ECE 315 Assignment 4

Arun Woosaree
1514457

April 2, 2020

1

DHCP and BOOTP cannot use TCP, because TCP requires both nodes to have unique IP addresses. When a DHCP client for example first starts up, it does not know which host or hosts it wants to connect to, so it sends a discover broadcast message to all hosts on the network. This broadcasting activity is not supported by TCP, because this activity has a one-to-many relationship as opposed to a one-to-one relationship. The purpose of DHCP and BOOTP is to obtain an IP address. That is, when the process is done, the computer will then have an IP address, meaning the computer does not have an IP address in the middle of the DHCP or BOOTP process.

Both work in a similar fashion, since DHCP is an extension of BOOTP.

First, there is a broadcast message, in which a client asks for an IP address.

Then, the broadcast message is picked up, and the BOOTP/DHCP server responds with information that the client needs such as the client's IP address, subnet mask, default gateway address, the IP address and host name of the BOOTP/DHCP server. This is also broadcasted, since the client does not have an IP address next.

The client then picks up this information, and processes it.

2

Similarities

- From the user perspective there is not a huge difference between the two strategies of displaying dynamic webpages
- both methods run scripts on computers to display dynamic content

Differences

- where the code is run. As the name suggests, client-side scripts run on the user's computer, instead of the server. Server-side scripts run on a remote server, instead of the user's computer.

Server-side scripting

Advantages

1. the user does not have to install plugins (for example, Flash)
2. scripts are hidden from the point of view of the user. They only see the HTML that is outputted. This can help with protecting intellectual property, especially if the code is not obfuscated
3. In general, load times are quicker since the website and its content is already generated/pre-computed so the client does not have to do any additional processing.
4. More control over what code is executed. The client does not have the ability to modify the scripts that are run.

Disadvantages

1. more investment is needed in server compute resources, since scripts are processed on the server side
2. does not scale well with many users. As your website gets more traffic, you need to spend more resources on server hardware compared to client side scripting

Client-side scripting

Advantages

1. more interactivity, since user actions can be processed immediately without requiring a trip to the server and back
2. less network usage, since user actions are processed locally, meaning the server has to send and receive less data
3. less server compute resources required compared to server side, since the processing is done client-side. As the website gains more users, not as much processing resources need to be added (although bandwidth would still be an issue)

Disadvantages

1. Depending on the scripting language chosen, the user's browser may not support it, meaning they would not be able to interact with the website
2. Different browsers may have different implementations of the scripting language, which adds some overhead for the developers, since more cases need to be considered and as a result, more testing to support users with different browsers

3. Load times may be slower since the client must run the script to generate the site and its contents

4. Less control over what gets executed. A malicious client might be able to modify the code and run it in an unexpected way

5. the client's computer may be too slow to run the scripts in a reasonable time, leading to a poor user experience

6. a client who has not refreshed the website in a long time may be running an older script which was previously updated

7. proprietary code can potentially be exposed to users if it is not obfuscated

3

1. lwIP is written in C, which is a highly portable language with broad compiler support for multiple platform targets

2. lwIP is designed such that the software is partitioned between a generic TCP/IP stack core which is buffered with 3 adaptation layers

3. the first adaptation layer deals with the hardware, it is a bunch of drivers which abstracts the hardware to make it appear the same since the same access functions are made available

4. the second adaptation layer deals with varying operating systems. It makes very few assumptions about the operating system to allow for compatibility with a wide range of operating systems

5. the third adaptation layer serves as an API which provides function calls that implement sockets for higher level applications

4

FIFO buffers are used at the sending and receiving ends to make sure that the buffers can hold data if there are short-term mismatches in the rate that the data is transferred. The order in which the data is sent/received is preserved, and the data does not have to be re-transmitted.

In contrast, flow control is used to avoid long-term mismatches in the data rates. The receiver gives feedback to the sender which lets the sender know whether it should slow down or speed up the rate at which it is sending data. It can also tell the sender to stop sending data if it is overwhelmed, and resume sending data.

FIFO buffers and flow control are used together decouple the data rates between two communicating computers.

Without data buffers, and only using flow control, the system would be less tolerant of short-term mismatches. More re-transmissions of data would be required, since small fluctuations in the sending and receiving rate of data means that the sender may not send the data at the correct rate, and the receiver may not be ready to receive data at the correct rate which would result in some incoming data being lost.

Without flow control, if the data rate is too fast, the receiver's buffer could get filled and we'd have a similar issue, where the receiver is not ready to receive data, and the lost data has to be re-transmitted. Alternatively, if the data rate is too slow, the sender's buffer could fill up, and the receiver's buffer would be mostly empty, being underutilized since a higher data rate would be more ideal.

5

When establishing the connection, the client sends its sequence number (a value between 0 and the 32-bit maximum unsigned integer). When the server receives the client's request, it replies with its own sequence number, and an acknowledgement indicating the next segment number it is expecting from the client. The client receives the response from the server, and sends a (pure) acknowledgement indicating the next segment number it is expecting from the server. At this point, the TCP connection is established, and the server is ready to listen for a request from the client. The client is then free to send a request with its current sequence number to the server, and when the server receives the segment, it responds with an acknowledgement indicating the next segment it is expecting to receive. This process is repeated until the connection is closed.

