RIPE 88, IoT WG

22/05/2024

UNIVERSITY of York

Investigating Security Vulnerability in IoT Thread Network

Dr Poonam Yadav

poonam.yadav@york.ac.uk

SYSTRONLab

UKRI Engineering and Physical Sciences Research Council

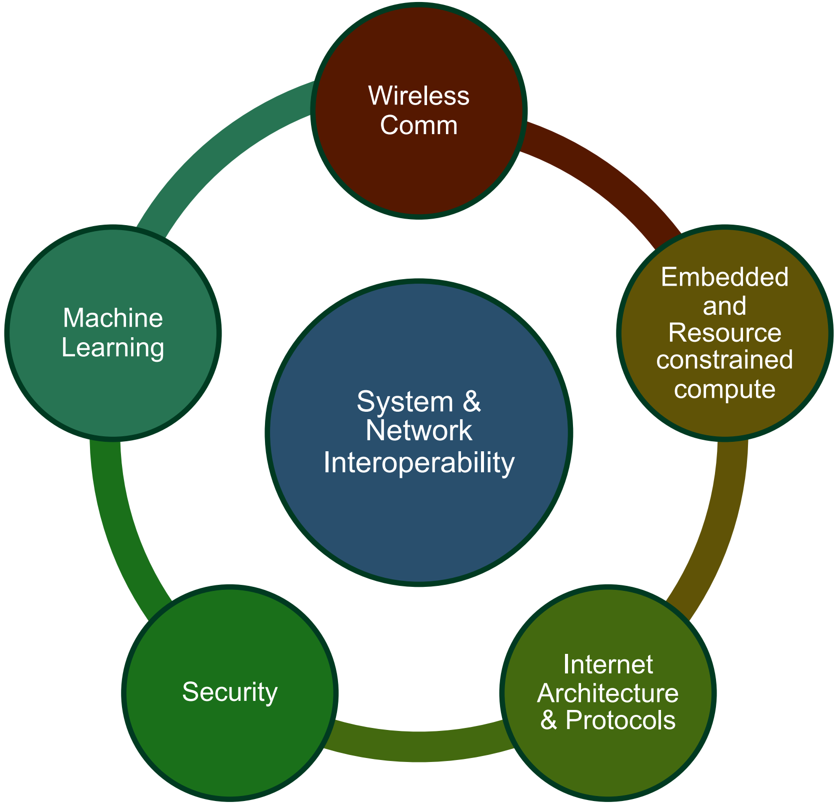
Department for Science, Innovation, & Technology

REMOTE Resilient Secure TinyEdge REACH

CHEDDAR

SYSTRON Lab

We are the System and Network Interoperability (SYSTRON) Lab, exploring distributed systems, interoperability, and network technologies.



SafetyNet Project

- Building Secure and Resilient IoT networks, specifically focused on Home IoT ecosystem
- Investigating vulnerability in protocols (Zigbee, Thread, Bluetooth and other networks)
- Investigating IoT device fingerprinting, e.g., identifying IoT devices uniquely or similar category of IoT devices in the network (without using device IP addresses)
  - IoT device behavioral fingerprint using their network traffic and ML
    a. The network traffic features used for training ML models can come from different layers of TCP/IP stack.
  - IoT device Physical Unclonable Function (PUF) based fingerprinting
- Detecting IoT device abnormal network behavioral change using the ML model (fingerprints). The abnormal network behavioral change can happen due to DoS or similar attacks or any other faults. Different fingerprints can be used for detecting different attacks and faults.

SafetyNet Project – Contd.

Prevention of attacks and Privacy leakage

Secure by design

--- Reducing attack surface by deploying MUD (IETF 8520). We investigates efficiency of MUD ecosystem (rule enforcement - IP table vs eBPF program), MUD extension, MUD user interface for accountability and transparency

--- Building secure device/user authentication, authorization and access control mechanisms, data/message encryption

--- Proactive traffic filtering and blocking through firewalls.

