**Lab Exercise 1 – 2 IPv4**

**Objective**

To learn about the details of IP (Internet Protocol). IP is the network layer protocol used throughout the Internet. We will examine IP version 4, since it is ubiquitously deployed, while the IP version 6 is partly deployed. Review those sections before doing this lab.

**Requirements**

**Wireshark:** This lab uses the Wireshark software tool to capture and examine a packet trace.

**wget / curl:** This lab uses wget (Linux and Windows) and curl (Mac) to fetch web resources. wget and curl are command-line programs that let you fetch a URL. Unlike a web browser, which fetches and executes entire pages, wget and curl give you control over exactly which URLs you fetch and when you fetch them. Under Linux, wget can be installed via your package manager. Under Windows, wget is available as a binary; look for download information on <http://www.gnu.org/software/wget/>. Under Mac, curl comes installed with the OS. Both have many options (try “wget --help” or “curl --help” to see) but a URL can be fetched simply with “wget URL” or “curl URL ”.

**traceroute / tracert:** This lab uses “traceroute” to find the router level path from your computer to a remote Internet host. traceroute is a standard command-line utility for discovering the Internet paths that your computer uses. It is widely used for network troubleshooting. It comes pre-installed on Window and Mac, and can be installed using your package manager on Linux. On Windows, it is called “tracert”. It has various options, but simply issuing the command “traceroute www.bit.edu.cn” will cause your computer to find and print the path to the remote computer (here [www.bit.edu.cn](http://www.bit.edu.cn)).

**Turn in**

Hand in the trace file you captured and exercise report including your answers to the questions and figures you drew.

**Step 1: Capture a Trace**

1. Pick a URL at a remote server and check that you can fetch the contents with wget or curl, e.g., “wget http://www.bit.edu.cn/” or “curl http://www.bit.edu.cn/”. This will fetch the resource and either writes it to a file (wget) or to the screen (curl). With wget, you want a single response with status code “200 OK”. If the fetch does not work then try a different URL; keep in mind that you may be referring to a URL by a shortcut for which browsers must do work to find the intended content, e.g.,http://mit.edu may really be http://web.mit.edu/index.html. If no URLs seem to work then debug your use of wget/curl or your Internet connectivity.
2. Perform a traceroute to the same remote server to check that you can discover information about the network path. On Windows, type, e.g., “tracert www.bit.edu.cn”. On Linux / Mac, type, e.g., “traceroute www.bit.edu.cn”. If you are on Linux / Mac and behind a NAT (as most home users or virtual machine users) then use the –I option (that was a capital i) to traceroute, e.g., “traceroute –I www.bit.edu.cn”. This will cause traceroute to send ICMP probes like tracert instead of its usual UDP probes; ICMP probes are better able to pass through NAT boxes. A successful example is shown below; save the output as you will need it for later steps. Note that traceroute may take up to a minute to run. Each line shows information about the next IP hop from the computer running traceroute towards the target destination. The lines with “\*”s indicate that there was no response from the network to identity that segment of the Internet path. Some unidentified segments are to be expected. However, if traceroute is not working correctly then nearly all the path will be “\*”s. In this case, try a different remote server, experiment with traceroute.

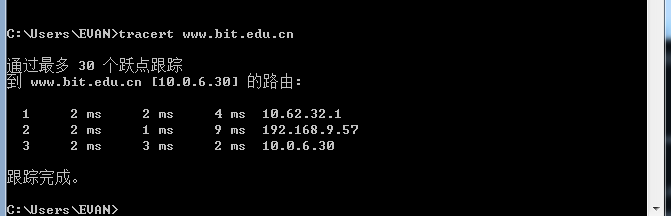


Figure 1: Running traceroute (as tracert on Windows)

1. Launch Wireshark and start a capture with a filter of “**tcp port 80**“. Make sure to check “enable network name resolution”. We use this filter to record only standard web traffic. Name resolution will translate the IP addresses of the computers sending and receiving packets into names. It will help you to recognize whether the packets are going to or from your computer. Your capture window should be similar to the one pictured below, other than our highlighting. Select the interface from which to capture as the main wired or wireless interface used by your computer to connect to the Internet. If unsure, guess and revisit this step later if your capture is not successful. Uncheck “capture packets in promiscuous mode”. This mode is useful to over-hear packets sent to/from other computers on broadcast networks. We only want to record packets sent to/from your computer. Leave other options at their default values. The capture filter, if present, is used to prevent the capture of other traffic your computer may send or receive. On Wireshark 1.8, the capture filter box is present directly on the options screen, but on Wireshark 1.9, you set a capture filter by double-clicking on the interface.



Figure 2: Setting up the capture options

1. After the capture is started, repeat the wget/curl command above. This time, the packets will also be recorded by Wireshark.
2. After the command is complete, return to Wireshark and stop the trace. You should now have a short trace similar to that shown in the figure below, along with the output of a traceroute you ran earlier to the corresponding server.

