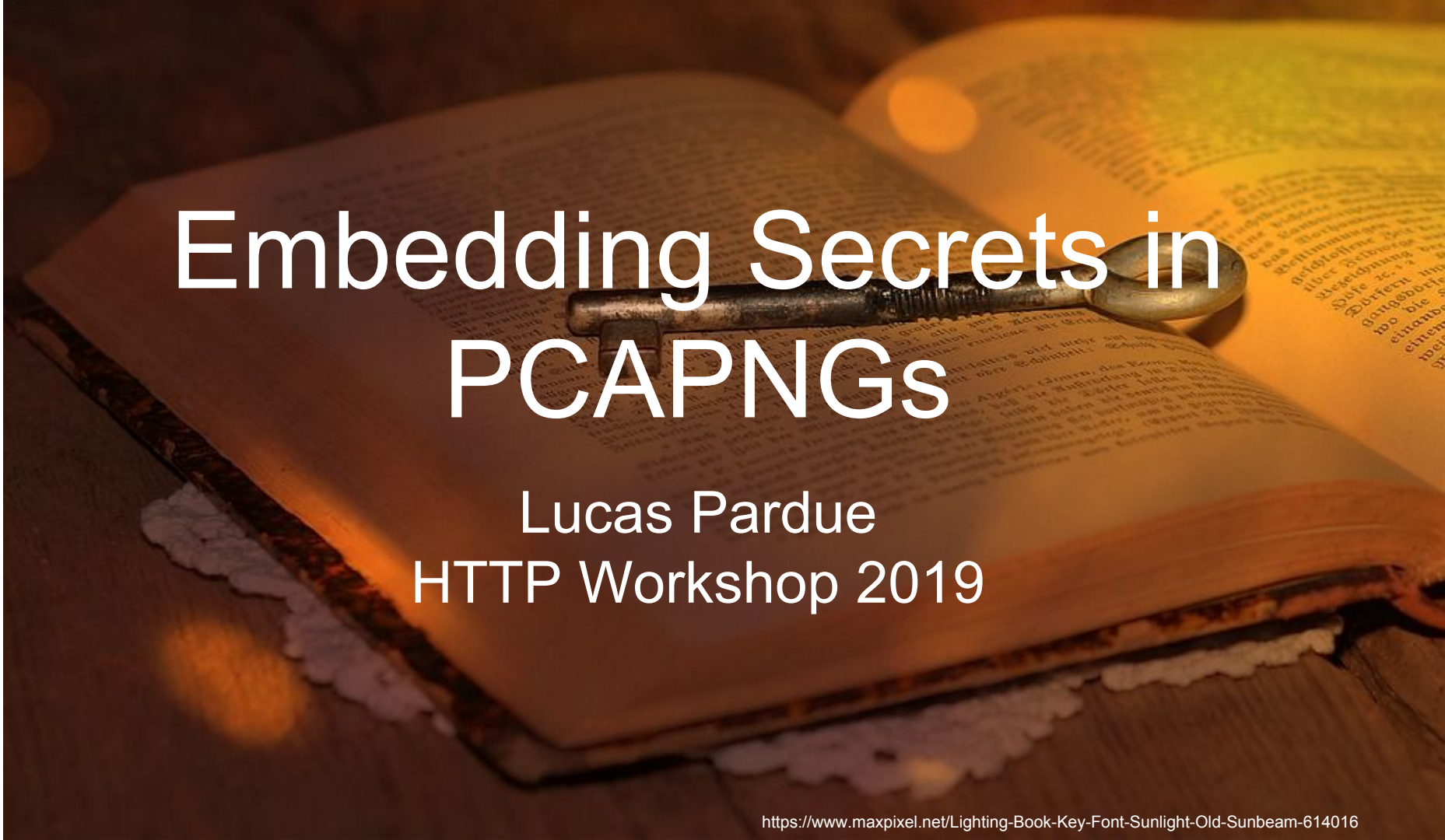
Embedding Secrets in PCAPNGs

Lucas Pardue

HTTP Workshop 2019



https://www.maxpixel.net/Lighting-Book-Key-Font-Sunlight-Old-Sunbeam-614016

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

Using SSL key log file in Wireshark

- 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).

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

Application and library support

- 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 callback[3].
- 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 jSSLKeyLog[4].

[3] https://github.com/Lekensteyn/mbedtls/commit/68aea15
[4] http://jsslkeylog.sourceforge.net

Unsupported applications for SSLKEYLOGFILE

- Windows native TLS library is Secure Channel (SChannel). Feature request for Microsoft Edge browser is pending[7].
- Extracting secrets from SChannel is not impossible (but neither easy) though[8].
- 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

Alternative ways to get the secret

- Force RSA key exchange (disable forward-secret cipher suites).
- Setup a fake CA and force traffic through a proxy like mitmproxy[9], 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 security[10].
- If you are really serious about a passive, nearly undetectable attack from a hypervisor, see the TeLeScope experiment[11].

---

[9] http://docs.mitmproxy.org/en/stable/dev/sslkeylogfile.html

[10] Durumeric 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/

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

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.