George Mavroeidis

COMP 445 M Theory Assignment 2

For Dr. Sandra Cespedes

1. **Explain why network applications only need to be written for the hosts they are running on despite needing to pass data through the network core to provide data communication services.**

Network applications are only written to run on different end-systems and communicate with each other over the network. For example, on a Web Application, two programs are communicating together: 1) the browser program running in the user’s host and 2) the web server program running as a host. Therefore, when writing a program, it has to be written to run on multiple end hosts (desktop, tablet, laptop, etc). A key point to remember is that it is not needed to write software that runs on network-core devices (routers, switches, etc). Even if you wanted to, you wouldn’t be able to do so. Network-core devices function at lower layers, specifically at the network layer and bellow. Hosts are able to only modify/create the Application layer, which is the first one. Thus, programs can only be written at the Application layer and communicate through it.

1. **HTTP provides two connections methods: persistent and non-persistent. While there are clear advantages of persistent-HTTP, it is rather unclear whether non-persistent can be of any use. Find out why these two modes of connections are provided. Specifically, you should find out some of the advantages and disadvantages for each of them, hence justifying their concurrent existence.**

HTTP uses both persistent and non-persistent approaches to transfer files between a server and a client.

Non-persistent connection is a TCP connection that is closed after the server sends the object, therefore, the connection does not persist for other objects. Each TCP connection transports exactly one request message and one response message. The round-trip time (RTT) for non-persistent connections includes packet-propagation delays, packet-queuing delays in intermediate routers and switches, and packet-processing delays (see 1.4). There is a three-way handsake, where there are two RTT’s: one for initiating TCP connection and one for requesting a file. Additional time is added for the transmission time at the server of the file. Non-persistent connections have some disadvantages. Each requested object allocates a brand-new connection, meaning more TCP buffers must be allocated in both client and server. Additionally, delivery delay is a burden, since there are two RTTs.

Persistent connection is a TCP connection that allows for subsequent requests and responses to be sent over the same connection between client and server. Multiple pages residing on the same server can be sent from the server to the same client over a single persistent TCP connection. These requests for objects can be made back-to-back, without waiting for replies to pending requests (pipelining). The default mode of HTTP uses persistent connections with pipelining. Main advantage are lower CPU and memory usage, meaning less buffer sizes on client and server, as well as allowing pipelining of requests/responses, and reducing network congestion. A disadvantage is that resources may be kept occupied even when are not used, meaning they may not be available to other clients and their requests.

1. **Given what we know about transmission delay from Chapter 1 and the following:**

* **A web client and a web server separated by one intermediary router.**
* **Each link carries data at 1Mbps with an MTU of 8192 bytes**
* **Each link is 1000 km long with a propagation speed of 250000 km/s**
* **Zero processing and queueing delays**
* **Connection management packets (packets used for opening a connection, confirming a connection and acknowledging receipt of information) contain no data and all packets have 125 bytes of overhead on top of any data they may contain.**
* **The web client makes an HTTP request to the server that is 250 bytes long.**
* **The requested data is 4 kilobytes in size with 500 bytes of overhead for the HTTP status and headers**

1. **What is the RTT to open the connection?**
2. **How long would it take for a web client to complete the request? You can ignore any time needed to close the connection.**

**A)**

Answer is 0.02 seconds RTT to open connection.

**B)**

In addition to the RTT of opening connection, another round trip has to be made to request and receive the file. We have an **HTTP request** that is 250 bytes long and the **requested data** to be received are 4 kilobytes + 500 bytes of overhead for the HTTP status and headers

Client-To-Server HTTP Request:

Server-To-Client Requested Data:

Total time to complete request:

1. **Given what we know about the request-response nature of the HTTP protocol. Is it possible for a web server to provide data to a web client without the client making a request? How would a web application (such as a social media application) using HTTP be able to provide near real-time content updates?**

A Web cache—also called a proxy server—is a network entity that satisfies HTTP requests on the behalf of an origin Web server. The Web cache has its own disk storage and keeps copies of recently requested objects in this storage. A user’s browser can be configured so that all of the user’s HTTP requests are first directed to the Web cache. Once a browser is configured, each browser request for an object is first directed to the Web cache. Typically, a Web cache is purchased and installed by an ISP. For example, a university might install a cache on its campus network and configure all of the campus browsers to point to the cache. Or a major residential ISP (such as Comcast) might install one or more caches in its network and preconfigure its shipped browsers to point to the installed caches.

