IoT TLS: Why It Is Hard

David Brown
What is IoT

“The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.”

— TechTarget
The 5 Worst Examples of IoT Hacking and Vulnerabilities in Recorded ...
https://www.iotforall.com › Insights ›
IoT hacking can be extremely effective, producing DDoS attacks that can cripple our infrastructure, systems, and way of life. Here are the 5 worst examples.
The 5 Worst Examples of IoT Hacking and Vulnerabilities in Recorded ...
https://www.iotforall.com › Insights ⇓

IoT hacking can be extremely effective, producing DDoS attacks that can cripple our infrastructure, systems, and way of life. Here are the 5 worst examples.
5 Worst Examples

- The Mirai Botnet
- The Hackable Cardiac Devices from St. Jude
- The Owlet WiFi Baby Heart Monitor Vulnerabilities
- The TRENDnet Webcam Hack
- The Jeep Hack
“IoT Security is not Interesting”

— James Mickens
Harvard University,
Associate Professor,
Authority on All Things
“TLS is the only good thing we have”

— James Mickens
Harvard University,
Associate Professor,
Authority on All Things
Raspberry Pi

- Memory: GBs
- Flash: GBs
- CPU: GHz
Tiny devices
- Memory: 10s KB
- Flash: 100s KB
- CPU: 10s MHz
Middle Devices
- Memory: 100s Kb
- Flash: 1Mb
- CPU: 10-100 MHz
How Does TLS?
Network Layers

- Application
- HTTP
- TCP
- IP
- Hardware
- Physical

- Application
- HTTP
- TCP
- IP
- Hardware
- Physical
Network Layers + TLS

Application
HTTPs
TLS
TCP
IP
Hardware
Physical
TLS Handshake

Client Hello
- Client Random
- Cipher Suites

Server Hello
- Server Random
- Cipher Suite
  - Server Cert
  - Server Key Exchange
  - Server Hello Done

Client Key Exchange

Change Cipher

Encrypted Handshake

Encrypted Handshake
Handshake Requirements

- Ciphersuite agreement
- Verification of certificate, **not optional**

“TLS done incorrectly is worse than not using it at all. At least with no TLS you know that the communication is insecure.” — hallway talk at ICMC18
Implementation Requirements

- Memory
- Time
- Randomness
Traditional TLS API

- Init library
  - ctr_drbg
  - ssl
  - ssl_config
  - x509_crt
- open socket
- set bio
- tls_handshake
- tx
- rx
- application
  - tls_read
  - tls_write
Improving Layering

- **Stream abstraction**
  - Common in higher level languages
  - Same API for TLS and non-TLS

- **Put under Socket API**
  - Not really done in Linux (really, not done in Linux)
  - Keeps same API
  - The layering is wrong, though
Zephyr’s Approach

● Second approach
● Already offloading support, including one that has TLS
● Abstractions are “scary”
API Mismatch

Socket API
- connect
- send
- receive
- ...

Network receive wakes all waiters

- tls read
  - would-block
  - available
  - return

- tls write
  - would-block
  - available
  - return

wait for ...

wait for ...
Where are we now?

- Video of a demo?
- Zephyr network API changes
- JWT, time, MQTT