ARM mbed Security Solution

Eric Yang
Staff Application Engineer
ARM Holdings Ltd

July 19 - 21, 2016
ARM mbed Overview
**ARM mbed Products**

**mbed OS**  
Software and Tools platform for IoT devices, pre-integrated with mbed Cloud

**mbed Client**  
Support for mbed Cloud across different platforms and operating systems

**mbed Enabled**  
Interoperability for hardware and tools that can be trusted

**End to end security**

**Key Technologies**
- Thread
- BLE
- 6LoWPAN

**mbed Cloud**  
Secure, scalable, efficient device management services  
COAP, HTTP, REST

**Connect**  
Global IoT connectivity and management

**Provision**  
Secure management of device assets

**Update**  
Cost-effective device update and maintenance

© ARM 2015
Development Focus

Accessibility
High-level MCU APIs
Instant Access Tools

Scalability
Platform Portability
Developer Collaboration

Technology
Mesh Connectivity
IoT Security

Platform
Software Integration
Device Services

*m In many cultures 4 is unlucky. 2+3=5!*
Faster time to market for secure solutions
Trust Enables Business

- Embeding trust into system design is essential in facilitating future business

- IoT will not scale without trust and security
Trusted Device Services Enable Data Analytics

- **Device Onboarding**: Easy intuitive self-install and registration
- **Device Configuration**: Match configurations to service subscriptions
- **Device Monitoring**: Real-time visibility into the status of all connected devices
- **Remote Connectivity**: Apps with secure, real-time remote access
- **Security Analytics**: Detecting and responding to anomalous behavior
- **Firmware Updates**: Policy driven firmware manager
Internet Protocol to the Edge

- Internet Security
  - Widely deployed and proven

- Modern 32-bit MCUs easily handle IP stacks

- Non-standard approaches are a risk
  - Repeat past mistakes

---

**IoT Application**
- Binary | JSON – IPSO Objects
  - REST APIs
  - CoAP | HTTP
  - DTLS | TLS
  - UDP | TCP
  - IPv4 | IPv6 | 6LoWPAN

---

**Device Management**

---

© ARM 2015
mbed OS: Architected Security from the Ground Up

**Lifecycle Security**
mbed™ OS Secure Identity, Config and Update

**Communication Security**
mbed TLS

**Secure Code Compartments**
mbed OS uVisor on ARMv7-M MPU
Lifecycle Security

Communication Security

Device Security

mbed Device Server
- Web Interface
- Lifecycle Management
- mbed TLS
- mbed Device Interface

mbed OS
- IP Stack
- mbed TLS
- Device Management: L2M2M
- uVisor: Secure Device Services
## Communication Security: mbed TLS

- Leveraging established standards, best practice
- Fully-fledged SSL / TLS / DTLS Library
- Developer friendly: Clean API and documentation
- High assurance: Extensive testing
- Practical: Flexible, small footprint

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TLS/DTLS, etc</td>
<td>AES, etc</td>
<td>ECDHE, ECDSA, etc</td>
<td>SHA, etc</td>
<td>Entropy pool, CTR_DEBUG, etc</td>
<td>✔</td>
</tr>
</tbody>
</table>

### mbed Device Server
- Web Interface
- Lifecycle Management
- mbed TLS
- mbed Device Interface

### mbed OS
- IP Stack
- mbed TLS
- Device Management: L2M2M
- uVisor: Secure Device Services
Traditional MCU Flat Security Model

- IoT devices include significant software complexity
- With flat security all code/data lives in a shared address space
- Bugs in any code can lead to a security flaw
- Code base is too large for exhaustive validation
- If unauthorized code updates Flash then it may become impossible to remotely recover a device
Device Security: Secure Partitioning for MCUs

- Split memory into private critical and public uncritical
- Small private footprint enables exhaustive verification
- Public code never sees keys/secrets

- Vulnerabilities on public side can’t affect private side
- Private side can verify integrity of the public side
- Public code can’t write code directly to Flash
- Private side can reliably recover device to clean state
Device Security Services in mbed OS

- **mbed uVisor**
  - The ARM Cortex-M CPU Architecture enforces access privileges
  - The Memory Protection Unit (MPU) enforces CPU memory partitioning
  - HAL enforces partitioning for other bus masters

- **Secure Storage**
  - Identity; Keys; Certificates

- **Cryptographic Services**
  - TRNG; Any code that needs to access a key directly

- **Agile Security**
  - FOTA

- **Lifecycle Security**
  - Onboarding; Configuration; Updates; Analytics

Security Foundation
- Cryptography
- Key Management
- Secure Identity
- ...

