

# Report

*by* Fathi Sharif

---

|                |                                                           |                 |       |
|----------------|-----------------------------------------------------------|-----------------|-------|
| FILE           | 4105_FATHI_SHARIF_REPORT_349477_1589560837.DOCX (405.62K) |                 |       |
| TIME SUBMITTED | 19-MAY-2021 07:14PM (UTC+0100)                            | WORD COUNT      | 3546  |
| SUBMISSION ID  | 153968988                                                 | CHARACTER COUNT | 18431 |

**Student name:** Sharif Fathi

**Module:** Socket Programming and Concurrency

**Module code:** ACCA6028

## **Task 1 – Report:**

It was asked to write a research-driven essay of 3000 words which focuses on different topics and will go through all topics one by one.

The first topic that will do a research on is Transmission Control Protocol (TCP), before understanding how this technology operates, we need to know what TCP is. Found out that Transmission Control Protocol is a communication protocol allows application programs and devices to communicate by using a network. TCP was created to send packets using the internet and making sure that data was delivered successfully.

TCP is used because this technology is one of rules that defines the internet and the reason that is used TCP is to digitally communicate within network and make sure end-to-end data delivery [1].

When TCP is delivering a data, it takes several actions to guarantee that the data is delivered and here are the following actions that TCP takes.

### Data ordering:

When TCP is delivering a data, it uses sequence numbers to identify data packets and TCP does this so it can rebuild the stream of bytes in the order they were submitted. The receiver also sends a data to let know the publisher the data has arrived, so the publisher can continue to publish and remove buffers for the already received data.

### Error detection:

TCP does error detection and it is done using a checksum field for data. If the data that was sent is corrupted, then there is a high chance that the checksum will fail, and it uses CRC to detect error at layer 2 which will make sure that corrupted data is statistically unlikely.

### Data de-duplication:

Found out that TCP does de-duplication which means when data is send each packet has a sequence, and this is easy for the receiver to easily discard packets that was already been received.

### Resending lost packets:

TCP resends packets that gets lost, and they can get lost by times out or a triplicate ACK is sent.

### Flow and congestion control:

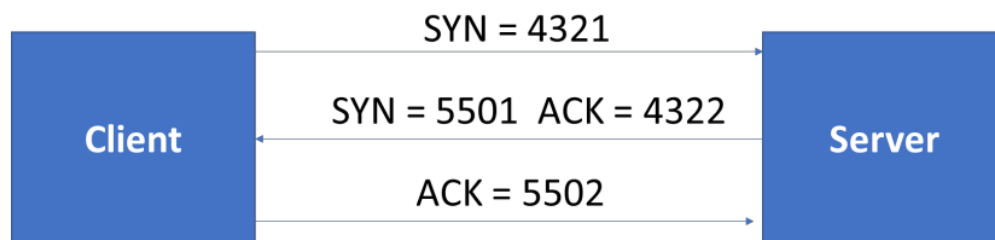
TCP uses something known as flow control which make sures that the sender is not sending data too fast because receiver needs to receive and process the data. This goal is achieved by the receiver indicating how much data the receiver will accept before letting know the sender the data was received, and in this case, it is the responsibility of the sender to wait [2].

### Three-way handshake:

TCP uses a process known as three-way handshake and is process that is used in TCP network to make connection between the server and client. This process has three steps that make sures that the server and client are exchanging synchronization and acknowledgment packets before sending/receiving data.

This process it was designed in a way that allows the server/client to initiate, so this allows you to transfer TCP socket connections between server and client at same time [3].

| Message | Description                                                                                              |
|---------|----------------------------------------------------------------------------------------------------------|
| SYN     | SYN allows you to initiate a connection between server and client.                                       |
| ACK     | ACK allows the other side to confirm that it has received the SYN.                                       |
| SYN-ACK | SYN request from local device and ACK server acknowledges by sending SYN-ACK request back to the client. |



### Two Generals' Problem:

I found out that Two Generals' Problem is an unsolvable problem in computer science that allows two party to communicate each other over an unreliable channel.

The problem is that there are two armies, Alice oversees army A and Bob oversees army B. So, the two armies they have an order of attacking and destroying castle.