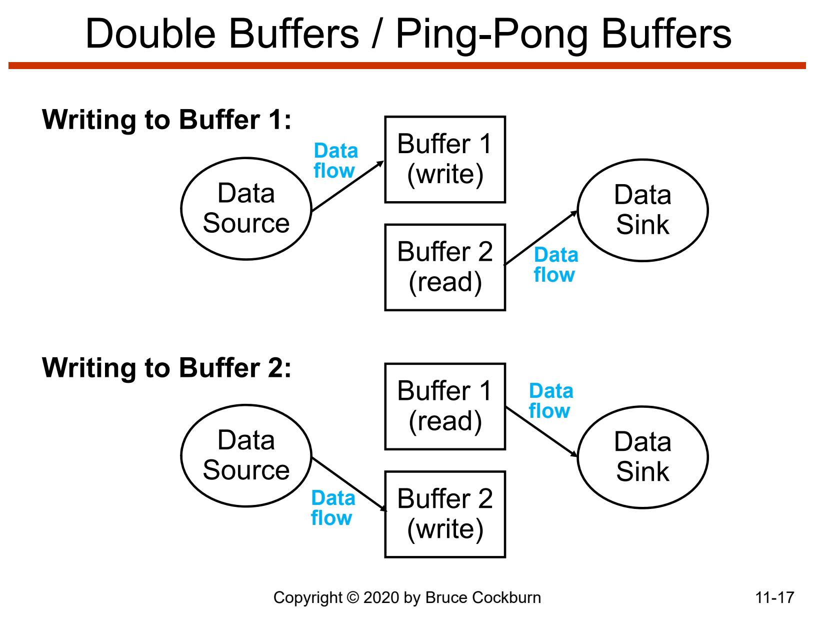
For example, let's say that a client wanted to connect to a server.

1. Client sends SYN with its sequence number. Let's say this number is 100
2. The server then sends a SYN with its own sequence number. Let's say this number is 200
3. The server also sends an acknowledgement to the client, saying it's expecting the next segment to be 101
4. The client receives the server's SYN, and responds with a pure acknowledgement saying it expects the next segment number to be 201.
5. the client sends a segment with segment number 101, and increments its own sequence number

6

A ping-pong buffer is a double buffer that can be used to overlap I/O operations with data processing to speed up a device. The idea is that the first buffer holds some old, but complete data which the reader can access, while the other buffer holds some partial, incomplete but newer data which the writer is writing to.

When the writer is finished, the buffers switch and "ping-pong", so that the reader is now reading from the second buffer, and the writer is writing to the first buffer. This process repeats itself. An illustration from the notes is below:



The advantage of a ping-pong buffer is that the reader and the writer can access data stored in their respective buffers in any arbitrary order, as opposed to a FIFO buffer for example, which requires that the data is read in the same order as it was written in.

For example, if an application was doing matrix operations, the elements of a matrix could be written column-by-column, while the reader could read the matrix row-by-row.

Another example application where a ping-pong buffer may be useful is in streaming video. The data in one buffer can be sent to a graphics card to display the video to the user, while at the same time, another buffer is being written to by the network card, where the video stream is coming from.

7

When the data arrives out of order from the perspective of the receiver, it is stored, and presented to the process in the correct order.

Out-of-order packets are detected by the receiving networking hardware by checking the sequence portion of the TCP header. This is a 32-bit field which comes after first 32 bits. (That is, bits 32-63) in the header. The purpose of the sequence field is store a number which allows the bytes which are sent to be ordered. The starting number is agreed upon when setting up the connection, and subsequent segments have an increasing segment number in their respective header fields.

If a later segment arrives before the one which is expected, the receiver can hold on to it while it waits for the next segment which it is expecting to arrive. Say, for example the receiver was waiting for a segment, and the segment number for argument's sake is 901. Due to different routes taken by the datagrams, segment 903 arrives first instead. The receiver would send an acknowledgement saying it is expecting segment 901. This lets the sender know that a datagram was received, but it was not the right one. (If 3 duplicate acknowledgements arrive, it is assumed the datagram was lost and needs to be re-transmitted). Then, say segment 902 arrives. This is again out of order, and the receiver would send another acknowledgement asking for segment 901. Finally, when segment 901 arrives, the receiver has segments 901, 902, and 903, so it sends an acknowledgement asking for segment 904, which lets the sender know that the previous segments have been received.

8

A circular buffer can be created using the modulo feature of the eDMA, in which the size of the queue is a power of 2. It is limited to a size of 0 bytes (where the buffer is disabled) to maximum of 256 Kbytes. The SMOD and the DMOD field descriptors are used to specify the source and destination buffer sizes, respectively. These are available in bits 15-12 and 11-8 in the DCRn field descriptors. (where n is 0, 1, 2, or 3). The buffer base address will be located on a boundary of the buffer size, and the value of the boundary is based on the initial source/destination address. (Source for SMOD, destination for DMOD These are in the SAR and DAR registers, respectively).