Leeds Beckett University
Faculty of Arts, Environment & Technology

BSc (Hons) Computing
Academic Year 2013-2014
Joanne Kennedy C3369865
Networking:
Accessing X-Stream from a remote location

Contents

Introduction

During this report I was investigating the behaviour of the TCP/IP protocol stack when accessing x-stream from a remote location such as my home. I used Wireshark to analyze the packet data and I discussed in detail the background processes which occurred at each layer of the protocol stack. During this examination I studied specific protocols and the changes which occurred to the packet headers as the data passed through the network.

Application Layer

The application layer is the highest level of the protocol stack; this layer makes the communication between the applications and the transport protocols. The application protocol used for the given task was HTTP which is the specific protocol for web pages, on top of SSL/TSL, which then makes it HTTPS; HTTPS was most likely selected for x-stream because it prevents security breaches by encrypting the data flow between the client and server.

DNS also works on the application layer and was used when accessing x-stream, DNS queries were sent out in order to resolve the web address https://x-stream.leedsmet.ac.uk/ into an IP address, a standard query response was then returned and the client could then send a connection request to the server. (Figure 1)

Figure 1

2916	16:56:37.475314000	192.168.0.5	192.168.0.1	DNS	91 Standard query 0x260c A calendar.student.leedsmet.ac.uk
2917	16:56:37.475538000	192.168.0.5	192.168.0.1	DNS	87 Standard query 0xd01b A mail.student.leedsmet.ac.uk
2939	16:56:37.492176000	192.168.0.1	192.168.0.5	DNS	155 Standard query response 0x260c CNAME ghs.google.com CNAME ghs.l.google.com A 173.194.66.121
2951	16:56:37.499956000	192.168.0.1	192.168.0.5	DNS	151 Standard query response 0xd01b CNAME ghs.google.com CNAME ghs.l.google.com A 173.194.66.121
3263	16:56:37.924684000	192.168.0.5	192.168.0.1	DNS	78 Standard query 0xd154 A www.leedsmet.ac.uk
3264	16:56:37.924684000	192.168.0.5	192.168.0.1	DNS	71 Standard query 0x26d4 A twitter.com
3265	16:56:37.926854000	192.168.0.5	192.168.0.1	DNS	80 Standard query 0x359f A platform.twitter.com
3268	16:56:37.936247000	192.168.0.1	192.168.0.5	DNS	142 Standard query response 0xd154 A 160.9.244.58 A 160.9.134.58 A 160.9.134.59 A 160.9.244.59
3269	16:56:37.940374000	192.168.0.1	192.168.0.5	DNS	135 Standard query response 0x26d4 A 199.59.148.10 A 199.16.156.230 A 199.16.156.38 A 199.16.156.6
3270	16:56:37.941386000	192.168.0.1	192.168.0.5	DNS	148 Standard query response 0x359f CNAME platform.twitter.com.tw.map.fastly.net A 199.96.57.6
3401	16:56:38.518730000	192.168.0.5	192.168.0.1	DNS	85 Standard query 0xf09b A cdn.syndication.twimg.com
3407	16:56:38.527434000	192.168.0.1	192.168.0.5	DNS	175 Standard query response 0xf09b CNAME san.twitter.com.edgekey.net CNAME e5903.g.akamaiedge.net A 2.21.113.224
3469	16:56:38.660047000	192.168.0.5	192.168.0.1	DNS	73 Standard query 0xd33c A pbs.twimg.com
3478	16:56:38.674611000	192.168.0.1	192.168.0.5	DNS	139 Standard query response 0xd33c CNAME wildcard.twimg.com.tw.map.fastly.net A 199.96.57.7
3496	16:56:38.696898000	192.168.0.5	192.168.0.1	DNS	64 Standard query 0x028e A t.co
3535	16:56:38.721944000	192.168.0.5	192.168.0.1	DNS	64 Standard query 0x028e A t.co
3560	16:56:38.741230000	192.168.0.1	192.168.0.5	DNS	96 Standard query response 0x028e A 199.16.156.11 A 199.16.156.75
4318	16:56:39.234634000	192.168.0.5	192.168.0.1	DNS	88 Standard query 0x99d0 A libraryonline.leedsmet.ac.uk
4319	16:56:39.256996000	192.168.0.1	192.168.0.5	DNS	120 Standard query response 0x99d0 A 160.9.134.32 A 160.9.134.33
4323	16:56:39.283874000	192.168.0.5	192.168.0.1	DNS	80 Standard query 0xab43 A myhub.leedsmet.ac.uk
4324	16:56:39.294066000	192.168.0.1	192.168.0.5	DNS	123 Standard query response 0xab43 CNAME myhub-web.vm.leedsmet.ac.uk A 160.9.247.62