- The Castle is located on top of a mountain. The two armies have each set up camp on opposite ends of the mountain.
- It is impossible for both armies to reach the other end of the mountain without crossing the top.
- They both must attack at same time to take the castle and defeat the enemy successful.

Before they attack the castle, Alice will decide the time of the attack and send a messenger to Bob, and Bob will send a response back to Alice camp using a messenger. When they both agree the timing of the attack, they will attack the castle together.

When it comes to computers, they have unsound communication and if we replace the armies with computers, is the same Analogy. TCP depend on something known as 4-way handshake in order terminate a connection [4].

### Maximum transmission unit:

In computer networking, maximum transmission unit is basically a measurement that is representing the size of the largest protocol data that can a network-connected device will accept [5].

After doing a lot of research, found out that MTU size of the network can really have impact on performance and by using a large MTU sizes allows the operating system to send fewer packets of a larger size to reach the same network data. Found out that larger packets usually reduce the processing required in the OS, which will assume that the workload allowing large message to be sent. For example, larger MTU size is not useful, if the workload is only sending small messages.

You could use a largest MTU size if you are using the adapter and network that supports it. For example, if you are using ATM adapters, the default MTU size of 9180 is a lot more efficient than using a MTU size of 1500 bytes [6].

PDU:

PDU stands for protocol data unit, so basically, it is a specific block of information transferred over a network. Found out that PDU is used in reference to the OSI model, because it describes the different types of data that is being transferred from each layer [7].

OSI model stand for Open Systems Interconnection and after doing a lot of research found out that it was created by the International Organization for Standardization which allows communication systems to communicate by using the standard protocol. OSI model it was created, so it can provide a standard for the different computers system can communicate with one another.

OSI model is known as a universal language for computer networking and the way that OSI model works is by splitting up a communication system into 7 layers, and they on top of each other.

1. Physical layer – raw bits one or zero transmitted physically through the hardware.
2. Data link layer – It formats the data on the network.
3. Network layer – This layer is the layer that decides which physical path the data will take.
4. Transport layer – This layer is the layer that is responsible for transmits data by using transmission protocols including TCP and UDP.
5. Session layer – This layer is responsible for maintaining connection and controlling ports ad sessions.
6. Presentation layer – This layer makes sures the data is in a usable format and is where data encryption occurs.
7. Application layer – This layer is the human-computer [8].

### IP fragmentation:

After doing a lot of research, found out that IP fragmentation is an Internet Protocol process and what does is that it breaks packets into pieces and the reason that it does this is by passing through a link with a smaller maximum transmission unit [9].

When someone sends an IP packet through a network, but it cannot be larger than the maximum size supported by the local network. Found out that the size defines the network is data link and IP MTUs which are usually the same. When packets are firstly transmitted through the network supporting one MTU may need to be routed across networks with a smaller MTU.

For example, lets just that the packet size exceeds a lower MTU the data that is in the packet need be fragmented if it is possible. Which means it was broken into smaller pieces, so it can be carried within new packets which equal to or smaller than lower MTU. Found out that this is known as Fragmentation and you should know that the data in these fragments is then usually reassembled when they reach their destination.

Found out that fragmentation allows the transport layer protocols to be unaware of the underlying network architecture and it also reducing overheads.

On the other hand, IP, and the higher layer protocols, they work across variable and diverse network paths and it does this without the need of path discovery protocol.

Just like any other technology, Fragmentation has few disadvantages and think they are the reason the people avoid using it. The first one is that if it loses a single fragment, it will result that all the other fragments have to resent where a reliable transport layer protocol like TCP is in use. The second disadvantages of fragmentation are that it the first fragment it only contains the high layer headers, and this can cause problem with firewalls, middle-box and routers [10].

### Synchronicity in network protocols:

After doing research, found out that synchronous is a data transfer method, which is in a constant stream of data which is signals accompanied by timing signals. You should know that it makes sures the transmitter and the receiver to synchronized with each other and This communication methods are often used when a large data is being moved from one location to another location.

### Asynchronicity in network protocols:

The other hand there is asynchronous which is known as start/stop transmission, and the way that it works is by sending data from the sender to the receiver by using a flow control method. You should know that the asynchronous does not need to use a timing to synchronize data from one location to another location.

