COMPUTER SYSTEMS – NETWORKS ASSIGNMENT

Name: Matthew Dyson **College:** Van Mildert
Year: 1st – 2008/2009 **User ID**: kklh54
Date: Sunday, 02 May 2010

QUESTION 1

"Client-Server architecture" is a structure that forms a vast amount of networks, whereby a link is established between two nodes on a network in order to transfer data. The client machine sends a request out to the server for information, which the server then provides. Besides this transfer of information, the server will generally not perform any other tasks, whereas the client will do all the processing of information and the actual computing involved. This kind of network architecture is extremely useful in a business environment (1), where information needs to be accessed from many different workstations.

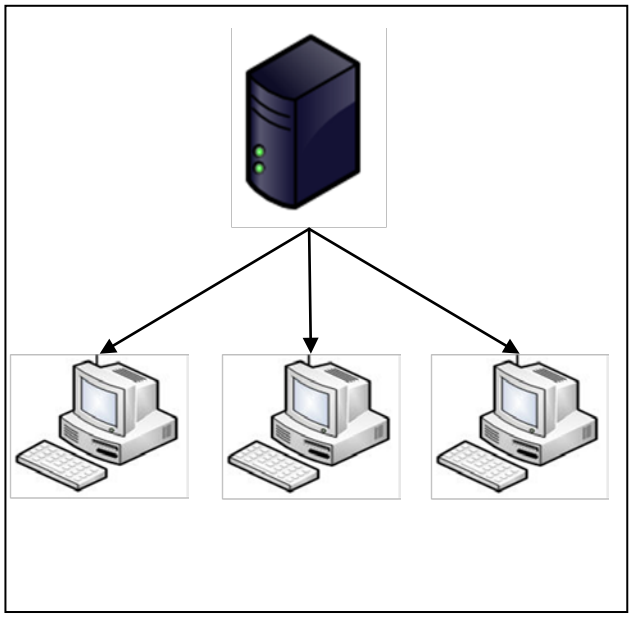

FIGURE 1 - CLIENT-SERVER ARCHITECTURE

What makes this style ideally suited to large scale networks is the data continuity that can be achieved by a central file store. If (for example) a piece of data is required by multiple people on a network, and a copy of this data is stored on each individual workstation, then a user updating said data will not affect the data held on any other machine on the network. However, if this information is stored on a central server then any user on the network can see changes instantly, and there is a hugely reduced risk of data redundancy.

Another important aspect of the "Client-Server architecture" is the easy adaptability of networks using this structure (2). As shown in Fig. 1, many clients can be linked to a single server, and clients can be added and removed without any effect to other clients.

QUESTION 2

Packet switching networks transfer information from source to destination by breaking up the data into small chunks (packets) of data, which contain 'header' information explaining to the receiver what data is contained within the specific packet (3). Although these packets are fairly universal in structure, there are differences between the two methods of packet switching – virtual circuit and datagram.

Virtual circuit networks begin data transmission by determining a static route from the source to the destination, which may pass through any number of intermediate nodes. Once this route has been established, each node has information on where it is receiving information from, and where it is passing this information on to (4). For this reason, the information required within the header of the packet is greatly reduced (compared to datagram, as described below), and hence less packets are required to transfer the same amount of data (assuming static packet size). However, due to the initial setup phase and a disconnect phase at the end of the transmission, the time a virtual circuit is in use is larger than a datagram transmission, which can lead to issues with congestion, causing delay problems.

Datagram networks do not rely on a static route between routers in order to transfer data, instead creating an ad-hoc connection in order to route towards the destination (5). As with virtual circuits, the transmission data is split into packets, although with larger headers containing all the information such as order, destination and total size. This information is then send out en-masse as it is processed, using an algorithm designed to find a router closer to the destination, and then transmitting the data to this intermediate point. Each router will, in turn, inspect the header of the packet it receives, and re-transmits it another step closer, until the packet arrives at the destination. Once all packets arrive at the destination, they are re-assembled according to the information in their headers, and can then be used.

This method of transfer has obvious advantages over virtual circuits, as any network delays or outages can be avoided, however since packets are being sent via multiple routes, the chance of some packet loss is increased. Therefore, methods need to be implemented alongside this protocol to allow for the re-transmission of lost packets to ensure that all data is received.

