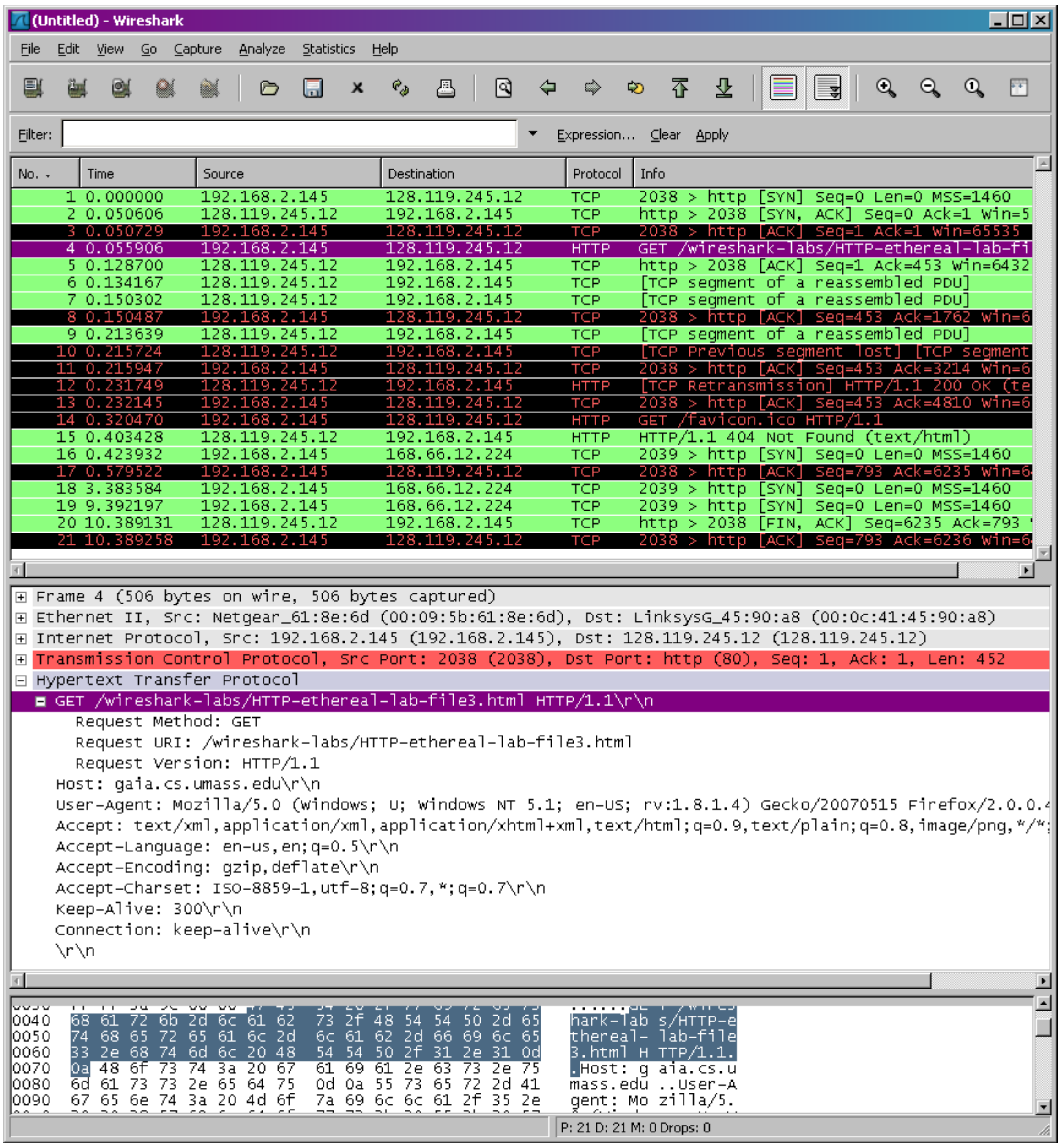
Solution:

Do the following[21]:

- First, make sure your browser's cache is empty. To do this under Mozilla Firefox V3, select *Tools->Clear Recent History* and check the box for Cache. For Internet Explorer, select *Tools->Internet Options->Delete Files.* Start up the Wireshark packet sniffer

[21] If you are unable to run Wireshark live on a computer, you can download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file *ethernet--ethereal-trace-1*. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the ethernet-ethereal-trace-1 trace file. You can then use this trace file to answer the questions below.

- Enter the following URL into your browserhttp://gaia.cs.umass.edu/wireshark-labs/HTTP-ethereal-lab-file3.html
  Your browser should display the rather lengthy US Bill of Rights.
- Stop Wireshark packet capture. First, find the packet numbers (the leftmost column in the upper Wireshark window) of the HTTP GET message that was sent from your computer to gaia.cs.umass.edu, as well as the beginning of the HTTP response message sent to your computer by gaia.cs.umass.edu. You should see a screen that looks something like this (where packet 4 in the screen shot below contains the HTTP GET message)


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

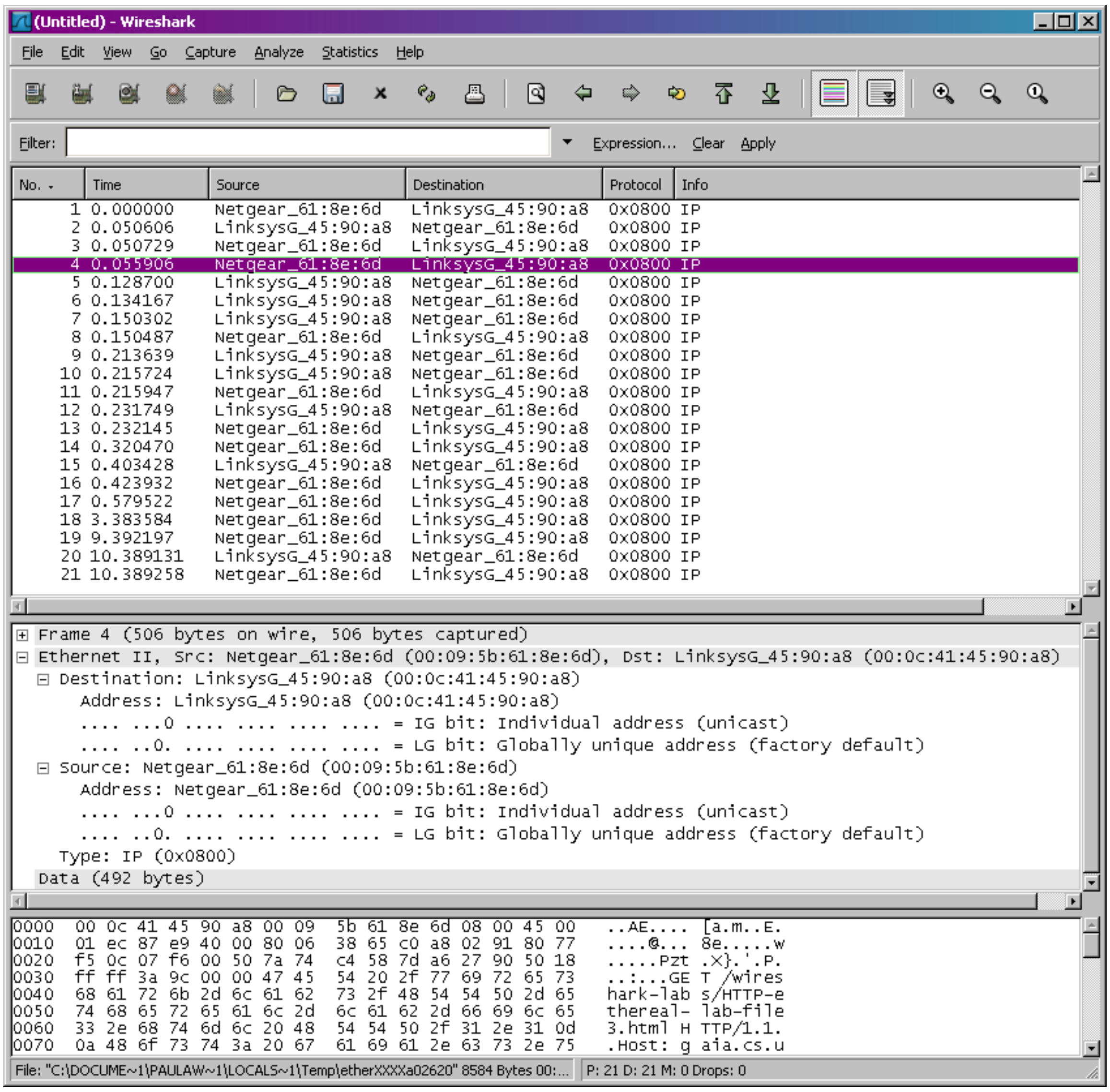
```
No.  Time        Source           Destination      Protocol  Info
  1 0.000000    192.168.2.145    128.119.245.12   TCP       2038 > http [SYN] Seq=0 Len=0 MSS=1460
  2 0.050606    128.119.245.12   192.168.2.145    TCP       http > 2038 [SYN, ACK] Seq=0 Ack=1 Win=5
  3 0.050729    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=1 Ack=1 Win=65535
  4 0.055906    192.168.2.145    128.119.245.12   HTTP      GET /wireshark-labs/HTTP-ethereal-lab-fi
  5 0.128700    128.119.245.12   192.168.2.145    TCP       http > 2038 [ACK] Seq=1 Ack=453 Win=6432
  6 0.134167    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
  7 0.150302    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
  8 0.150487    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=1762 Win=6
  9 0.213639    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
 10 0.215724    128.119.245.12   192.168.2.145    TCP       [TCP Previous segment lost] [TCP segment
 11 0.215947    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=3214 Win=6
 12 0.231749    128.119.245.12   192.168.2.145    HTTP      [TCP Retransmission] HTTP/1.1 200 OK (te
 13 0.232145    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=4810 Win=6
 14 0.320470    192.168.2.145    128.119.245.12   HTTP      GET /favicon.ico HTTP/1.1
 15 0.403428    128.119.245.12   192.168.2.145    HTTP      HTTP/1.1 404 Not Found (text/html)
 16 0.423932    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 17 0.579522    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=793 Ack=6235 Win=6
 18 3.383584    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 19 9.392197    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 20 10.389131   128.119.245.12   192.168.2.145    TCP       http > 2038 [FIN, ACK] Seq=6235 Ack=793
 21 10.389258   192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=793 Ack=6236 Win=6
```

```
⊞ Frame 4 (506 bytes on wire, 506 bytes captured)
⊞ Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
⊞ Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 128.119.245.12 (128.119.245.12)
⊞ Transmission Control Protocol, Src Port: 2038 (2038), Dst Port: http (80), Seq: 1, Ack: 1, Len: 452
⊟ Hypertext Transfer Protocol
  ⊟ GET /wireshark-labs/HTTP-ethereal-lab-file3.html HTTP/1.1\r\n
      Request Method: GET
      Request URI: /wireshark-labs/HTTP-ethereal-lab-file3.html
      Request Version: HTTP/1.1
    Host: gaia.cs.umass.edu\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4) Gecko/20070515 Firefox/2.0.0.
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    \r\n
```

```
0040  68 61 72 6b 2d 6c 61 62  73 2f 48 54 54 50 2d 65   hark-lab s/HTTP-e
0050  74 68 65 72 65 61 6c 2d  6c 61 62 2d 66 69 6c 65   thereal- lab-file
0060  33 2e 68 74 6d 6c 20 48  54 54 50 2f 31 2e 31 0d   3.html H TTP/1.1.
0070  0a 48 6f 73 74 3a 20 67  61 69 61 2e 63 73 2e 75   .Host: g aia.cs.u
0080  6d 61 73 73 2e 65 64 75  0d 0a 55 73 65 72 2d 41   mass.edu ..User-A
0090  67 65 6e 74 3a 20 4d 6f  7a 69 6c 6c 61 2f 35 2e   gent: Mo zilla/5.
```

