

Net Game Development

by Steffan Benson

FILE	1348_STEFFAN_BENSON_NET_GAME_DEVELOPMENT_178155_2090972 544.DOCX (64.38K)		
TIME SUBMITTED	05-APR-2019 01:09PM (UTC+0100)	WORD COUNT	1058
SUBMISSION ID	104191487	CHARACTER COUNT	7642

Network Multiplayer Games

Assignment Title: Net Game Development

Name: Steffan John Benson

Student Number: P151818

Lecturer Name: Mike Dacey

Submission Deadline: 05/04/2019

Contents

Introduction	3
1. Network protocols	3
1.1. TCP	3
1.2. UDP	3
1.3. Comparison of TCP and UDP	3
2. Multithreading	4
3. Implementation	4
3.1. TCP	4
References	6

Figures

Figure 1 - Server listening for connection.....	4
Figure 2 - Server with message received and printed	5
Figure 3 - Client failing to receive information from the server	5

Introduction

The purpose of this project is to develop a networked web game utilising TCP and UDP protocols, while also utilising multithreading principles. In order to achieve this, a client system and a server system must be developed. The client will handle all rendering/game visuals as well as user input systems. The server will handle all data handling and calculations for both communication and for the game.

1. Network protocols

Network protocols are separated into several distinctive layers. TCP and UDP protocols fall into the 4th later, the transport layer. This layer commonly deals with packet transmission between devices in a network. TCP protocols handle communications between hosts and servers, providing flow control of data transmission. UDP is used for more specific purposes.

Both TCP and UDP will be discussed below.

1.1. TCP[1][2]

Transmission Control Protocol, or TCP, is a set of protocols for use in connection and communication between network devices via the internet or a private network. TCP utilised the client to server communication model, where the server provides a service to the client device. TCP protocols specify the way in which data is exchanged between 2 devices by breaking information into packets, addressing the packets and transmitting them. The protocols are also responsible for the routing of the packets and the collection of the data packets at the destination device. TCP networks are easy to manage and recover quickly from devices/network failures on either end of the network.

TCP has several advantages, chief of which is its modularity. As a non-proprietary system, it is compatible with all device types and operating systems, meaning TCP connections can be established between any device. As well as this, TCP is highly scalable, which allows it to find the most efficient path through any network it handles.

1.2. UDP[3]

UDP, or User Datagram Protocol, is a communications protocol similar to TCP. Its primary use is in low-latency connections between network devices. As opposed to the host-to-host communication system established by TCP, UDP runs a process-to-process form of communication. UDP has no form of flow control implemented in its protocols and is considered a connectionless protocol. A feature of UDP that isn't included in the IP layer is the provision of port numbers to distinguish user requests, as well as the optional capability to check if data was received intact.

1.3. Comparison of TCP and UDP[4]

TCP and UDP are both network protocols which are responsible for data packet allocation and transmission between a client device and server network. TCP is considered to be more robust than UDP, as it is able to automatically detect connection or device failure and resend data packets accordingly. UDP is considered to be more efficient than TCP, as it requires a smaller data overhead and is generally faster when compared to TCP. Both protocols are widely used in common networking systems, often alongside one another for specific purposes.

2. Multithreading[5][6][7]

Multithreading is the process of managing processes by multiple users or multiple processes by a single user. Most commonly, multithreading refers to the process of handling multiple processes simultaneously by a single device. Multithreading eliminates the need for multiple copies of a process to be included in a program, reducing the computational complexity of the program.

In a programme that runs sequentially, if an input is required or a complex calculation is being performed, the following processes would be required to wait until the previous process had been completed. This time delay is removed when multithreading, as it allows interactive programs to continue running, even if the program is waiting for an input or a complex calculation to complete. This provides scalability to the system, as processes in the system can be divided into several smaller processes and performed on different processors. Thanks to modern processor architecture, thousands of threads can be handled at once to reduce the time taken to run an entire programme.

3. Implementation

3.1. TCP

In the TCP implementation, the server and the client are setup individually, however the server must be run first in order for the system to operate correctly. Once the Server is running, it will begin listening for connection from a client.