Web caching has seen deployment in the Internet for two reasons. First, a Web cache can substantially reduce the response time for a client request, particularly if the bottleneck bandwidth between the client and the origin server is much less than the bottleneck bandwidth between the client and the cache. If there is a high-speed connection between the client and the cache, as there often is, and if the cache has the requested object, then the cache will be able to deliver the object rapidly to the client, basically, turning it into a real-time solution. Second, as we will soon illustrate with an example, Web caches can substantially reduce traffic on an institution’s access link to the Internet. By reducing traffic, the institution (for example, a company or a university) does not have to upgrade bandwidth as quickly, thereby reducing costs. Furthermore, Web caches can substantially reduce Web traffic in the Internet as a whole, thereby improving performance for all applications.

1. **Consider a local network with a 1 Gbps access link to the Internet. Hosts on this network are accessing resources with an average size of 2 MB from a distant server at an average rate of 50 requests per second.**

**A) What is the link utilization of the access link to the internet?**

**B) Suppose these requests were to go through a caching proxy server. What would the link utilization to the internet be if half of all requests were served from the cache of the proxy server?**

**C) Suppose the proxy server could not serve any content from cache (due to the same resources never being needed for example.) Would this still provide any advantages to the end user? Would there be any disadvantages to the end user?**

**A)**

**B)**

**C)**

Proxy servers still play a vital role regardless if caching is not an option that is being used. Proxy servers provide security and protect local users from breachers, ensuring anonymity and internet usage control.

1. **E-mail requires both sender and receiver mail servers to communicate directly. Further, all communications must be made between these two servers using SMTP.**

**A) Is it possible to allow intermediate servers as part of this communications? If so, what are the main advantages of doing so? If no, why is that infeasible, or what are the disadvantages of utilizing it if it was feasible?**

**B) Considering webmail, are there any cases when SMTP between the two mail servers can be replaced by HTTP? Explain why, or why not.**

**A)**

SMTP is the principal application-layer protocol for Internet electronic mail. SMTP has two sides: a client side, which executes on the sender’s mail server, and a server side, which executes on the recipient’s mail server. Basically, SMTP transfers messages from senders’ mail servers to the recipients’ mail servers. SMTP is much older than HTTP. The body (not just the headers) are restricted to simple 7-bit ASCII formats.

It is important to observe that SMTP does not normally use intermediate mail servers for sending mail, even when the two mail servers are located at opposite ends of the world. If the receiver’s mail server is down, the message remains in the sender’s mail server and waits for a new attempt. This can be insecure and it is a process that can easily be manipulated or hacked. Fake spam emails can be sent to the address specified without the receiver giving the confirmation for it, unlike HTTP requests that have a round trip just for that.

**B)**

Both protocols are used to transfer files from one host to another: HTTP transfers files (also called objects) from a Web server to a Web client (typically a browser); SMTP transfers files (that is, e-mail messages) from one mail server to another mail server. In addition, both use persistent connections.

One difference is that HTTP is a pull protocol and SMTP is a push protocol. This means that in HTTP, the TCP connection is initiated by the receiver, while in SMTP, the TCP connection is initiated by the sender. Another difference is that SMTP requires each message (body and headers) to be in 7-bit ASCII format, while HTTP has no such restrictions. A third one is that HTTP encapsulates each object in its own HTTP response message, while SMTP places all of the message’s objects into one message.

