CNT 4700C Computer Networks Fundamentals, Fall 2019

Instructor: Prof. Ahmed Helmy

Homework 1: Internet Architecture & Application Layer

Due Date: Sept 24th, through canvas

Instructions: Be precise and to the point. Many questions require answers using a sentence or two). Some questions will ask you to elaborate, use visual aids or graphs, or show traces/code. Use your own words and phrases, do not copy from any other source.

This is an example solution. It is not the only solution, nor is it expected to match the words provided in students' solutions. It is also detailed beyond what is expected from students to provide a reference and discussion of the concepts beyond just answering the questions. The extra parts are labeled [Extra].

Q1. A. What is a network protocol?

note: This is discussed in Ch1 (slide 1-10), and Ch2 (slide 2-13) among others.

Answer: A network protocol is a comprehensive description of the rules governing the communication between the network devices implementing the protocol. It includes rules about when and how to send and receive message, and how to prepare and process these messages, and the associated timers, state/memory, tables, and flags/variables. A protocol is detailed in a 'protocol specification'. The protocol specification also states the format of the messages, every field and bit in the message, and the semantics of those fields.

[Extras: example Internet protocols include HTTP, SNMP, TCP, IP, 802.11, among others.

Protocols should be 'complete' but they rarely are, leaving some cases uncovered in which the behavior of a protocol is 'unspecified' or 'under-specified', which could lead to unexpected/unpredictable behavior. This could be a vulnerability for potential attacks.]

B. What is the body that standardizes the Internet protocols?

Answer: The Internet Engineering Task Force (IETF) standardizes Internet protocols through Request For Comments (RFCs) and Internet Drafts (IDs).

C. In which language are the protocol specifications largely written? What is the main problem with that format?

Answer: This was discussed in the lecture in the first couple of lectures.

The main language is 'English'! The problem with English (or any *natural language* for that matter) is that it is not always precise and has ambiguities in terms of interpretation, which may lead to different software implementations that may not be interoperable (say between vendors of routers or networking devices). Thus, different parts of the Internet may not be able to communicate correctly or efficiently due to this difference. This is in contrast to some form of more precise representation as in pseudo code, finite state machines, or any computer language (reference code). Openness by itself does not lead to interoperability, but precision in representation is also necessary.

[extra] in class we also talked about IETF sessions/discussions, and the lengthy discussions that people have about the 'may', 'should' and 'must' clauses in the standards (RFCs, or Internet drafts) regarding optional, recommended and required actions in the protocols.

(small point) The fact that it's in English and not accessible to non-English speakers is a small point (not emphasized in class), but translations exist and English is becoming quite standard for many main publications in the world.

D. [Extra:] Write a simple protocol spec for the following task: consider two people playing a game where one has a secret number from 1 to 10 and the other is attempting to guess it. But the two people can only communicate by each alternatively saying a single number.

Answer: Example protocol (any other reasonable example outlining the messages sent and received and the actions taken should be acceptable, and will be assessed based on details):

General description: there are two roles in this protocol, a ‘guesser’ that guesses the secret number then processes the response, and a ‘verifier’ that receives the guess from the guesser, processes it and sends back a response. It includes two messages, the ‘guess’ and the ‘verify’. These messages are triggered and processed as follows:

- Guesser side:
    - Sending a ‘guess’ message: the guesser *must* send a ‘guess’ message to start the game, with the number ‘0’ included therein. To send other guesses in response to a ‘verify’ message, the guesser picks a number between 1 and 10 to include in the guess message. It also sends a guess message after the verifier’s respond is ‘0’. The number included in the message *may* be generated randomly. The guesser *should* remember previous numbers generated with wrong answer to avoid repeating the same mistake.
    - Receiving a ‘verify’ message: upon receiving a verify message, the number in it is checked. If it is ‘0’ then the previous guess is wrong and another guess message is sent. If it is ‘1’ then the previous guess is correct and the game is stopped.
- Verifier side:
    - Receiving a ‘guess’ message: upon receiving a guess message, the number in it is checked. If it is ‘0’ then this is a start of a new game, and the verifier *must* send a verify message with ‘0’ and *must* generate a *secret* number between 1 and 10 and stores this number. The *secret* number *may* be generated randomly. Otherwise, if the number in the guess message is between 1 and 10, then store it in the ‘guessed’ number variable and generate/send a verify message.
    - Sending a ‘verify’ message: the message includes a ‘verify’ number. The verify number is determined by comparing the ‘guessed’ number to the stored *secret* number. If the numbers are equal, then include ‘1’ in the verify message, otherwise include ‘0’.

