

作业四说明

要求：

文件名：学号+姓名+作业4

文件标题：学号+姓名+作业 4

回答文档中的所有问题。

In this lab, we'll investigate the IP protocol, focusing on the IP datagram. We'll do so by analyzing a trace of IP datagrams sent and received by an execution of the `traceroute` program (the `traceroute` program itself is explored in more detail in the Wireshark ICMP lab). We'll investigate the various fields in the IP datagram, and study IP fragmentation in detail.

1. Capturing packets from an execution of traceroute

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.

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

- **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,

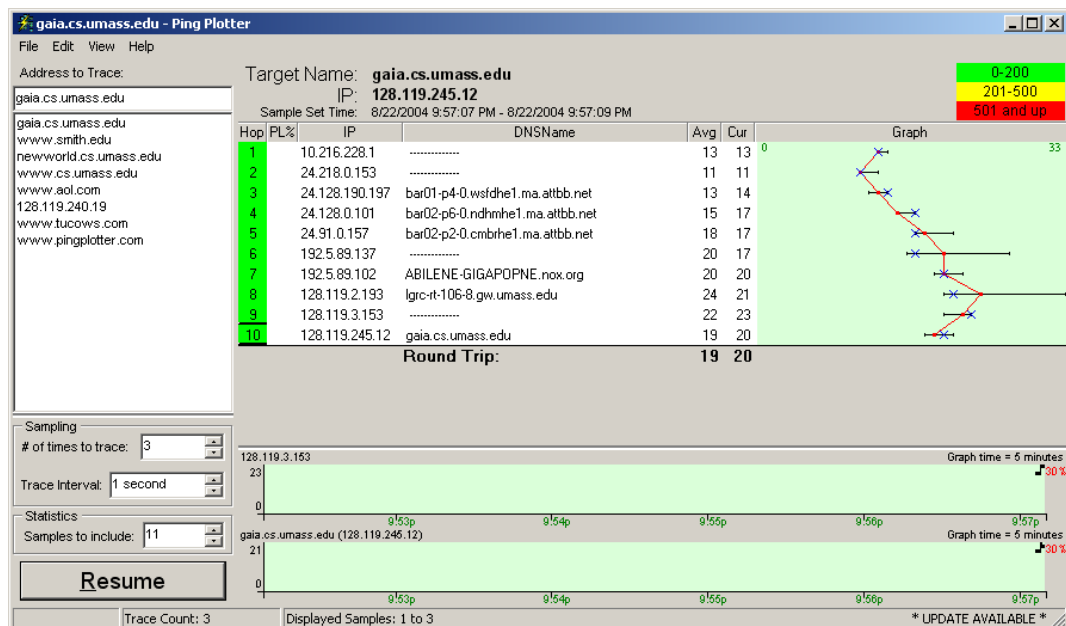
after waiting *Trace Interval* amount of time. The value of *Trace Interval* and the number of intervals can be explicitly set in *pingplotter*.

- **Linux/Unix/MacOS.** With the Unix/MacOS `traceroute` 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 `tracert` 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.

2. A look at the captured trace

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 Unix machine 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 *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. 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.

ip-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: Expression... Clear Apply

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|------------------|---------------|----------|--------|--|
| 1 | 0.000000 | Telebit_73:8d:ce | Broadcast | ARP | 60 | who has 192.168.1.117? Tell 192.168.1.104 |
| 2 | 4.866867 | 192.168.1.100 | 192.168.1.1 | UDP | 174 | Source port: 30955 Destination port: ssdp |
| 3 | 4.868147 | 192.168.1.100 | 192.168.1.1 | UDP | 175 | Source port: 30955 Destination port: ssdp |
| 4 | 5.363536 | 192.168.1.100 | 192.168.1.1 | UDP | 174 | Source port: 30955 Destination port: ssdp |
| 5 | 5.364799 | 192.168.1.100 | 192.168.1.1 | UDP | 175 | Source port: 30955 Destination port: ssdp |
| 6 | 5.864428 | 192.168.1.100 | 192.168.1.1 | UDP | 174 | Source port: 30955 Destination port: ssdp |
| 7 | 5.865461 | 192.168.1.100 | 192.168.1.1 | UDP | 175 | Source port: 30955 Destination port: ssdp |
| 8 | 6.163045 | 192.168.1.102 | 128.59.23.100 | ICMP | 98 | Echo (ping) request id=0x0300, seq=20483/848, ttl=1 |
| 9 | 6.176826 | 10.216.228.1 | 192.168.1.102 | ICMP | 70 | Time-to-live exceeded (Time to live exceeded in transit) |
| 10 | 6.188629 | 192.168.1.102 | 128.59.23.100 | ICMP | 98 | Echo (ping) request id=0x0300, seq=20739/849, ttl=2 |
| 11 | 6.202957 | 24.218.0.153 | 192.168.1.102 | ICMP | 70 | Time-to-live exceeded (Time to live exceeded in transit) |
| 12 | 6.208597 | 192.168.1.102 | 128.59.23.100 | ICMP | 98 | Echo (ping) request id=0x0300, seq=20995/850, ttl=3 |
| 13 | 6.234505 | 24.128.190.197 | 192.168.1.102 | ICMP | 70 | Time-to-live exceeded (Time to live exceeded in transit) |

Frame 8: 98 bytes on wire (784 bits), 98 bytes captured (784 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: 128.59.23.100 (128.59.23.100)

Version: 4

Header Length: 20 bytes

Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))

Total Length: 84

Identification: 0x32d0 (13008)

Flags: 0x00

Fragment offset: 0

0000 00 06 25 da af 73 00 20 e0 8a 70 1a 08 00 45 00 ..%.s. .p...E.

0010 00 54 32 d0 00 00 01 01 2d 2c 0c a8 01 66 80 3b ..T2.....f..;

0020 17 64 08 00 17 ca 03 00 50 03 37 32 20 aa aa aa ..d.....P.T2...

0030 aa aa aa aa aa aa aa aa aa aa aa aa aa aa aa

0040 aa aa aa aa aa aa aa aa aa aa aa aa aa aa aa

0050 aa aa aa aa aa aa aa aa aa aa aa aa aa aa aa

0060 aa aa

Frame (frame), 98 bytes | Packets: 380 Displayed: 380 Marked: 0 Load time: 0:00.006 | Profile: Default

- What is the IP address of your computer?
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?

Fragmentation

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

10. 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? 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?
11. 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 there more fragments? How can you tell?
12. 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.

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