CNT5106C Computer Networks, Fall 2014
Instructor: Prof. Ahmed Helmy
Homework #1
On Internet Architecture and Application Layer

Q1. (10 points) What is the difference between a flat network architecture and a hierarchical network architecture? Discuss pointing out the advantages and disadvantages of each.

Ans. 1. (This is an example answer citing routing. Other answers with similar/reasonable arguments may use other functionality [e.g., name/address mapping/resolution] and that would be acceptable) (see slides 35-45 of Ch1 lecture, mainly slide 42)
In a flat architecture all the nodes are at the same level, and there are no special nodes or border routers (for example) at a higher level. An algorithm running on routers in a flat architecture can produce a shortest path that is optimal (w.r.t. the metric used). A flat architecture responds to dynamics by informing (propagating control messages) throughout the network to route around the failures.

In a hierarchical architecture on the other hand, nodes may have different designations and functionality (e.g., such as the border router), and there are designated boundaries between levels of the hierarchy (e.g., autonomous systems). A node in the lower level of the hierarchy needs to use a node at the higher level to communicate. Dynamics in one part of the network need not be propagated throughout the network.

An advantage of the flat architecture is that it achieves very high quality routing paths (e.g., shortest paths), whereas the hierarchical routing is usually sub-optimal (due to crossing the border routers). It also does not suffer the overhead of hierarchy establishment and maintenance.

An advantage of the hierarchical architecture is that it isolates the dynamics to the part of the network that is failing (or changing); i.e., to the domain with these route dynamics.
It achieves more scalability due to its ability to aggregate routing prefixes per domain.
It achieves flexibility of deployment as every domain may potentially have its own intra-domain routing.

Q2. (10 points: 5+5) What is the Internet's architecture? Discuss from: I- an administrative point of view and II- a routing point of view, pointing out concrete examples of the advantages of such architecture for the Internet.

Ans. 2. The Internet architecture is hierarchical from several points of view: (slides 35-45 Ch1)

I- (slides 35-40 in Chapter 1 lectures) From administrative point of view there are three levels of hierarchy: A. Tier 1 provides the backbone connectivity including large ISPs (e.g., Verizon, AT&T) and large content providers (e.g., Google, Akamai), B. Tier 2 providers include smaller ISPs such as regional providers that interface with customers, and C. Tier 3 includes last hop access networks for customers such as industrial and educational campuses.
The tiered administrative architecture provides flexibility for the different tiers to have their own policies, internal work model and external peering and business

relationship with other peers and children/parents in the hierarchy. This architecture also can add new businesses at different tiers, merge others, or have others leave this architecture; i.e., it supports the evolution of the Internet from the admin point of view. One drawback is that the policies and business relations may not correspond always to good paths and routing becomes suboptimal.

II- (slide 41 in Ch1 lecture) From a routing point of view there are 2 levels of hierarchy: A. The intra-domain routing (such as RIP (distance vector) unicast routing or OSPF (link state) unicast routing), that operates between routers within an autonomous system (AS) and B. The inter-domain routing represented by the border gateway protocol (BGP), that operates between the border routers connecting different autonomous systems.

This architecture supports diverse (and competing) variations of unicast routing within the autonomous systems, while keeping the policies implementation and enforcement at the border routers. Scalability is achieved by aggregating routing prefixes per domain (or AS), hence the entries in the backbone border routers are reduced dramatically (allowing for support of more number of domains, with faster and efficient performance). Also, the routing updates (due to link and router failures, changes and upgrades) occur locally and are not propagated across domains, which provides for better routing stability than a flat routing architecture (which usually becomes unstable with scale).

Q3. (10 points: 5+5) Derive an expression for the calculation of delays in computer-related (telephone and data) networks. Customize such expression to calculate delays in: I- circuit switching and II- packet switching, noting which of the terms are static or dynamic (and under which conditions is it dynamic).