Q2. A. What are the basic paradigms (or models) of communication? (mention 3)

Answer: This was discussed in the application layer context, and programming network applications, in the 1st 12 slides of ch2.

The three models are: 1- client-server model, 2- peer-to-peer model, 3- hybrid (using both client-server and peer-to-peer)

B. What are the processes are needed to support all these paradigms?

Answer: only two processes are needed in all these models: the client process and the server model. [Extra: in client-server model the client process runs at the end-system (computer, laptop) and the server process runs on the server. In peer-to-peer model, each end-system runs both processes. They hybrid model encompasses both the other models.]

Q3. What are the advantages and disadvantages of:

A. Hierarchical network architectures
Ch1 slide 1-44

- Advantages
  - ❖ Isolates and scopes internal dynamics: dampens oscillations, providing stability to the overall network
  - ❖ Supports scalability: aggregation/summary per domain for smaller, more efficient routing tables
  - ❖ Allows for flexibility: domains deploy different protocols, and policies
- Disadvantages
  - ❖ Overhead of establishing and maintaining the hierarchy (esp. for mobile, dynamic networks)
  - ❖ Sub-optimality of routing

B. Protocol layering
Ch1 slide 1-51 (two advantages and two disadvantages are enough)

Advantages:

- explicit structure allows identification, relationship of complex system's pieces
  - ❖ layered reference model for discussion
- modularization eases maintenance and updating of the networking system
  - ❖ change of implementation of layer's service transparent to rest of system
  - ❖ change in one layer doesn't affect rest of system (in theory)

Disadvantages:

- overhead added to the traffic (in terms of packet headers at every layer)
- isolation between layers prevents cross-layer optimization if needed (particularly applicable in the context of mobile networking and the Internet of Things (IoT)
- change in one layer can sometimes affect upper layers unintentionally (think TCP over wireless links)

C. Stateless protocols

Ch2 slide 2-22 (2 in each is enough)
Advantages

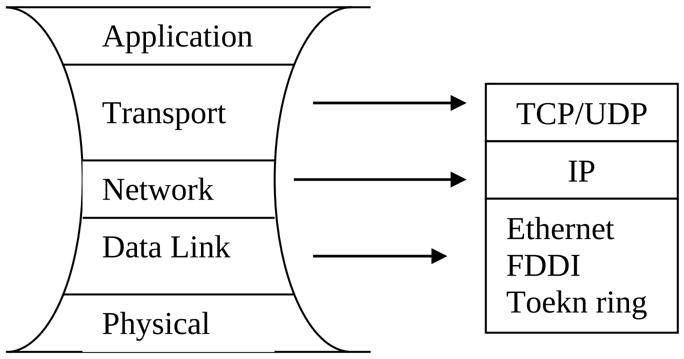
- past history (state) does not need to be maintained (as in stateful protocols)
- it is more resilient to crashes and failures (by contrast, in stateful protocols if server/client crashes, their views of "state" may be inconsistent, must be reconciled)
- It is simpler than stateful protocols (protocols that maintain "state" are complex!)

Disadvantages:

- because there is no requirement to maintain/remember previous state, the setup is needed with every new connection (which may mean re-entering credentials, passwords, etc).
- state can added via 'cookies', which brings up some issues of privacy and security
- since the state is not kept, more overhead is needed to 'carry' the state in the messages since it is not stored.

Q4. A. What is the *hour-glass* Internet model? (use simple drawing to illustrate)

Ch1 slide 1-59



The Internet *hour-glass* model, has a wide top and bottom and a thin waist

B. What does the Internet *thin waist* mean?

It means there has been little change in the protocols in the middle of the protocol stack, particularly at the IP layer. On the other hand, there has been much development and evolution at the applications layer and the physical layer to accommodate new access technologies (such as wireless, optical, etc.) and application models (such as peer-to-peer, CDN-based streaming, among many others).

C. [Extra:] Give two examples of protocols at the *thin waist* of the hourglass? Why are these protocols at the *thin waist*?

Two examples are IP routing protocols (like BGP, RIP, OSPF) , and IP multicast (like PIM-SM, MOSPF).

These protocols provide basic routing functionality for the packet-switched Internet technology, and have not changed much over the years, mainly to allow for other layers to grow and provide more services to the users (in terms of applications or connectivity).

