LeafSense Scouts

Robot-to-Robot Secure Communication

Tomiwa Adeyemi

6ENT1180 Robot Communications
University of Hertfordshire

Theme: Agricultural Disease Surveillance

Grounded in my FYP on MobileNetV2 plant disease classification

- Two scout robots patrol a crop field
- Field Scout detects disease → transmits events
- Base Scout logs to EEPROM → sends encrypted ACK
- 3-byte core payload (zone, disease, severity)
- 256-zone structured field map in 8 KB EEPROM
- AES-encrypted Zigbee radio link (XCTU EE=1)

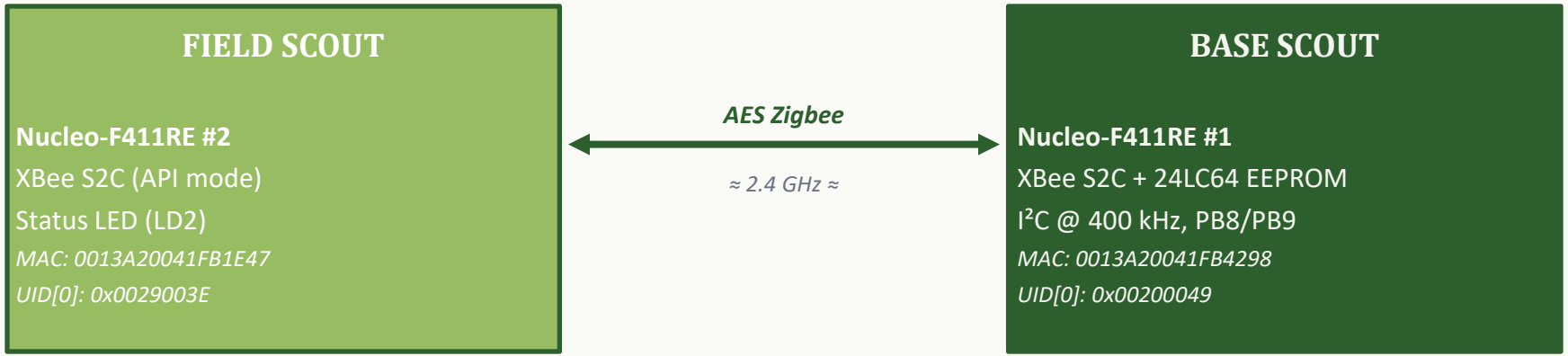
FYP Connection

MobileNetV2 classifier outputs the disease ID and severity bytes used in this protocol. In deployment the Field Scout would run the pruned model on-device, pushing results to the Base over encrypted Zigbee.

2	10	4
Nucleo-F411RE	zones demo'd	security layers

System Architecture

Two-node R2R over Zigbee API mode, four-layer security stack



Protocol Stack (top-down)

App: LeafSense payload + AES-128 + nonce + UID check
Zigbee API frames (0x10 TX / 0x90 RX, hand-rolled)
XBee AES-128 radio encryption (XCTU EE=1, EO=2)
IEEE 802.15.4 physical radio

Security — Four Defence Layers

Defence in depth: each layer addresses a different threat

Layer	Mechanism	Defends Against
1	Zigbee AES-128 (XCTU EE=1, EO=2)	Passive radio eavesdropping
2	App-layer AES-128 ECB + 2-byte nonce	Defence-in-depth if Zigbee key extracted
3	UID allow-list (STM32 96-bit factory UID)	Cloned Nucleo with stolen firmware
4	Monotonic nonce tracking per peer	Replay attacks (captured-frame rebroadcast)

Why ECB is acceptable here: *each block contains a unique nonce, so no two plaintext blocks are ever identical — neutralising ECB's usual correlation weakness. The 10 zero-padding bytes act as an implicit integrity tag (decrypt to non-zero → reject).*

Security Demo — UID Rejection

Negative test: impostor UID blocked even with valid encryption

Scenario

Changed **`FIELD_SCOUT_UID0`** to **`0xDEADBEEF`**

Field Scout continues transmitting valid encrypted frames with its real UID.

Every frame rejected by Base at the identity check:

COM7 (Base)

```
[BASE] AUTH FAIL:
  decryption/integrity
  check rejected
[BASE] AUTH FAIL:
  decryption/integrity
  check rejected
[BASE] AUTH FAIL:
  decryption/integrity
  check rejected
```

✓ No events logged. No ACKs sent. LED enters AUTH_FAIL rapid-blink pattern. *Screenshot saved:* `security_layer3_uid_rejection_proof.png`

EEPROM — Structured 8 KB Field Map

256 pages × 32 B. Page 0 = config. Pages 1-255 = one zone each.

Address	Size	Purpose
`0x0000 – 0x001F`	`32 B`	Config page (magic, schema, event count, UID)
`0x0020 – 0x1FFF`	`32 B × 255`	Zone pages (one per field zone)

Per-zone 32-byte layout:

- Latest disease ID + severity (2 B)
- Event count for this zone (2 B)
- Max severity ever seen (1 B)
- Sender MAC low byte (1 B)
- Ring buffer: last 13 detections (26 B)

Design Note

Each zone lives in one 32-byte page — so writes never cross page boundaries. This avoids the 24LC64's page-wrap bug where writes past a boundary wrap to the page start rather than advancing.

Reflection

What worked, where I struggled, what I learned

Worked Well

- Bidirectional R2R with hand-rolled Zigbee API frames
- Receive state-machine parser (robust byte-level)
- Four-layer security with negative-test proof
- Non-blocking LED state machine (6 patterns)
- 18-source Harvard literature base

Challenges

- EEPROM persistence — writes silently dropped
- Diagnosed via hex-dump of page 0 on boot
- Root cause: WP path (resistor or chip)
- No spare components to isolate component
- In-session logging still correct

Next Steps

- Persistent nonce in EEPROM (once fixed)
- STM32 RDP Level 2 — prevent JTAG key read
- Hardware Secure Element for key storage
- Retry + forward-error correction on ACK
- Field trial with real MobileNetV2 classifier

Key Resources

Technical foundations and domain literature

Datasheet	**Digi XBee S2C User Guide (90002002)** *API frame structure (p. 171), checksum formula (p. 197)*
Datasheet	**Microchip 24LC64 I²C Serial EEPROM Datasheet** *Page write limits, WP pin behaviour, I²C timing*
Standard	**NIST FIPS 197 — Advanced Encryption Standard** *AES-128 specification and approved use*
Reference	**STM UM1724 — Nucleo-F411RE user manual (Rev 14)** *Arduino header pinout mapping (D-pins ↔ PORT/PIN)*
Paper	**Kamilaris & Prenafeta-Boldú (2018) — Deep learning in agriculture** *Grounds the disease-surveillance theme in published research*

.

Thank you

Submission package

- Source files (.c.txt + .h.txt for each module)
- Security design document (threat model + layers)

Tomiwa Adeyemi • 6ENT1180