LLMNR queries were also sent to allow IPv4 to perform name resolutions for hosts on the same local link. (Wikipedia, 2013)

Transport Layer

The NetBIOS Name Server was also used to send out 3 name queries with UDP on port 137, they were broadcast packets which sent queries of type NB, but no response queries were returned.

In order to perform the DNS queries UDP was used on port 53, this ensures a quicker response from the name server. Because I had previously cleared my cache before capturing the packets, the 'non-authoritative' flag was not set. The DNS queries were of type A and did not contain any answers

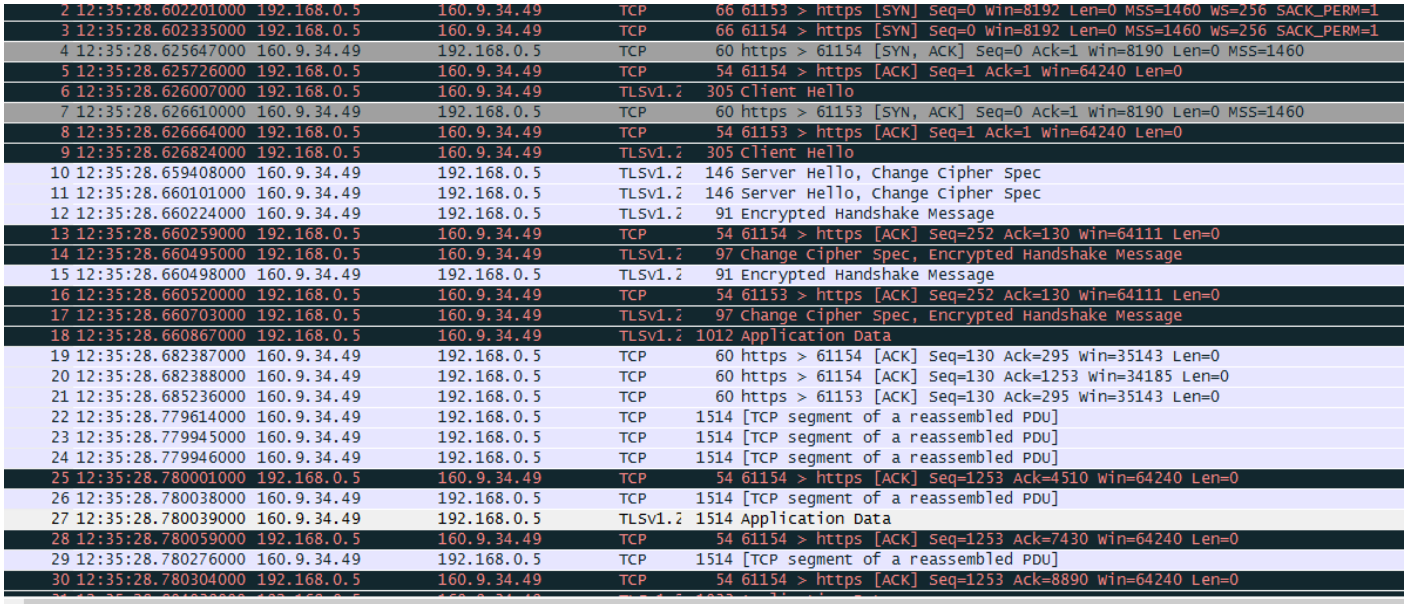
where as the response queries were of type A and contained between 3-4 answers per response. The answers contained a name, type, class, TTL, data length and an IP address.