Q5. Can the Internet provide guarantees for message delivery and bandwidth? Provide reasons for your answer [hint: you may compare/contrast your answer with the telephone network]

No, in general the current Internet cannot provide guarantees for delivery and bandwidth, since it does not have reservations or admission control. With the lack of these two mechanisms, the load/traffic can potentially exceed the network capacity, leading to congestion, loss, delays and reduced bandwidth. This also is attributed to the on-demand resource allocation in packet-switch statistical multiplexing (see Q1 for more details).

By contrast, the telephone network performs admission control (to limit demand, and prevent exceeding network capacity), and reservations (in TDM or FDM). Thus, the telephone network can provide guarantees to those flows/calls admitted into the network.

[Extra: Circuit-switched telephone networks, however, cannot provide absolute guarantees to admit users into the network (if admission control decides it cannot handle new calls, then the user is ‘blocked’ [e.g., ‘all circuits are busy, try your call again later’]. With respect to blocking, telephone networks can only provide statistical guarantees, which is achieved through ‘trunking theory’ and design, using models of network access and call duration.]

Q6. A. How were the original Internet requirements met through its design?

Ch1 slide 1-66

- Scalability & economic access:
  - ❖ Resource sharing, reduce reservations, allow for higher utilization
  - ❖ Use of packet switching (statistical multiplexing) instead of circuit switching
- Robustness:
  - ❖ Re-routing around failures
  - ❖ Stateless connections, dynamic routing
- Reliablility:
  - ❖ Timed retransmission, based on acks, seq. #s
- Evolvablility:
  - ❖ Minimize complexity in the network and push functionality to the edges (*end-to-end* principles)

B. What are the two main requirements that you see missing from the original design that are much needed today?

Security and support for seamless mobility

Q7. How does the Internet scale its routing tables?

Ch1 slide 1-43 and 1-72, in addition to in-class discussions

The Internet has a hierarchical structure for routing, around autonomous-systems (AS). The addresses are assigned in a location-specific manner using prefixes in those autonomous systems. Routers need only store prefix information for efficient routing, without knowing individual addresses. This effectively reduces the number of entries (and storage/processing) needed in the routers from billions of entries to thousands. This aggregation is one major way to scale the Internet.

Q8. Give an example in which wireless mobile networks may be more secure than wired networks. [hint: you can use examples given in the class/lecture].

This was also discussed during the lecture, when talking about security towards the end of Ch 1.

This is not the only example, but the 'continuous connectivity' of the wired internet, although usually a very big plus, can also provide vulnerabilities, especially in scenarios of 'spreading of malware' like worms [this relates to a later question about worms, with a video link provided to watch for that "extra" (optional) question].

The example of worms is that they spread very fast in the Internet due to good connectivity. In wireless mobile networks, in which connectivity is intermittent (like delay/disruption tolerant networks [DTN], which we touched upon in class), worms can only spread to 'islands' of networks that are well connected. Due to the delay in connectivity (depending on future mobility) the rest of the network can be secured against the worm/virus to avoid further infection in the future.

Another example has to do with the 'spatial' nature of wireless connections (vs. the relational, location-independent, connection of the wired Internet). In wireless networks, having a peer-to-peer connection means that the other device is 'nearby', and one can use out-of-band information, such as face-to-face meeting, to validate one's identity. This is not possible with the wired connections, where users can impersonate others on social media without verification.

The spatial vs relational nature of wireless vs wired networks was also discussed in class.

Q9. A. What is statistical multiplexing and what is the value it provides over TDM? Give example with numbers to support your argument(s).

Covered in Ch1 slides 1-63 to 1-71, along with a numeric example to compare between the two. It was repeatedly discussed in class as one of the main concepts leading to many implications that we will cover (including congestion control, dynamic routing, reliability, etc.).

Statistical multiplexing (stat mux) assigns capacity ‘on-demand’ and so can utilize the resources to the maximum and can allow a much larger number of users into the network than does TDM. TDM pre-allocates capacity, reserves resources and applies admission control. It can only allow users whose maximum load does not exceed network capacity, whereas stat mux can allow users to potentially exceed the network capacity at times (hoping that the average load is less than the network capacity). Also, stat mux does not need admission control, call setup or fixed routing as does TDM.

As for numbers a network with 1Mbps capacity link, multiplexing users with 100kbps bursts, but active 10% of the time: TDM can only allow 10users, but stat mux can allow many more but with probability of exceeding the network capacity (for 35users, that probability is ~0.0004 (slide 1-71). Other sound examples are also acceptable.

