

SSL decryption using Wireshark

Peter Wu

peter@lekensteyn.nl

<https://lekensteyn.nl>

January 12, 2016

Overview

Introduction

SSL/TLS

SSL Decryption using Wireshark

Conclusion

Wireshark: network protocol analyzer

The screenshot displays the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for file operations, navigation, and analysis. A filter bar at the top shows the filter expression "ssl and http".

The main packet list pane shows several captured packets. The selected packet (No. 3407) is highlighted in blue. The details pane below shows the expanded view of this packet, revealing the underlying protocols: Internet Protocol Version 4, Transmission Control Protocol, Secure Sockets Layer, and Hypertext Transfer Protocol. The HTTP details show a POST request to "/metrics" with various headers including Host, User-Agent, Accept, and Referer.

No.	Time	Source	Destination	Protocol	Length	Info
3407	8.1362...	10.9.0.2	accounts.fir...	HTTP	1881	POST /metrics HTTP/1.1 (text/...
3424	8.1721...	dcky6u1m8u6el...	10.9.0.2	HTTP	1525	HTTP/1.1 200 OK (PNG)
3620	8.3243...	accounts.firef...	10.9.0.2	HTTP	865	HTTP/1.1 200 OK (application/...
5862	11.215...	10.9.0.2	shavar.prod...	HTTP	582	POST /downloads?client=Firefox...
5867	11.398...	shavar.prod.mo...	10.9.0.2	HTTP	447	HTTP/1.1 200 OK (application/...
5887	11.525...	10.9.0.2	tracking-pro...	HTTP	476	GET /mozstd-track-digest256/14...

```

> Internet Protocol Version 4, Src: 10.9.0.2 (10.9.0.2), Dst: accounts.firefox.com (52.26.19...
> Transmission Control Protocol, Src Port: 42370, Dst Port: 443, Seq: 2077, Ack: 261185, Len...
  Secure Sockets Layer
    > TLSv1.2 Record Layer: Application Data Protocol: http
  Hypertext Transfer Protocol
    > POST /metrics HTTP/1.1\r\n
      Host: accounts.firefox.com\r\n
      User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
      Accept-Language: en-US,en;q=0.5\r\n
      Accept-Encoding: gzip, deflate\r\n
      Referer: https://accounts.firefox.com/?utm_campaign=fxa-embedded-form&utm_medium=referra...
  
```

Secure Sockets Layer (ssl), 1815 bytes Packets: 10782 · Displayed: 114 (1.1%) · Load time: 0:2.33 Profile: Default

Why decrypt SSL with Wireshark?

- Debug applications that use SSL.
- Packet captures contain a full view of all network traffic.
- Wireshark supports many (application) protocols.

Methods for obtaining plaintext

Active:

- MITM, replace certificate.

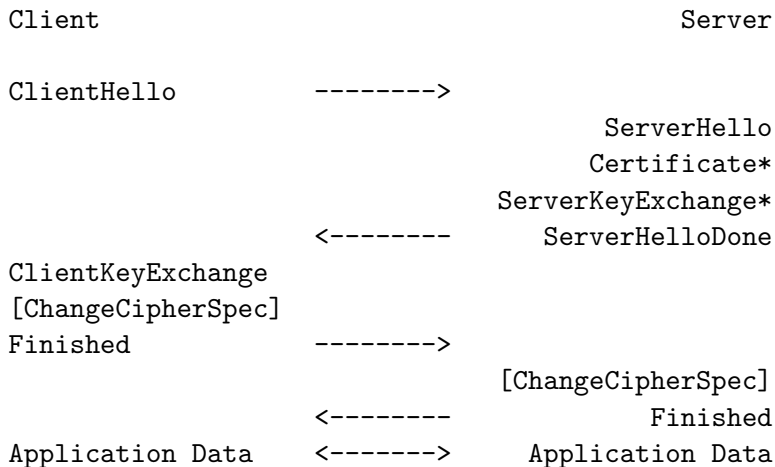
Passive:

- Option 1: after decryption (e.g. Web developer tools)
- Option 2: obtain secrets and capture packets (Wireshark)

SSL protocol overview

- SSLv3/TLS: basically the same protocol.
- Handshake establishing master secret (“session key”).
- Master secret is used for symmetric encryption of Application Data (HTTP, SMTP, etc.).

