TCP / UDP

TCP

Transmission Control Protocol

In TCP, the client must first contact the server and request to connect to socket which the server has been configured to listen on. The server accepts the connection and creates a new socket for communicating with the client, so there can simultaneously be several connections at once. In TCP, there is error detection and data is received in the order it is sent. This error detection comes at the cost of speed, when compared to UDP.

TCP connections are established with the following steps

1. **Bind** socket to listen on server
2. **Create** request from client to connect to socket
3. **Accept** connection request and provide the server with a unique socket for communicating with this new connection

A bytestream is transmitted over a connection oriented channel and is a constant stream of data from a source to a destination. Since this happens on a connection oriented channel, bytestreams are constant and are only lost if the connection is closed.

A few protocols which use TCP connections are HTTP, HTTPS, SMTP, POP, and FTP.

TCP SSL Handshake

First, we should distinguish between a SSL *session* and *connection*.

A session can have multiple connections at any given time. SSL sessions are cached by the browser, typically until the browser is closed entirely but this may vary depending on browser and configurations. SSL sessions are also remembered by the server itself, which is also configurable on the back-end of the server but typically these sessions can last anywhere from 10 minutes to 10 hours. When a session is established, *a master secret is created for the client, along with two random values - one for the client from the server, and one for the server from the client.*

When a new connection is established within a valid session, *new symmetric keys are established for-each connection*. This is because the symmetric keys are derived from the master secret and

the random values provided by both the client and server. Using these three values, symmetric keys are created for-each connection under any given session. Typically, a server will close connections after 1-2 minutes of inactivity and thus a new handshake will be required in order to establish a new connection.

Symmetric keys are only stored in RAM. This means if you shut down your device you can guarentee that a new handshake will occur the next time a HTTPS request is made to the server.

The steps required for a successful TCP SSL Handshake are seen below

1. Client sends `hello` message to the server, including a TLS version supported by the client, the cipher suites supported by the client, and a string of random bytes called the `client random`
2. The server responds with a `hello` message sent to the client, including the SSL certificate, the cipher suite used by the server, and a random string of bytes called the `server random`
3. The client verifies the SSL certificate with the Certificate Autority that issued the certificate, confirming the server is who it claims to be
4. The client sends one final random string of bytes called the `master secret`, which is encrypted using the server's public key which was provided with the SSL certificate from the previous steps
5. The server decrypts the `master secret` string using it's private key
6. Both the server and the client produce a `session key` using the `client random`, `server random`, and `master secret`. Since both the client and the server produce their own `session key` from the same ingredients, they both arrive at the same result and have matching `session keys`
7. The client sends a `finished` message encrypted with it's session key
8. The server sends a `finished` message encrypted with it's session key
9. Symmetric encryption has been established, and the handshake has completed. Secure communication can continue between the client and the server.

It is worth noting that depending on the cipher suite selected by the server in step 2, asymmetric encryption may or may not be used in the steps above. This means that steps 4-6 can vary depending on the cipher suite.

UDP

User Datagram Protocol

In UDP, there is no connection between a client and a server, packets may be sent out of order, and packets may be lost. The data keeps transmitting regardless, the sender using an IP and port which the recipient can derive from the datagram. UDP does not check for errors, and as a result has a faster speed than TCP connections. Since UDP is conectionless, there are no steps to establish a UDP connection.

UDP would be preferred in situations where data transfer does not stop if a segment is lost, like streaming a video or playing an online multiplayer game

A datagram is transmitted over a connectionless communication channel and represents just one part of a message being sent. Datagrams can be lost during transmission and resent.

Resources and Links

Cloudflare - TLS SSL Handshake

SSL.com - SSL/TLS Handshake

StackOverflow - SSL Session & Connection

Revision #2
Created 25 March 2022 02:35:41 by Shaun Reed
Updated 25 March 2022 13:07:50 by Shaun Reed