Post-Quantum

Cryptography Conference

Post-Quantum Firmware Signing in IoT: Practical PQC-FOTA Implementation

Tan Wai Kaey
IoT & Cybersecurity Engineer

KEYFACTOR CRYPTO4A SSL.com ENTRUST HID

October 28 - 30, 2025 - Kuala Lumpur, Malaysia

PKI Consortium

Post-Quantum Firmware Signing in IoT:

Practical PQC-FOTA Implementation

A practical implementation of PQC-FOTA for secure, future-proof firmware in IoT devices against evolving quantum threats.

Crypto-agility

Why do we need Firmware Signing in IoT?

With the emergence of **quantum threats**, traditional signing methods like RSA and ECDSA pose significant risks.

Agenda

- IoT Constraints
- PQC Integration
- Bootloader Verification
- Hybrid Signing
- PQC Migration Roadmap

IoT Constraints

Classical Cryptographic Algorithms

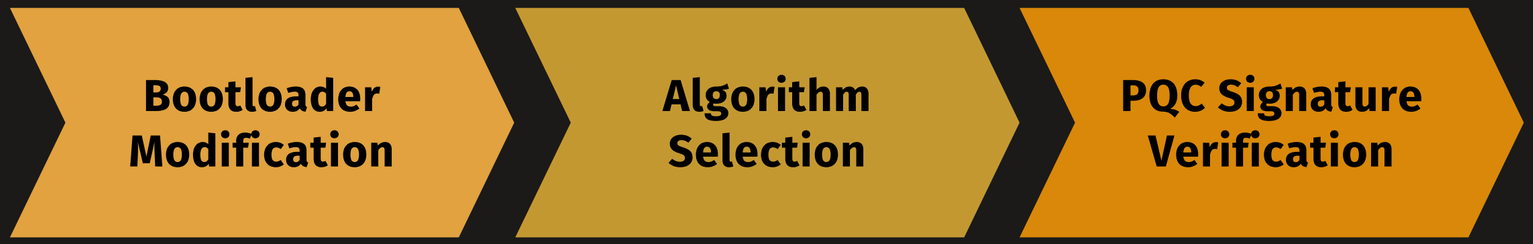
FOTA Compatibility

Resource & Performance Limits

PQC Integration

Firmware Signing [Python]

ESP32 Secure Boot Verification [C]



Bootloader Verification [C]

```c
#include <oqs/oqs.h>

OQS_STATUS status = OQS_SIG_verify(
    "ML-DSA-65",
    firmware_data,
    firmware_len,
    signature,
    sig_len,
    public_key
);

if (status == OQS_SUCCESS) {
    printf("PQC verification passed\n");
} else {
    printf("Verification failed\n");
}
```

OQS C API:

OQS_SIG_verify()

Demo

```
┌──(kali㉿kali)-[~/esp-idf/tools]
└─$ # Generate with default (ML-DSA-44)
python3 sign_firmware_pqc.py gen

# Generate with ML-DSA-65
python3 sign_firmware_pqc.py gen ML-DSA-65

# Sign firmware.bin with ML-DSA-65
python3 sign_firmware_pqc.py sign ML-DSA-65 firmware.bin firmware.sig

# Verify signature
python3 sign_firmware_pqc.py verify ML-DSA-65 firmware.bin firmware.sig

[+] Generated ML-DSA-44 keypair
    Private key: secure_boot_signing_key_ML-DSA-44.pkl
    Public key : signature_verification_key_ML-DSA-44.pkl
[+] Generated ML-DSA-65 keypair
    Private key: secure_boot_signing_key_ML-DSA-65.pkl
    Public key : signature_verification_key_ML-DSA-65.pkl
[+] Signed firmware.bin with ML-DSA-65, signature written to firmware.sig
[+] Verification SUCCESS for firmware.bin using ML-DSA-65
```

```
┌──(kali㉿kali)-[~/esp-idf/tools]
└─$ echo "tampered" >> firmware.bin
python3 sign_firmware_pqc.py verify Falcon-512 firmware.bin firmware.sig

[-] Verification FAILED for firmware.bin using Falcon-512
```

Hybrid Signing (ECDSA + MLDSA)

Header layout example:

Verification Policy:

1. **Verify both (strict mode)**
2. **Accept either (compatibility mode)**

PQC Integration Roadmap

Hybrid Signing Transition

Start with ECDSA + PQC signatures for compatibility with existing systems.

PQC-Only Secure Boot & FOTA

Move fully to PQC signatures for firmware verification and updates.

Full PQC Adoption

Expand PQC to mutual authentication and TLS stack

Q&A