Privacy by design

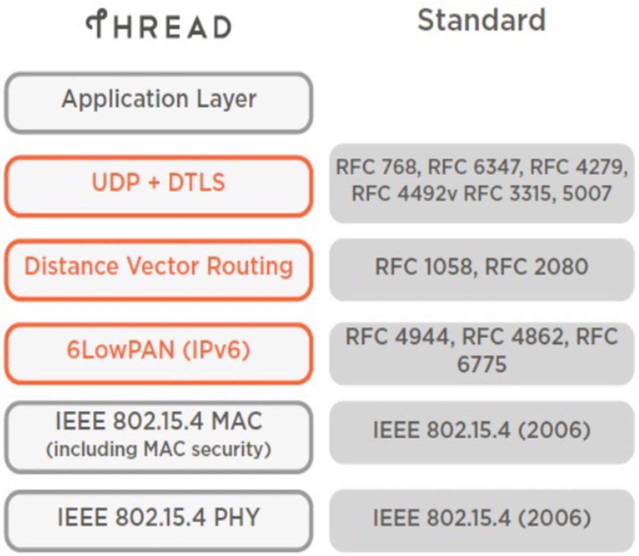
--- User data encryption, anonymization, differential privacy, multi-factor authorization, Secure multi-party computation, Federated Learning

Accountability and transparency

--- Providing Users' control over their devices by providing more transparency and tools.

Thread Network

- On July, 2014, the Thread group was launched with just one aim in mind: to provide the best method for connecting and controlling gadgets in the house and buildings.
- Thread is an IPv6-based mesh networking protocol developed by industry-leading technology companies for connecting products around the home and in buildings to each other, to the internet and to the cloud.
- The Thread stack is an open standard that is built upon a collection of existing Institute for Electrical and Electronics Engineers (IEEE) and Internet Engineering Task Force (IETF) standards, rather than a whole new standard.
- Thread networks are simple to install, highly secure, scalable to hundreds of devices and developed to run on low-power IEEE 802.15.4 chipsets.

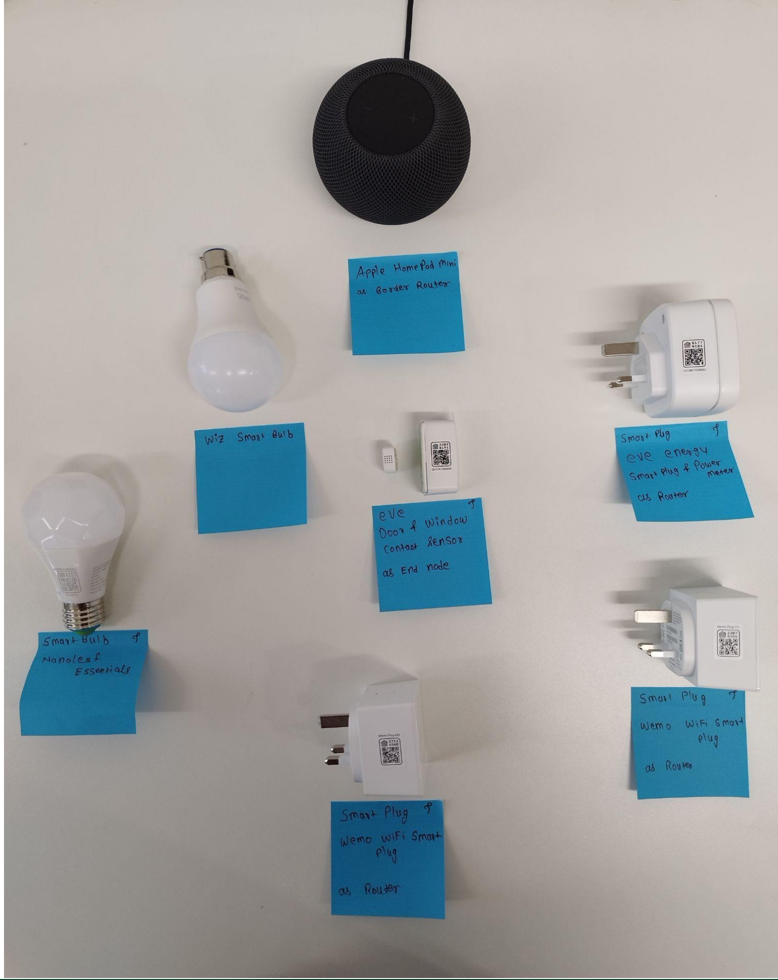


https://www.silabs.com/documents/public/user-guides/ug103-11-fundamentals-thread.pdf