P: 21 D: 21 M: 0 Drops: 0

- Since this lab is about Ethernet and ARP, we're not interested in IP or higher-layer protocols. So let's change Wireshark's "listing of captured packets" window so that it shows information only about protocols below IP. To have Wireshark do this, select *Analyze->Enabled Protocols.* Then uncheck the IP box and select *OK*. You should now see an Wireshark window that looks like:


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time        Source              Destination         Protocol Info
     1 0.000000    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
     2 0.050606    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
     3 0.050729    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
     4 0.055906    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
     5 0.128700    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
     6 0.134167    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
     7 0.150302    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
     8 0.150487    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
     9 0.213639    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
    10 0.215724    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
    11 0.215947    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    12 0.231749    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
    13 0.232145    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    14 0.320470    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    15 0.403428    LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
    16 0.423932    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    17 0.579522    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    18 3.383584    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    19 9.392197    Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
    20 10.389131   LinksysG_45:90:a8   Netgear_61:8e:6d    0x0800   IP
    21 10.389258   Netgear_61:8e:6d    LinksysG_45:90:a8   0x0800   IP
```

```
⊞ Frame 4 (506 bytes on wire, 506 bytes captured)
⊟ Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
  ⊟ Destination: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
      Address: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
  ⊟ Source: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
      Address: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    Type: IP (0x0800)
  Data (492 bytes)
```

```
0000  00 0c 41 45 90 a8 00 09  5b 61 8e 6d 08 00 45 00   ..AE.... [a.m..E.
0010  01 ec 87 e9 40 00 80 06  38 65 c0 a8 02 91 80 77   ....@... 8e.....w
0020  f5 0c 07 f6 00 50 7a 74  c4 58 7d a6 27 90 50 18   .....Pzt .X}.'.P.
0030  ff ff 3a 9c 00 00 47 45  54 20 2f 77 69 72 65 73   ..:...GE T /wires
0040  68 61 72 6b 2d 6c 61 62  73 2f 48 54 54 50 2d 65   hark-lab s/HTTP-e
0050  74 68 65 72 65 61 6c 2d  6c 61 62 2d 66 69 6c 65   thereal- lab-file
0060  33 2e 68 74 6d 6c 20 48  54 54 50 2f 31 2e 31 0d   3.html H TTP/1.1.
0070  0a 48 6f 73 74 3a 20 67  61 69 61 2e 63 73 2e 75   .Host: g aia.cs.u
```