Remainder of mbed OS
- Scheduler
- HAL + Drivers
- Connectivity stack(s)
- ...

Isolated

Uncritical

mbed uVisor
The uVisor Boot Process on the ARMv7-M Architecture

- uVisor initialized first in boot process
  - Private stack and data sections
  - Private data sections in flash for storing secrets
- Relocation of interrupts vector table into secure memory
- Initialization of memory protection unit based on box ACL's
  - Whitelist approach – only necessary peripherals are accessible to each box
  - Each box has private .bss data and stack sections
- De-privilege execution, continue boot unprivileged to initialize C/C++ libraries
The uVisor memory model

- uVisor allocates protected per-box stacks and detects under-/overflows during operation
- Main box memory accessible to all boxes
- All remaining per-Box data sections are protected by default:
  - Secure Per-Box Context Memory
  - Shared data/peripherals with other boxes on demand
- uVisor resolves ACLs during boot and identifies ACL collisions
- uVisor code sections visible to everybody
- Empty flash memory is made available to the system as configuration storage – write access only through configuration API
Lifecycle Security and OMA LWM2M

Device Identity and config. for Bootstrap

Assign devices to a particular owner and service vendor

Registered with service and ready to use

Device Support (for a specific IoT service vendor)

Device requests credentials and configuration for service

Manufacturing Support

Bootstrap Support

Device requests credentials and configuration for service

Factory Distribution

Delivery Setup

Usage Update Reprovision?

Reprovision?

Can choose to reassign device to new service or owner
Lifecycle Security: Enabled with mbed Device Server

- **Security Administration**
  - Provides management of security material via a User Interface or API
  - Population of endpoint X.509 certificates
  - Management of black list

- **Web Interface Security**
  - TLS-based HTTPs transport with certificates

- **Device Interface Security**
  - DTLS-based CoAP transport
  - X.509 certificate mode (TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8)
Summary

- IoT deployments will not scale without trust

- Very few developers have strong security experience

- mbed IoT Device Platform provide a comprehensive security foundation
  - Device Security
  - Communications Security
  - Lifecycle Security

- ARM are adopting, leading and supporting the evolution of internet security standards for IoT
Secure IoT: Example Use Cases
Thank you
Case Study: TLS Protocol

- We started with PolarSSL
- Which became mbed TLS on mbed OS
- Which we split up into:
  - TLS
  - X.509
  - Crypto
- Which we can now split into a critical and exposed part: Low Attack Surface
- Authentication and encryption keys are protected against malware
- Malware can’t interfere without knowing the encryption or signing keys
Case Study: Secure Server Communication

- Trusted messages contain commands, data or firmware parts
- Box security not affected by communication stack exploits or infections outside of trusted box
- Payload delivery is agnostic of protocol stack
- Resilient box communication over the available channels
  - Ethernet, CAN-Bus, USB, Serial
  - Bluetooth, Wi-Fi, ZigBee, 6LoWPAN

IoT Device owned by user. Initial identity provisioned by System Integrator
Messages delivered agnostic of communication stack
Case Study: Secure Remote Firmware Updates

- Firmware manifest format augments existing firmware formats with safety and security features
  - Downgrade attacks are prevented
  - Devices or device classes directly addressed by updates to improve safety
  - Efficient firmware distribution in mesh networks
  - Support for partial & chained firmware updates
- New firmware applied after all blocks are received, decrypted and verified
- Trusted side also updates itself using a boot loader