Thread General Characteristics

- Simple network installation, start-up, and operation
- Secure – authorisation and encryption at network/application layer
- Small and large home networks
- Large commercial networks
- Bi-directional service discovery and connectivity
- Range
- No single point of failure
- Low power
- Cost-effective

Ref: https://www.silabs.com/documents/public/user-guides/ug103-11-fundamentals-thread.pdf

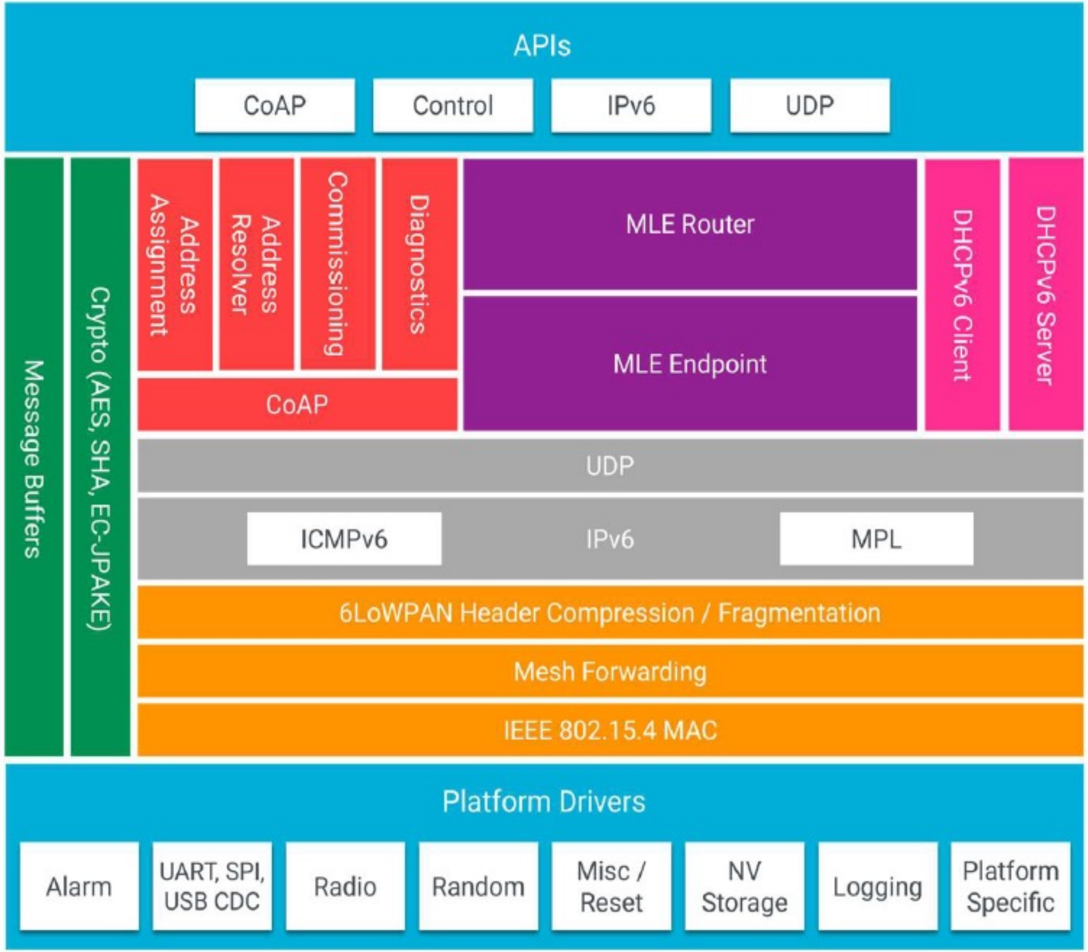


Open Thread

- OpenThread released by Google is an open-source implementation of Thread®. Google has released OpenThread to make the networking technology used in Google Nest products more broadly available to developers, in order to accelerate the development of products for the connected home and commercial buildings

- OpenThread implements all Thread networking layers (IPv6, 6LoWPAN, IEEE 802.15.4 with MAC security, Mesh Link Establishment, Mesh Routing) and device roles, as well as Border Router support.