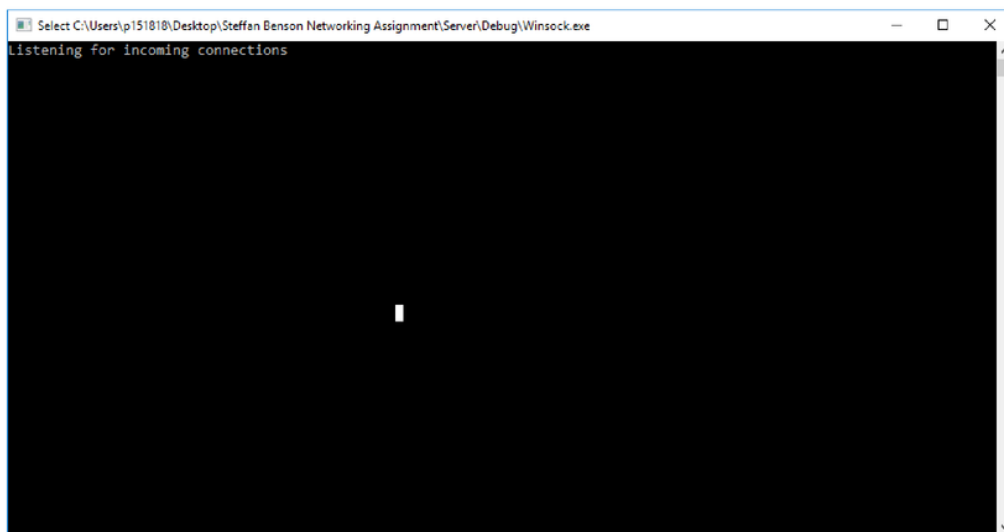
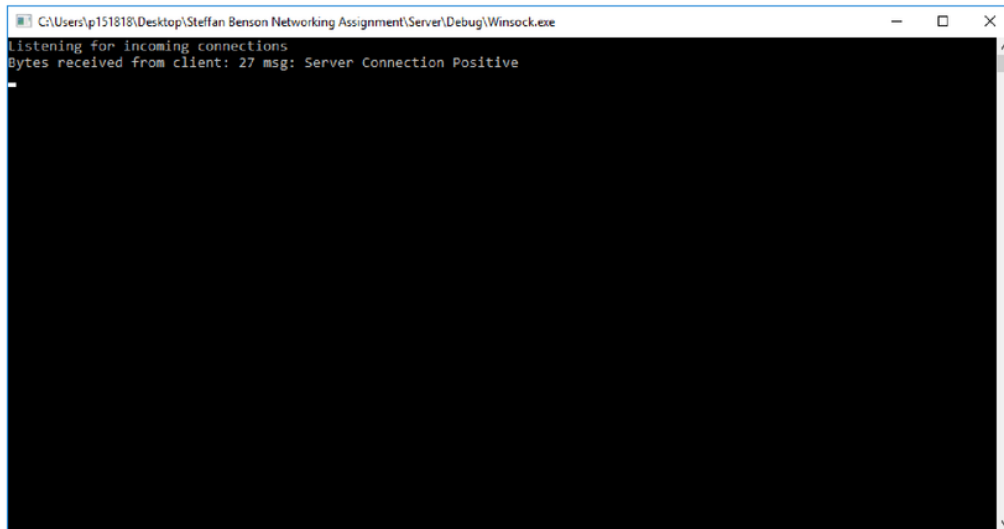