```
File: "C:\DOCUME~1\PAULAW~1\LOCALS~1\Temp\etherXXXXa02620" 8584 Bytes 00:...   P: 21 D: 21 M: 0 Drops: 0
```

In order to answer the following questions, you'll need to look into the packet details and packet contents windows (the middle and lower display windows in Wireshark).

Select the Ethernet frame containing the HTTP GET message. (Recall that the HTTP GET message is carried inside of a TCP segment, which is carried inside of an IP datagram, which is carried inside of an Ethernet frame; reread section 1.5.2 in the text if you find this encapsulation a bit confusing). Expand the Ethernet II information in the packet details window. Note that the contents of the Ethernet frame (header as well as payload) are displayed in the packet contents window.

Answer the following questions, based on the contents of the Ethernet frame containing the HTTP GET message. Whenever possible, when answering a question you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[22] to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

[22] What do we mean by "annotate"? If you hand in a paper copy, please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate.

- What is the 48-bit Ethernet address of your computer?
- What is the 48-bit destination address in the Ethernet frame? Is this the Ethernet address of gaia.cs.umass.edu? (Hint: the answer is *no*). What device has this as its Ethernet address? [Note: this is an important question, and one that students sometimes get wrong. Re-read pages 468-469 in the text and make sure you understand the answer here.]
- Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?
- How many bytes from the very start of the Ethernet frame does the ASCII "G" in "GET" appear in the Ethernet frame?

Next, answer the following questions, based on the contents of the Ethernet frame containing the first byte of the HTTP response message.

- What is the value of the Ethernet source address? Is this the address of your computer, or of gaia.cs.umass.edu (Hint: the answer is *no*). What device has this as its Ethernet address?
- What is the destination address in the Ethernet frame? Is this the Ethernet address of your computer?
- Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?

How many bytes from the very start of the Ethernet frame does the ASCII "O" in "OK" (i.e., the HTTP response code) appear in the Ethernet frame?

Activity 2:

The Address Resolution Protocol

In this section, we'll observe the ARP protocol in action. We strongly recommend that you re-read section 5.4.1 in the text before proceeding.

ARP Caching

Recall that the ARP protocol typically maintains a cache of IP-to-Ethernet address translation pairs on your comnputer the *arp* command (in both MSDOS and Linux/Unix) is used to view and manipulate the contents of this cache. Since the *arp* command and the ARP protocol have the same name, it's understandably easy to confuse them. But keep in mind that they are different - the *arp* command is used to view and manipulate the ARP cache contents, while the ARP protocol defines the format and meaning of the messages sent and received, and defines the actions taken on message transmission and receipt.

Let's take a look at the contents of the ARP cache on your computer:

- **MS-DOS.** The *arp* command is in c:\windows\system32, so type either "*arp*" or "*c:\windows\system32\arp*" in the MS-DOS command line (without quotation marks).
- **Linux/Unix/MacOS.** The executable for the arp command can be in various places. Popular locations are /sbin/arp (for linux) and /usr/etc/arp (for some Unix variants).

The Windows arp command with no arguments will display the contents of the ARP cache on your computer. Run the *arp* command.

- Write down the contents of your computer's ARP cache. What is the meaning of each column value?

In order to observe your computer sending and receiving ARP messages, we'll need to clear the ARP cache, since otherwise your computer is likely to find a needed IP-Ethernet address translation pair in its cache and consequently not need to send out an ARP message.

- **MS-DOS.** The MS-DOS *arp –d ** command will clear your ARP cache. The *–d* flag indicates a deletion operation, and the * is the wildcard that says to delete all table entries.

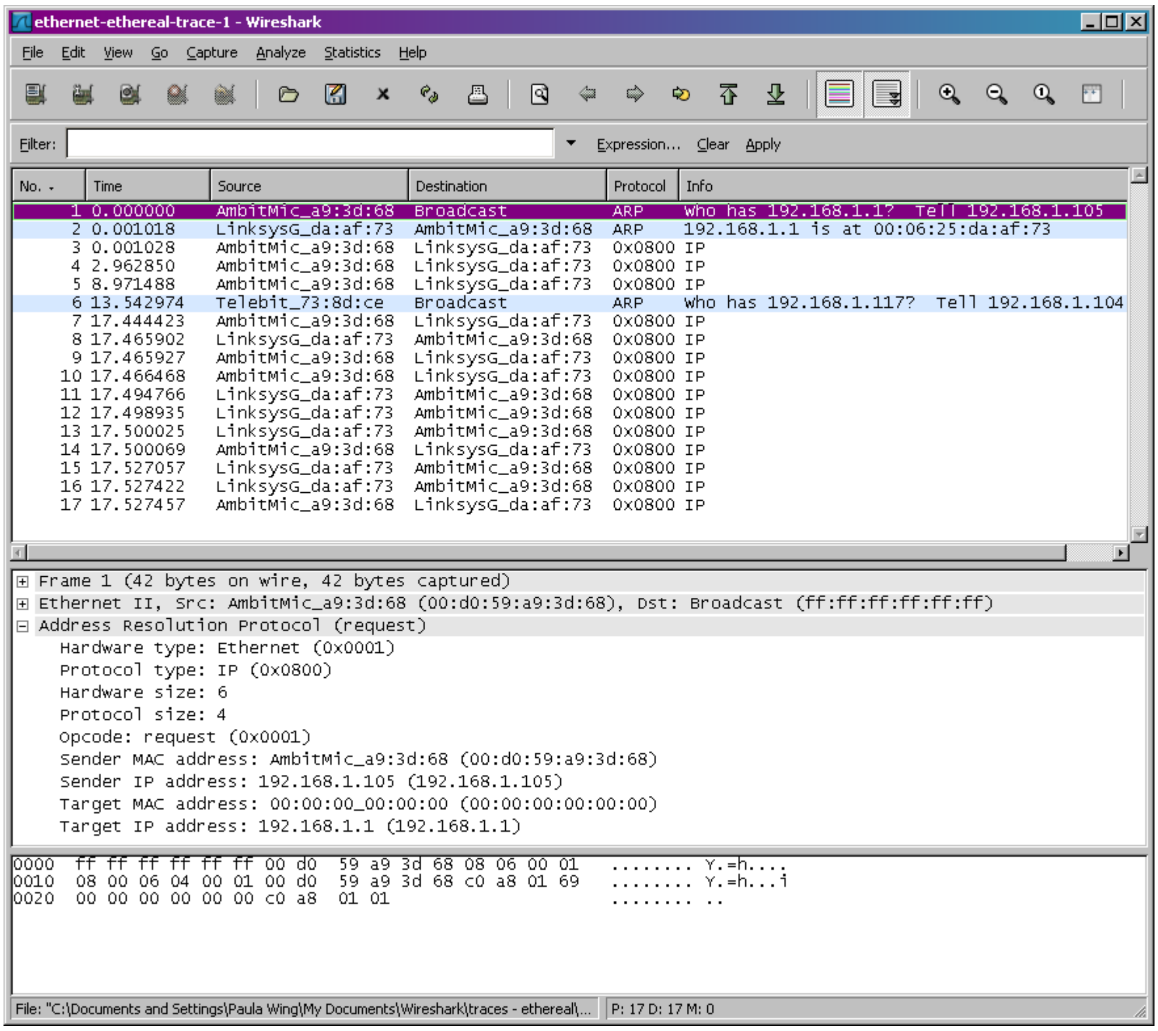
- **Linux/Unix/MacOS.** The *arp –d ** will clear your ARP cache. In order to run this command you'll need root privileges. If you don't have root privileges and can't run Wireshark on a Windows machine, you can skip the trace collection part of this lab and just use the trace discussed in the earlier footnote.

Solution:

Do the following[23]:

- Clear your ARP cache, as described above.
- Next, make sure your browser's cache is empty. To do this under Mozilla Firefox V3, select *Tools->Clear Recent History* and check the box for Cache. For Internet Explorer, select *Tools->Internet Options->Delete Files.*
- Start up the Wireshark packet sniffer
- Enter the following URL into your browserhttp://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-lab-file3.html
  Your browser should again display the rather lengthy US Bill of Rights.
- Stop Wireshark packet capture. Again, we're not interested in IP or higher-layer protocols, so change Wireshark's "listing of captured packets" window so that it shows information only about protocols below IP. To have Wireshark do this, select *Analyze->Enabled Protocols.* Then uncheck the IP box and select *OK*. You should now see an Wireshark window that looks like:

[23]The *ethernet-ethereal-trace-1* trace file in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip was created using the steps below (in particular after the ARP cache had been flushed).


ethernet-ethereal-trace-1 - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time        Source               Destination          Protocol  Info
  1 0.000000    AmbitMic_a9:3d:68    Broadcast            ARP       who has 192.168.1.1?  Tell 192.168.1.105
  2 0.001018    LinksysG_da:af:73    AmbitMic_a9:3d:68    ARP       192.168.1.1 is at 00:06:25:da:af:73
  3 0.001028    AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
  4 2.962850    AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
  5 8.971488    AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
  6 13.542974   Telebit_73:8d:ce     Broadcast            ARP       who has 192.168.1.117?  Tell 192.168.1.104
  7 17.444423   AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
  8 17.465902   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
  9 17.465927   AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
 10 17.466468   AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
 11 17.494766   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
 12 17.498935   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
 13 17.500025   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
 14 17.500069   AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
 15 17.527057   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
 16 17.527422   LinksysG_da:af:73    AmbitMic_a9:3d:68    0x0800 IP
 17 17.527457   AmbitMic_a9:3d:68    LinksysG_da:af:73    0x0800 IP
```

```
⊞ Frame 1 (42 bytes on wire, 42 bytes captured)
⊞ Ethernet II, Src: AmbitMic_a9:3d:68 (00:d0:59:a9:3d:68), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
⊟ Address Resolution Protocol (request)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    Sender MAC address: AmbitMic_a9:3d:68 (00:d0:59:a9:3d:68)
    Sender IP address: 192.168.1.105 (192.168.1.105)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.1.1 (192.168.1.1)
```

```
0000  ff ff ff ff ff ff 00 d0  59 a9 3d 68 08 06 00 01   ........ Y.=h....
0010  08 00 06 04 00 01 00 d0  59 a9 3d 68 c0 a8 01 69   ........ Y.=h...i
0020  00 00 00 00 00 00 c0 a8  01 01                     ........ ..
```

File: "C:\Documents and Settings\Paula Wing\My Documents\Wireshark\traces - ethereal\... P: 17 D: 17 M: 0

3) Stage v (verify)

Home Activities:

Activity 1:

In the example above, the first two frames in the trace contain ARP messages (as does the 6th message). The screen shot above corresponds to the trace referenced in footnote 1. Answer the following questions:

- What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP request message?
- Give the hexadecimal value for the two-byte Ethernet Frame type field. What upper layer protocol does this correspond to?
- Download the ARP specification from ftp://ftp.rfc-editor.org/in-notes/std/std37.txt. A readable, detailed discussion of ARP is also at http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/arp.html.
    1. How many bytes from the very beginning of the Ethernet frame does the ARP *opcode* field begin?
    2. What is the value of the *opcode* field within the ARP-payload part of the Ethernet frame in which an ARP request is made?
    3. Does the ARP message contain the IP address of the sender?
    4. Where in the ARP request does the “question” appear – the Ethernet address of the machine whose corresponding IP address is being queried?
- Now find the ARP reply that was sent in response to the ARP request.

1. How many bytes from the very beginning of the Ethernet frame does the ARP *opcode* field begin?
2. What is the value of the *opcode* field within the ARP-payload part of the Ethernet frame in which an ARP response is made?
3. Where in the ARP message does the "answer" to the earlier ARP request appear – the IP address of the machine having the Ethernet address whose corresponding IP address is being queried?
4. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP reply message?
5. Open the ethernet-ethereal-trace-1trace file in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip. The first and second ARP packets in this trace correspond to an ARP request sent by the computer running Wireshark, and the ARP reply sent to the computer running Wireshark by the computer with the ARP-requested Ethernet address. But there is yet another computer on this network, as indicated by packet 6 – another ARP request. Why is there no ARP reply (sent in response to the ARP request in packet 6) in the packet trace?