Ref: https://openthread.io/



IEEE802.15.4 Standard

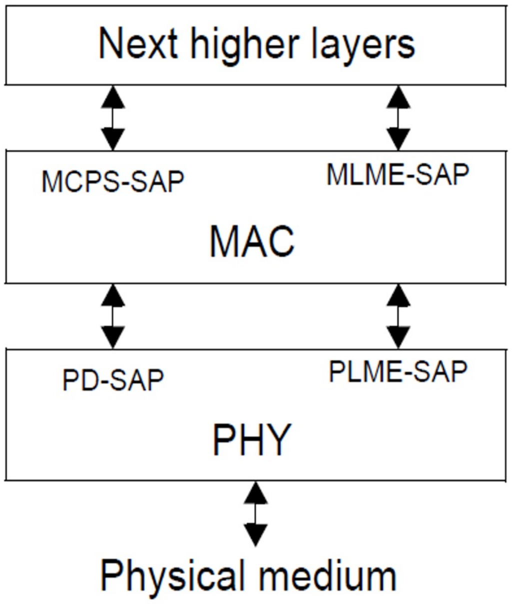
-- A low-rate wireless personal area network (LR-WPAN)

-- Allows wireless connectivity in applications with limited power and relaxed throughput requirements.

-- Ease of installation, reliable data transfer, low cost implementation.

-- A device has a single radio interface that implements an IEEE Std 802.15.4 MAC and PHY.

-- Thread protocol stack implements IEEE802.15.4 for its PHY and MAC layers.



LR-WPAN device architecture

IEEE802.15.4 PHY Features

PHY layer responsibilities include:

- modulation and demodulation,
- switching between transmitter and receiver,
- fragmentation,
- scrambling,
- interleaving, and
- error correction coding.

IEEE802.15.4 PHY Features

Bits: 0-2	3	4	5	6	7-9	10-11	12-13	14-15
Frame Type	Security Enabled	Frame Pending	Ack. Request	PAN ID Compression	Reserved	Dest. Addressing Mode	Frame Version	Source Addressing Mode

Frame Control Field (FCF) Structure

IEEE802.15.4 PHY Features

- When the Frame Control Field (FCF) is received, the driver checks if the length of the frame is valid, and it verifies the frame type and version.
- When the destination address fields (PAN ID and address) are present and received, the driver checks if the frame is destined to this node (broadcast or unicast).
- When the entire frame is received, the driver verifies if the FCS field contains a valid value.
- A received frame includes a timestamp captured when the last symbol of the frame is received. The timestamp can be used to support synchronous communication like CSL or TSCH.

If all checks are passed, the driver passes the received frame to the MAC layer.

Idle Sequence

Receive Sequence

Transmit Sequence

Continuous CCA

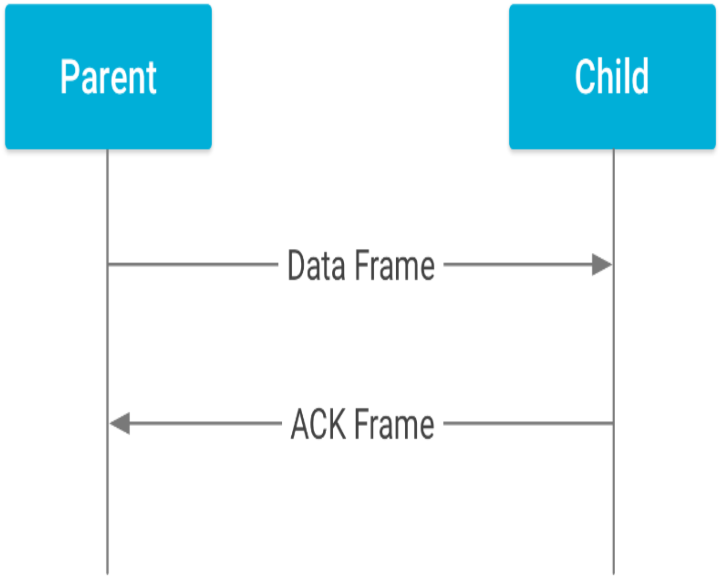
TR Sequence

IEEE802.15.4 PHY Features

Sending Automatically ACK frames

This automatically created ACK frame complies with IEEE 802.15.4-2006: 7.2.2.3 or IEEE 802.15.4-2015: 6.7.2 and 6.7.4.2. This frame is sent exactly 192 microseconds after a data frame is received.