1. **SMTP uses 7-bit ASCII for email messages which means that messages can only contain ASCII characters however email messages can contain binary data such as image and among other types of attachments which contain byte values outside the normal range of ASCII. How is this possible? Are there advantages or disadvantages to how this is done? Consider aspects such as protocol overhead and the presence of “legacy” systems in the network in your answer.**

SMTP uses 7-but ASCII for email messages; therefore, any object must be converted into 7-bit ASCII before sending it. It restricts the body (not just the headers) of all mail messages to simple 7-bit ASCII. This restriction made sense in the early 1980s when no one was e-mailing large attachments or large image, audio, or video files and emails were not a common thing across the world. But today, in the multimedia era, the 7-bit ASCII restriction can be limiting. It requires binary multimedia data to be encoded to ASCII before being sent over SMTP; and vice versa after SMTP transport. HTTP does not require such restrictions.

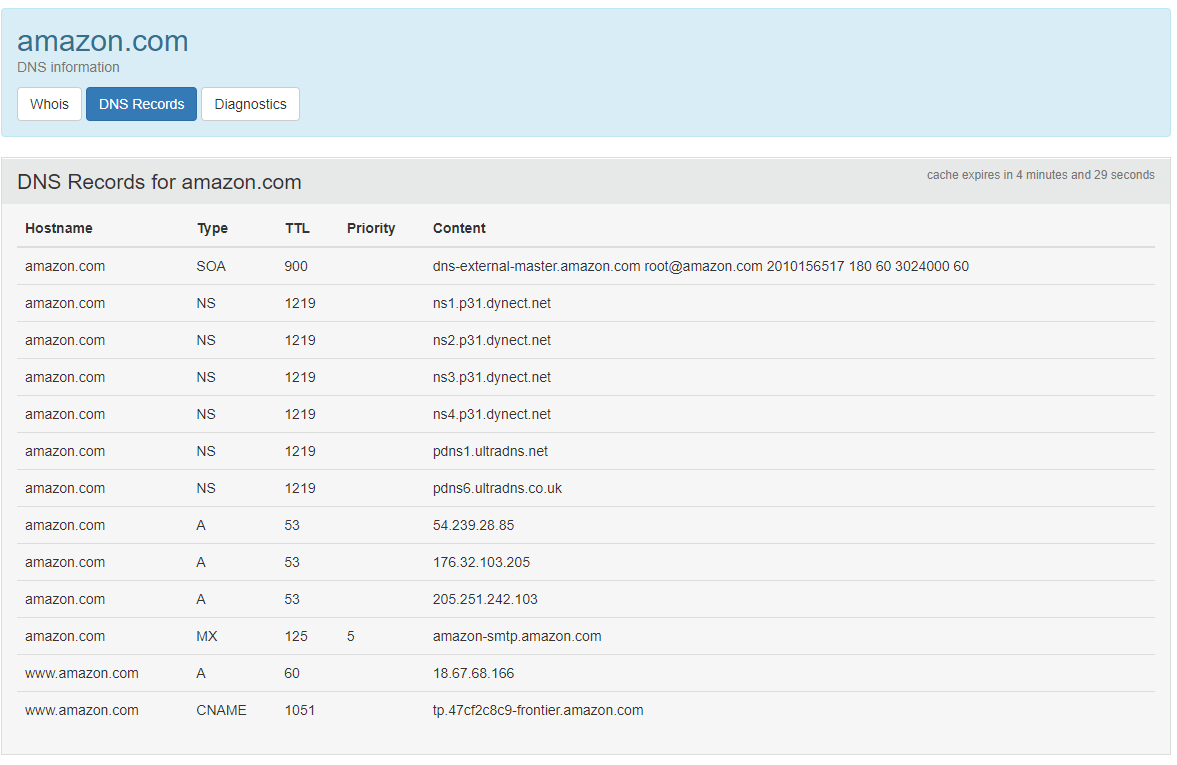
1. **A successful attack to the Internet DNS would be devastating. Explain what type of attacks can be made towards DNS. Why, to-date, such attacks in practice have not been successful? In your answer, you should look at the role caching has and why such technique has not only proven to provide better performance, which is its original goal, but also protection against security attacks.**

