

Research_paper_Leon_Mathias

by Leon Mathias

Submission date: 02-May-2023 03:30PM (UTC+0100)

Submission ID: 204337089

File name: 24756_Leon_Mathias_Research_paper_Leon_Mathias_499749_898363308.docx (212.98K)

Word count: 4008

Character count: 22163

Leon Mathias

2013015

Computing (Software engineering)

Student University of Wales Trinity St David,
Swansea, Wales

L6 - Socket Programming and Concurrency
(ACCA6028)

Date of submission: 02/05/2023

Table of Contents

Introduction	3
Transmission Control Protocol.....	3
Maximum transmission unit	4
Synchronicity and asynchronicity in network protocols.....	5
Interoperability	7
Conclusion.....	8
References	8
Figure 1: TCP three-way handshake [5]	4
Figure 2: MTU fragmentation	5

Introduction

In this paper, the Transmission Control Protocol is to be researched along with its relevant position in the TCP/IP model. How this protocol operates and attempts to guarantee the delivery of data reliably and accurately by using the three-way handshake as well as how it is affected by the Two Generals problem.

The maximum transmission unit shall be explored and the impact it has on transferring data over a network. The way protocol data unit affects the maximum transmission unit as well as its position in the OSI model. Also explored is how fragmentation is used to ensure data is able to be transferred. This paper also aims to explore the use of Synchronicity and asynchronicity in network protocols and where each is used. Additionally how it is used in I/O dispatch models. This paper will explore the concept of interoperability in networked applications and how it can be best achieved through different approaches.

Transmission Control Protocol

Transmission Control Protocol (TCP) is a protocol that is used in the transfer layer of the TCP/IP model and the OSI model. The TCP/IP model which is a networking model that defines how data is transmitted over a network from devices connected to the internet to communicate with one another. The TCP/IP model uses a four-layer procedure to work [1]. The layers are as follows.

- Application layer which is responsible for providing network services to user applications by using different protocols such as The Hypertext Transfer Protocol (HTTP) for web browsing or Simple Mail Transfer Protocol (SMTP) to send emails.
- Transfer layer uses TCP or User Datagram Protocol (UDP) to ensure and provide reliable communication between the original device and its intended destination.
- Internet Layer/Network Layer uses the Internet Protocol (IP) to ensure the data is accurately sent by controlling and routing the traffic.
- Network Access Layer described how data is physically transmitted through the network as well as defining the Mac address to ensure the data is sent to the correct physical machine.

As stated above on the transfer layer as well as using TCP, UDP can be used. One of the main differences between these protocols is UDP does not is less reliable as it is a connectionless protocol which does not check to see if packets are sent and received meaning it continuously sends packets even if they are not being received [2]. Whereas TCP establishes a reliable connection between the devices.

The way TCP provides reliable data transfer is by implementing many different techniques. Each packet of data sent using TCP is acknowledged by the server to ensure the delivery has been received [3] and if the packets of the sender fail to receive an acknowledgement from the receiver, then the data can be retransmitted to ensure no data is missing [4]. But before sending data between devices a connection is made between the devices that the data will be sent by using a three-way handshake in which the client and server send three messages to each other which can be seen in Figure 1. The three-way handshake works by the client sending a Synchronization Sequence Number (SYN) which will contain a sequence number which is a 32-bit random number, an ACK number and an SYN number uses control flags which are 1-bit which I set to 1 when either machine wants to establish a connection or 0 to terminate a connection.

As an example the initial sequence number=1234, the ACK=0 and the SYN=1 which tells the server to open a connection. The second step is the server sends a SYN-ACK with its own sequence number for example 4321, the ACK=1235 which is the client's sequence number +1 so that the client knows

the server has acknowledged the request, the SYN is 1 as it is asking the client to open a connection. The final step is for the client to send an ACK to the server letting the server know the request has been received so the sequence number is the same as the ACK sent from the server which is 1235, the ACK is 4322 which is the sequence number from the server +1 and the SYN is 0 as there is no connection request being sent only an acknowledgement. Now once all the messages have been received the two-way connection is successfully established between the client and server. As well as establishing a connection the TCP protocol also closes the connection. This is done in a similar way as the connections are made but a FIN is sent rather than a SYN.