The ACK frame is sent only if the received frame passes all the filter steps, even in promiscuous mode, and if the ACK request bit is present in the FCF of the received frame.



https://openthread.io/

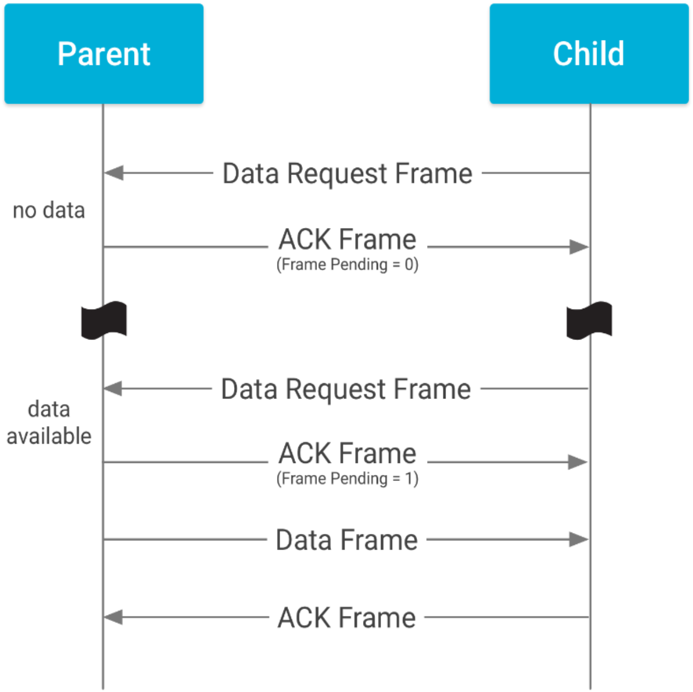
IEEE802.15.4 PHY Features

Sending Automatically ACK frames

The driver handles the pending bit as follows, depending on the protocol used:

Thread mode:

- If the driver matches the source address with an entry in the array, the pending bit is set (1).
- If the array does not contain an address matching the source address, the pending bit is cleared (0).



https://openthread.io/

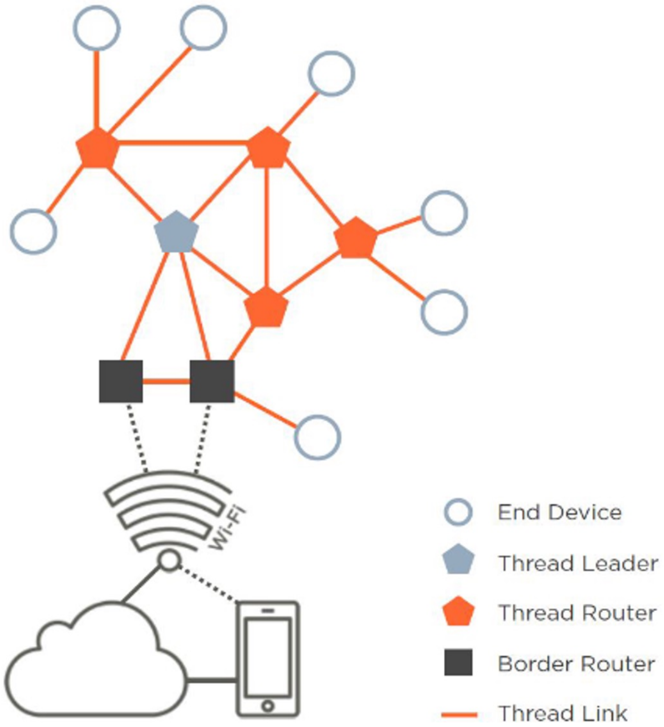
Thread Network Architecture

The Thread network is comprised of two types of Thread devices:

Full Thread Device (FTD)
- versatile, can act as network Leader, Router or End Device
- an FTD device can perform the role of Border Router, a gateway to other networks (Wi-Fi, Ethernet, etc)

Minimal Thread Device (MTD)
- least requirements on device power and resources (usually battery powered)
- normally configured to act as End Device only

The Thread's mesh networking topology makes the wireless system more reliable by enabling message forwarding between radio nodes.