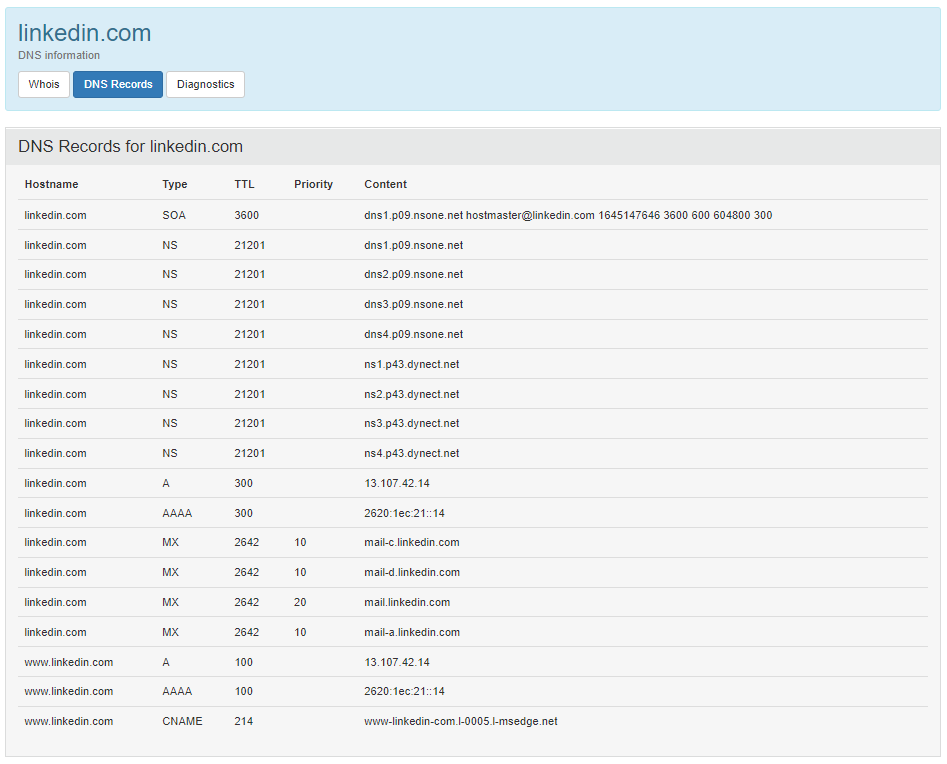
The most common way to attack DNS is by DDoS attacks that flood the DNS root server with a deluge of packets or send a deluge of DNS queries to to-level-domain servers. The severity of such attacks would be partially mitigated by caching in local DNS servers. Also, many of the DNS root servers can be protected by packet filters that block ICMP ping messages directed to them. In the DNS poisoning attack, the attacker sends bogus replies to a DNS server, tricking the server into accepting bogus records into its cache. Either of these attacks could be used, for example, to redirect an unsuspecting Web user to the attacker’s Web site.

1. **Regarding DNS and whois database:**
   1. **What is a whois database?**

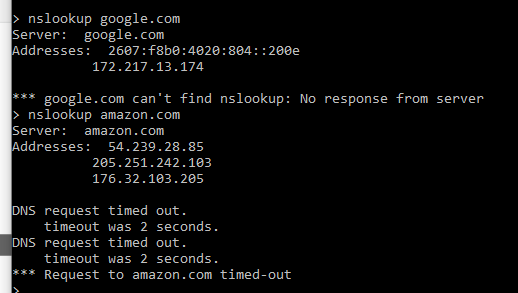
WhoIs domain lookup is a database that store the registered users of a domain name, IP address and other information. By searching it you can discover a number of things about the domain's ownership and registration.