This type of method sends 8 bits at each time and the way that it works is by sending character to the start bit, before the transmission process starts. You should know that this method when it sends the character, it sends the stop bit and the character bits the start/stop bits, have total bits of 10 bits.

Let us go through the key differences between these two methods and the synchronous it is a method that transfers data in endless stream of data signals which uses a timing signals to transfer data and this transfer method has sender/receiver which uses a flow control method.

The synchronous is method which allows you to transfer data, but you as user must wait until the data it is fully send before getting a response from the server. Where other hand asynchronous method does not make the user wait until the data is send before getting a response from the server.

You should know the way that synchronous method send data and it does it is by sending the data in form of blocks. The other hand asynchronous method it works by sending the data in the form of character.

Found out that synchronous is method that is faster than asynchronous because asynchronous is a slow method. The synchronous method is expensive, on the other hand asynchronous is cheaper than synchronous.

The way that synchronous transmission works is by separating the clocking lines that is used when distance between the DTE and DCE is short. You should know that synchronous method uses a clocking electrical system when it is sending and receiving phase, and this is makes sures that the communication is synchronized. When there are devices which communicate each other synchronously by using separate clocking channels.

The way that asynchronous works is by using two bits that is known as start bit (0) and stop bit (1). You should know that there will be a time delay between start bit (0) and stop bit (1) and sender/receiver might be function at different clock frequencies.

Will go through the benefits of synchronous method and the synchronous method can transfer a larger data. What so great about synchronous is that it allows you to have a real-time communication between connected devices and it has bytes that it used to transmit without a gap between the next byte, it can reduce the time errors.

Let us go through the advantages of asynchronous and you should know that it has many advantages for example this method is a very flexible method. This method uses asynchronization between the receiver and sender is unnecessary. This method allows you

to send signals from the sources which can have different bit rates, it can also resume when the data byte transmission is available, and it has mode of Transmission which found out it is easy for implementation.

The asynchronous transmission methods have disadvantages and here are main disadvantages. The first one is that asynchronous transmission has bits known as start and stop bits which are required to be used. The asynchronous transmission has timing error which might take place as it is difficult to determine synchronicity, this method uses transmission which is slower rate, and it might create false recognition due to the noise on the channel.

You should know that synchronous Transmission has few disadvantages and one of the main one is that the data that is received, it depends on the receiver's ability to count the received bits correctly [11].

#### I/O models:

After doing research, found out I/O models is method which is used to control the program flow of the code which can deal with the network input and output. You should know that Winsock have few functions which allow you to design an I/O strategy.

The reason that you need I/O model is that it helps you when you are sending or receiving data the operation that have asked for might not be complete immediately, especially these networks that is slow compared to the other local operation. There are ways you can handle this and one of the ways that can handle this is by waiting to try again or you should write a program that waits until the operation is done. You should know that the best choice, it always depends on the structure and requirements of your program.

You should know that originally, Berkeley sockets uses blocking I/O model which means that all the socket functions operate synchronously, it will not return before the operation is completed. This type of behavior is in the windows environment, which mean user input and the output will still be processed even while network operations may arise. To solve this type of problem you could use non-blocking sockets [12].

#### Interoperability for networked applications:

Found out that network interoperability has increased because of how important it is in the telecommunications industry with the new network functions virtualization and software-defined network. These types of technologies made a lot of customers free from vendors locking in by replacing the hardware with an open-source software, which can help you as company to work together separate themselves and deliver a great new service.

Interoperability allows you to communicate two or more networks, application, and components. When it comes to telecommunications industry, interoperability means that the networking of telecom services over multi-vendor, multi-carrier inter-connections. Which means as your company networks keep growing and with new technologies like 5G and IoT coming out now days, it is important that they can interoperate to allow end-to-end communication.

So, the question is how you can achieve network Interoperability and will talk about how you can achieve this goal. Now days devices work in conjunction with each other by adhering to a common protocol standard. You should know that common protocol standards are like a recipe for software engineers and manufacturers, and they take notes, so they can use the necessary ingredients to achieve the goal of interoperability into different systems and products.