Once the network understood that we were using the web application layer protocol 'HTTP/HTTPS' it then initiated a TCP connection with the server https://x-stream.leedsmet.ac.uk/ on port 443 which is the default port for HTTPS. This was displayed on Wireshark through the 'three way handshake' whereby three connection messages were transmitted, and the appropriate flags were set.

Once the connection was established the SSL/TLS connection could be established. The client initiated the connection by sending a 'Client Hello' message to the server, when exploring the packet details you can see that the values of Ethernet II, IPv4 and TCP are consistent with the TCP connection previously analyzed, the details of the cipher suites supported by the client are also present and have been sent to the server. (Wikiversity, 2012; Figure 2)

The sever sends a TCP acknowledgement message to the client, the client responds with an acknowledgement message and another 'Client Hello' message. The server then returns with a 'Sever Hello Change Cipher Spec' message, this ensures the connection is established, returns the list of cipher suites chosen that it supports and tells the client to set up the cipher suites previously agreed. It then sends an 'Encrypted Handshake Message' to the client, the client then acknowledges this message and returns the 'Change Cipher Spec, Encrypted Handshake Message' to the server. The server responds with another 'Encrypted Handshake Message,' the client computer again acknowledges this messages and returns the 'Change Cipher Spec, Encrypted Handshake Message' to the server along with the application data. The server uses TCP and PDU to disassemble information being transferred into many segments before returning the application data to the client. (Wikiversity, 2012; Microsoft, 2014; University of Birmingham, 2014; Superuser, 2011; Figure 2)

Figure 2



The transport layer passes the segments and packets down to the IP layer where in this case Internet Protocol Version 4 is being used. The 'don't fragment' flag has been set, meaning if a packet did need fragmenting it would be automatically dropped. IPv4 then converts the packets and segments into IP datagram's and determines the IP addresses, it then routes the converted datagram's from

node to node until they reach their final destination node. (Oracle Docs, 2010; Lecture Slides, 2014; Irv Englander, 2009)

The IP layer adds control data to the packet header including the source and destination IP addresses, with the source address being my IP address which the router has assigned to my home computer 192.168.0.5 and the destination address being the internet hosts IP address 160.9.34.49. It also shows us the IP header length which in this case is 20 bytes, a checksum field which checks for errors and a protocol field which shows us which data the datagram is encapsulating. In this investigation we can see the if type field has a value of oxo800 which means IPv4, we can also see that it has added the differentiated services field which is set to its default value of 0 which means best effort. (Oracle Docs, 2010; Lecture Slides, 2014)

Routers carefully track web requests; they read the destination address from the IP header and check the routing table in order to forward the packets to the next hop. The average TTL for the packets sent from client to server is 128 and from the server to the client it is 237, this ensures the packets do not remain on the network for too long. (Oracle Docs, 2010; Lecture Slides, 2014)

It is also the IP layer where ARP was implemented; the IP layer needs to convert the IP addresses into MAC addresses before sending them down to the link layer because the link layer works primarily with physical addresses rather than IP addresses, in order to make this conversion the telecommunication protocol ARP was needed. ARP sent out broadcast queries with the IP address to each node on the local network, a response was then returned with the destination MAC address, because of the ARP cache table the broadcast process was only needed during the first packet, this speeded up the process and the MAC address was then sent in a frame to the link layer. (Irv Englander, 2009)

Link Layer

In this layer information added to the header includes the source and destination MAC addresses, the data length field, the data field and a field which confirms integrity. In order to move the datagram's received a link layer protocol must be used. The link layer protocol provides a number of actions to ensure integrity and successful transition; this includes error checking, retransmission of frames and packets, flow control and random access. However the link layer protocol used in accessing x-stream is Ethernet, meaning it does have error detection and it can cancel packets however Ethernet does not provide flow control or retransmission because of the short wires it means things such as errors are highly unlikely. (J.F. Kurose, K.W. Ross, 2009; Wikipedia, 2014)

Physical Layer

This is the final layer which shows us the physical data which has been sent. Information added to the header includes the interface id, the encapsulation type which in this case is Ethernet, the time of arrival, the frame number, frame length and the capture length. This layer then transmits frames as a series of signals over the local media. This layer allows us to read the data passed however because in this case the SSL application has been used it shows the encrypted application data field instead of a data field, and this ensures me that the login and password has been successfully encrypted. (Lecture Slides)