[Extra detailed explanation: This directly relates to the context of packet switching (vs circuit switching) and Time Division Multiplexing (TDM) [as employed in circuit switching] vs statistical multiplexing as employed by packet switching.

For TDM the resources are reserved (pre-allocated) after the admission control of a circuit switched network (or virtual circuit, which we haven't covered in class yet). Then once the time slot is reserved it cannot be shared with other flows or calls. The routing is done at 'call setup' time and is fixed throughout the duration of the flow. There is no packet-by-packet or per-hop routing decisions, and the routes are implicitly determined by the time slot. Failures lead to call termination (tear-down), with a need for another call setup to re-establish the connection.

By contrast statistical multiplexing provides "on-demand" allocation, so there is no pre-allocation or reservation of the resources, no admission control and no call setup per se. You can allow a much larger number of users into the network through statistical sharing of the resources, especially that they're not likely to all be 'on' at the same time, but the data packets are likely to arrive in 'bursts' with enough inter-burst time to admit other flows.]

B. What is the main disadvantage of statistical multiplexing?

With the lack of admission control (which limits the load on the network), statistical multiplexing allows the load on the network to (even momentarily) exceed the network capacity. In those cases, congestion occurs, with the need to queue packets inside the network (i.e., in the switches and routers). Queues can buildup, leading to delays, jitters, and loss. Hence the need for congestion control and traffic management in the network. This also leads to the inability to provide absolute 'guarantees' in the 'best effort' packet-switched Internet.

[Extra: smaller points: Because the routing is no longer implicit in the timing, the routing decisions are done hop-by-hop using IP addresses (in the IP header), and the packets may be routed through different paths. Processing of the packets is more complex in the routers and the headers are larger. Failures along the path are addressed through dynamic routing without the need for connection tear-down or re-establishment. Dynamic routing is more complex than static/fixed routing.]

Q10. A. What is a DDoS attack? And why is it harder to control than a DoS attack?

Distributed denial of service attack, where a large number/network of computers (botnet) can be compromised and used to launch a synchronized attack on a target machine/server. Usually each of the attacking computers is not injecting enough traffic to be detected, but the collective synchronized traffic can be quite harmful to limit the resources at the attacked server from performing its function efficiently (or at all).

DDoS is harder to control than DoS (which usually involves a single attacker), since a single attacker can be detected and prevented from injecting more traffic much more easily than a distributed system of many computers.

Q11. [Extra:] What is the first Internet worm, and how did it harm the Internet? [hint: Watch video link posted on canvas]

The first worm in 1988 was the Morris worm developed by Robert Morris (for research/testing/experimentation reasons, not for malicious reasons). It was an experiment that went out of control unexpectedly. It consisted of code that copied itself from one computer and networked device to another passively (without the need for human intervention). It led to the infection of a very large number of computers connected to the Internet at that time, sometimes preventing those machines from booting up or functioning correctly. [Note: Robert Morris is currently a well-respected researcher and Prof. of CS at the MIT CSAIL lab, working on wireless mobile networking among other projects.]

Q12. A. Why is UDP preferred over TCP for IP-telephony/VoIP (like Skype)?

VoIP is an application that is sensitive to delays and delay jitters (variations), more than it is sensitive to loss (it can tolerate 10% or more loss in many cases and still be usable). UDP provides less delays than TCP, as it does not require connection (handshake), and does not perform reliability (re-transmission) or congestion control (regulation within packet windows). Hence UDP matches VoIP requirements better than TCP.

B. Why would Skype sometimes use TCP? Give two reasons.

Many firewalls block or rate-limit the UDP traffic, for security reasons. Since UDP traffic is not regulated (like TCP's), it can be used in many attacks. In those cases, if the VoIP over UDP is blocked then TCP may be used to get around the firewall restriction.

Skype uses several steps to setup the connection (using client-server model to contact the Skype server, then contacting the super-nodes to conduct search). TCP can be used in those phases.

Q13. Would an application that needs congestion control ever use UDP? Give two examples to support your argument.

Yes.

First, an application that needs congestion control without the need for reliability and re-transmissions, may want to implement congestion control at the application layer over UDP.

Second, an application that needs congestion control that is different from that provided by TCP (which is window-based), may want to implement its own protocol (say rate-adaptation) at the application layer over UDP.