Creating these standards as software engineers are not easy process. These standards were created by different organizations and institutions that is always working together, so they can test. The ETSI which is stands for European Telecommunications Standards Institute, for example there is NFV event that provides an outlet telecommunications vendor and open-source community representatives to measure the level of interoperability of their implementations, and make sure they stand by NFV specifications which is now being defined by ETSI.

Network interoperability has benefits because it allows you to achieve the network interoperability, for example users can communicate with many devices by using one terminal. If your company do not want to be locked as customer to one provider, your company can choose equipment from different manufacturers based on the price and performance [13].

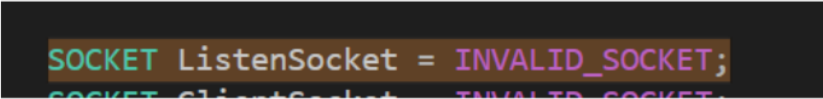
**Task 2 – implementation (50%):**

It was asked to create a network application using C++, using sockets, to build the application of my choosing. I have picked to build chat application and it was used Winsock. This application should be an application which is supporting multiple simultaneous users and it also avoids I/O blocking through multithreading. The application should be incorporate low-level data transport through sockets.

The application will be console application, it will have these features:

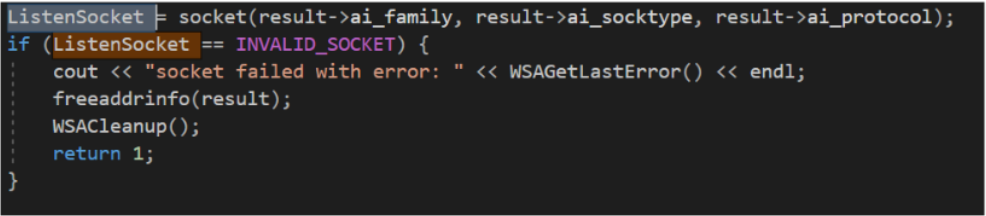
- Join the server.
- Send a message.
- Leave the server.

I have created a socket for the server using Winsock, if look at the code there is socket function which is returns it value to the listenSocket variable. The server application is using first IP address return by the call known as getaddrinfo which matches the address family, socket type and protocol specified in the hints parameter. When the application runs it requests IPv4 address, so the ListenSocket can listen.



```
SOCKET ListenSocket = INVALID_SOCKET;
```

As you can see the screenshot bellow, there is an if statement that checks for errors, so it can make sure that the socket is valid socket.



```
ListenSocket = socket(result->ai_family, result->ai_socktype, result->ai_protocol);
if (ListenSocket == INVALID_SOCKET) {
    cout << "socket failed with error: " << WSAGetLastError() << endl;
    freeaddrinfo(result);
    WSACleanup();
    return 1;
}
```

The server needs binding a socket, the reason that it needs this is because the server must accept the client connections, and it needs to bound to the network address within the system. Bellow there is a code that shows how the bind socket that was created to an IP address and port. The client application is using the IP address and port to connect to the host network. When the bind function gets called, and There is information that was returned by the function that known as getaddrinfo which is no longer needed.

```
iResult = bind(ListenSocket, result->ai_addr, (int)result->ai_addrlen);
if (iResult == SOCKET_ERROR) {
    cout << "bind failed with error: " << WSAGetLastError() << endl;
    freeaddrinfo(result);
    closesocket(ListenSocket);
    WSACleanup();
    return 1;
}

else
{
    cout << "The TCP listening socket" << endl;
}
```

When the socket is bound to an IP address and port on the system, the server will wait for you IP address and port for all the incoming connection requests. If look at the code bellow there is function known as listen function, which has parameters which creates the socket and a value of backlog and the backlog parameter is set to SOMAXCONN and the value constant the Winsock provider for the socket to allows a maximum reasonable number of the pending connections which is in the queue. The if statement checks if the value is returning errors.

```
iResult = listen(ListenSocket, SOMAXCONN);
if (iResult == SOCKET_ERROR) {
    cout << "listen failed with error: " << WSAGetLastError() << endl;
    closesocket(ListenSocket);
    WSACleanup();
    return 1;
}

else
{
    cout << "listening..." << endl;
}
```

After when the socket listens for a connection, the server application needs to handle connection request on that socket.