QUESTION 3

HyperText Transfer Protocol (HTTP) is a pre-defined set of rules used for transferring data across the internet between clients and web servers (6). This works by a client sending commands across a network to a server which holds specific files, and then the server returning appropriate information back to the client. The information returned could be anything from plain text HTML, to image data, to movies or any other file. Web servers work by having a 'daemon' (a program running in the background) which waits for a HTTP request and then handles them as appropriate.

When using a web browser, inputting a URL or clicking a link causes the browser to generate a HTTP request, which could look like the following:

```
GET / HTTP/1.1
Host: www.google.com
```

This request will then be sent via the internet (TCP/IP) to the host – www.google.com. Once the host machine receives the request, the running HTTP daemon will generate a response, which would be similar to:

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.3.7 (Unix)  (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Content-Length: 10254
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head>
...
```

The rest of the HTML for the requested page would then be returned.

The first line of the server response includes a 'HTTP status code' (7) (in this case – '200 OK'). This defines the type of response that the server is giving to the request. For instance, if the server could not find the file requested, the status code would be '404 Not Found'. These status codes are universal to every web client and server, as defined by the Network Working Group in the memo RFC 2616 (8), and handle any eventuality for the outcome of the request from access being denied to the file being moved. This document also defines the use of requests (use of GET and POST for instance, two different methods of transferring information into a request), and the different optional header information that can be returned.

HTTP is extremely necessary in order to standardize the internet, so that every web server and client can intercommunicate and receive the required response. If there were no standard for such requests, then the internet would not be as free and open as it is today.

QUESTION 4

One of the more common connection orientated services used on the internet the Transmission Control Protocol (TCP), which lends itself to the standard for communication on the internet – TCP/IP. Connection orientated services are designed to transfer information in such a way to guarantee delivery as opposed to speed (9), hence being more useful when transporting packets where speed is not as essential as the client receiving accurate data. This is achieved by the recipient passing back a message to the host detailing that the information has been received, which requires more information to be passed along with the packet. This increase in header information means that less data can be transmitted in a single packet, leading to a slower transmission speed, as well as a larger overhead on the connection medium due to the handshaking going on between client and server. Services working in this manner also require the packets to be received in the correct order, which can lead to re-transmission of data which requires even more use of transmission time.

Connectionless services such as UDP (User Datagram Protocol), however, do not provide any medium for the client to send receipt information back to a server. This leads to a smaller header size within each packet (not as much user data required), which means that connectionless services transmit data a lot faster. These protocols are generally used for streaming media over the internet, where guaranteed delivery is not as essential as a quick transmission speed. For this reason, anything sent via connectionless services (videos, for example) generally have a lower quality than those sent via connection based protocols. The speed at which connectionless services transmit data means they are commonly used within denial of service attacks (10).

BIBLIOGRAPHY

1. **White, David.** What is Client-Server architecture? *WiseGeek.* [Online] [Cited: 28 January 2009.] http://www.wisegeek.com/what-is-client-server-architecture.htm.

2. **Exforsys Inc.** Client Server Architecture. [Online] [Cited: 29 January 2009.] http://www.exforsys.com/tutorials/client-server/client-server-architecture.html.

3. **The Linux Information Project.** Packet Switching Definition. [Online] [Cited: 28 January 2009.] http://www.linfo.org/packet_switching.html.

4. **Fairhurst, Gorry.** Datagrams Versus Virtual Circuits. [Online] [Cited: 28 January 2009.] http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/vcanddg.html.

5. —. Datagram Networks. [Online] [Cited: 30 January 2009.] http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/datagrams.html.

6. **Tech Target.** What is HTTP? [Online] [Cited: 29 January 2009.] http://searchwindevelopment.techtarget.com/sDefinition/0,,sid8_gci214004,00.html.

7. **Network Working Group.** Status Code Definitions. *RFC 2616, Section 10.* [Online] [Cited: 29 January 2009.] http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html.

8. —. RFC 2616. [Online] [Cited: 29 January 2009.] http://www.ietf.org/rfc/rfc2616.txt.

9. **Rodriguez, Erik.** TCP vs. UDP. [Online] [Cited: 30 January 2009.] http://www.skullbox.net/tcpudp.php.

10. **Criscuolo, Paul J.** Distributed Denial of Service. [Online] [Cited: 29 January 2009.] http://www.doecirc.energy.gov/ciac/documents/CIAC-2319_Distributed_Denial_of_Service.pdf.