Activity 2:

1. EX-1. The arp command:
2. arp -s InetAddrEtherAddr
3. allows you to manually add an entry to the ARP cache that resolves the IP address InetAddr to the physical address EtherAddr. What would happen if, when you manually added an entry, you entered the correct IP address, but the wrong Ethernet address for that remote interface?
4. EX-2. What is the default amount of time that an entry remains in your ARP cache before being removed. You can determine this empirically (by monitoring the cache contents) or by looking this up in your operation system documentation. Indicate how/where you determined this value.

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

LAB # 13

Statement Purpose:

- Configuring VLAN using Layer 2 device
- Make some simple Packet Tracer scenarios

Activity outcomes:

- Students will have gained the understanding of Virtual LANs that is partitioned and isolated broadcast domain at layer 2.
- Students will be able to overcome the broadcast problem in LAN environment.

Instructor Note:

-

1) StageJ(Journey)

Introduction

Modern switches use virtual local-area networks (VLANs) to improve network performance by separating large Layer 2 broadcast domains into smaller ones. VLANs can also be used as a security measure by controlling which hosts can communicate. In general, VLANs make it easier to design a network to support the goals of an organization.
VLAN trunks are used to span VLANs across multiple devices. Trunks allow the traffic from multiple VLANS to travel over a single link, while keeping the VLAN identification and segmentation intact.
In this lab, you will create VLANs on both switches in the topology, assign VLANs to switch access ports, verify that VLANs are working as expected, and then create a VLAN trunk between the two switches to allow hosts in the same VLAN to communicate through the trunk, regardless of which switch the host is actually attached to.

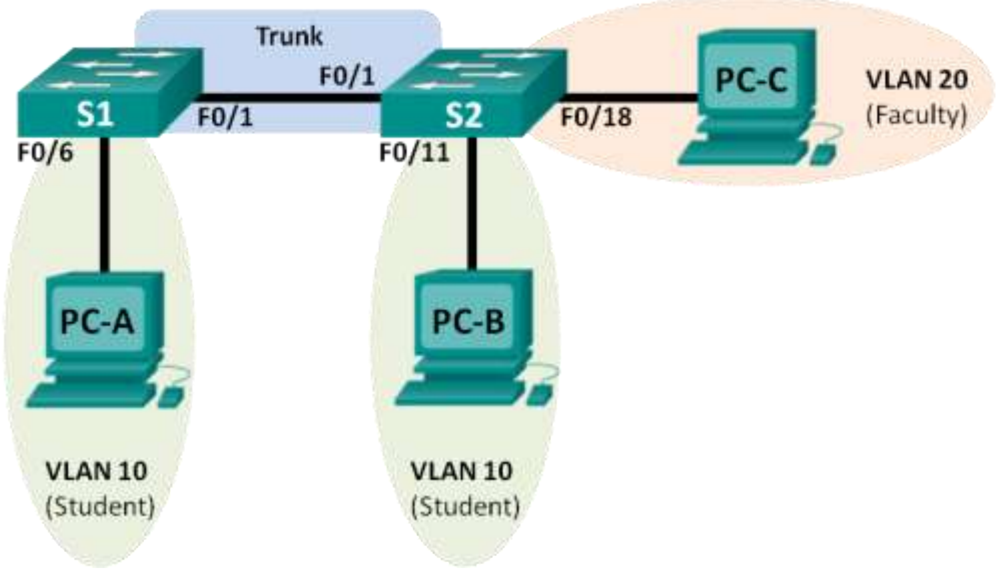
2) Stage a1 (apply)

Lab Activities:

Activity 1:

Configuring VLANs (Subnets) using Layer 2 device.

Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway
S1	VLAN 1	192.168.1.11	255.255.255.0	N/A
S2	VLAN 1	192.168.1.12	255.255.255.0	N/A
PC-A	NIC	192.168.10.3	255.255.255.0	192.168.10.1
PC-B	NIC	192.168.10.4	255.255.255.0	192.168.10.1
PC-C	NIC	192.168.20.3	255.255.255.0	192.168.20.1

Objectives

Part 1: Build the Network and Configure Basic Device Settings

Part 2: Create VLANs and Assign Switch Ports

Part 3: Maintain VLAN Port Assignments and the VLAN Database

Part 4: Configure an 802.1Q Trunk between the

Switches Part Part 5: Delete the VLAN Database

Required Resources

- 2 Switches (Cisco 2960)
- 3 PCs (Windows 7, Vista, or XP with terminal emulation program, such as Tera Term/Putty)
- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet cables as shown in the topology

Part 1: Build the Network and Configure Basic Device Settings

In Part 1, you will set up the network topology and configure basic settings on the PC hosts and switches.

Step 1: Cable the network as shown in the topology.

Attach the devices as shown in the topology diagram, and cable as necessary.

Step 2: Initialize and reload the switches as necessary.

Step 3: Configure basic settings for each switch.

a. Disable DNS lookup.

b. Configure device name as shown in the topology.

c. Assign **class** as the privileged EXEC password.

d. Assign **cisco** as the console and vty passwords and enable login for console and vty lines.

e. Configure **logging synchronous** for the console line.

f. Configure a MOTD banner to warn users that unauthorized access is prohibited.

g. Configure the IP address listed in the Addressing Table for VLAN 1 on both switches.

h. Administratively deactivate all unused ports on the switch.

i. Copy the running configuration to the startup configuration.

Step 4: Configure PC hosts.

Refer to the Addressing Table for PC host address information.

Step 5: Test connectivity.

Verify that the PC hosts can ping one another.

Note: It may be necessary to disable the PCs firewall to ping between PCs.

Can PC-A ping PC-B? __________ Yes

Can PC-A ping PC-C? __________ No

Can PC-A ping S1? __________ No

Can PC-B ping PC-C? __________ No

Can PC-B ping S2? __________ No

Can PC-C ping S2? __________ No

Can S1 ping S2? __________ Yes

If you answered no to any of the above questions, why were the pings unsuccessful?

____

____

Pings were unsuccessful when trying to ping a device on a different subnet. For those pings to be successful, a default gateway must exist to route traffic from one subnet to another.

Part 2: Create VLANs and Assign Switch Ports

In Part 2, you will create student, faculty, and management VLANs on both switches. You will then assign the VLANs to the appropriate interface. The **show vlan** command is used to verify your configuration settings.

Step 1: Create VLANs on the switches.

a. Create the VLANs on S1.

```
S1(config)# vlan 10
```

```
S1(config-vlan)# name
Student
S1(config-vlan)# vlan 20
S1(config-vlan)# name
Faculty
S1(config-vlan)# vlan 99
S1(config-vlan)# name
Management
S1(config-vlan)# end
```

b. Create the same VLANs on S2.

c. Issue the **show vlan** command to view the list of VLANs on S1.

```
S1# show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- ----------------------------
---
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11,
Fa0/12
                                                Fa0/13, Fa0/14, Fa0/15,
Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19,
Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23,
Fa0/24
                                                Gi0/1, Gi0/2
10   Student                          active
20   Faculty                          active
99   Management                       active
```

What is the default VLAN? ___________

VLAN 1

What ports are assigned to the default VLAN? _______

All switch ports are assigned to VLAN 1 by default.

Step 2: Assign VLANs to the correct switch interfaces.

a. Assign VLANs to the interfaces on S1.

1) Assign PC-A to the Student VLAN.

```
S1(config)# interface f0/6
S1(config-if)# switchport mode
access  S1(config-if)#  switchport
access vlan 10
```

2) Move the switch IP address VLAN 99.

```
S1(config)# interface vlan 1
S1(config-if)# no ip
address
S1(config-if)# interface vlan 99
```

```
S1(config-if)# ip address 192.168.1.11 255.255.255.0
S1(config-if)# end
```

b. Issue the **show vlanbrief** command and verify that the VLANs are assigned to the correct interfaces.

```
S1# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- ----------------------------
---
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/11, Fa0/12,
Fa0/13
                                                Fa0/14, Fa0/15, Fa0/16,
Fa0/17
                                                Fa0/18, Fa0/19, Fa0/20,
Fa0/21                                                 Fa0/22, Fa0/23,
Fa0/24, Gi0/1
                                                Gi0/2
10   Student                          active    Fa0/6
20   Faculty                          active
99   Management                       active
```

c. Issue the **show ip interfaces brief** command.

```
S1# show ip interface brief
Interface                          IP-Address           OK?  Method  Status
Protocol
Vlan1                  unassigned      YES unset  upup
Vlan99                 192.168.1.11    YES manualup                        down
FastEthernet0/1        unassigned      YES unset  upup
FastEthernet0/2        unassigned      YES unset  administratively down down
FastEthernet0/3        unassigned      YES unset  administratively down down
FastEthernet0/4        unassigned      YES unset  administratively down down
FastEthernet0/5        unassigned      YES unset  administratively down down
FastEthernet0/6        unassigned      YES unset  upup
<output omitted>
```

What is the status of VLAN 99? Why? ________________

The status of VLAN 99 is up/down, because it has not been assigned to an active port yet.

d. Use the Topology to assign VLANs to the appropriate ports on S2.

e. Remove the IP address for VLAN 1 on S2.

f. Configure an IP address for VLAN 99 on S2 according to the Addressing Table.

g. Use the **show vlan brief** command to verify that the VLANs are assigned to the correct interfaces.

```
S2# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- ----------------------------
---
```

```
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/12,
Fa0/13
                                                Fa0/14, Fa0/15, Fa0/16,
Fa0/17
                                                Fa0/19, Fa0/20, Fa0/21,
Fa0/22
                                                Fa0/23, Fa0/24, Gi0/1, Gi0/2
10   Student                          active    Fa0/11
20   Faculty                          active    Fa0/18
99   Management                       active
```

Is PC-A able to ping PC-B? Why? __________

No. Interface F0/1 is not assigned to VLAN 10, so VLAN 10 traffic will not be sent over it.