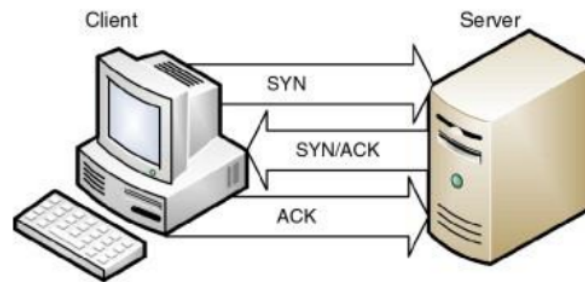


Figure 1: TCP three-way handshake [5]

A problem that TCP faces is that both the client and the server cannot know whether the request has been acknowledged simultaneously therefore synchronization cannot be guaranteed. This is known as the two generals' problem [6], this problem is if computer A wants to communicate across an unsecured network to computer B the only way for computer A to know that computer B got the message is for computer B to send a message to computer A and the only way for computer B to know that the message is delivered is to get a response from computer A and so forth [7]. The issue is data may be lost while being sent therefore there is no way for both computers to know 100% that the message has been received successfully [8]. It is important for a programmer to understand this problem as the applications developed should be designed to handle such failures and potential loss of packets as it leads to more reliable network applications.

Referring to Figure 1 and the example above there is no way for the client to know that the server has received the final ACK after it is sent to the server because the server does not respond. Therefore, without the server receiving the final ACK it will assume that the client did not receive the SYN and the client will open its connection, but the server will not know this.

Maximum transmission unit

The maximum transmission unit (MTU) defines the largest size packet that can be transmitted over the network. Therefore, it has a big impact on the transfer of data over a network. TCP uses MTU is measured in bytes which dictates the maximum amount of data [9].

As explained above the TCP/IP model contains four layers. The OSI model contains seven layers [10], where the application layer of the TCP/IP is defined as three different layers Application, Presentation and Session. It has the same Transport layer where TCP is used as well as the same Network layer. Although the Network Access layer is split into the Data link layer and the Physical layer. Each layer in the model uses Protocol Data Unit (PDU). The size of the PDU in each layer of the OSI model affects the MTU size [11]. This is because PDU defines the different data types that are transferred from each layer of the OSI model as the data changes between the layers. The data is not

altered but is transformed as it is transferred between the layers. The PDU for each layer of the OSI model is listed below [12].

- The Session, Presentation and Application Layers PDU are all referred to as Data.
- Transport layer PDU is referred to as the segment based on TCP which contains a TCP header containing the receivers and the sender's TCP port numbers.
- Network layer PDU is referred to as packet which contains the TCP header as well as the network header containing the IP or logical addresses of the receiver and sender.
- Data link layer PDU is referred to as frame, the frame contains both the TCP header and network header as well as the Medium Access Control (MAC) addresses of the receiver and sender.
- Physical layer PDU is referred to as bit. So, this layer contains bits for the transmission of the data.

As stated above the size of the PDU affects the MTU. If the PDU size is larger than the MTU size of the network protocol, then the data must be fragmented. This means the data is broken into smaller or equally sized new packets that are lower than the MTU. As an example the usual MTU for ethernet frames on the data link layer is 1500 Bytes [13] Therefore if a 3500-byte Ethernet packet arrives it must be fragmented to fit the MTU of 1500 see Figure 2. It will be fragmented into two 1500-byte packets and one 500. Any data sent over the network will have to match the lowest MTU if one of the routers on the network has a lower MTU than any of the others [14]. If the packet is lower than the MTU then it will be sent in one packet and will not use fragmentation.

