Networking (Extended) HTTP Protocol and Options

Task 1:

curl is used to create statements for a number of different protocols to more easily access data from other locations.

Website 1 – www.cs.bham.ac.uk

Output HTTP:

Trying 147.188.192.42:80...
** TCP_NODELAY set*
** Connected to www.cs.bham.ac.uk (147.188.192.42) port 80 (#0)*
> GET / HTTP/1.1
> Host: www.cs.bham.ac.uk
> User-Agent: curl/7.66.0
*> Accept: */**
>
** Mark bundle as not supporting multiuse*
< HTTP/1.1 200 OK
< Date: Fri, 15 Nov 2019 14:06:56 GMT
< Server: Apache
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/html; charset=utf-8
<
<!doctype html>
<html>
Full HTML for the webpage follows
</html>
** Connection #0 to host www.cs.bham.ac.uk left intact*

This first curl shows a standard HTTP request and from further research seems to be a very common output, the request sent and the returned data with no errors (200 OK).

Output HTTPS:

** Trying 147.188.192.42:443...*
** TCP_NODELAY set*
** Connected to www.cs.bham.ac.uk (147.188.192.42) port 443 (#0)*
** ALPN, offering h2*

** ALPN, offering http/1.1*
** successfully set certificate verify locations:*
** CAfile: /etc/ssl/certs/ca-certificates.crt*
* CApath: none*
** TLSv1.3 (OUT), TLS handshake, Client hello (1):*
** TLSv1.3 (IN), TLS handshake, Server hello (2):*
** TLSv1.2 (IN), TLS handshake, Certificate (11):*
** TLSv1.2 (IN), TLS handshake, Server key exchange (12):*
** TLSv1.2 (IN), TLS handshake, Server finished (14):*
** TLSv1.2 (OUT), TLS handshake, Client key exchange (16):*
** TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):*
** TLSv1.2 (OUT), TLS handshake, Finished (20):*
** TLSv1.2 (IN), TLS handshake, Finished (20):*
** SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384*
** ALPN, server did not agree to a protocol*
** Server certificate:*
** subject: C=GB; ST=West Midlands; L=BIRMINGHAM; O=University of Birmingham; OU=School of Computer Science; CN=www-lb-2.cs.bham.ac.uk*
** start date: Feb 5 14:51:14 2018 GMT*
** expire date: Feb 5 15:01:00 2021 GMT*
** subjectAltName: host "www.cs.bham.ac.uk" matched cert's "www.cs.bham.ac.uk"*
** issuer: C=BM; O=QuoVadis Limited; CN=QuoVadis Global SSL ICA G3*
** SSL certificate verify ok.*
> GET / HTTP/1.1
> Host: www.cs.bham.ac.uk
> User-Agent: curl/7.66.0
*> Accept: */**
>
** Mark bundle as not supporting multiuse*
< HTTP/1.1 200 OK
< Date: Fri, 15 Nov 2019 15:56:49 GMT
< Server: Apache
< Vary: Accept-Encoding
< Transfer-Encoding: chunked
< Content-Type: text/html; charset=utf-8
<

<!doctype html>
<html>
Full HTML for the webpage follows
</html>
* *Connection #0 to host www.cs.bham.ac.uk left intact*
The second curl of the same website but with HTTPS ends very similarly to the first but the query is proceeded by a TLS handshake to provide security. This is done by both client and server providing and reviewing authentication to/from the opposite party such as ceritificates and keys.

Website 2 - www.bbc.co.uk
Output HTTP:
* *Trying 212.58.249.210:80...*
* *TCP_NODELAY set*
* *Connected to www.bbc.co.uk (212.58.249.210) port 80 (#0)*
> GET / HTTP/1.1
> Host: www.bbc.co.uk
> User-Agent: curl/7.66.0
*> Accept: */**
>
* *Mark bundle as not supporting multiuse*
< HTTP/1.1 301 Moved Permanently
< Server: nginx
< X-BBC-No-Scheme-Rewrite: 1
< X-Cache-Action: HIT
< X-Cache-Hits: 5466
< Vary: X-BBC-Edge-Scheme
< Cache-Control: public, max-age=3600
< X-Cache-Age: 2014
< Content-Type: text/html
< Date: Fri, 15 Nov 2019 14:22:55 GMT
< Location: https://www.bbc.co.uk/
< Content-Length: 162
< Connection: Keep-Alive
<
<html>
<head><title>301 Moved Permanently</title></head>
<body>

* <h1>301 Moved Permanently</h1> *
*<hr> nginx *
</body>
</html>
* *Connection #0 to host www.bbc.co.uk left intact*
The BBC website would not allow access to HTTP, this aligns with accessing the website itself where the user is automatically redirected to the HTTPS site, this shows a move by website owners to increase security.

Output HTTPS:
This output was incedibly large and so I decided to not copy it over but it shows a similar TLS handshake. Once completed all a massive amount of html is shown (This could be considered normal for such a large company). Interestingly a large amount of javascript is located within the html rather than being contained in seperate files, this may be done because it is more efficient on resources or for a number of other reasons, it does raise questions of security however if all scripts are easily readable then coding errors could possibly be found quicker by malicious entities.

Further findings
The website www.facebook.com did not produce the same output as with www.bbc.co.uk, instead it displayed 302 Found, this is another way of redirecting the user to HTTPS.

Task 2:
The command
telnet www.cs.bham.ac.uk 80
is entered to connect and once connected the Query
GET / HTTP/1.1
Host: www.cs.bham.ac.uk
*Accept: */**
is entered. The first line of the Query denotes that nature of the request, the GET command is used to request data from the server hosting the HTTP, the / indicates the top index and the HTTP/1.1 denotes version of HTTP used. The second line denotes host which is necessary to ensure you connect to the correct website as several

may be hosted to the same IP address. Lastly, the final line states that any content type is accepted by the request.

Another query was used to retrieve the research page from the website
GET /research/ HTTP/1.1
Host: www.cs.bham.ac.uk
*Accept: */**
The only change is to attach a web location to the top index, research/ was added to the backslash in place to transfer to the webpage. An interesting occurrence was that when the closing backslach was not added, the request would return a 301 error.

Task 3:
As shown below when connecting to www.batten.eu.org, DNS queries are performed initially to locate the webpage. Once found a TCP Handshake occurs. The SYN, SYN-ACK, ACK packets initialise the connection from port 43104 of the client to the server port 80 (The commonly used port for HTTP web services). The GET request sent via curl is then completed with acknowledgements occuring during the transfer. Finally the FIN-ACK packets are sent to and from the two devices to sever the connection between the two devices. The connection that has occurred also shows what would happen with a 301 error, there are a lot of packets being sent for such a small amount of data and the connection is severed afterwards, leading to a new one needing to be formed. This is very ineffecient and UDP would be a better choice in this instance.