Is S1 able to ping S2? Why? ________

No. The IP addresses for the switches now reside in VLAN 99. VLAN 99 traffic will not be sent over interface F0/1.

Part 3: Maintain VLAN Port Assignments and the VLAN Database

In Part 3, you will change VLAN assignments to ports and remove VLANs from the VLAN database.

Step 1: Assign a VLAN to multiple interfaces.

a. On S1, assign interfaces F0/11 – 24 to VLAN 10.

```
S1(config)# interface range f0/11-24
S1(config-if-range)# switchport mode access
S1(config-if-range)# switchport access vlan 10
S1(config-if-range)# end
```

b. Issue the **show vlan brief** command to verify VLAN assignments.

```
S1# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -----------------------------
--
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Gi0/1, Gi0/2
10   Student                          active    Fa0/6, Fa0/11, Fa0/12, Fa0/13
Fa0/14, Fa0/15, Fa0/16, Fa0/17
Fa0/18, Fa0/19, Fa0/20, Fa0/21
Fa0/22, Fa0/23, Fa0/24
20   Faculty                          active
99   Management                       active
```

c. Reassign F0/11 and F0/21 to VLAN 20.

```
S1(config)# interface range f0/11, f0/21
```

```
S1(config-if-range)# switchport access vlan 20
S1(config-if-range)# end
```

d. Verify that VLAN assignments are correct.

```
S1# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -----------------------------
--
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Gi0/1, Gi0/2
10   Student                          active    Fa0/6, Fa0/12, Fa0/13, Fa0/14
                                                Fa0/15,     Fa0/16,     Fa0/17,
Fa0/18

                                                Fa0/19,     Fa0/20,     Fa0/22,
Fa0/23

                                                Fa0/24
20   Faculty                          active    Fa0/11, Fa0/21
99   Management                       active
```

Step 2: Remove a VLAN assignment from an interface.

a. Use the **noswitchport access vlan** command to remove the VLAN 10 assignment to F0/24.

```
S1(config)# interface f0/24
S1(config-if)# no switchport access vlan
S1(config-if)# end
```

b. Verify that the VLAN change was made.

Which VLAN is F0/24 is now associated with? _______

VLAN 1, the default VLAN.

```
S1# show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -----------------------------
--
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/24, Gi0/1, Gi0/2
10   Student                          active    Fa0/6, Fa0/12, Fa0/13, Fa0/14
                                                Fa0/15,     Fa0/16,     Fa0/17,
Fa0/18

                                                Fa0/19,     Fa0/20,     Fa0/22,
Fa0/23
20   Faculty                          active    Fa0/11, Fa0/21
99   Management                       active
```

Step 3: Remove a VLAN ID from the VLAN database.

a. Add VLAN 30 to interface F0/24 without issuing the VLAN command.

```
S1(config)# interface f0/24
S1(config-if)# switchport access vlan 30
% Access VLAN does not exist. Creating vlan 30
```

Note: Current switch technology no longer requires that the **vlan** command be issued to add a VLAN to the database. By assigning an unknown VLAN to a port, the VLAN adds to the VLAN database.

b. Verify that the new VLAN is displayed in the VLAN table.

```
S1# show vlan brief


VLAN Name                             Status    Ports
---- -------------------------------- --------- ----------------------------
---
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Gi0/1, Gi0/2
10   Student                          active    Fa0/12, Fa0/13, Fa0/14,
Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18,
Fa0/19
                                                Fa0/20, Fa0/22, Fa0/23
20   Faculty                          active    Fa0/11, Fa0/21
30   VLAN0030                         active    Fa0/24
99  Management                       active
```

What is the default name of VLAN 30? __________

VLAN0030

c. Use the **no vlan 30** command to remove VLAN 30 from the VLAN database.

```
S1(config)# no vlan 30
S1(config)# end
```

d. Issue the **show vlan brief** command. F0/24 was assigned to VLAN 30.

After deleting VLAN 30, what VLAN is port F0/24 assigned to? What happens to the traffic destined to the host attached to F0/24? ________

Port F0/24 is not assigned to any VLAN. This port will not transfer any traffic.

```
S1# show vlan brief


VLAN Name                             Status    Ports
---- -------------------------------- --------- ----------------------------
---
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Gi0/1, Gi0/2
10   Student                          active    Fa0/12, Fa0/13, Fa0/14,
Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18,
Fa0/19
                                                Fa0/20, Fa0/22, Fa0/23
20   Faculty                          active    Fa0/11, Fa0/21
99   Management                       active
```

e. Issue the **no switchport access vlan** command on interface F0/24.

```
S1(config)# interface f0/24
S1(config-if)# no switchport access vlan
S1(config-if)# end
```

Issue the **show vlan brief** command to determine the VLAN assignment for F0/24. To which VLAN is F0/24 assigned? _________

VLAN 1

```
S1# show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- ------------------------------
--
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/24, Gi0/1, Gi0/2
10   Student                          active    Fa0/6, Fa0/12, Fa0/13, Fa0/14
                                                Fa0/15,    Fa0/16,    Fa0/17,
Fa0/18
                                                Fa0/19,    Fa0/20,    Fa0/22,
Fa0/23
20   Faculty                          active    Fa0/11, Fa0/21
99   Management                       active
```

Note: Before removing a VLAN from the database, it is recommended that you reassign all the ports assigned to that VLAN.

Why should you reassign a port to another VLAN before removing the VLAN from the VLAN database?

____
____

The interfaces assigned to ainterfaces assigned to a VLAN that is the removed from the VLAN database are unavailable for use until they are reassigned to another VLAN. This can be a tricky thing to troubleshoot as trunked interfaces do not show up in the port list as well (Part 4 contains more information about trunked interfaces).

Part 4: Configure an 802.1Q Trunk Between the Switches

Note: By default, all VLANs are allowed on a trunk. The **switchport trunk** command allows you to control what VLANs have access to the trunk. For this lab, keep the default settings which allows all VLANs to traverse F0/1.

Manually configure trunk interface F0/1.

The **switchport mode trunk** command is used to manually configure a port as a trunk. This command should be issued on both ends of the link.

a. Change the switchport mode on interface F0/1 to force trunking. Make sure to do this on both switches.

```
S1(config)# interface f0/1
S1(config-if)#  switchport  mode
trunk

S2(config)# interface f0/1
S2(config-if)# switchport mode trunk
```

b. Issue the **show interfaces trunk** command to view the trunk mode. Notice that the mode changed from **desirable** to **on**.

```
S2# show interfaces trunk
```

```
Port        Mode             Encapsulation  Status        Native vlan
Fa0/1       on               802.1q         trunking      99

Port        Vlans allowed on trunk
Fa0/1       1-4094

Port        Vlans allowed and active in management domain
Fa0/1       1,10,20,99

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/1       1,10,20,99
```

Part 5: Delete the VLAN Database

In Part 5, you will delete the VLAN Database from the switch. It is necessary to do this when initializing a switch back to its default settings.

Step 1: Determine if the VLAN database exists.

Issue the **show flash** command to determine if a **vlan.dat** file exists in flash.

```
S1# show flash

Directory of flash:/

2   -rwx        1285   Mar 1 1993 00:01:24 +00:00  config.text
3   -rwx       43032   Mar 1 1993 00:01:24 +00:00  multiple-fs
4   -rwx           5   Mar 1 1993 00:01:24 +00:00  private-config.text
5   -rwx    11607161   Mar 1 1993 02:37:06 +00:00  c2960-lanbasek9-mz.150-
    2.SE.bin
6   -rwx         736   Mar 1 1993 00:19:41 +00:00  vlan.dat

32514048 bytes total (20858880 bytes free)
```

Note: If there is a **vlan.dat** file located in flash, then the VLAN database does not contain its default settings.