Handshake overview



Simplified SSL handshake (adapted from RFC 5246 (TLS 1.2))

Relevant security parameters

- Client Hello: Client Random, list of supported cipher suites.
- Server Hello: Server Random, selected cipher suite.
- ServerKeyExchange: needed for DHE cipher suites.
- ClientKeyExchange: **encrypted** pre-master secret.
- (Public key from Certificate is only used for authentication.)
- Master secret: calculated from pre-master secret and two Randoms.

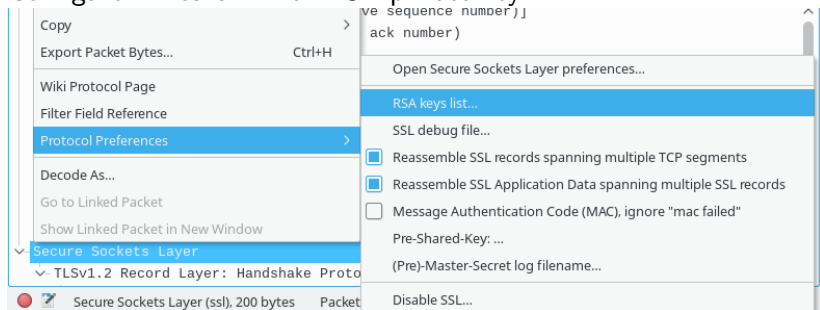
Pre-master secret

Value of pre-master secret depends on key exchange algorithm:

- RSA: 48 random bytes from **client**, encrypted using public key from Server Certificate.
- Diffie-Hellman: compute shared secret using parameters from Server Certificate (or ServerKeyExchange) and ClientKeyExchange.

Decryption using private RSA key of server

Configure Wireshark with RSA private key¹:



Limitations:

- Does not work with Diffie-Hellman key exchanges.
- Requires RSA private key of the **server**, i.e. cannot be used for decryption of traffic as a client.

¹See https://wiki.wireshark.org/SSL#Preference_Settings

SSL key logfile

- Text file containing (pre-)master secrets from SSL libraries².
- Configure file in Wireshark preferences: Edit → Preferences; Protocols → SSL; (Pre-)Master Secret log filename.
- Works also for clients!
- Supported by Firefox (via NSS), Chrome (via patched BoringSSL library).
- Also supported by cURL when built with NSS.
- For other libraries: dump keys from memory.

²Documented at https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS/Key_Log_Format

Example with Firefox (and other NSS users)

- Set environment variable `SSLKEYLOGFILE` to the output file before starting a program:
 - `SSLKEYLOGFILE=$PWD/premaster.txt firefox`
- Set file in Wireshark preferences.
- Start live capture.

Example with OpenSSL

- Does not support SSLKEYLOGFILE method.
- Solution: intercept OpenSSL library calls and dump keys³.
- Implementations⁴:
 - LD_PRELOAD approach: teach OpenSSL SSLKEYLOGFILE by injecting code.
 - `SSLKEYLOGFILE=premaster.txt`
`LD_PRELOAD=./sslkeylog.so curl https://example.com`
 - Debugger (GDB) approach: allows extraction of keys from running programs.

³<https://security.stackexchange.com/questions/80158/extract-pre-master-keys-from-an-openssl-application>

⁴<https://git.lekensteyn.nl/peter/wireshark-notes/tree/src> ▶

Concluding remarks

- Remember that RSA keys cannot be used for decryption of SSL sessions using DH key exchanges.
- SSL keylog files (`SSLKEYLOGFILE`) also works for DH key exchanges and can be used on clients too (Firefox, Chrome).
- Use the latest version (currently Wireshark 2.0.1) when possible, fixes various bugs.

Other resources

- Main website: <https://www.wireshark.org/>
- Q&A: <https://ask.wireshark.org/>
- IRC: #wireshark at Freenode
- Wireshark wiki on SSL: <https://wiki.wireshark.org/SSL>
- Sample captures: https://wiki.wireshark.org/SampleCaptures#SSL_with_decryption_keys