Ans. 3. (slides 63, 66-68, and 80-83 in Ch1 slides)
The general expression for delay in a network (whether circuit or packet switched) is:

Overall Delay $D=d_{setup} + N.d_{nodal}$, where d_{setup} is the setup delay, N is the number of nodes along the path, and d_{nodal} is the delay at each node given by:

$d_{nodal}=d_{trans}+d_{prop}+d_{proc}+d_{queue}$ (as in slide 83 Ch1)

Hence, we get overall delay:

$D=d_{setup} + N.(d_{trans}+d_{prop}+d_{proc}+d_{queue})$

[Note: This assumes identical delay at each node for simplicity. If we do not make this assumption, an even more general expression is: $D = d_{setup} + \sum_{\forall i \in N} d_{nodal\_i}$

I. For circuit switching d_{setup} is significant while $d_{proc}+d_{queue}$ are negligible, and we get: $D=d_{setup} + N.(d_{trans}+d_{prop})$
The setup delay occurs once per connection; only during the pre-call phase. The transmission and propagation delays are generally not highly dynamic.

II. For packet switching d_{setup} becomes negligible, while d_{queue} becomes significant. We also can still neglect d_{proc}, and we get:
$D=N.(d_{trans}+d_{prop}+d_{queue})$
The queuing delay is certainly dynamic and is complex to calculate (generally using the queuing theory principles and techniques). The transmission delays are

generally not highly dynamic for wired networks. For wireless networks, especially with those using rate adaptation (e.g., 802.11) the transmission delay may vary widely. If the path changes then N itself may vary, and the characteristics of the links may also change.

Q4. (8 points) Discuss the following statement, pointing out to what extent is it true or false: We can never guarantee services over computer networks, even in circuit switching or the packet switched Internet, when the demand exceeds the available capacity the network goes into congestion and quality guarantees are lost.

Ans. 4. (slide 60-66, 80-84 Ch 1) This statement is false and inaccurate. The argument needs to be made for circuit switching and packet switching separately. In circuit switching, resources are reserved for every call after an admission control process, using time (or frequency) division multiplexing (TDM or FDM). In that sense, the overall load on the network and the individual loads on the paths are controlled and are designed never to exceed the maximum capacity (by using slot pre-allocation for sources). A new call that would lead to exceeding the capacity of the network is blocked and not allowed into the network (or is delayed until resources become available). So congestion never builds up in such a network and service can be guaranteed, especially given the uniform load that voice calls put on the network.

For packet switching, there is no admission control or resource reservation, as statistical multiplexing is used (rather than TDM). Hence, inherently the load on the network can exceed its capacity, leading to congestion, queue build up, excessive delays and losses. Subsequently, services cannot be guaranteed currently in a packet switched network, such as the Internet.

Q5. (15 points: 12+3) What were some of the main design requirements/principles during the original design of the Internet? (Mention at least four).

If you are to re-design the Internet today, which new design requirements would you include as very high priority (excluding the original requirements) and why. [Include at least two].

Ans. 5. (slide 64 Ch1) The main design requirements and principles during the original design of the Internet include:

1 Scalability & economic access:
  - Resource sharing, reduce reservations, allow for higher utilization
  - This was achieved by the use of packet switching (statistical multiplexing) instead of circuit switching

2 Robustness:
  - Re-routing around failures
  - This was achieved by using stateless connections (using IP) and dynamic routing (using dynamic shortest path techniques for unicast routing).

3 Reliablility:
  - Timed retransmission, based on acks, seq. #s
  - This was established by introducing TCP end-to-end at the transport layer (in addition to hop-by-hop flow control as/if needed at the data-link layer)

4 Evolvable:
  - Minimize complexity in the network and push functionality to the edges

(end-to-end principle)

You can easily notice that 'mobility' and 'security' were not among those in the list above. In today's Internet with the proliferation of mobile technologies (including laptops, tablets, smartphones, PDAs, etc.) and the commercial use of the Internet in almost every aspect of our lives it becomes imperative to design for mobility and security.

Q6. (10 points) Security: Why is it not sufficient to offer security at the lowest layer of the protocol stack (i.e., the physical layer)?

- Give two examples of attacks that cannot be prevented using encryption and secret keys.