Step 2: Delete the VLAN database.

a. Issue the **delete vlan.dat** command to delete the vlan.dat file from flash and reset the VLAN database back to its default settings. You will be prompted twice to confirm that you want to delete the vlan.dat file. Press Enter both times.

```
S1# delete vlan.dat
Delete filename
[vlan.dat]?  Delete
flash:/vlan.dat? [confirm]
S1#
```

b. Issue the **show flash** command to verify that the vlan.dat file has been deleted.

```
S1# show flash

Directory of flash:/
```

```
2   -rwx        1285   Mar 1 1993 00:01:24 +00:00  config.text
3   -rwx       43032   Mar 1 1993 00:01:24 +00:00  multiple-fs
4   -rwx           5   Mar 1 1993 00:01:24 +00:00  private-config.text
5   -rwx    11607161   Mar 1 1993 02:37:06 +00:00  c2960-lanbasek9-mz.150-
    2.SE.bin
32514048 bytes total (20859904 bytes free)
```

To initialize a switch back to its default settings, what other commands are needed?

____

____

To get a switch back to its default settings, the **erase startup-config** and **reload** commands need to be issued after the **delete vlan.dat** command.

Reflection

1. What is needed to allow hosts on VLAN 10 to communicate to hosts on VLAN 20?

____

Answers will vary, but Layer 3 routing is needed to route traffic between VLANs.

2. What are some primary benefits that an organization can receive through effective use of VLANs?

____

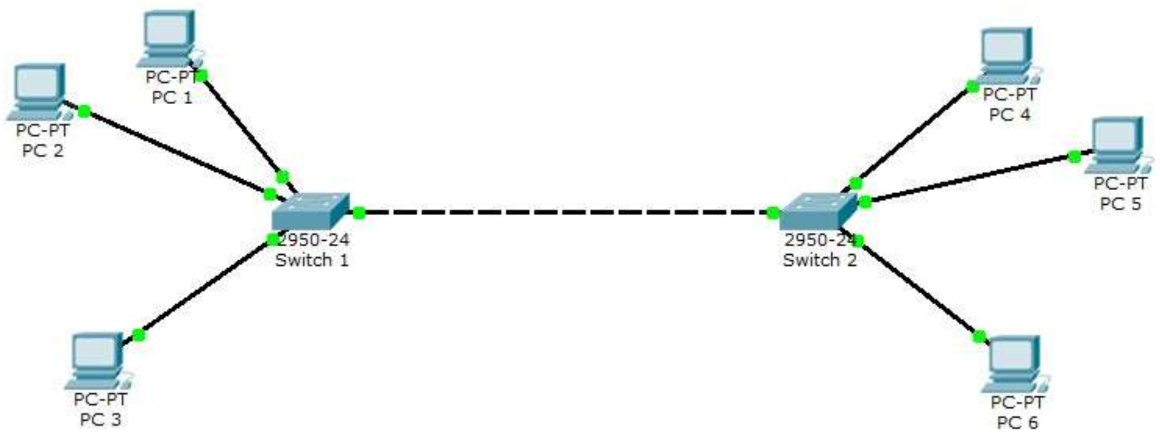
Answer will vary, but VLAN benefits include: better security, cost savings (efficient use of bandwidth and uplinks), higher performance (smaller broadcast domains), broadcast storm mitigation, improved IT staff efficiency, simpler project and application management.

3. Stage v (verify)

Home Activities:

Activity 1:

Configure the topology for VLAN:



PC configurations:

PC1 - >192.168.1.2 VLAN 2

PC2 - >192.168.1.3 VLAN 2

PC3 - >192.168.1.4 VLAN 3

PC4 - >192.168.1.6 VLAN 3

PC5 - >192.168.1.7 VLAN 3

PC6 - >192.168.1.8 VLAN 2

4) Stage a2 (assess)

Submit the home activity before next lab

LAB # 14

Statement Purpose:

Investigate the 802.11 wireless network protocols.

Activity Outcomes:

Students will have better understanding of the 802.11 protocol in terms of its various frames, data transfer mechanism and association/disassociation mechanism.

Instructor Note:

As pre-lab activity, read Chapter 6 from the book (Kurose, Jim, and Keith Ross. "Computer Networking: A Top Down Approach, 2012."), and also as given by your theory instructor.

1) StageJ(Journey)

Introduction

A wireless LAN (WLAN or WiFi) is a data transmission system designed to provide location-independent network access between computing devices by using radio waves rather than a cable infrastructure

In the corporate enterprise, wireless LANs are usually implemented as the final link between the existing wired network and a group of client computers, giving these users wireless access to the full resources and services of the corporate network across a building or campus setting.

The widespread acceptance of WLANs depends on industry standardization to ensure product compatibility and reliability among the various manufacturers.

The 802.11 specification [**IEEE Std 802.11 (ISO/IEC 8802-11: 1999)**] as a standard for wireless LANS was ratified by the Institute of Electrical and Electronics Engineers (IEEE) in the year 1997. This version of 802.11 provides for 1 Mbps and 2 Mbps data rates and a set of fundamental signaling methods and other services. Like all IEEE 802 standards, the 802.11 standards focus on the bottom two levels the ISO model, the physical layer and link layer (see figure below). Any LAN application, network operating system, protocol, including TCP/IP and Novell NetWare, will run on an 802.11-compliant WLAN as easily as they run over Ethernet. The major motivation and benefit from Wireless LANs is increased mobility. Untethered from conventional network connections, network users can move about almost without restriction and access LANs from nearly anywhere.

The other advantages for WLAN include cost-effective network setup for hard-to-wire locations such as older buildings and solid-wall structures and reduced cost of ownership-particularly in dynamic environments requiring frequent modifications, thanks to minimal wiring and installation costs per device and user. WLANs liberate users from dependence on hard-wired access to the network backbone, giving them anytime, anywhere network access. This freedom to roam offers numerous user benefits for a variety of work environments, such as:

- Immediate bedside access to patient information for doctors and hospital staff
- Easy, real-time network access for on-site consultants or auditors
- Improved database access for roving supervisors such as production line managers, warehouse auditors, or construction engineers
- Simplified network configuration with minimal MIS involvement for temporary setups such as trade shows or conference rooms
- Faster access to customer information for service vendors and retailers, resulting in better service and improved customer satisfaction
- Location-independent access for network administrators, for easier on-site troubleshooting and support
- Real-time access to study group meetings and research links for students

2) Stage a1 (apply)

Lab Activities:

Activity 1:

Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file Wireshark_802_11.pcap. This trace was collected using AirPcap and Wireshark running on a computer in the home network of one of the authors, consisting of a Linksys 802.11g combined access point/router, with two wired PCs and one wireless host PC attached to the access point/router. The author is fortunate to have other access points in neighboring houses available as well. In this trace file, we'll see frames captured on channel 6. Since the host and AP that we are interested in are not the only devices using channel 6, we'll see a lot of frames that we're not interested in for this lab, such as beacon frames advertised by a neighbor's AP also operating on channel 6. The wireless host activities taken in the trace file are:

•The host is already associated with the *30 Munroe St* AP when the trace begins.

•At *t = 24.82*, the host makes an HTTP request to http://gaia.cs.umass.edu/wiresharklabs/alice.txt. The IP address of gaia.cs.umass.edu is 128.119.245.12.

•At *t=32.82, t*he host makes an HTTP request to http://www.cs.umass.edu, whose IP address is 128.119.240.19.

•At *t = 49.58,* the host disconnects from the *30 Munroe St* AP and attempts to connect to the *linksys_ses_24086*. This is not an open access point, and so the host is eventually unable to connect to this AP.

•At *t=63.0* the host gives up trying to associate with the *linksys_ses_24086 AP,* and associates again with the *30 Munroe St* access point.
Once you have downloaded the trace, you can load it into Wireshark and view the trace using
the *File* pull down menu, choosing *Open*, and then selecting the Wireshark_802_11.pcap trace file.

Recall that beacon frames are used by an 802.11 AP to advertise its existence. To answer some of the questions below, you'll want to look at the details of the "IEEE 802.11" frame and subfields in the middle Wireshark window. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout27 to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question.

1. What are the SSIDs of the two access points that are issuing most of the beacon frames in this trace?

2. What are the intervals of time between the transmissions of the beacon frames the *linksys_ses_24086* access point? From the *30 Munroe St*. access point? (Hint: this interval of time is contained in the beacon frame itself).

3. What (in hexadecimal notation) is the source MAC address on the beacon frame from *30 Munroe St*? Recall from Figure 6.13 in the text that the source, destination, and BSS are three addresses used in an 802.11 frame. For a detailed discussion of the 802.11 frame structure, see section 7 in the IEEE 802.11 standards document (cited above).

