**Decrypting TLS traffic for troubleshooting**

**When the Key Exchange method is RSA, if you know the server’s private key, you can decrypt all traffic to and from that server. Instructions are here:**[**https://wiki.wireshark.org/SSL**](https://wiki.wireshark.org/SSL)

This method will become less and less accessible as time goes on. TLS 1.3 mandates that only protocols with Forward Secrecy (FS, or PFS) be used, mostly DH and ECDHE. In these protocols the complete session key cannot be derived from the packet capture even if we know the private keys. Therefore, there is no easy way to decrypt the traffic. Proxy servers and IDSs that need to inspect the traffic in its unencrypted form must resort to performing Man in the Middle (MITM) attacks against their clients. The system administrator must configure the clients so that they trust the proxy server or IDS certificate.

However, if you want to decrypt the traffic from your own workstation so you can see the HTTP traffic unencrypted and troubleshoot HTTP problems, you can do it. The Chrome and Firefox browsers allow you to enable logging of the session keys, so you can use them to decrypt the traffic. Again, this only works for traffic to and from a workstation you control. The method is described here: <https://jimshaver.net/2015/02/11/decrypting-tls-browser-traffic-with-wireshark-the-easy-way/>

I have had mixed results with this method. Perhaps 25% of the TLS traffic is decrypted and the rest is not. Part of it could be that the TLS protocol is much more complicated than we have discussed in class. It uses methods to save information from previous sessions so that connection establishment can be faster. If the information from previous sessions is not available in the SSLKEYLOGFILE, Wireshark may not be able to decrypt it. Feel free to experiment but there won’t be an assignment using this method, at least not yet.