The socket connection must be accepted, and to accept the connection I have call an object call known as ClientSocket for accepting connection from clients. When it comes to a normal server application that could be used by many clients, the server would be designed to listen for many connections that is coming from many clients. For example, if you want to build high-performance server for a company, the best choice would be to use multiple threads to handle the multiple client connections.

There are different ways of programming techniques that you could use in Winsock to listen for many client connections. The most popular technique that is been used is loop that checks for connection requests using the listen function.

```
// Accept a client socket
ClientSocket = accept(ListenSocket, NULL, NULL);
if (ClientSocket == INVALID_SOCKET) {
    cout << "accept failed with error: " << WSAGetLastError() << endl;
    closesocket(ListenSocket);
    WSACleanup();
    return 1;
}

else
{
    cout << "Accept a client socket" << endl;
}
```

When it comes to creating the client application, first I have declared an addrinfo object which contains a sockaddr structure and initialize values. The client application needs the internet address family which is unspecified so that either an IPv6 or IPv4 address can be returned.

```
struct addrinfo *result = NULL,
    *ptr = NULL,
    hints;
```

```
ZeroMemory(&hints, sizeof(hints));
hints.ai_family = AF_UNSPEC;
hints.ai_socktype = SOCK_STREAM;
hints.ai_protocol = IPPROTO_TCP;
```

There is function known as `getaddrinfo` which requests the IP address for the server name passed on the command line. There is a TCP port which is on the server that the client that will connect to is defined by `DEFAULT_PORT` as 27015 in this sample.

```
// Resolve the server address and port
iResult = getaddrinfo(argv[1], DEFAULT_PORT, &hints, &result);
if (iResult != 0) {
    cout << "getaddrinfo failed with error: " << iResult << endl;
    WSACleanup();
    return 1;
}
```

When it comes to creating a socket for connecting to the server, below there is screenshots.

```
ConnectSocket = socket(ptr->ai_family, ptr->ai_socktype,
    ptr->ai_protocol);
if (ConnectSocket == INVALID_SOCKET) {
    cout << "socket failed with error: " << WSAGetLastError() << endl;
    WSACleanup();
    return 1;
}
```

Below there is a screenshot of if statement that checks if the client is connecting to the server.

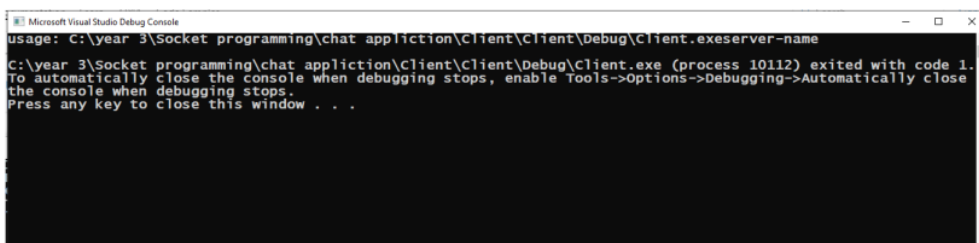
```
// Connect to server.
iResult = connect(ConnectSocket, ptr->ai_addr, (int)ptr->ai_addrlen);
if (iResult == SOCKET_ERROR) {
    closesocket(ConnectSocket);
    ConnectSocket = INVALID_SOCKET;
    continue;
}
break;
```

I have tested the server application, bellow there is a screenshot that shows it works.



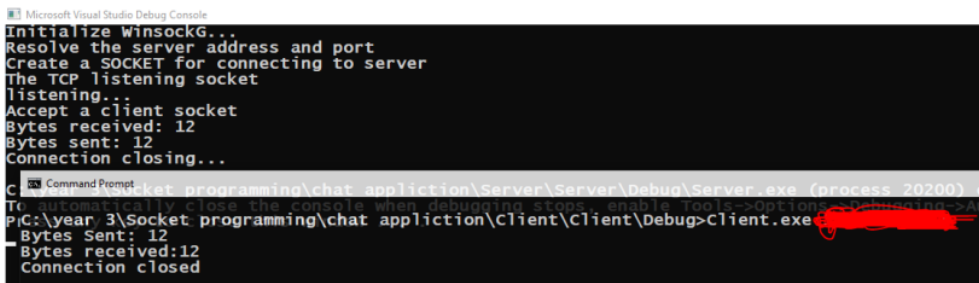
```
C:\year 3\Socket programming\chat application\Server\Server\Debug>Server.exe
Initialize WinsockG...
Resolve the server address and port
Create a SOCKET for connecting to server
The TCP listening socket
listening...
```