Q14. What are the identifiers needed for process communication across the network? Give example of a connection between two hosts. You can use a simple drawing to aid your answer.

The identifiers include a 4-tuple:

[source IP-address, source port #, destination IP-address, destination port #]

Q15. Someone suggested that 'HTTP' is a stateful protocol. Argue for or against this statement.

Ch2 slide 2-22 , cookies slides 2-28 & 2-29

HTTP is a stateless protocol. It does not maintain state at either end of the connection (client or server). It does not need or rely on any state for the correct operation of the protocol.

It is true that state can added to HTTP using cookies. However, that extra state (using cookies) is only added for user convenience and not for correctness. HTTP can still operate correctly without cookies, so it is considered a stateless protocol.

Q16. How do web caches/proxy servers help Internet performance? Explain its benefits from the user and network perspectives. [hint: explain using your understanding of elementary queueing theory and delays, and use graphs as needed]

Ch2 slides 2-30 to 2-35

- 1. reduce response time for client request (from the user perspective)
- 2. reduce traffic on an institution's access link (from the network perspective)
- 3. Internet dense with caches: enables "poor" content providers to effectively deliver content (so too does P2P file sharing)

Points 1 and 2 above are achieved with the local proxy cache, which (assuming reasonable cache hit ratio) avoids sending most of the traffic over the access link. So instead of having link utilization (or traffic intensity) close to 80-90% leading to exponential queueing delay (perhaps minutes of delay), the goal is to bring the traffic intensity with the help of the proxy cache close to 50-60% resulting in linear queueing delay (close to a few seconds or less in delay).

Q17. What is the effect of customized content, video streaming and encrypted traffic (e.g., HTTPS) on proxy caching?

This was discussed in class and the slides in the context of 'cachable' objects.

Proxy caching takes advantage of similar content being accessed by users in the same network, and the ability to cache such content.

For customized content, it becomes less likely that users in the network will access similar content, rendering the cached content less useful (than cases of regular non-customized content), and the cache hit ratio much less.

For video streaming, the content can be copyrighted and needs authorization for access, and with encrypted traffic the content is not accessible. In both these cases the content cannot be cached due to technical or licensing restrictions. Hence, the proxy cache is less effective in these cases.

Q18. A. When is peer-to-peer (p2p) network architecture needed or preferred?

In peer-to-peer networks it is not necessary to use a server, and so the reliance/dependence on 3rd party is removed. In situations where passing through the 3rd party server creates problems then p2p is preferred, including the following scenarios:

- Server access is considered costly and the cost can be avoided by bypassing the server
- Information exchanged is deemed sensitive/private and should not pass through a 3rd party
- The server has limited resources and cannot scale well with the number of resources or the files served. Peer-to-peer networks provide a more scalable approach.

B. What is the main problem in the p2p design?

Since the client-server architecture is no longer valid for p2p, the 'search' problem becomes the main problem. [Extra: Such problem is solved in the client-server architecture through: 1- DNS mapping from the name to the server IP address, and 2- querying the server for resources and files.]

C. Suggest a solution to the problem in B above.

One suggestion is enough. Three examples are given below.

One solution is to add structure to the p2p network via an 'overlay network' or a distributed hash table (DHT), then use such structure to resolve queries. In Gnutella, the overlay network is used to query peers, then their peers, and so on, using a scoped flooding mechanism, to perform the search.

In tracker-less Bittorrent, the query uses the DHT algorithm to send the query using hashing that maps the resource to node(s) ID(s) that redirect the query to nodes with the resource or those who know where the resource is.

Another approach is to use a server to resolve queries then use the p2p network to exchange information. This approach was used in Napster, and (in the form of tracker) in Bittorrent.

Q19. Use '*traceroute*' and '*ping*' commands/tools to measure and analyze delays in the Internet:

A. Use *traceroute* to measure delays between your location and an overseas location (e.g., www.eurecom.fr). Show the trace and annotate it showing the transoceanic link.
B. Identify machines/routers along the way with:
1. less than1ms delay, 2. 2–10ms delay, 3. 11–100ms delay, more than 100ms delay

then *ping* those machines for 15seconds each and analyze their delays

C. Identify the locations of the machines and reason about the differences in delays [hints: look at the traceroute example in the lecture/book and perform something similar. *traceroute* is called *tracert* on windows. On some machines you need to be super user (sudo) to run traceroute. You may run the commands from your machine or from a UF machine (e.g., storm.cise.ufl.edu), so try and see what works for you.]

