September 2021

TRANSPORT LAYER SECURITY PATH TRACER

David McGrew

Soumya Kalahasti

Vineeth Joseph

TRANSPORT LAYER SECURITY PATH TRACER

AUTHORS:
David McGrew
Soumya Kalahasti
Vineeth Joseph

ABSTRACT

Today, most of the traffic that traverses the Internet is encrypted. Users (e.g., clients) and servers are able to exchange data securely using the Transport Layer Security (TLS) protocol. However, there will likely be one or more proxies in the path between a client and a server and those proxies are able to change some of the security parameters based on, for example, a network security policy. As a result, a client may not know exactly what is happening in the middle. To address these types of challenges, techniques are presented herein that support an extension to the handshake protocol that can request a 'trace' feature along a network path. All of the different TLS entities in the network can recognize the extension and add any changes that they are making to the upstream proposal. Advantages of the techniques presented herein include, among other things, helping to troubleshoot the TLS policy end-to-end.

DETAILED DESCRIPTION

Most of the traffic that currently traverses the Internet is encrypted. Clients and servers are able to exchange data securely using the Transport Layer Security (TLS) protocol. A client and a server negotiate a cipher suite and exchange key parameters to be able to derive a symmetric key for bulk encryption purposes. During such a call flow the client thinks that it is talking directly to the server. In reality, there will be one or more proxies in the path between the client and the server and those proxies are able to change some of the security parameters based on, for example, the network security policy. Proxies can be either transparent or terminating. The client may not know exactly what is happening in the middle. If the client would like to know the exact treatment that his or her connection is getting across the network, today there is no way to know those particulars. Also, if an administrator would like to know what proxies are on a path and

what changes are being made between a client and a server from a TLS perspective (e.g., for debugging purposes) today there is no way to know those details.

An exemplary scenario, which follows the narrative that was presented above, may be depicted as:

```
Client ---- Proxy1 ---- Internet ----- Proxy2 ---- Server
```

To address the types of challenges that were described above, techniques are presented herein that support an extension to the handshake protocol that can request a 'trace' feature along a network path. All of the different TLS entities in the network can recognize the extension and add any changes that they are making to the upstream proposal. Ultimately, a server receives the modified proposal from a client and recognizes the 'trace' extension in the client Hello and copies that information to the server Hello and sends it back towards the client. On the way back, the server Hello may be modified (by, for example, middle boxes) and those changes will be recorded along the path in the extension data. A client receives the full trace of negotiation parameters along with an indication of who changed what. If an entity does not understand the identified trace extension, they can just ignore it and send it as is without dropping it.

According to aspects of the techniques presented herein, a TLS extension with a type that is selected from within the range of codes reserved for private use (65,282 through 65,535), such as '65,299' for examples discussed herein, may support the tracing of TLS parameter changes throughout a network path. A client may send the type 65,299 extension in the client Hello indicating that it would like to see any of the TLS parameters that are changed during the traversal to the destination by any proxies or intermediate nodes in the path. A server receives the client Hello, sees the type 65,299 extension, copies the data as it is from the type 65,299 extension, and appends it as an extension in the server Hello. The server Hello is received by the client which then parses the information.

The format that is used in the client Hello and the server Hello, according to aspects of the techniques presented herein, is as follows:

```
<65,299> (extension type)
  Total length of the extension
  Proxy Id
    Type (client Hello/server Hello)
    Added parameter (+, parameter _id)
    Subtracted parameter (-, parameter _id)
```

Under aspects of the techniques presented herein an important assumption encompasses the absence of any 'bad actors' in the network. A 'bad actor' may include, for example, any proxy or intermediate nodes which might populate wrong or incorrect information or which can remove the type 65,299 extension from a server Hello or a client Hello.

Figure 1, below, depicts elements of the call flow that is associated with an example involving client Hello and server Hello artifacts being modified and recorded for an end-to-end TLS path trace.

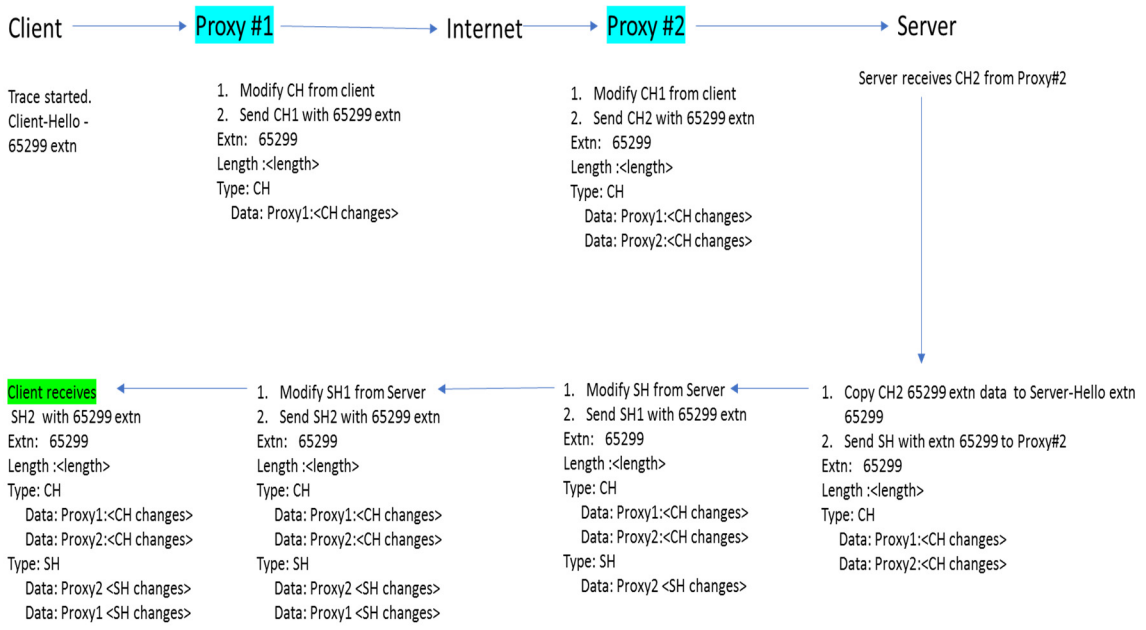

Figure 1: Exemplary Call Flow

Various of the techniques presented herein offer a number of advantages including, among other things:

- Serving as an excellent tool for administrators as they work to fine tune the policy of a network.
- Aiding in identifying potential security vulnerabilities on a path.
- Informing a user as to the proxies that are in a path. For example, a browser may provide this information to a user who is worried about the security of their sessions.

- Being relatively easy to implement.
- Being employable just for debugging purposes, thus avoiding any negative impact to data path performance.
- Working in both TLS 1.3 and prior versions or future versions such as, for example, the Quick UDP Internet Connections (QUIC) protocol.

It is also important to note potential disadvantages that may arise from use of aspects of the techniques presented herein. Such disadvantages may include, for example:

- One or more middle boxes may not honor the identified extension and either drop it or not do anything with it.
- An extension may become lengthy depending upon what information an entity chooses to include in it.

In summary, techniques have been presented herein that support an extension to the handshake protocol that can request a 'trace' feature along a network path. All of the different TLS entities in the network can recognize the extension and add any changes that they are making to the upstream proposal. Advantages of the techniques presented herein include, among other things, helping to troubleshoot the TLS policy end-to-end.