Conclusion

I have tried to analyze the data in a number of ways; I looked at a TCP flow graph (figure 3) which showed me the connection between the client and server in a graph and helped me to see the flow of data frames and the essential details including any packets that were dropped or re-transmitted. I also looked at a time/sequence graph (figure 4) where each dot represents a TCP segment. In this case you can see how solid black lines have formed which represents a series of packets sent back to back. I have manage to see for myself all the processes that occur in the background when accessing a simple web page such as x-stream, and all of the above happens in less than a few minutes. I believe with more knowledge of TCP/IP and Wireshark I could have analysed the data in much more detail however I do believe I have covered the relevant protocols used in each layer of the protocol stack. (J.F. Kurose, K.W. Ross, 2007; techrepublic, 2012)

Figure 3

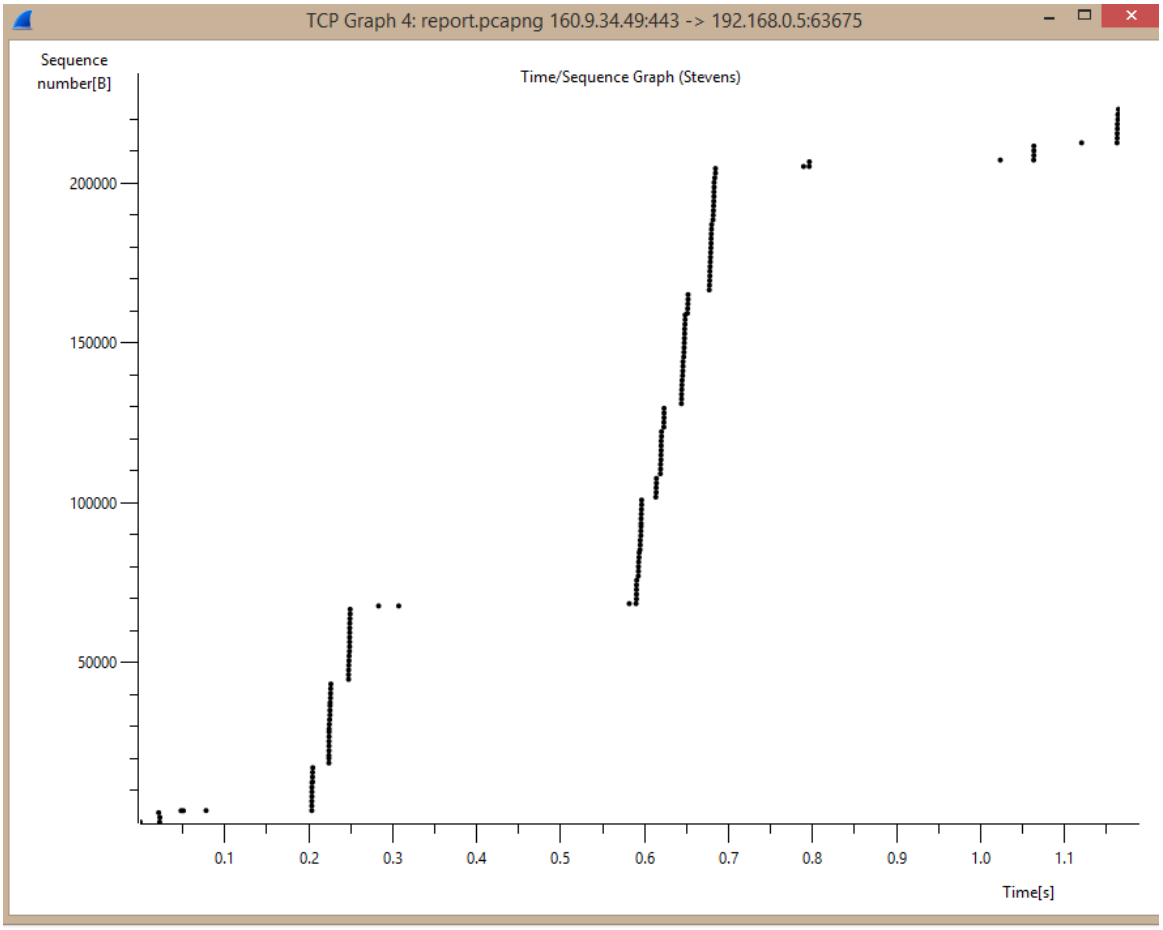
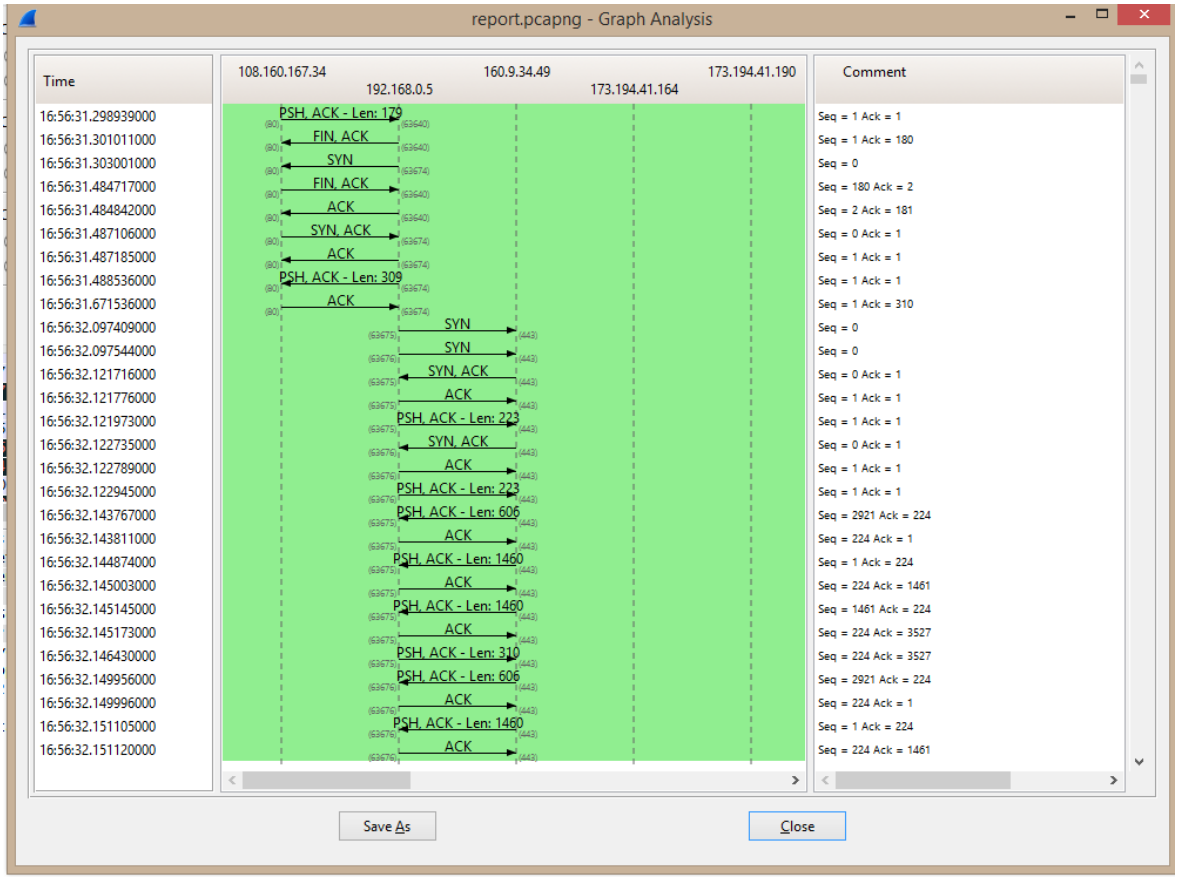