Designed to avoid single point of failure

https://openthread.io/

IPv6 / 6LoWPAN

- 6LoWPAN provides a compression mechanism that reduces the IPv6 header sizes sent over the air and thus reduces transmission overhead. The fewer bits that are sent over the air, the less energy is consumed by the device. Thread makes full use of these mechanisms to efficiently transmit packets over the 802.15.4 network.

The two RFC's provide more details on how fragmentation and header compression are accomplished in 6LoWPAN

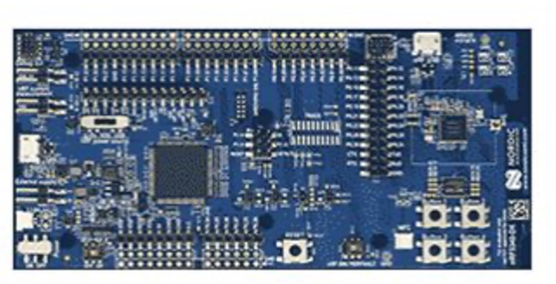
- RFC 4944 (https://tools.ietf.org/html/rfc4944) and
- RFC 6282 (https://tools.ietf.org/html/rfc6282)

Thread Network Vulnerability

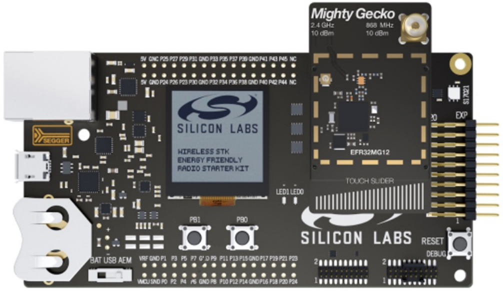
- Thread, like ZigBee and WirelessHart, uses a wireless radio system for networking by implementing the well-established IEEE802.15.4 protocol.
- Unliked wired systems, the network is vulnerable to radio jamming and RF interference. For example, a powerful radio signal (a simple unmodulated carrier wave) can overcome and interfere with the network.
- The IEEE802.15.4 PHY implements an algorithm (CSMA-CA) to detect channel RF energy, and hold-off pending data transmission until the channel is free, potentially stalling a Thread network.
- The MAC sub-layer in the IEEE802.15.4 architecture can be jammed with rogue data packet frames, reducing Thread network reliability and performance.

Thread Network Testbed

Thread-capable devices from Nordic Semiconductor and Silicon Laboratories were chosen to form the test Thread network. These companies combined currently occupy the majority market space for Thread chipsets.