4. What (in hexadecimal notation) is the destination MAC address on the beacon frame from *30 Munroe St*??

5. What (in hexadecimal notation) is the MAC BSS id on the beacon frame from *30 Munroe St*?

6. The beacon frames from the *30 Munroe St* access point advertise that the access point can support four data rates and eight additional "extended supported rates." What are these rates?

Solution:

1. What are the SSIDs of the two access points that are issuing most of the beacon frames in this trace? Answer SSID of first access points is 30 Munroe St. SSID of second access points is linksys12.

2. What are the intervals of time between the transmission of the beacon frames the linksys_ses_24086 access point? From the 30 Munroe St. access point?Answer intervals of time between the transmisson of the beacon frames the linksys_ses_24086 access point is Beacon Interval: 0.102400 [Seconds] .intervals of time between the transmisson of the beacon frames the 30 Munroe St. access point is Beacon Interval: 0.102400 [Seconds] .

3. What (in hexadecimal notation) is the source MAC address on the beacon frame from 30 Munroe St? Recall from Figure 6.13 in the text that the source, destination, and BSS are three addresses used in an 802.11 frame. For a detailed discussion of the 802.11 frame structure, see section 7 in the IEEE

802.11 standards document (cited above).Answer Source address: Cisco-Li_f7:1d:51 (00:16:b6:f7:1d:51).

4. What (in hexadecimal notation) is the destination MAC address on the beacon frame from 30 Munroe St??Answer Destination address: Broadcast (ff:ff:ff:ff:ff:ff).

5. What (in hexadecimal notation) is the MAC BSS id on the beacon frame from 30 Munroe St? Answer BSS Id: Cisco-Li_f7:1d:51 (00:16:b6:f7:1d:51).

6. The beacon frames from the 30 Munroe St access point advertise that the access point can support four data rates and eight additional "extended supported rates." What are these rates?Answer Supported Rates: 1.0(B) 2.0(B) 5.5(B) 11.0(B). Extended Supported Rates: 6.0(B) 9.0 12.0(B) 18.0 24.0(B) 36.0 48.0 54.0.

Activity 2:

Since the trace starts with the host already associated with the AP, let first look at data transfer over an 802.11 association before looking at AP association/disassociation. Recall that in this trace, at *t = 24.82*, the host makes an HTTP request to http://gaia.cs.umass.edu/wireshark-labs/alice.txt. The IP address of gaia.cs.umass.edu is 128.119.245.12. Then, at *t=32.82,* the host makes an HTTP request to http://www.cs.umass.edu.

1. Find the 802.11 frame containing the SYN TCP segment for this first TCP session (that downloads alice.txt). What are three MAC address fields in the 802.11 frame? Which MAC address in this frame corresponds to the wireless host (give the hexadecimal representation of the MAC address for the host)? To the access point? To the first-hop router? What is the IP address of the wireless host sending this TCP segment? What is the destination IP address? Does this destination IP address correspond to the host, access point, first-hop router, or some other network-attached device? Explain.

2. Find the 802.11 frame containing the SYNACK segment for this TCP session. What are three MAC address fields in the 802.11 frame? Which MAC address in this frame corresponds to the host? To the access point?To the first-hop router? Does the sender MAC address in the frame correspond to the IP address of the device that sent the TCP segment encapsulated within this datagram? (Hint: review Figure 5.19 in the text if you are unsure of how to answer this question, or the corresponding part of the previous question. It's particularly important that you understand this)

Solution:

1. Find the 802.11 frame containing the SYN TCP segment for this first TCP session (that downloads alice.txt). At what time is the TCP SYN sent?Answer TCP SYN sent 24.811093 Sec.What are three MAC address fields in the 802.11 frame? Which MAC address in this frame corresponds to the wireless host (give the hexadecimal representation of the MAC address for the host)? To the access point?To the first-hop router? Answer BSS Id: Cisco-Li_f7:1d:51 (00:16:b6:f7:1d:51). Source address: IntelCor_d1:b6:4f (00:13:02:d1:b6:4f). Destination address: Cisco-Li_f4:eb:a8 (00:16:b6:f4:eb:a8).What is the IP address of the wireless host sending this TCP segment? What is the destination IP address? Does this destination IP address correspond to the host, access point, first-hop router, or some other network-attached device? Answer IP address wireless host 192.168.1.109. Destination IP address 128.119.245.12.

2. Find the 802.11 frame containing the SYNACK segment for this TCP session. Awhat time is the TCP SYNACK received?Answer TCP received 24.827751 Sec.What are three MAC address fields in the 802.11 frame containing the SYNACK? Which MAC address in this frame corresponds to the host? To the access point?To the first-hop router? Answer Destination address: 91:2a:b0:49:b6:4f (91:2a:b0:49:b6:4f). BSS Id: Cisco-Li_f7:1d:51 (00:16:b6:f7:1d:51). Source address: Cisco-Li_f4:eb:a8 (00:16:b6:f4:eb:a8).Does the sender MAC address in the frame correspond to the IP address of the device that sent the TCP segment encapsulated within this datagram? Answer sender MAC address is MAC address of access point. it is not TCP segment.

Activity 3:

Recall from Section 6.3.1 in the text that a host must first *associate* with an access point before sending data. Association in 802.11 is performed using the ASSOCIATE REQUEST frame (sent from host to AP, with a frame type 0 and subtype 0, see Figure 6.13 in the text) and the ASSOCIATE RESPONSE frame (sent by the AP to a host with a frame type 0 and subtype of 1, in response to a received ASSOCIATE REQUEST). For a detailed explanation of each field in the 802.11 frame, see page 34 (Section 7) of the 802.11 spec at http://gaia.cs.umass.edu/wireshark-labs/802.11-1999.pdf.

1. What two actions are taken (i.e., frames are sent) by the host in the trace just after *t=49*, to end the association with the *30 Munroe St* AP that was initially in place when trace collection began? (Hint: one is an IP-layer action, and one is an 802.11-Page 93 of 96layer action). Looking at the 802.11 specification, is there another frame that you might have expected to see, but don't see here?

2. Examine the trace file and look for AUTHENICATION frames sent from the host to an AP and vice versa. How many AUTHENTICATION messages are sent from the wireless host to the *linksys_ses_24086* AP (which has a MAC address ofCisco_Li_f5:ba:bb) starting at around *t=49?* .

3. Does the host want the authentication to require a key or be open?

4. Do you see a reply AUTHENTICATION from the *linksys_ses_24086* AP in thetrace?

5. Now let's consider what happens as the host gives up trying to associate with the*linksys_ses_24086* AP and now tries to associate with the *30 Munroe St* AP. Look for AUTHENICATION frames sent from the host to and AP and vice versa. At what times are there an AUTHENTICATION frame from the host to the *30 Munroe St.* AP, and when is there a reply AUTHENTICATION sent from that AP to the host in reply? (Note that you can use the filter expression "wlan.fc.subtype == 11andwlan.fc.type == 0 and wlan.addr == IntelCor_d1:b6:4f" to display only theAUTHENTICATION frames in this trace for this wireless host.)

Solution:

1. What two actions are taken (i.e., frames are sent) by the host in the trace just after t=49, to end the association with the 30 Munroe St AP that was initially in place when trace collection began, and at what times are these frames sent?Answer time DHCP release is 49.583615 second. timeDeauthentication is 49.609617 second.

2. Examine the trace file and look for AUTHENICATION frames sent from the host to an AP and vice versa. When is the first AUTHENTICATION frame sent from the wireless host to the linksys_ses_24086 AP (which has a MAC address of Cisco_Li_f5:ba:bb) starting at around t=49? .

Answer the first AUTHENTICATION frame sent from the wireless host to the linksys_ses_24086 AP is49.638857 second.

3. Does the host want the authentication to require a key or be open?Answer Host want the authentication to require be open system .

4. Do you see a reply AUTHENTICATION from the linksys_ses_24086 AP in the trace?Answer None.

5. Now let's consider what happens as the host gives up (sometime after t = 63.0) trying to associate with the linksys_ses_24086 AP and now tries to associate with the 30 Munroe St AP. Look for AUTHENICATION frames sent from the host to and AP and vice versa. At what times are there an AUTHENTICATION frame from the host to the 30 Munroe St. AP, and when is there a reply AUTHENTICATION sent from that AP to the host in reply?Answer time AUTHENTICATION frame from the host to the 30 Munroe St. AP is 63.169071 second. time reply AUTHENTICATION sent from that AP to the host in reply is 63.169707 second.