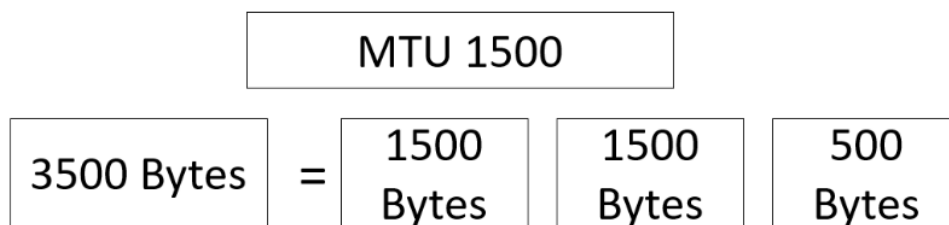


Figure 2: MTU fragmentation

It's important to ensure that the MTU size is set appropriately depending on the application requirements and specific network protocol to attempt to minimize fragmentation [15]. therefore, MTU size should be adjusted to optimize requirements. The data can be transferred faster and reduce the delay in the network if the MTU is smaller. Whereas having a larger MTU can take longer to be transmitted in a single connection although it does reduce overhead [16] as the packets do not need to be reassembled because protocol headers and other control information are contained in each smaller fragmented packet [17]. This is important to take into account while designing and developing network applications as by taking into account the MTU it is possible to adjust the size of the data to limit the amount of fragmentation to ensure better performance by the application.

Synchronicity and asynchronicity in network protocols

In terms of network protocols synchronicity and asynchronicity are two different approaches to communication. The main difference between these modes is the way that data is transferred, which provides advantages and disadvantages depending on the application [18].

Synchronous communication in terms of network protocols is where the sender and the receiver exchange data in real-time simultaneously [19]. In synchronous communication, the sender will wait for a response and only continue with the next exchange of data once a response is provided by the receiver. An advantage of synchronicity in network protocols is to ensure that the correct data is being sent and received. Also, it is simple to determine if and when the data transfer is complete this allows for any errors to be fixed. Because a response is needed to continue this causes one of the main disadvantages of synchronous communication in network protocols is that it can reduce performance, cause delays and slow down the overall data transfer process if the response takes too long.

The network protocol Hypertext Transfer Protocol (HTTP) commonly uses synchronous communication as once a request is made the sender cannot continue until a response is received [20]. Although this can be a process that can take milliseconds depending on latency the sender still cannot continue which is an issue when an HTTP request is blocked or fails.

Applications that may use synchronous communication are real-time video or audio transmission such as live streaming. Online gaming also uses synchronous communication as players must react to other players' actions in real-time, this is because all players must be synchronized.

Unlike synchronous communication, using asynchronous communication the sender does not need to wait for a response from the receiver in order to continue with the next exchange of data [18]. The sender and receiver do not need to be synchronized to exchange data. Therefore, it will continue to exchange data without getting confirmation it is being received. This can lead to issues as it is difficult to ensure that the correct data is being sent and received [21]. Meaning errors cannot be easily corrected [22]. On the other hand because asynchronous does not wait for a response to continue this can improve efficiency and performance as it can continue to send and receive requests simultaneously. Also, if there is a long response time it does not have such a big impact on the service in comparison to synchronous communication.

Two network protocols that use asynchronous communication are Simple Mail Transfer Protocol (SMTP) and File Transfer Protocol (FTP) which both are TCP/IP protocols. SMTP is most commonly used by email clients in order to send and receive mail. FTP is a protocol that is used to transfer files from a server to a client over a network. Asynchronicity is used in these protocols because there is no need for an immediate response from the receiver. Web servers must also be asynchronous as clients can connect and disconnect at anytime therefore, they should not wait for a response from every client.

In computer programming, input/output (I/O) dispatching is the process of handling input and output operations [23]. Synchronous and asynchronous are two approaches to I/O dispatching. The program uses a call-back function in asynchronous I/O dispatching which allows the program to continue executing [24] other tasks and then call-back is executed once that operation is completed. For example, if a large file is downloaded from the internet, the program will start the download and complete other tasks, and once the download is complete it will tell the program. Whereas synchronous I/O dispatching the operation must finish before anything else is allowed to be executed. This can lead to long delays to the application as the program is blocked until the I/O operation is complete.

As stated above using synchronous or asynchronous programming can have an impact on the performance of the application or system. Therefore, as a programmer, it is important to decide whether to use synchronous or asynchronous programming depending on the requirements of the

system or application that is being developed and by taking into consideration the advantages and disadvantages of each approach.

Interoperability

Interoperability is the ability of different network systems and applications to be able to communicate and connect with each other efficiently without any need for the end user to do anything or have any specific knowledge [25]. This is important to allow the systems to communicate and work together without compatibility issues no matter technology, platform, or vendor used. Failing to achieve interoperability can lead to a divide in technology where different systems or vendors cannot communicate with each other. It also could cause people to be forced into using a certain technology or eliminate the ability to communicate with other systems [26].

