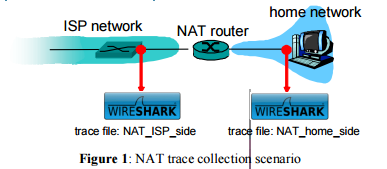
Wireshark Day 2

NAT: Download the wireshark-traces.zip file from the resources folder and open the NAT\_home\_side and NAT\_ISP\_side files in wireshark.



In the NAT\_home\_side file, filter out the packets so you only see HTTP, then answer the following questions.

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 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 NAT\_ISP\_side file, filter out the packets so you only see HTTP, then answer the following questions.

6. In the NAT\_ISP\_side trace file, find the HTTP GET message that 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 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 above?

7. 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.

8. 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 above?

9. 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 above?

ICMP:

1. Start a new capture, then go to command prompt and type in ping –n 10 [www.ust.hk](http://www.ust.hk) before ending the capture. Then, filter out ICMP.

* What is the IP address of your host? What is the IP address of the destination host?
* Why is it that an ICMP packet does not have source and destination port numbers?
* 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?
* 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?

2. Start another new capture and type in tracert [www.inria.fr](http://www.inria.fr) to command prompt before ending the capture.

* What is the IP address of your host? What is the IP address of the target destination host?
* 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?
* 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?
* Examine the ICMP error packet in your screenshot. It has more fields than the ICMP echo packet. What is included in those fields?
* 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?
* Within the tracert measurements, is there a link whose delay is significantly longer than others?

Ethernet and ARP: Clear out your browser’s internet history, then start up a new capture and go to <http://gaia.cs.umass.edu/wireshark-labs/HTTP-ethereal-lab-file3.html>. Stop the capture.

Answer the following questions based on the contents of the Ethernet frame containing the HTTP GET message.

1. What is the 48-bit Ethernet address of your computer?

2. What is the 48-bit destination address in the Ethernet frame? What device has this as its Ethernet address?

3. Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?

4. 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.

5. What is the value of the Ethernet source address? What device has this as its Ethernet address?

6. What is the destination address in the Ethernet frame? Is this the Ethernet address of your computer?

7. Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?

8. 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?

DHCP: Go into command prompt and type in ipconfig /release before you start up your wireshark capture. After you do that, start you capture and type in ipconfig /renew into command prompt. TO filter out unnecessary data, use the bootp filter.

1. Are DHCP messages sent over UDP or TCP?

2. What is the link-layer (e.g., Ethernet) address of your host?

3. What values in the DHCP discover message differentiate this message from the DHCP request message?

4. 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?

5. What is the IP address of your DHCP server?

6. 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 to clarify.

802.11: Open up the wireshark traces zip file and extract Wireshark\_802\_11.pcap, then open it up in wireshark. Note that this file was generated using 802.11g in infrastructure mode.

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?

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?

7. 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?

SSL: Either use the SSL capture file from the provided zip file or make your own capture by accessing a site that uses SSL. Ex: <https://drive.google.com>, <https://www.github.com>. Then, filter out the packets so you’re only viewing SSL.

1. For each of the first 8 Ethernet frames, identify the source of the frame (client or server), determine the number of SSL records that are included in the frame, and examine the SSL record types that are included in the frame. Be able to explain what these mean and how we can use them.

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. What are the three fields and their lengths?

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?

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 are session IDs used for?

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?

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?

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?

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?

Lab created by Mary Sawyer.