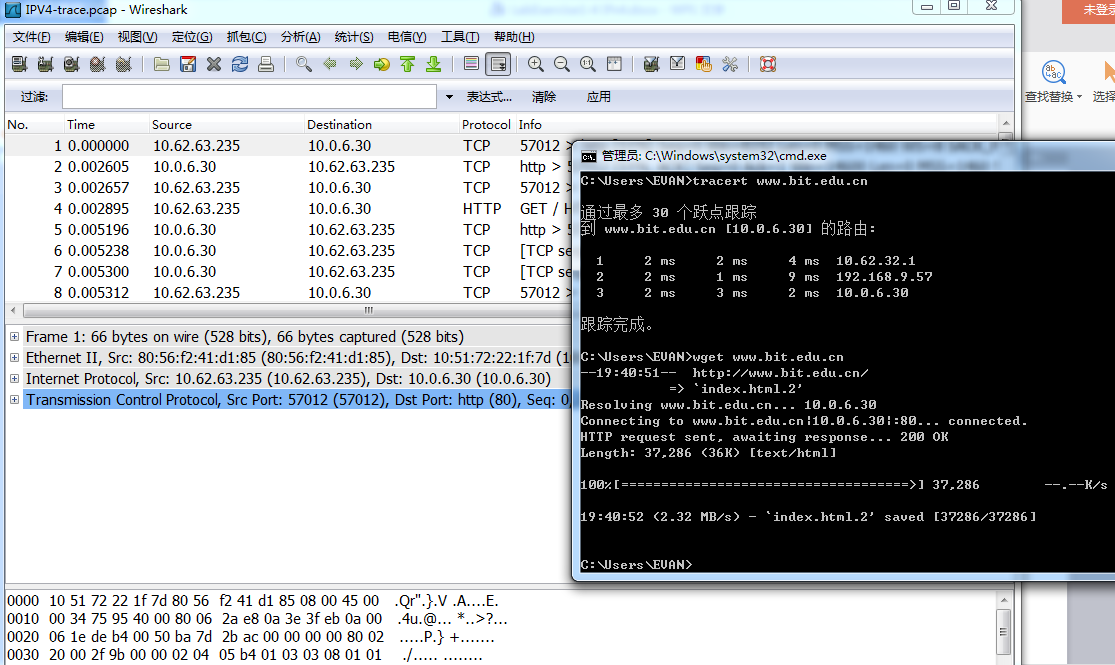


Figure 3: Trace of wget/curl traffic showing the details of the IP header

**Step 2: Inspect the Trace**

Select any packet in the trace and expand the IP header fields (using the “+” expander or icon) to see the details. You can simply click on a packet to select it (in the top panel). You will see details of its structure (in the middle panel) and the bytes that make up the packet (in the bottom panel). Our interest is the IP header, and you may ignore the other higher and lower layer protocols. When you click on parts of the IP header, you will see the bytes that correspond to the part highlighted in the bottom panel. We have expanded the IP header and clicked on all the IP header fields in the figure above.

**Step 3: IP Packet Structure**

To show your understanding of IP, sketch a figure of an IP packet you studied. It should show the position and size in bytes of the IP header fields as you can observe using Wireshark. Since you cannot easily determine sub-byte sizes, group any IP fields that are packed into the same bytes. Your figure can simply show the frame as a long, thin rectangle. Try not to look at the figure of an IPv4 packet in your text; check it afterwards to note and investigate any differences.

**By looking at the IP packets in your trace, answer these questions:**

1. What are the IP addresses of your computer and the remote server?

本机：10.62.63.235

[www.bit.edu.cn:10.0.6.30](http://www.bit.edu.cn:10.0.6.30)

1. Does the Total Length field include the IP header plus IP payload, or just the IP payload?

包括IP包和的头和负载两部分

1. How does the value of the Identification field change or stay the same for different packets? For instance, does it hold the same value for all packets in a TCP connection or does it differ for each packet? Is it the same in both directions? Can you see any pattern if the value does change?

根据包的发送IPdizhi 不同，在足够大的数字空间中，分别使用两段连续的数字序列来填充识别区，来自不同地址的识别码分别在自己的序列中连续变化

1. What is the initial value of the TTL field for packets sent from your computer? Is it the maximum possible value, or some lower value?

TTL=128s是最大的可能值

1. How can you tell from looking at a packet that it has not been fragmented? Most often IP packets in normal operation are not fragmented. But the receiver must have a way to be sure.

IP header 有一个标记字段FLAG根据这个字段，接受者可以分辨该IP包是否被分段化，FLAG =0X02时没有被分段，FLAG = 00时可能被分段了。

1. What is the length of the IP Header and how is this encoded in the header length field? Hint: no-tice that only 4 bits are used for this field, as the version takes up the other 4 bits of the byte. You may guess and check your text.

IP包：长度为20个字节

Header length部分占用了4位，提供了一个16进制数5描述了头的长度

**Step 4: Internet Paths**

The source and destination IP addresses in an IP packet denote the endpoints of an Internet path, not the IP routers on the network path the packet travels from the source to the destination. traceroute is a utility for discovering this path. It works by eliciting responses (ICMP TTL Exceeded messages) from the router 1 hop away from the source towards the destination, then 2 hops away from the source, then 3 hops, and so forth until the destination is reached. The responses will identify the IP address of the router. The output from traceroute normally prints the information for one hop per line, including the measured round trip times and IP address and DNS names of the router. The DNS name is handy for working out the organization to which the router belongs. Since traceroute takes advantage of common router implementations, there is no guarantee that it will work for all routers along the path, and it is usual to see “\*” responses when it fails for some portions of the path.

Using the traceroute output, **sketch a drawing** of the network path. Show your computer (lefthand side) and the remote server (righthand side), both with IP addresses, as well as the routers along the path between them numbered by their distance on hops from the start of the path. You can find the IP address of your computer and the remote server on the packets in the trace that you captured. The output of traceroute will tell you the hop number for each router.

To finish your **drawing**, label the routers along the path with the name of the real-world organization to which they belong. To do this, you will need to interpret the domain names of the routers given by traceroute. If you are unsure, label the routers with the domain name of what you take to be the organization. Ignore or leave blank any routers for which there is no domain name (or no IP address).

主机 10.62.54.132

Hop1：10.62.32.1

本地局域网内IP

Hop2:192.168.9.57

本地局域网内IP

Dst：10.0.6.30

www.bit.edu.cn

Figure 4: Logical structure of the network