Because there are many different devices, applications and systems that are used to exchange data it can be difficult to achieve interoperability across many different networks with different users, but this is what makes it important.

One way to help achieve interoperability is to use standardization of protocols and data formats that have been agreed upon by the different systems and applications. Using the more popular standards is even better to ensure that many different systems can communicate together without any trouble. By using different standards compatibility issues can be had making it difficult to achieve interoperability. The use of open standards can help ensure interoperability as these are publicly available and can be used by anyone [27]. Examples of open standards that can be used in networked applications are HTTP which makes building and deploying web-based services and applications easier as it provides a standardized way for different web applications to interact with one another [28]. Also, JavaScript Object Notation (JSON) is an open standard data format that is easy to write and read by both humans and computers making it widely used. JSON can also be understood by many programming languages meaning it is very popular. TCP/IP can also be used as a communication protocol because it is implemented by many vendors and on most machines [29].

Another way to help ensure interoperability is to have a well-designed Application Programming Interface (API) in order to allow software systems to interact and also allow developers to build applications that can communicate and interact with other systems more easily. The API should be standardized and kept consistent to attempt to achieve interoperability. In order for the APIs to be used by more developers which is important for interoperability, they should be easy to use and understand but also include clear documentation [30]. One of the benefits of using APIs is that they can be developed using different programming languages [31]. If two developers were using different languages as long as they both use the same API they can exchange data between their applications because APIs use a standardized interface.

To achieve interoperability of different systems and applications to ensure that they can work together as intended, compatibility testing should be done [32]. The main way this testing should be done is to test across a range of platforms and environments to ensure that the system can work with a variety of different software and hardware configurations [33]. Also, perform the tests to ensure that the system can handle different types of data inputs and outputs by using different data formats and protocols. Another problem that can lead to failure of interoperability is working with different versions of software, therefore it is important to test using different software versions.

Conclusion

To conclude, firstly TCP is an important protocol to ensure data is transferred correctly and accurately. Although the two general problems can cause issues the three-way handshake attempts to ensure the safe delivery of data by acknowledging when data is received and sent.

The MTU and PDU are important to consider while designing and developing network applications it is also good to consider the effects IP fragmentation can have.

There are many different benefits and drawbacks to both synchronicity and asynchronicity in network protocols therefore it is important to take into consideration these benefits and drawbacks while developing networked applications.

Although interoperability can be difficult to achieve it is an important factor to consider while developing networked applications and there are many ways to help achieve it.

References

- [1] K. R. Fall and W. R. Stevens, *TCP/IP illustrated, volume 1: The protocols*. addison-Wesley, 2011.
- [2] A. Mehdizadeh, M. Nagarajan, H. Harun, and M. Mohammadpoor, "Congestion Window Scaling Method to Optimize Delay in TCP/IP," *Wirel Pers Commun*, vol. 101, no. 4, pp. 2227–2239, 2018, doi: 10.1007/s11277-018-5813-3.
- [3] K.-C. Leung, V. O. K. Li, and D. Yang, "An Overview of Packet Reordering in Transmission Control Protocol (TCP): Problems, Solutions, and Challenges," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, 2007.
- [4] J. R. Sella Veluswami, K. Chinnusamy, K. Kumar, V.-C. Klinge, and S. Mohankumar, "Improvement of Transmission Control Protocol for High Bandwidth Applications," *Wirel Pers Commun*, vol. 117, no. 4, pp. 3359–3379, 2021, doi: 10.1007/s11277-021-08074-2.
- [5] E. Conrad, S. Misener, and J. Feldman, "Domain 4: Communication and Network Security (Designing and Protecting Network Security)," *CISSP Study Guide*, pp. 219–291, Jan. 2016, doi: 10.1016/B978-0-12-802437-9.00005-9.
- [6] E. Godard and E. Perdereau, "Back to the Coordinated Attack Problem: MSCS," *Mathematical Structures in Computer Science*, vol. 30, no. 10, pp. 1089–1113, Nov. 2020, doi: <https://doi.org/10.1017/S0960129521000037>.
- [7] V. Babaylov and G. Timofeeva, "Calculation of the optimal timeout of executing a SQL query for multi-master replication system," in *AIP Conference Proceedings*, Mar. 2022, p. 60001. doi: 10.1063/5.0100779.
- [8] T. Fevat and E. Godard, "Minimal Obstructions for the Coordinated Attack Problem and Beyond," in *ipdps*, IEEE, 2011, pp. 1001–1011. doi: 10.1109/IPDPS.2011.96.
- [9] R. R. Subbaraman, "Rate limitable and efficient discovery of path maximum transmission units," *International Journal of Communication Systems*, vol. 32, no. 6, p. e3905, Apr. 2019, doi: <https://doi.org/10.1002/dac.3905>.
- [10] S. Kumar, S. Dalal, and V. Dixit, "THE OSI MODEL: OVERVIEW ON THE SEVEN LAYERS OF COMPUTER NETWORKS," *International Journal of Computer Science and Information Technology Research*, vol. 2, pp. 461–466, Aug. 2014, Accessed: Mar. 20, 2023. [Online]. Available: www.researchpublish.com