Ans. 6. (slides 92-99 Ch 1)
There are many types of attacks and securing the lowest layer of the protocol stack does not address all of them. Security problems and attacks may occur before we reach the lowest layer, e.g., at the operating system or application levels, e.g., viruses, phishing, etc. Also, security implementation comes with high overhead, and depending on the security requirements of an application, some applications may not require security but still have to pay the cost of complexity [remember the physical layer is not aware of the application layer requirements]. Hence, the lower layer security may not be sufficient (nor required in some cases).

Examples include denial of service (DoS) attacks, distributed denial of service (DDoS) attacks and playback attacks. [other answers along similar lines may also be acceptable]

Q7. (12 points) DNS attacks: Discuss four different attacks on the DNS system. For each describe the characteristics or measures of DNS to thwart or ameliorate the severity of the attacks.

Ans. 7. (slides 40-50 Ch 2)
From slide 50

I. DDoS bandwidth-flooding attack of DNS root servers
- Attacker (using botnet) sends packets (ICMP datagrams) to root servers to overload them
- DNS root servers use packet filters that block ICMP messages/pings. Local caches bypass root

II. DDoS of top-level domain servers (e.g., .com)
- Severity of attack mitigated by local caching

III. Man-in-the-middle attack, cache poisoning
- Intercept queries, return bogus replies
- Hard to implement, effectiveness limited

IV. Using DNS to launch DDoS attack
- Trigger many queries using spoofed target address
- Limited effect, responses must be quite large in order to be effective.

Q8. (8 points) Discuss the adequacy of existing transport layer protocols (TCP and UDP) to support today's Internet applications and services, noting how application developers deal with these two options as transport layer. [Support your answer with example applications and their use of the transport layer].

Ans. 8. (slides 16-18 Ch 2) TCP provides a reliable transport (with retransmissions) and congestion/flow control. UDP does not provide reliability mechanisms, congestion or flow control. Hence, an application will have to choose between 100% reliability (with potential for increased delays and jitter [difference in delay]), or 0% reliability support. These two choices may not be adequate for many applications that may require some level of reliability but less than 100%, or those that need congestion control without the reliability (e.g., multimedia video streaming). What many developers do today is that if their application does not fit directly into the TCP model, then they use UDP and implement their own proprietary protocol to provide x% reliability and y% congestion control, etc.
[examples of TCP based apps include FTP, telnet, HTTP/web, email, and for UDP include some Internet telephony or voice over IP (VoIP) apps, and some multimedia streaming apps.]

Q9. (4 points) What are RFCs and why are they important?

Ans. 9. RFCs are ‘request for comments’ and provide the ‘standardization’ documents for the protocols and services of the Internet released by the IETF (Internet Engineering Task Force). They are important because they define the standard over which different vendors and implementers will deploy their equipment in the Internet. There are two aspects to its importance: 1- compliance to the standard, to adhere to safety and regulatory codes, and 2- providing interoperability between different implementations (by different vendors, or different versions of the same vendor).

Q10. (6 points) Compare and contrast the client-server model and the peer-to-peer model as paradigms for connection in the Internet, pointing advantages and disadvantages of each.

Ans. 10. (slides 56-60 Ch 2) Both models provide end-to-end communication at the application layer.
I- The client-server model assumes the existence and availability of powerful servers that are (almost) always on-line, awaiting clients’ requests.
Advantages: 1- The service is somewhat predictable, and 2- the search for content is simple (since the content is usually stored on the server).
Disadvantages: On the other hand, 1- the server provides a single-point-of-failure and 2- single-point-of-congestion and so may not be very scalable (unless replicated and/or distributed).
II- The peer-to-peer model contains only peers (i.e., end systems that would usually serve as clients in the client-server model) and every node serves as both client and server at the same time. Disadvantages: 1- P2P may not provide a very predictable service as the peers join and leave the network (churn), and 2- search for content becomes an issue as it is distributed and no one entity typically has full knowledge of where the content is. New algorithms for search, using unstructured or structured networks (e.g., DHTs) need to be introduced to address this problem. 3- The overlay network may provide suboptimal routes as it may not be aware of the underlying physical topology characteristics.
Advantages: 1- Since all the peers now participate in the uploading and downloading of content,

the network provides a much more scalable solution than the client-server model in terms of content distribution time for popular files. 2- The overlay network avoids the single-point-of-failure, and 3- it avoids the single-point-of-congestion or attack so it is more robust.

