TLS Session Key Extraction from Memory on iOS Devices

Research Project 2

Tom Curran <tom.curran@os3.nl>
Marat Nigmatullin <marat.nigmatullin@os3.nl>
Motivation

- Increase in TLS encryption on iOS devices
- Prevents blackbox testing
- Existing tools disable TLS and rely on jailbreak
- Is there an alternative approach?
Research Question

Is it possible to extract TLS session keys from the process memory of a device running iOS 9.0 or greater?

- How is TLS handled in iOS?
- Can it be done with jailbroken \textit{and} non-jailbroken devices?
TLS

- Cryptographic protocol, successor of SSL
- Provides confidentiality and authentication
- Uses the Record protocol
- Sub-protocols
  1. Handshake
  2. ChangeCipherSpec
  3. Application Data
  4. Alert
TLS in iOS

- Foundation Networking APIs
- TLS handled via *Secure Transport API*
- App Transport Security (ATS) (9.0+)
  - TLS 1.2
  - Forward secrecy
  - Key Exchange - ECDHE
  - Authentication - ECDSA or RSA
  - Mandatory on App Store from 2017

Source: Apple.com
iOS Network Stack

AFNetworking

HTTP

NSURLConnection

HTTP

NSURLSession

HTTP (iOS 7+)

UIWebView

Web View

CFNetworking

Networking

Secure Transport

SSL/TLS

BSD Sockets

Networking
TLS Handshake

- Establish shared secret
- Four phases:
  1. Exchange capabilities and agree on connection parameters
  2. Authentication
  3. Agree on shared secret
  4. Verify handshake messages
TLS Handshake

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. ServerHelloDone
6. ClientKeyExchange
7. [ChangeCipherSpec]
8. Finished
9. [ChangeCipherSpec]
10. Finished
Key Material

- Client Random
- Pre-master Secret
- Server Random

Master Secret

Key Block

- Client MAC Key
- Server MAC Key
- Client Write Key
- Server Write Key
- Client IV
- Server IV
What do we need?
Tools
• Dynamic instrumentation toolkit
• Live inspection of processes
• Scriptable
  • Execute own debug scripts *inside* another process
• Used for
  • Attaching to processes
  • Hooking functions
  • *Inspecting memory*
Frida

Frida thread

Instrumentation scripts

frida-agent.so

Target

Source: frida.re
Finding the secret
Targeting iOS Secure Transport
Targeting iOS Secure Transport

- Secure Transport API hides internal handshake operations
- Encryption *actually* handled by coreTLS library
- Source code for older versions available online
struct _tls_handshake_s {
    tls_protocol_version negProtocolVersion;
    tls_protocol_version clientReqProtocol;
    tls_protocol_version minProtocolVersion;
    tls_protocol_version maxProtocolVersion;
    ...
    uint8_t clientRandom[SSL_CLIENT_SRVR_RAND_SIZE];
    uint8_t serverRandom[SSL_CLIENT_SRVR_RAND_SIZE];
    tls_buffer preMasterSecret;
    uint8_t masterSecret[SSL_MASTER_SECRET_SIZE];
    ...
}

typedef struct _tls_handshake_s *tls_handshake_t;
CoreTLS source

tls_handshake.h

```c
int tls_handshake_internal_prf(tls_handshake_t ctx,
    const void *vsecret,
    size_t secretLen,
    const void *label,
    size_t labelLen,
    const void *seed,
    size_t seedLen,
    void *vout,
    size_t outLen);
```

- Main object
- Master secret
- MS length
Using Frida

FRIDA

argument[0]
tls_handshake_t ctx

Dump the memory (ctx, 400)

argument[1]
const void *vsecret

Reading the sequence of bytes(*vsecret, secretLen)

argument[2]
size_t secretLen

Master secret

Master Secret

???

???

???
Memory dump with Frida

tls_types.h

... 
TLS_1_0 = 0x0301
TLS_1_1 = 0x0302
TLS_1_2 = 0x0303
...

Master secret
Finding the Identifiers

- Captured packets with Wireshark whilst running Frida
- Compared hex outputs to match identifiers
Non-jailbroken devices
Compiling Frida into an Application

1. Jailbreak
2. Decrypt
3. Inject & Link
4. Resign
5. Deploy
6. Attach

FRIDA
Demo
Is it possible to extract TLS session keys from the process memory of a device running iOS 9.0 or greater?

- Yes, both with jailbroken and non-jailbroken
- TLS APIs in iOS rely on coreTLS library
- Relies on Frida, also possible with llmdb
Future Work

- iOS 10
- Support for OpenSSL in iOS?
- TLS 1.3 in Draft
With special thanks to

Cedric van Bockhaven @ Deloitte
Thank you for your attention!

Questions?