- [11] Peter Loshin, "What is a protocol data unit, and how does it work?," *TechTarget*, 2022. <https://www.techtarget.com/searchnetworking/definition/protocol-data-unit-PDU> (accessed Mar. 20, 2023).
- [12] M. E. Ilchenko, A. V Moshinskaya, and L. A. Urywsky, "Levels separation and merging in the OSI reference model for information-telecommunication systems," *Cybern Syst Anal*, vol. 47, no. 4, pp. 598–605, Jul. 2011, doi: <https://doi.org/10.1007/s10559-011-9340-4>.
- [13] M. Ashjaei, M. Behnam, L. Almeida, and T. Nolte, "MTU configuration for real-time switched Ethernet networks," *Journal of Systems Architecture*, vol. 70, pp. 15–25, 2016, doi: <https://doi.org/10.1016/j.sysarc.2016.04.009>.
- [14] S. Hogg, "MTU size issues, fragmentation, and jumbo frames: The maximum transmission unit (MTU) for a protocol sets the top limit on how big its packets can be, but the network devices they pass through, encapsulation and other factors can complicate things.," *Network World (Online)*, Oct. 2021, [Online]. Available: <https://www.proquest.com/trade-journals/mtu-size-issues-fragmentation-jumbo-frames/docview/2584046643/se-2?accountid=130472>
- [15] I. Hussain and J. Bashir, "Dynamic MTU: A smaller path MTU size technique to reduce packet drops in IPv6," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 9, pp. 7070–7088, Oct. 2022, doi: [10.1016/j.jksuci.2021.06.011](https://doi.org/10.1016/j.jksuci.2021.06.011).
- [16] K. Nattapong, O. Eiji, K. Takashi, and U. Shigeo, "Single tag scheme for segment routing in software-defined network," *Telecommun Syst*, vol. 74, no. 2, pp. 173–184, Jun. 2020, doi: <https://doi.org/10.1007/s11235-019-00645-w>.
- [17] N. Sharma, J. Kaur, and J. Yadav, "IMPLEMENTATION AND ANALYSIS OF FRAGMENTATION OF MPEG-4 TRAFFIC IN NS-2 INTEGRATED WITH EVALVID," *Oeconomics of Knowledge*, vol. 5, no. 2, p. 7, 2013, [Online]. Available: <https://www.proquest.com/scholarly-journals/implementation-analysis-fragmentation-mpeg-4/docview/1346916397/se-2?accountid=130472>
- [18] Priyank Gupta, "Synchronous vs. asynchronous communications: The differences | TechTarget," Aug. 18, 2021. <https://www.techtarget.com/searcharchitecture/tip/Synchronous-vs-asynchronous-communication-The-differences> (accessed Mar. 29, 2023).
- [19] J.-P. Vasseur and A. Dunkels, "Chapter 11 - Smart Object Hardware and Software," in *Interconnecting Smart Objects with IP*, J.-P. Vasseur and A. Dunkels, Eds., Boston: Morgan Kaufmann, 2010, pp. 119–145. doi: <https://doi.org/10.1016/B978-0-12-375165-2.00011-9>.
- [20] "Communication in a microservice architecture | Microsoft Learn." <https://learn.microsoft.com/en-us/dotnet/architecture/microservices/architect-microservice-container-applications/communication-in-microservice-architecture> (accessed Mar. 29, 2023).
- [21] X. Fafoutis, A. Di Mauro, M. D. Vithanage, and N. Dragoni, "Receiver-initiated medium access control protocols for wireless sensor networks," *Computer Networks*, vol. 76, pp. 55–74, 2015, doi: <https://doi.org/10.1016/j.comnet.2014.11.002>.
- [22] A. Damian, C. Drăgoi, A. Militaru, and J. Widder, "Communication-Closed Asynchronous Protocols," in *Computer Aided Verification*, I. Dillig and S. Tasiran, Eds., Cham: Springer International Publishing, 2019, pp. 344–363.