Q11. (6 points) What do we mean by the 'thin waist' of the Internet? And why was this term used to describe the Internet? Discuss advantages and/or disadvantages of such phenomenon.

Ans. 11. (slide 57 Ch 1) In the 'hour glass' model of the Internet, the protocol stack is represented using stacked boxes/rectangles, the width of each is a function of the number of protocol innovation and activity at that layer. Hence, the resulting shape becomes wide at the top (the application layer, with many new applications and p2p networks introduced recently), and bottom (the physical and data link layers, where many new technologies for wired and wireless connectivity are introduced). At the same time, the middle part (the waist) at the network layer (IP layer) becomes narrow or 'thin' due to the lack of new protocols and activity recently in this layer. Hence the name 'thin waist'.

Many researchers attribute this 'thin waist' characteristic of the Internet protocol stack with the stability and flexibility/evolvability of the Internet. An advantage is that it keeps the central (routing) architecture of the Internet intact and stable, and provides space to introduce newer services at the upper layers and newer access technologies at the lower layers. The routing semantics and addressing stays the same and the autonomous system (AS borders) and domains still provide address prefix aggregation for scalability. A disadvantage is that evolving the core of the network (i.e., the routing structure) becomes very difficult and some of the services that may need to be implemented in the IP layer will have to be implemented at higher layers with less optimality and efficiency (e.g., resource reservations and priority support, or multi-path routing).

Q12. (6 points) What is the 'end-to-end' principle of Internet design? Discuss advantages and disadvantages of such principle. Has this principle been violated (e.g., through layer violation) since the inception of the Internet? [Provide examples if so].

Ans. 12. (slide 64, 102 Ch 1) The end-to-end principle of Internet design suggests to minimize the complexity inside the network as much as possible and push the functionality as far out to the edges as possible. The network would then be simple and only process layer 3 and below. This allows for more control and functionality at the edges (which are usually easily upgradable PCs and end nodes vs. network routers), which allows for flexibility and evolvability of the network functionality and services (e.g., as we have seen in p2p networks). In some cases it relieves the network from lots of complexity so it can be efficient and fast. The main question is which functions to push to the edges vs. the network. A downside could be that the network itself cannot guarantee services since it is best effort with minimal functionality, and cannot provide new services (e.g., security).

Some examples in which this principle has been violated include cache proxies (slides 32-35 Ch 2), security firewalls, and NAT boxes.

Q13. (4 points) In slide 1-68 (the 68th slide of chapter 1 discussed in class), explain the probability of '0.0004' when 35 users are active.

The system of packet switched (statistically multiplexed) network or link with capacity of 1Mbps, with N sources each with 100kbps and active 10% of the time. The capacity of this system will be exceeded when more than 10 users are active at the same time. If we express the number of users that are active at the same time by a random variable called *x*, and the probability that a source is active is *p*=10%, then we want to get the probability P(*x*>10). Assuming the sources are independent and identically distributed, then we can look at this system as having N=35 experiments, each could have an outcome of success '*on*' with 10% or failure '*off*' with 90%, then we can express the probability of having *x* of these experiments as success '*on*' using the binomial distribution, with

$$P(X=x)=\binom{N}{x}p^x(1-p)^{N-x}, \text{ where } \binom{N}{x}=\frac{N!}{(N-x)!.x!}$$

The probability that more than 10 sources are on at the same time is given by:

$$P(X>10)= \sum_{x=11}^{N} P(X=x)$$

[using a binomial calculator or any other tool/program]
For *p*=10%=0.10, *N*=35, we get *P*(*X* > 10)= 0.0004243.

Q14. (16 points) Napster, Gnutella, Bittorent, and Skype are examples of peer-to-peer file distribution networks. Discuss their architecture and the advantages and disadvantages of such architecture.

Ans. 14. (slides 56-73 Ch 2) Napster: slide 58-59, Gnutella: slide 60-62, Bittorent: slide 68-70, Skype: slide 71-73.

Q15. (6 points) Give an example of a stateless protocol. Can such a protocol be made stateful? How and what would be the purpose?