* 1. **Use various whois databases on the Internet to obtain the names of two DNS servers. Indicate which whois databases you used.**

****

****

* 1. **Use nslookup on your local host to send DNS queries to three DNS servers: your local DNS server and the two DNS servers you found in part (b). Try querying for Type A, NS, and MX reports. Summarize your findings.**

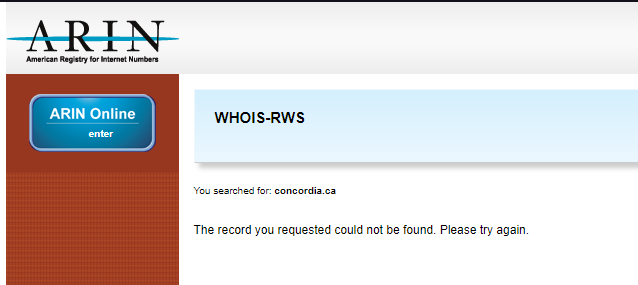
****

* 1. **Use nslookup to find a Web server that has multiple IP addresses. Does the Web server of your institution (school or company) have multiple IP addresses?**

****

No, concordia.ca only has one IP address.

* 1. **Use the ARIN whois database to determine the IP address range used by your university.**

****

* 1. **Describe how an attacker can use whois databases and the nslookup tool to perform reconnaissance on an institution before launching an attack.**

Attackers can use the tools mentioned above (whois and nslookup) to collect the IP address(es) and attack. Secure connections are always required. Concordia uses login info and secure connection as well.

* 1. **Discuss why whois databases should be publicly available**

We need these tools available to the public to identify one another and communicate, data verification and allow the world to communicate together. If not publicly available, finding domain data is hard.

1. **Client-Server architecture can be used for file transfer; however, it is assumed/said to be inferior in comparison to P2P for such operations. Considering a small number of interacting hosts that need to share files, is it true that client-server would perform badly? Explain clearly your answer. You must indicate why the number of interacting hosts/peers is significant in determining whether client-server is suitable for file transfer.**

For client-server architecture, the distribution time increases linearly and without bound as the number of peers increases. In P2P, the minimal distribution time is not only always less than the distribution time of the client-server architecture, but it also reaches a bound that decreases the linear rate with the number of peers increasing. Therefore, applications with the P2P architecture can be self-scaling. This scalability is a direct consequence of peers being redistributors as well as consumers of bits.

1. **BitTorrent uses a trading scheme referred to as tit-for-tat. While some researchers argued deficiencies in that scheme, others argued that if it was not for tit-for-tat, it is likely that BitTorrent would not even exist today. Explain both points of view. In other words, you should indicate the disadvantages of the scheme, as well as why it is still crucial to have such a scheme, or an alternative one, for BitTorrent to exist.**

The BitTorrent ecosystem is wildly successful, with millions of simultaneous peers actively sharing files in hundreds of thousands of torrents. BitTorrent would likely not exist, if it had been designed without tit-for-tat. The majority of users would have been freeriders, meaning no fast file sharing strategy. Main criticism is that mistaking chokes can make cooperation difficult. Sometimes, choking a mistakenly uncooperative user can lead to some imbalances within peers. Thanks to optimal unchoking, this allows for less strict choking and giving a chance to uncooperative peers to reciprocate an objective.

1. **Suppose you are streaming your favorite video content from your favorite video provider. The video is playing at a high quality and then suddenly the quality drops for a few seconds without any noticeable gaps in playback. What could have caused this to happen? How did the video playing software handle this event?**

Dynamic Adaptive Streaming over HTTP (DASH) allows for videos to be streamed to clients in different quality levels to allow on-demand streaming. The video is encoded into several versions, with each one having a different bit rate. The client dynamically requests chunks from a high-rate version and when the bandwidth drops, it selects the lower bandwidth versions. The client selects different chunks one at a time with HTTP GET request messages. DASH also allows a client to adapt to the available bandwidth if the available end-to-end bandwidth changes during the session. This feature is particularly important for mobile users, who typically see their bandwidth availability fluctuate as they move with respect to the base stations.

**END NOTE: The answers have been studied and retrieved from the book or the class slides (check course outline for book reference).**