- [23] "Synchronous and Asynchronous I/O - Win32 apps | Microsoft Learn," Mar. 20, 2021. <https://learn.microsoft.com/en-us/windows/win32/fileio/synchronous-and-asynchronous-i-o> (accessed Mar. 29, 2023).
- [24] "Understanding the Windows I/O System | Microsoft Press Store," Sep. 15, 2012. <https://www.microsoftpressstore.com/articles/article.aspx?p=2201309&seqNum=3> (accessed Mar. 29, 2023).
- [25] M. L. Zeng, "Interoperability," *Knowledge Organization : KO*, vol. 46, no. 2, pp. 122–122–146, 2019, doi: <https://doi.org/10.5771/0943-7444-2019-2-122>.
- [26] A. Park, M. Wilson, K. Robson, D. Demetis, and J. Kietzmann, "Interoperability: Our exciting and terrifying Web3 future," *Bus Horiz*, 2022, doi: <https://doi.org/10.1016/j.bushor.2022.10.005>.
- [27] "Open Standards principles - GOV.UK." <https://www.gov.uk/government/publications/open-standards-principles/open-standards-principles> (accessed Mar. 29, 2023).
- [28] T. Katayama *et al.*, "The DBCLS BioHackathon: standardization and interoperability for bioinformatics web services and workflows," *J Biomed Semantics*, vol. 1, pp. 1–19, 2010, doi: <https://doi.org/10.1186/2041-1480-1-8>.
- [29] S. Bishop *et al.*, "Engineering with Logic: Rigorous Test-Oracle Specification and Validation for TCP/IP and the Sockets API," *J. ACM*, vol. 66, no. 1, Dec. 2018, doi: [10.1145/3243650](https://doi.org/10.1145/3243650).
- [30] S. Meeus *et al.*, "Recommendations for interoperability among infrastructures," *RIO*, pp. 59–70, Oct. 2022, doi: <https://doi.org/10.3897/rio.8.e96180>.
- [31] N. Puspitasari, E. Budiman, Y. N. Sulaiman, and M. B. Firdaus, "Microservice API Implementation For E-Government Service Interoperability," *J Phys Conf Ser*, vol. 1807, no. 1, Apr. 2021, doi: <https://doi.org/10.1088/1742-6596/1807/1/012005>.
- [32] G. da Silva Serapião Leal, W. Guédria, and H. Panetto, "Interoperability assessment: A systematic literature review," *Comput Ind*, vol. 106, pp. 111–132, Apr. 2019, doi: [10.1016/J.COMPIND.2019.01.002](https://doi.org/10.1016/J.COMPIND.2019.01.002).
- [33] T. Rings, P. Poglitsch, S. Schulz, L. Serazio, and T. Vassiliou-gioles, "A generic interoperability testing framework and a systematic development process for automated interoperability testing," *International Journal on Software Tools for Technology Transfer*, vol. 16, no. 3, pp. 295–313, Jun. 2014, doi: <https://doi.org/10.1007/s10009-013-0281-2>.

Research_paper_Leon_Mathias

GRADEMARK REPORT

FINAL GRADE

54/100

GENERAL COMMENTS

Instructor

A good effort in the report. Topics covered well, with a huge number of sources provided of high academic credibility.

The implmenetation is a limited update of my code. This is a bit underwhelming, but a reasonable report outlining some of the features.

Part 1 (report): 75

Part 2 (implementation): 33

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10