Answer:

```
A- storm:16% traceroute www.eurecom.fr
traceroute to www.eurecom.fr (193.55.113.240), 30 hops max, 60 byte packets
 1  _gateway (128.227.205.193)  0.503 ms  0.441 ms  0.481 ms
 2  * * *
 3  ssrb230a-nexus-msfc-1-v21-1.ns.ufl.edu (128.227.236.9)  0.456 ms  0.407 ms  0.314 ms
 4  ssrb230a-pel-asr9001-1-v15-1.ns.ufl.edu (128.227.236.203)  1.173 ms  1.301 ms  1.426 ms
 5  * * *
 6  ssrb230a-ewan-msfc-1-v16-1.ns.ufl.edu (128.227.236.205)  1.054 ms  1.135 ms  1.096 ms
 7  ctx36-pel-msfc-1-te15-1.ns.ufl.edu (128.227.236.175)  1.162 ms  1.819 ms  1.798 ms
 8  renet-flrcore-108-59-26-114.rtr.net.flrnet.org (108.59.26.114)  4.318 ms  4.178 ms  4.111 ms
 9  jax-flrcore-asr9010-1-hu0701-1.net.flrnet.org (108.59.31.150)  6.863 ms  8.081 ms  7.786 ms
10  prov-i2-atla-renet-1070.net.flrnet.org (108.59.25.21)  6.615 ms  6.565 ms  6.308 ms
11  et-3-3-0.4079.rtsw.atla.net.internet2.edu (162.252.70.42)  13.900 ms  13.550 ms  13.493 ms
12  ae-4.4079.rtsw.wash.net.internet2.edu (198.71.45.7)  25.044 ms  25.176 ms  24.953 ms
13  internet2-gw.mx1.lon.uk.geant.net (62.40.124.44)  99.824 ms  99.569 ms  99.501 ms
14  ae6.mx1.lon2.uk.geant.net (62.40.98.37)  100.731 ms  100.586 ms  100.488 ms
15  ae5.mx1.par.fr.geant.net (62.40.98.179)  110.408 ms  107.075 ms  107.223 ms
16  renater-lb1-gw.mx1.par.fr.geant.net (62.40.124.70)  107.144 ms  107.065 ms  107.047 ms
17  te0-6-0-4-lyon1-rtr-001.noc.renater.fr (193.51.177.219)  122.085 ms  193.51.180.61
(193.51.180.61)  124.681 ms te0-3-1-0-lyon1-rtr-001.noc.renater.fr (193.51.177.65)  124.692 ms
18  xe0-0-1-marseille1-rtr-131.noc.renater.fr (193.51.177.17)  119.891 ms xe1-0-1-marseille1-rtr-
131.noc.renater.fr (193.51.177.222)  119.798 ms  119.786 ms
19  te1-2-sophia-rtr-021.noc.renater.fr (193.51.177.21)  121.780 ms  121.727 ms  138.205 ms
20  eurocom-valbonne-gi9-7-sophia-rtr-021.noc.renater.fr (193.51.187.17)  121.580 ms  121.568
ms  121.538 ms
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

in #13 above the delay jumps from ~25ms to ~100ms, going from the US to the UK across a transoceanic link

B- 1. less than1ms delay, 2. 2–10ms delay, 3. 11–100ms delay, more than 100ms delay

#1 above is the gateway (the first hop router) with less than 1ms delay

#3 is a router at uf with less than 1ms delay

#8 to #10 routers are outside ufl.edu but are in the florida net (i.e., in-state) with delays from 2ms-10ms

#11 and 12 are routers outside of Florida but within the US (before crossing the ocean) with delays between 11-99ms delays

routers 13 and beyond are routers in Europe (UK, France) with delays larger than 99ms

ping to #3 above results in:

```
storm:17% ping 128.227.236.9
PING 128.227.236.9 (128.227.236.9) 56(84) bytes of data.
64 bytes from 128.227.236.9: icmp_seq=1 ttl=253 time=0.549 ms
64 bytes from 128.227.236.9: icmp_seq=2 ttl=253 time=0.687 ms
64 bytes from 128.227.236.9: icmp_seq=3 ttl=253 time=0.623 ms
64 bytes from 128.227.236.9: icmp_seq=4 ttl=253 time=0.694 ms
64 bytes from 128.227.236.9: icmp_seq=5 ttl=253 time=46.4 ms
64 bytes from 128.227.236.9: icmp_seq=6 ttl=253 time=0.564 ms
64 bytes from 128.227.236.9: icmp_seq=7 ttl=253 time=320 ms
64 bytes from 128.227.236.9: icmp_seq=8 ttl=253 time=0.592 ms
64 bytes from 128.227.236.9: icmp_seq=9 ttl=253 time=0.592 ms
64 bytes from 128.227.236.9: icmp_seq=10 ttl=253 time=0.601 ms
64 bytes from 128.227.236.9: icmp_seq=11 ttl=253 time=276 ms
64 bytes from 128.227.236.9: icmp_seq=12 ttl=253 time=0.546 ms
64 bytes from 128.227.236.9: icmp_seq=13 ttl=253 time=0.603 ms
64 bytes from 128.227.236.9: icmp_seq=14 ttl=253 time=0.572 ms
64 bytes from 128.227.236.9: icmp_seq=15 ttl=253 time=96.2 ms
64 bytes from 128.227.236.9: icmp_seq=16 ttl=253 time=0.596 ms
^C
--- 128.227.236.9 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15226ms
rtt min/avg/max/mdev = 0.546/46.651/320.319/98.591 ms
```

as can be observed the delay varies widely from 0.55-320ms

ping to # 10 above

```
storm:18% ping 108.59.25.21
PING 108.59.25.21 (108.59.25.21) 56(84) bytes of data.
64 bytes from 108.59.25.21: icmp_seq=1 ttl=56 time=6.86 ms
64 bytes from 108.59.25.21: icmp_seq=2 ttl=56 time=7.83 ms
```

```
64 bytes from 108.59.25.21: icmp_seq=3 ttl=56 time=6.83 ms
64 bytes from 108.59.25.21: icmp_seq=4 ttl=56 time=8.90 ms
64 bytes from 108.59.25.21: icmp_seq=5 ttl=56 time=9.85 ms
64 bytes from 108.59.25.21: icmp_seq=6 ttl=56 time=7.11 ms
64 bytes from 108.59.25.21: icmp_seq=7 ttl=56 time=7.74 ms
64 bytes from 108.59.25.21: icmp_seq=8 ttl=56 time=7.62 ms
64 bytes from 108.59.25.21: icmp_seq=9 ttl=56 time=6.88 ms
64 bytes from 108.59.25.21: icmp_seq=10 ttl=56 time=7.14 ms
64 bytes from 108.59.25.21: icmp_seq=11 ttl=56 time=6.84 ms
64 bytes from 108.59.25.21: icmp_seq=12 ttl=56 time=6.86 ms
64 bytes from 108.59.25.21: icmp_seq=13 ttl=56 time=6.74 ms
64 bytes from 108.59.25.21: icmp_seq=14 ttl=56 time=6.74 ms
64 bytes from 108.59.25.21: icmp_seq=15 ttl=56 time=6.82 ms
64 bytes from 108.59.25.21: icmp_seq=16 ttl=56 time=6.90 ms
^C
--- 108.59.25.21 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15082ms
rtt min/avg/max/mdev = 6.744/7.358/9.857/0.856 ms
```

the delay varies to this router but not as much as to #1 above, with much less deviation

ping to #12 above

```
storm:19% ping 198.71.45.7
PING 198.71.45.7 (198.71.45.7) 56(84) bytes of data.
64 bytes from 198.71.45.7: icmp_seq=1 ttl=54 time=25.6 ms
64 bytes from 198.71.45.7: icmp_seq=2 ttl=54 time=25.4 ms
64 bytes from 198.71.45.7: icmp_seq=3 ttl=54 time=25.6 ms
64 bytes from 198.71.45.7: icmp_seq=4 ttl=54 time=25.6 ms
64 bytes from 198.71.45.7: icmp_seq=5 ttl=54 time=25.8 ms
64 bytes from 198.71.45.7: icmp_seq=6 ttl=54 time=25.4 ms
64 bytes from 198.71.45.7: icmp_seq=7 ttl=54 time=25.4 ms
64 bytes from 198.71.45.7: icmp_seq=8 ttl=54 time=25.5 ms
64 bytes from 198.71.45.7: icmp_seq=9 ttl=54 time=25.6 ms
64 bytes from 198.71.45.7: icmp_seq=10 ttl=54 time=25.5 ms
64 bytes from 198.71.45.7: icmp_seq=11 ttl=54 time=25.5 ms
64 bytes from 198.71.45.7: icmp_seq=12 ttl=54 time=25.3 ms
64 bytes from 198.71.45.7: icmp_seq=13 ttl=54 time=25.7 ms
64 bytes from 198.71.45.7: icmp_seq=14 ttl=54 time=35.3 ms
64 bytes from 198.71.45.7: icmp_seq=15 ttl=54 time=25.6 ms
64 bytes from 198.71.45.7: icmp_seq=16 ttl=54 time=25.5 ms
64 bytes from 198.71.45.7: icmp_seq=17 ttl=54 time=25.8 ms
^C
--- 198.71.45.7 ping statistics ---
```