I have tested the client application, bellow there is a screenshot that shows it works



```
Microsoft Visual Studio Debug Console
usage: C:\year 3\Socket programming\chat application\Client\Client\Debug\Client.exe server-name
C:\year 3\Socket programming\chat application\Client\Client\Debug\Client.exe (process 10112) exited with code 1.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close
the console when debugging stops.
Press any key to close this window . . .
```

To send data to the server, I have run my command line using my computer IP address, here is the screenshot.



```
Microsoft Visual Studio Debug Console
Initialize WinsockG...
Resolve the server address and port
Create a SOCKET for connecting to server
The TCP listening socket
listening...
Accept a client socket
Bytes received: 12
Bytes sent: 12
Connection closing...

C:\year 3\Socket programming\chat application\Server\Server\Debug>Server.exe (process 20200) e
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Au
pC:\year 3\Socket programming\chat application\Client\Client\Debug>Client.exe [redacted]
Bytes Sent: 12
Bytes received:12
Connection closed
```

## References

- [1] fortinet, [Online]. Available: <https://www.fortinet.com/resources/cyberglossary/tcp-ip>. [Accessed 1 5 2021].
- [2] K. Kilbride-Singh, 15 Oct 2019. [Online]. Available: <https://ably.com/blog/good-enough-not-good-enough-realtime-data>. [Accessed 1 5 2021].
- [3] guru99, [Online]. Available: <https://www.guru99.com/tcp-3-way-handshake.html>. [Accessed 1 5 2021].
- [4] Brian Kim, Aug 13 2018. [Online]. Available: <https://medium.com/@brianjckim/two-generals-problem-5eadb3f395ed>. [Accessed 1 5 2021].
- [5] cloudflare, [Online]. Available: <https://www.cloudflare.com/en-gb/learning/network-layer/what-is-mtu/>. [Accessed 1 5 2021].
- [6] ibm, [Online]. Available: <https://www.ibm.com/docs/en/aix/7.2?topic=tuning-mtu-size-performance-impacts>. [Accessed 1 5 2021].
- [7] techterms, [Online]. Available: <https://techterms.com/definition/pdu>. [Accessed 1 5 2021].
- [8] cloudflare, [Online]. Available: <https://www.cloudflare.com/en-gb/learning/ddos/glossary/open-systems-interconnection-model-osi/>. [Accessed 5 5 2021].
- [9] wikipedia, [Online]. Available: [https://en.wikipedia.org/wiki/IP\\_fragmentation](https://en.wikipedia.org/wiki/IP_fragmentation). [Accessed 5 5 2021].
- [10] Steven Iveson, [Online]. Available: <https://packetpushers.net/ip-fragmentation-in-detail/>. [Accessed 5 5 2021].
- [11] guru99, [Online]. Available: <https://www.guru99.com/difference-between-synchronous-and-asynchronous-transmission.html>. [Accessed 12 5 2021].
- [12] madwizard, [Online]. Available: <https://www.madwizard.org/programming/tutorials/netcpp/5>. [Accessed 12 5 2021].
- [13] N. Cranford, 19 April 2018. [Online]. Available: <https://www.rcrwireless.com/20180419/fundamentals/the-role-of-network-interoperability-in-telecommunications>. [Accessed 12 5 2021].

# Report

## GRADEMARK REPORT

### FINAL GRADE

40/100

### GENERAL COMMENTS

#### Instructor

The report here is rather too dependent on other sources - modest chunks are taken from source.

A basic implementation is included which covers some sample chat-type code.

**Report (100): 55**

**Implementation (100): 25**

**Module average: 40**

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

---

PAGE 13

---

PAGE 14

---

PAGE 15

---