Figure 1 - Server listening for connection

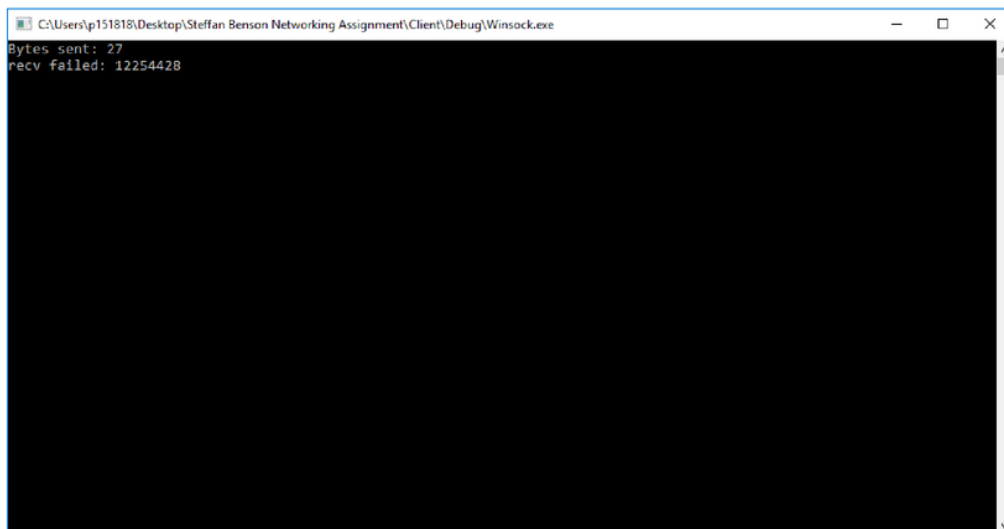
Once the client is running, it will begin searching for a server to connect to. Once the client has established a connection with the server, it will separate a hardcoded message into data packets and transmit them to the server IP, displaying the number of bytes sent in the command prompt. The server will count the number of bytes of data received and print the message in the server command prompt.



```
C:\Users\p151818\Desktop\Steffan Benson Networking Assignment\Server\Debug\Winsock.exe
listening for incoming connections
Bytes received from client: 27 msg: Server Connection Positive
```

Figure 2 - Server with message received and printed

Once the message has been received, the server will attempt to send a response. The message is not sent however, which is made evident by the server not displaying the number of bytes being sent. The client will attempt to receive information from the server but will fail as the connection between the client and the server is closed.



```
C:\Users\p151818\Desktop\Steffan Benson Networking Assignment\Client\Debug\Winsock.exe
Bytes sent: 27
recv failed: 12254428
```

Figure 3 - Client failing to receive information from the server

References

- [1] G. M, Rouse. K, "What is TCP/IP (Transmission Control Protocol/Internet Protocol)? - Definition from WhatIs.com," *TechTarget*, 2019. [Online]. Available: <https://searchnetworking.techtarget.com/definition/TCP-IP>. [Accessed: 05-Apr-2019].
- [2] Unknown, "TCP/IP protocols," *IBM*. [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/ssw_aix_71/com.ibm.aix.networkcomm/tcpip_protocols.htm. [Accessed: 05-Apr-2019].
- [3] M. M, Rouse. G, Lawton. C, "What is UDP (User Datagram Protocol)? - Definition from WhatIs.com," *TechTarget*, 2018. [Online]. Available: <https://searchnetworking.techtarget.com/definition/UDP-User-Datagram-Protocol>. [Accessed: 05-Apr-2019].
- [4] H. C, "What's the Difference Between TCP and UDP?," *How-To-Geek*, 2017. [Online]. Available: <https://www.howtogeek.com/190014/htg-explains-what-is-the-difference-between-tcp-and-udp/>. [Accessed: 05-Apr-2019].
- [5] R. M, "What is multithreading? - Definition from WhatIs.com," *TechTarget*, 2005. [Online]. Available: <https://whatis.techtarget.com/definition/multithreading>. [Accessed: 05-Apr-2019].
- [6] Unknown, "What is Multithreading? - Definition from Techopedia," *Techopedia*. [Online]. Available: <https://www.techopedia.com/definition/24297/multithreading-computer-architecture>. [Accessed: 05-Apr-2019].
- [7] Unknown, "Operating System | Benefits of Multithreading - GeeksforGeeks," *Hotjar*. [Online]. Available: <https://www.geeksforgeeks.org/operating-system-benefits-multithreading/>. [Accessed: 05-Apr-2019].

Net Game Development

GRADEMARK REPORT

FINAL GRADE

25/100

GENERAL COMMENTS

Instructor

UDP Server (30) 5

TCP Server (30) 13

Process Sync (20) 0

Work Independently (20) 7

Report does not provide much analysis or evaluation of the code.

Only a TCP version is provided and this code is very short and basic. No application is provided and the communications between client and server is very limited.

I will give you some marks for discussing UDP.

No multi threading provided.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5