Ans. 15. (slides 20-31 Ch 2) [with focus on slides 22, 29-31] HTTP is a stateless protocol, as it inherently includes all the information needed in the messages exchanged and does not store information at the server side. It can be turned into a stateful protocol using cookies (where information about the connection and/or user preferences are stored at the server side, and a pointer (or ID) is stored at the client side).
This would help users re-establish connections easily, without having to re-enter authentication, preferences and other information, which can improve the user experience for many on-line services (e.g., shopping, banking, etc.).

Q16. (10 points) Discuss the concept of 'push' vs. 'pull' architecture as relates to p-2-p networks, discussing the advantages and disadvantages of each. Provide another architecture as relates to the data push or pull schemes? What would be the trade-off for such architecture?

Ans. 16. (slides 56-73 Ch 2) Data push occurs when the data is sent from the senders (producers) of the data to other places in the network preemptively (in anticipation of queries and search).

This scheme has a high cost when the data is added to (inserted into) the network, but low cost for the search. An example of this scheme could be Napster where the data (mapping between the node ID and the content) is pushed to a server, then the search only requires a query to the server.
Data pull occurs when the data is not pushed ahead of time, but based on an explicit request from a querier/receiver (consumer) the data is sought and 'pulled' (i.e., transferred on demand). This scheme saves cost when the data is inserted into the network, but has a high cost for the search. Gnutella is one example in which the pull scheme is mainly used.
Another architecture could include a hybrid of both (perhaps based on a hierarchy), where the data is pushed within a certain scope away from the senders/producers, then pulled the rest of the way on-demand to consumers/receivers. It could also be adaptive (so the scope to which the data is pushed/pulled can be made as a function of the data dynamics). The trade-off is between cost of the data push and the cost of the search. For dynamic objects/content that is likely to be inserted many times into the network before it gets queried (e.g., sensed data) the pull mechanism is likely to be more efficient, while for more static content that is likely to be popular (i.e., inserted once and queried many times) then the push mechanism should be more efficient, in general.
[other reasonable answers providing alternative architectures, such as hierarchical, a mix of push/pull using client-server models as in SMTP, with discussion of the pull/push interaction is acceptable]

Q17. (4 points) How many sockets (minimum) would a UDP server need to support 'n' connections? What about a TCP server?

26. Ans. 17. (slides 79-87 Ch 2) (book pg 156-166) With the UDP server, there is no welcoming socket, and all data from different clients enters the server through this one socket. With the TCP server, there is a welcoming socket, and each time a client initiates a connection to the server, a new socket is created. Thus, to support n simultaneous connections, the server would need *n+1* sockets.

Q18. (4 points) What's a DHT and why is it used in p-2-p networks?

Ans. 18. (slide 74-77 Ch 2) DHT stands for distributed hash table and is a structured (algorithmic) way of accessing the data by consistently storing and retrieving it from different nodes using the same process/algorithm using hash functions or tables. It is utilized in p2p networks to add structure to the overlay network, which aims to simplify and add efficiency to the process of searching for content and files in the p2p network.

Q19. (6 points) In the circular DHT example in Chapter 2 (slide 77, or book page 153) if peer 3 learns that peer 5 left the p-2-p network. How does peer 3 update its successor state information? Which peer is now its first successor? Its second successor?

Ans. 19. (slide 74-77 Ch2, book pg 151-156) focus on pg 155-156: Peer 3 learns that peer 5 has just left the system, so Peer 3 asks its first successor (Peer 4) for the identifier of its immediate successor (peer 8). Peer 3 will then make peer 8 its second successor.

Q20. (6 points) In the same DHT if a new peer '6' wants to join the DHT and initially knows peer 15's IP address, what steps are taken?

Ans. 20. (slide 74-77 Ch2, book pg 151-156) focus on pg 155-156: Peer 6 would first send peer 15 a message, saying "what will be peer 6's predecessor and successor?" This message gets forwarded through the DHT until it reaches peer 5, who realizes that it will be 6's predecessor and that its current successor, peer 8, will become 6's successor. Next, peer 5 sends this predecessor and successor information back to 6. Peer 6 can now join the DHT by making peer 8 its successor and by notifying peer 5 that it should change its immediate successor to 6.