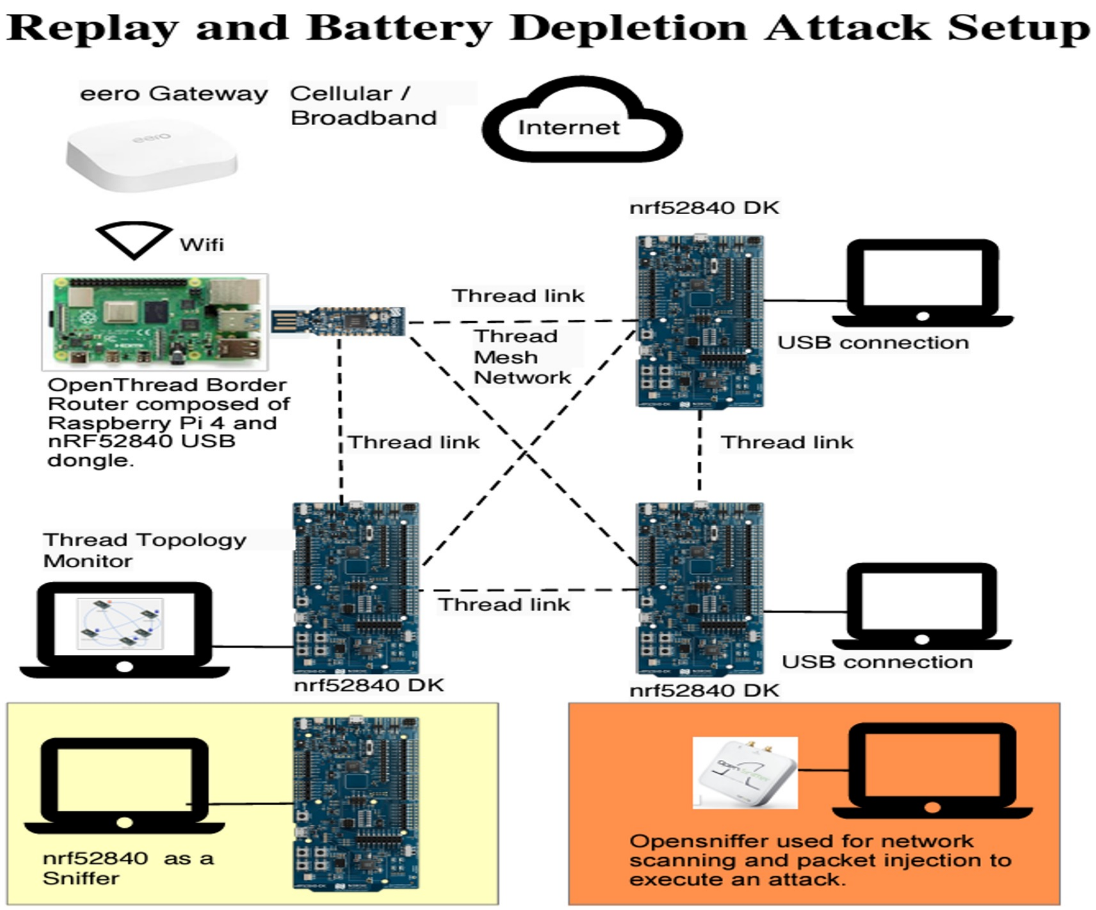


nRF5340DK (nRF5340 SoC)



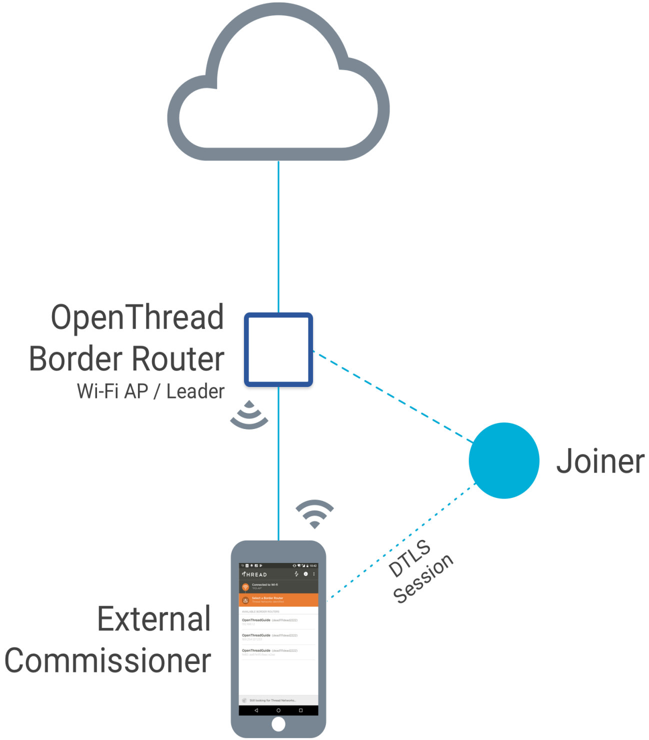
SLWSTK6006B (EFR32MG12 SoC)