Figure 4


report.pcapng - Graph Analysis
Time
108.160.167.34
192.168.0.5
160.9.34.49
173.194.41.164
173.194.41.190
Comment
16:56:31.298939000 PSH, ACK - Len: 179 Seq = 1 Ack = 1
16:56:31.301011000 FIN, ACK Seq = 1 Ack = 180
16:56:31.303001000 SYN Seq = 0
16:56:31.484717000 FIN, ACK Seq = 180 Ack = 2
16:56:31.484842000 ACK Seq = 2 Ack = 181
16:56:31.487106000 SYN, ACK Seq = 0 Ack = 1
16:56:31.487185000 ACK Seq = 1 Ack = 1
16:56:31.488536000 PSH, ACK - Len: 309 Seq = 1 Ack = 1
16:56:31.671536000 ACK Seq = 1 Ack = 310
16:56:32.097409000 SYN Seq = 0
16:56:32.097544000 SYN Seq = 0
16:56:32.121716000 SYN, ACK Seq = 0 Ack = 1
16:56:32.121776000 ACK Seq = 1 Ack = 1
16:56:32.121973000 PSH, ACK - Len: 223 Seq = 1 Ack = 1
16:56:32.122735000 SYN, ACK Seq = 0 Ack = 1
16:56:32.122789000 ACK Seq = 1 Ack = 1
16:56:32.122945000 PSH, ACK - Len: 223 Seq = 1 Ack = 1
16:56:32.143767000 PSH, ACK - Len: 606 Seq = 2921 Ack = 224
16:56:32.143811000 ACK Seq = 224 Ack = 1
16:56:32.144874000 PSH, ACK - Len: 1460 Seq = 1 Ack = 224
16:56:32.145003000 ACK Seq = 224 Ack = 1461
16:56:32.145145000 PSH, ACK - Len: 1460 Seq = 1461 Ack = 224
16:56:32.145173000 ACK Seq = 224 Ack = 3527
16:56:32.146430000 PSH, ACK - Len: 310 Seq = 224 Ack = 3527
16:56:32.149956000 PSH, ACK - Len: 606 Seq = 2921 Ack = 224
16:56:32.149996000 ACK Seq = 224 Ack = 1
16:56:32.151105000 PSH, ACK - Len: 1460 Seq = 1 Ack = 224
16:56:32.151120000 ACK Seq = 224 Ack = 1461
Save As
Close

Bibliography

Wikiversity (2012) **Wireshark/HTTPS** [Online] Avaliable from:
< http://en.wikiversity.org/wiki/Wireshark/HTTPS > [Accessed 14 May 2014]

Microsoft (2014) **TLS Handshake Protocol** [Online] Avaliable from:
< http://msdn.microsoft.com/en-gb/library/windows/desktop/aa380513(v=vs.85).aspx>
[Accessed 14 May 2014]

University of Birmingham (2014) **About SSL/TLS** [Online] Avaliable from:
<https://www.cs.bham.ac.uk/~mdr/teaching/modules03/security/students/SS8a/SSLTLS.html>
[Accessed 14 May 2014]

Superuser (2011) **TCP segment of a reassembled PDU** [Online] Avaliable from:
< http://superuser.com/questions/255157/tcp-segment-of-a-reassembled-pdu> [Accessed 14 May 2014]

Oracle Docs (2010) **Data Encapsulation and the TCP/IP Protocol Stack** [Online] Avaliable from:
< http://docs.oracle.com/cd/E19455-01/806-0916/ipov-32/index.html> [Accessed 14 May 2014]

Wikipedia (2013) **Link-Local Multicast Name Resolution** [Online] Avaliable from:
<http://en.wikipedia.org/wiki/Link-local_Multicast_Name_Resolution>[Accessed 15 May 2014]

Techrepublic (2012) **Using the flow graph feature on Wireshark** [Online] Avaliable from:
< http://www.techrepublic.com/blog/linux-and-open-source/using-the-flow-graph-feature-on-wireshark/> [Accessed 15 May 2014]

Wikipedia (2014) **Data Link Layer** [Online] Avaliable from:
< http://en.wikipedia.org/wiki/Data_link_layer> [Accessed 21 May 2014]

J.F. Kurose, K.W. Ross, (2007) **Computer Networking – a top down approach**, 4th edition

J.F. Kurose, K.W. Ross, (2009) **Computer Networking – a top down approach**, 5th edition, international edition

Irv Englander (2009) **The Architecture of Computer Hardware, Systems Software & Networking** – an information technology approach, 4th edition, international student version