```
17 packets transmitted, 17 received, 0% packet loss, time 16062ms
rtt min/avg/max/mdev = 25.349/26.171/35.301/2.287 ms
```

the standard deviation is bigger than that of #10 but smaller than that of #3

ping to #20 above:

```
storm:20% ping 193.51.187.17
PING 193.51.187.17 (193.51.187.17) 56(84) bytes of data.
64 bytes from 193.51.187.17: icmp_seq=1 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=2 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=3 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=4 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=5 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=6 ttl=237 time=122 ms
64 bytes from 193.51.187.17: icmp_seq=7 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=8 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=9 ttl=237 time=121 ms
64 bytes from 193.51.187.17: icmp_seq=10 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=11 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=12 ttl=237 time=120 ms
64 bytes from 193.51.187.17: icmp_seq=13 ttl=237 time=121 ms
64 bytes from 193.51.187.17: icmp_seq=14 ttl=237 time=121 ms
64 bytes from 193.51.187.17: icmp_seq=15 ttl=237 time=127 ms
64 bytes from 193.51.187.17: icmp_seq=16 ttl=237 time=128 ms
64 bytes from 193.51.187.17: icmp_seq=17 ttl=237 time=125 ms
64 bytes from 193.51.187.17: icmp_seq=18 ttl=237 time=123 ms
64 bytes from 193.51.187.17: icmp_seq=19 ttl=237 time=123 ms
^C
--- 193.51.187.17 ping statistics ---
19 packets transmitted, 19 received, 0% packet loss, time 18072ms
rtt min/avg/max/mdev = 120.499/122.155/128.948/2.448 ms
```

all the delay is 120ms and above, mostly attributed to propagation delay across the atlantic (and back) which occurs at ~2x10^8m/s (electromagnetic wave speed). Variance is relatively low.

C- It is clear that those machines with min delay (less than 1ms) are either in the local network (gateway) or are inside of the UF campus (ufl.edu domain). Those with slightly higher delays are outside of ufl.edu but part of Florida flrnet. Then delays increase slightly outside of Florida but inside the US when we cross internet2. Finally, delay increases sharply when crossing the Atlantic to the UK then France, with delays exceeding 99ms.

The locations can be easily identified by looking at the router/machine address name, particularly the domain name.

Q20. [Extra:] Visit the wireshark website at wireshark.org, read the user's manual (https://www.wireshark.org/docs/wsug_html_chunked/), then answer these questions:

A. What is wireshark?

Wireshark is a network packet sniffer and analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible.

[Extra, detailed:

Wireshark is a packet sniffing tool for observing the messages exchanged between executing protocol entities.

A packet sniffer captures (obtains copies of) messages being sent/received from/by computers on the local network;

it stores and displays the contents of the various protocol fields in these captured messages. It is a passive tool, as it observes messages being sent and received by applications and protocols running on the computer, but never sends packets itself. Similarly, received packets are never explicitly addressed to wireshark. Instead, a the tool receives a copy of packets that are sent/received from/by application and protocols executing on the machine on which it runs.

The second component of a packet sniffer is the packet analyzer, which displays the contents of all fields within a protocol message.]

B. What are some of intended purposes? (mention four)

Here are some reasons people use Wireshark:

Network administrators use it to troubleshoot network problems

Network security engineers use it to examine security problems

QA (quality assurance) engineers use it to verify network applications

Developers use it to debug protocol implementations

People use it to learn network protocol internals

C. What are two unintended purposes?

[hints: install wireshark and start using it to prepare for future hwks. Read intro posted on canvas.]

Here are some things Wireshark does not provide:

Wireshark isn't an intrusion detection system. It will not warn you when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Wireshark might help you figure out what is really going on.

Wireshark will not manipulate things on the network, it will only "measure" things from it. Wireshark doesn't send packets on the network or do other active things (except domain name resolution, but that can be disabled).