3) Stage v (verify)

Home Activities:

Activity 1:

Recall from Section 6.3.1 in the text that a host must first *associate* with an access point before sending data. Association in 802.11 is performed using the ASSOCIATE REQUEST frame (sent from host to AP, with a frame type 0 and subtype 0, see Figure 6.13 in the text) and the ASSOCIATE RESPONSE frame (sent by the AP to a host with a frame type 0 and subtype of 1, in response to a received ASSOCIATE REQUEST). For a detailed explanation of each field in the 802.11 frame, see page 34 (Section 7) of the 802.11 spec at http://gaia.cs.umass.edu/wireshark-labs/802.11-1999.pdf.
An ASSOCIATE REQUEST from host to AP, and a corresponding ASSOCIATE RESPONSE frame from AP to host are used for the host to associated with an AP. At what time is there an ASSOCIATE REQUEST from host to the *30 Munroe St* AP? When is the corresponding ASSOCIATE REPLY sent? (Note that you can use the filter expression "wlan.fc.subtype< 2 and wlan.fc.type == 0 and wlan.addr ==IntelCor_d1:b6:4f" to display only the ASSOCIATE REQUEST and ASSOCIATERESPONSE frames for this trace.)

What transmission rates is the host willing to use? The AP? To answer thisquestion, you will need to look into the parameters fields of the 802.11 wireless LANmanagement frame.

Our trace contains a number of PROBE REQUEST and PROBE RESPONSE frames.What are the sender, receiver and BSS ID MAC addresses in these frames? What isthe purpose of these two types of frames? (To answer this last question, you'll needto dig into the online references cited earlier in this lab).

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.

LAB # 15

Statement Purpose:

1.1. Investigate the Secure Sockets Layer (SSL) protocol, focusing on the SSL records sent over a TCP connection.
1.2. We'll do so by analyzing a trace of the SSL records sent between your host and an ecommerce server.
1.3. We'll investigate the various SSL record types as well as the fields in the SSL messages.

Activity Outcomes:

Students will gain better understanding of SSL.

Instructor Note:

As pre-lab activity, read Chapter 8 from the book (Kurose, Jim, and Keith Ross. "Computer Networking: A Top Down Approach, 2012."), and also as given by your theory instructor.

1) StageJ(Journey)

Introduction

The Transmission Control Protocol/Internet Protocol (TCP/IP) governs the transport and routing of data over the Internet. Other protocols, such as the HyperText Transport Protocol (HTTP), Lightweight Directory Access Protocol (LDAP), or Internet Messaging Access Protocol (IMAP), run "on top of" TCP/IP in the sense that they all use TCP/IP to support typical application tasks such as displaying web pages or running email servers. The SSL protocol runs above TCP/IP and below higher-level protocols such as HTTP or IMAP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to an SSL-enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

These capabilities address fundamental concerns about communication over the Internet and other TCP/IP networks: SSL server authentication allows a user to confirm a server's identity. SSL-enabled client software can use standard techniques of public-key cryptography to check that a server's certificate and public ID are valid and have been issued by a certificate authority (CA) listed in the client's list of trusted CAs. This confirmation might be important if the user, for example, is sending a credit card number over the network and wants to check the receiving server's identity. SSL client authentication allows a server to confirm a user's identity. Using the same techniques as those used for server authentication, SSL-enabled server software can check that a client's certificate and public

ID are valid and have been issued by a certificate authority (CA) listed in the server's list of trusted CAs. This confirmation might be important if the server, for example, is a bank sending confidential financial information to a customer and wants to check the recipient's identity.

An encrypted SSL connection requires all information sent between a client and a server to be encrypted by the sending software and decrypted by the receiving software, thus providing a high degree of confidentiality. Confidentiality is important for both parties to any private transaction. In addition, all data sent over an encrypted SSL connection is protected with a mechanism for detecting tampering-that is, for automatically determining whether the data has been altered in transit. The SSL protocol includes two sub-protocols: the SSL record protocol and the SSL handshake protocol. The SSL record protocol defines the format used to transmit data. The SSL handshake protocol involves using the SSL record protocol to exchange a series of messages between an SSL-enabled server and an SSL-enabled client when they first establish an SSL connection.

This exchange of messages is designed to facilitate the following actions:

- Authenticate the server to the client.
- Allow the client and server to select the cryptographic algorithms, or ciphers, that they both support.
- Optionally authenticate the client to the server.
- Use public-key encryption techniques to generate shared secrets.
- Establish an encrypted SSL connection.
- For more information about the handshake process, see "The SSL Handshake."

2) Stage a1 (apply)

<u>Lab Activities:</u>

Activity 1:

The first step is to capture the packets in an SSL session. To do this, you should go to your favorite e-commerce site and begin the process of purchasing an item (but terminating before making the actual purpose!). After capturing the packets with Wireshark, you should set the filter so that it displays only the Ethernet frames that contain SSL records sent from and received by your host. (An SSL record is the same thing as an SSL message.) You should obtain something like screenshot on the previous page.

Your Wireshark GUI should be displaying only the Ethernet frames that have SSL records. It is important to keep in mind that an Ethernet frame may contain one or more SSL records. (This is very different from HTTP, for which each frame contains either one complete HTTP message or a portion of a HTTP message.) Also, an SSL record may not completely fit into an Ethernet frame, in which case multiple frames will be needed to carry the record. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout28 to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question

1. For each of the first 8 Ethernet frames, specify the source of the frame (client or server), determine the number of SSL records that are included in the frame, and list the SSL record types that are included in the frame. Draw a timing diagram between client and server, with one arrow for each SSL record.

2. Each of the SSL records begins with the same three fields (with possibly different values). One of these fields is “content type” and has length of one byte. List all three fields and their lengths.

Solution:

1. For each of the first 8 Ethernet frames, specify the source of the frame (client or server), determine the number of SSL records that are included in the frame, and list the SSL record types that are included in the frame. Draw a timing diagram between client and server, with one arrow for each SSL record.answer

2. Each of the SSL records begins with the same three fields (with possibly different values). One of these fields is “content type” and has length of one byte. List all three fields and their lengths.answer Content Type : Handshake(22) has length 1 byte Version : TLS 1.0 (0×0301) has length 2 byte Length : 111 has length 2 byte ClientHello Record

Activity 2:

ClientHello Record:

3. Expand the ClientHello record. (If your trace contains multiple ClientHello records, expand the frame that contains the first one.) What is the value of the content type?

4. Does the ClientHello record contain a nonce (also known as a “challenge”)? If so, what is the value of the challenge in hexadecimal notation?

5. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?

Solution:

3. Expand the ClientHello record. (If your trace contains multiple ClientHello records, expand the frame that contains the first one.) What is the value of the content type?answer value of the content type is Handshake (22)

4. Does the ClientHello record contain a nonce (also known as a “challenge”)? If so, what is the value of the challenge in hexadecimal notation?answer value of the challenge in hexadecimal notation is48ca936dccacffd6d73613ac9ed9bb1fe52ca43424577b37b16d26fdf e14ef98 .

5. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?answer yes, ClientHello record advertise the cyber suites it supports does. , So the first lisetd suite is Cipher Suite: TLSRSAWITHAES128CBCSHA (0×002f), So Public-key algorithm is RSA Symmetric-key algorithm is AES 128 Bit Cipher Block Chaining Hash algorithm is Secure Hash Algorithm ServerHello Record.

3) Stage v (verify)

<u>Home Activities:</u>

Activity 1:

Your Wireshark GUI should be displaying only the Ethernet frames that have SSL records. It is important to keep in mind that an Ethernet frame may contain one or more SSL records. (This is very different from HTTP, for which each frame contains either one complete HTTP message or a portion of a HTTP message.) Also, an SSL record may not completely fit into an Ethernet frame, in which case multiple frames will be needed to carry the record. Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout28 to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question

ServerHello Record:
6. Locate the ServerHello SSL record. Does this record specify a chosen cipher suite? What are the algorithms in the chosen cipher suite?

7. Does this record include a nonce? If so, how long is it? What is the purpose of the client and server nonces in SSL?
8. Does this record include a session ID? What is the purpose of the session ID?

9. Does this record contain a certificate, or is the certificate included in a separate record. Does the certificate fit into a single Ethernet frame?
Client Key Exchange Record:

10. Locate the client key exchange record. Does this record contain a pre-master secret? What is this secret used for? Is the secret encrypted? If so, how? How long is the encrypted secret? Change Cipher Spec Record (sent by client) and Encrypted Handshake Record:

11. What is the purpose of the Change Cipher Spec record? How many bytes is the record in your trace?

12. In the encrypted handshake record, what is being encrypted? How?

13. Does the server also send a change cipher record and an encrypted handshake record to the client? How are those records different from those sent by the client?
Application Data

14. How is the application data being encrypted? Do the records containing application data include a MAC? Does Wireshark distinguish between the encrypted application data and the MAC?

15. Comment on and explain anything else that you found interesting in the trace.

4) Stage a2 (assess)

Assignment:

For this student will submit Lab Assignment before the deadline.