MUD (D)TLS profiles for IoT devices

draft-reddy-opswg-mud-tls-00

IETF 105, Montreal

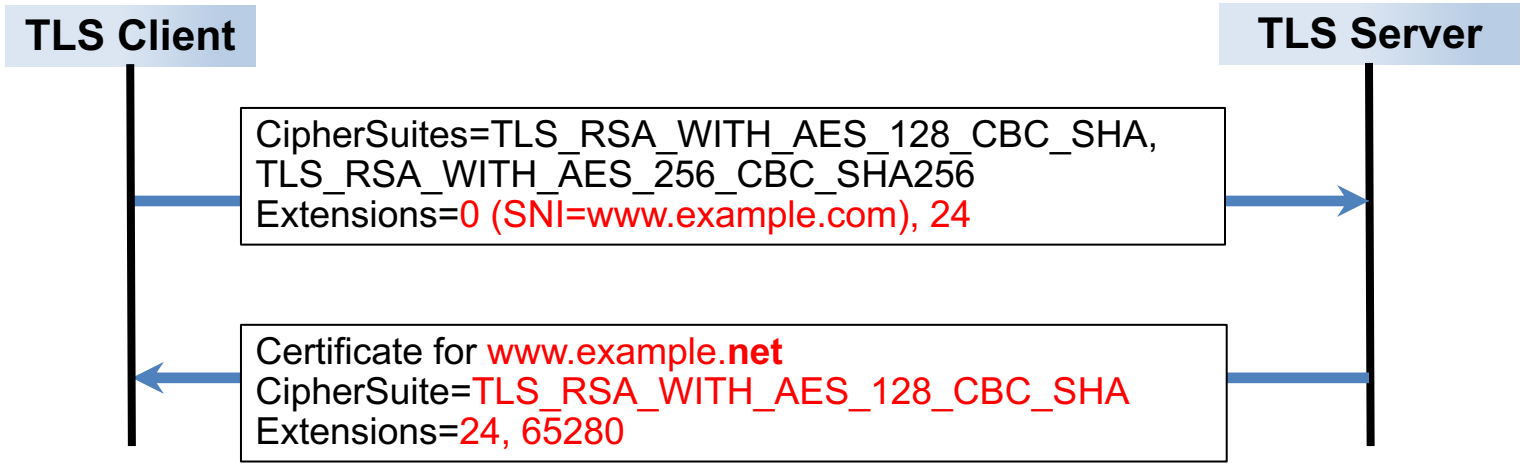
July 2019

T. Reddy (McAfee)
D. Wing (Citrix)

MUD (D)TLS Goal

- We propose extending MUD to describe TLS interactions

TLS handshake inspection



Malware TLS difers from legitimate software ([1])

- SNI and SAN mismatch
- DGA pattern in SNI or SAN
- Offered/Selected Ciphersuites
- Diversity of TLS extensions

Detect broken TLS

- Best-practice failure (e.g., RFC7525)
  - Expired certificates
  - SNI/SAN mismatch
  - Poor-quality cipher suites
- Re-use of same private key ([2])

([1]) “Deciphering Malware's use of TLS (without Decryption)”, https://arxiv.org/abs/1607.01639
([2]) “Millions of IoT Devices Using Same Hard-Coded CRYPTO Keys”, http://thehackernews.com/2015/11/iot-device-crypto-keys.html

Observable (D)TLS profile parameters

- IoT devices have constrained TLS usage patterns
  - One or few TLS crypto suites to reduce memory footprint
- Adding a new skill (“check weather”) changes server with same TLS parameters
  - Tested with Amazon Echo

- Profiled easily-obtained home Things:
  - Amazon Echo (Echo Spot, Echo Dot, Echo Show and Echo Plus)
  - Kindle eBook Reader
  - Google Chromecast, Google Home, Google Home Mini

Solution overview

- Extends MUD to observable TLS/DTLS profile parameters

```
module: reddy-opsawg-mud-tls-profile
  augment /mud:mud/mud:from-device-policy:
    +--rw client-profile
       +--rw tls-profiles* [protocol-version supported_versions]
          +--rw protocol-version          uint16
          +--rw supported_versions        boolean
          +--rw encryption-algorithms*    encryption-algorithm
          +--rw compression-methods*      compression-method
          +--rw extension-types*          extension-type
          +--rw acceptlist-ta-certs*      ct:trust-anchor-cert-cms
          +--rw SPKI-pin-sets*            SPKI-pin-set
          +--rw SPKI-hash-algorithm       ct:hash-algorithm-t
          +--rw supported-groups*         supported-group
          +--rw signature-algorithms*     signature-algorithm
          +--rw client-public-keys
          |  +--rw key-exchange-algorithms*     key-exchange-algorithm
          |  +--rw client-public-key-lengths*   client-public-key-length
          +--rw SNI-mismatch-allowed?     boolean
          +--rw server-name*              inet:domain-name
          +--rw actions
             +--rw forwarding    identityref
```

TLS 1.3 Inspection

- TLS 1.3 encrypts handshake, allowing inspection of few parameters
  - ClientHello ciphers and extensions (e.g., SNI)
  - ServerHello cipher and extensions
- Fuller inspection requires active participation in TLS 1.3
  - TLS-inspecting middlebox (yuck)
  - Passive observation sufficient

draft-reddy-opsawg-mud-tls-00

- Comments and suggestions are welcome