

The background of the slide is a photograph of an open, old book. The pages are yellowed with age and contain faint, illegible text. A dark, ornate metal key lies horizontally across the center of the open pages. The scene is dramatically lit from the upper right, creating a strong warm glow and casting long, soft shadows. The overall mood is one of mystery and discovery.

Embedding Secrets in PCAPNGs

Lucas Pardue
HTTP Workshop 2019



- ▶ Text file with master secrets².
- ▶ Works for any cipher, including RSA and DHE.
- ▶ Clients can use this too!
- ▶ Set environment variable SSLKEYLOGFILE before starting Firefox or Chrome. The variable is only read during startup, so restart if necessary.
- ▶ Format: CLIENT_RANDOM <Client Hello Random> <master secret>.

```
# SSL/TLS secrets log file, generated by NSS
CLIENT_RANDOM 5f4dad779789bc5142cacf54f5dafba0a06235640796f40048ce4d0d1df63ad8 a4d69a3fa4222d6b6f2492e66dca2b1fc4e2bc143df849ad45eff9
CLIENT_RANDOM c2407d5ba931798e3a35f775725fb3e5aefcb5804bb50271fe3bd5fb19c90061 e419759e7b44f766df6defe6b656eda3d430754044773b6fc0a91e
CLIENT_RANDOM abec6cf83ea1dcb135b21fd94bc0120dd6a37dca0fcd96efd8989d05c51cc3ab 5b4d525dfe3168132d388881033633c2aba99346c25ae8163f2191
CLIENT_RANDOM dffe2c85a7d6f3c3ec34ba52ea710f0f1649e58afa02f9824d983ea74f07900e fdb58d49482f876f200ce680b9d6987434e3aca54d203fc57cc588
CLIENT_RANDOM fbf40ada961093cd917fba97bfffef7c4b0bbf57a0cf90626dee417d3d12b3755 6b4e313d6be9316c42f47ddd3ceeeef9743825bd3c3bb25ec9ac73c
CLIENT_RANDOM 2b8184f7642df4bb5979ad9a623690b08f392deb94fdb64b00d7dc78b711638b dfdbe9f4d6949ee02489eb39b2c8d7770c12928becaf0ac1e34ed
CLIENT_RANDOM 7e4340c76c720d39c98e761697be0f32e1c79c6c04ade05a3f29325ac9cae612 1dfe402b85560048ae278b78febe83ee1640785b969c328d94a785
```

²File format at https://developer.mozilla.org/NSS_Key_Log_Format



- ▶ Configure file in Wireshark preferences: Edit → Preferences; Protocols → TLS; (Pre-)Master Secret log filename. (Protocol name is SSL before Wireshark 3.0.)
- ▶ Key log file is also read during a live capture. And if the file is removed and a new file is written, the new key log file is automatically read.
 - ▶ Caveat: key log is read while processing ChangeCipherSpec. If key is written too late, trigger a redissection (e.g. change a preference or (Un)ignore a packet).



- ▶ Any application built using NSS and GnuTLS enable key logging via the SSLKEYLOGFILE environment variable.
- ▶ Applications using OpenSSL 1.1.1 or BoringSSL d28f59c27bac (2015-11-19) can be configured to dump keys:

```
void SSL_CTX_set_keylog_callback(SSL_CTX *ctx,
    void (*cb)(const SSL *ssl, const char *line));
```
- ▶ ARM Mbed TLS using a debug callback³.
- ▶ cURL supports many TLS backends, including NSS, GnuTLS and OpenSSL. Key logging with OpenSSL/BoringSSL is possible since curl 7.58.0.
- ▶ Java applications can use jSSLKeyLog⁴.

³<https://github.com/Lekensteyn/mbedtls/commit/68aea15>

⁴<http://jsslkeylog.sourceforge.net>



- ▶ Windows native TLS library is Secure Channel (SChannel). Feature request for Microsoft Edge browser is pending⁷.
- ▶ Extracting secrets from SChannel is not impossible (but neither easy) though⁸.
- ▶ Apple macOS applications use SecureTransport, also not supported.

⁷<https://wpdev.uservoice.com/forums/257854-microsoft-edge-developer/suggestions/16310230-ssl-key-logging-aka-sslkeylogfile>

⁸<https://www.blackhat.com/docs/us-16/materials/us-16-Kambic-Cunning-With-CNG-Soliciting-Secrets-From-SChannel.pdf>

<https://lekensteyn.nl/files/wireshark-ssl-tls-decryption-secrets-sharkfest18eu.pdf>



- ▶ Force RSA key exchange (disable forward-secret cipher suites).
- ▶ Setup a fake CA and force traffic through a proxy like mitmproxy⁹, OWASP Zap, Fiddler or Burp Suite.
- ▶ All of these methods can be detected by the client. Certificate pinning can also defeat the custom CA method.
- ▶ The proxy interception method may also weaken security¹⁰.
- ▶ If you are really serious about a passive, nearly undetectable attack from a hypervisor, see the TeLeScope experiment¹¹.

⁹<http://docs.mitmproxy.org/en/stable/dev/sslkeylogfile.html>

¹⁰Durumeric et. al., The Security Impact of HTTPS Interception,
<https://jhalderm.com/pub/papers/interception-ndss17.pdf>

¹¹<https://conference.hitb.org/hitbsecconf2016ams/sessions/telescope-peering-into-the-depths-of-tls-traffic-in-real-time/>

Embedding session secret into a pcapng

Shipping in Wireshark 3.0+

Use case:

Store TLS decryption secrets in a pcapng file to avoid the need for extra out-of-band configuration of Wireshark. [Bug 9616](#) implemented this through a packet capture comment, but using pcapng blocks for this is preferable.

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6

Manual steps needed

For new files captured *in* Wireshark, Wireshark will not automatically copy decryption secrets (for example, those found using the RSA private key file or those read from the TLS Key Log File preference).

2. Create the key log file while triggering some traffic. For example, using curl 7.58.0 (built with OpenSSL):

`https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6`

`SSLKEYLOGFILE=some.keys curl https://example.com`

1. Start capture

Start the capture, for example using the GUI or CLI. For example, to capture traffic with "example.com" using the default network interface:

```
dumpcap -w some.pcapng -f "host example.com"
```

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6

2. Generate traffic

Create the key log file while triggering some traffic. For example, using curl 7.58.0 (built with OpenSSL):

```
SSLKEYLOGFILE=some.keys curl https://example.com
```

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6

3. Stop and add secrets

Stop the capture and save it to a file (some.pcapng). Add the TLS secrets from some.keys to a new file (some-with-keys.pcapng):

```
editcap --inject-secrets tls,some.keys some.pcapng some-with-keys.pcapng
```

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6

4. Success

Now you can load some-with-keys.pcapng in Wireshark 3.0 or newer without requiring any further configuration and have its TLS contents decrypted automatically.

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=15252#c6

Wrap up

Capturing bytes on wire and sharing them as captured is useful.

Other methods for exporting encrypted session traffic are kind of sucky, more often than not they mangle some important property of the session or lose fidelity

Sharing encryption secrets out of band is possible but is not incredibly user-friendly

In-band secrets solve all of the above challenges. This is still WIP and has some rough edges.

Thanks to Peter Wu (Lekensteyn) for the work and all his content borrowed for this presentation.