IoT Thread Network Setup 1



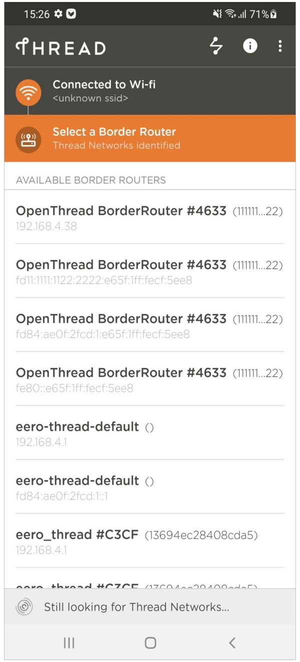
Poonam Yadav, Nirdesh Sagathia, Dan Wade:
Demo: Battery Depletion Attack Through Packet Injection on IoT Thread Mesh Network. IEEE COMSNETS 2024

IoT Thread Network New Device Commissioning

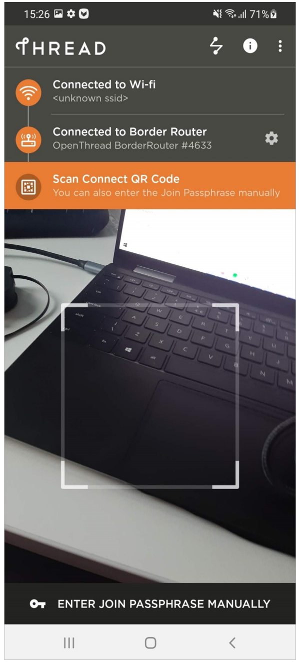


Poonam Yadav, Nirdesh Sagathia, Dan Wade:
Demo: Battery Depletion Attack Through Packet Injection on IoT Thread Mesh Network. IEEE COMSNETS 2024

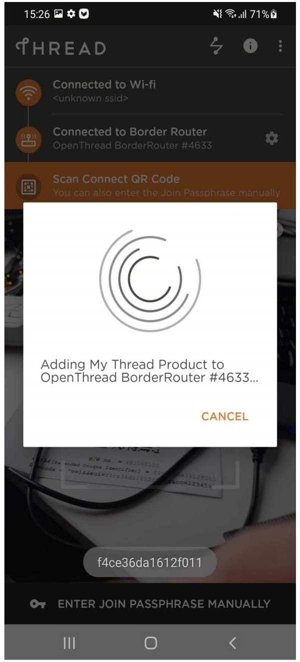
IoT Thread Network New Device Commissioning



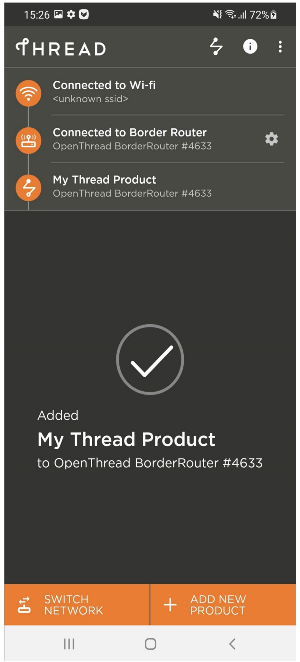
Step 1) Select the border router and enter the passphrase



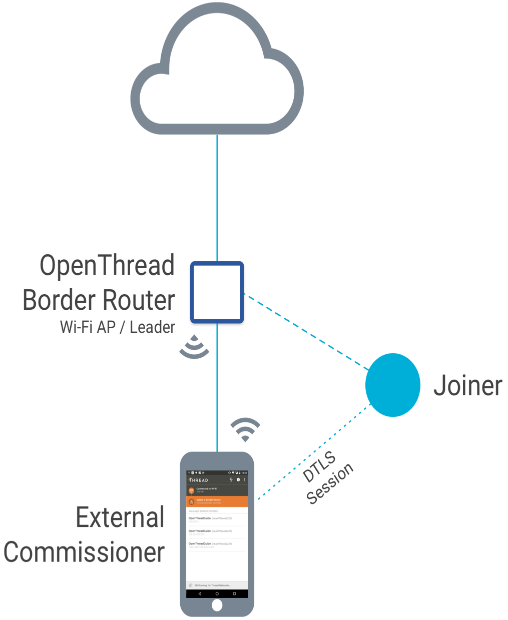
Step 2) Scan the QR code of the new device



Step 3) When this shows, run the joiner commands on the new device



Step 4) Wait for join process to complete and device is added



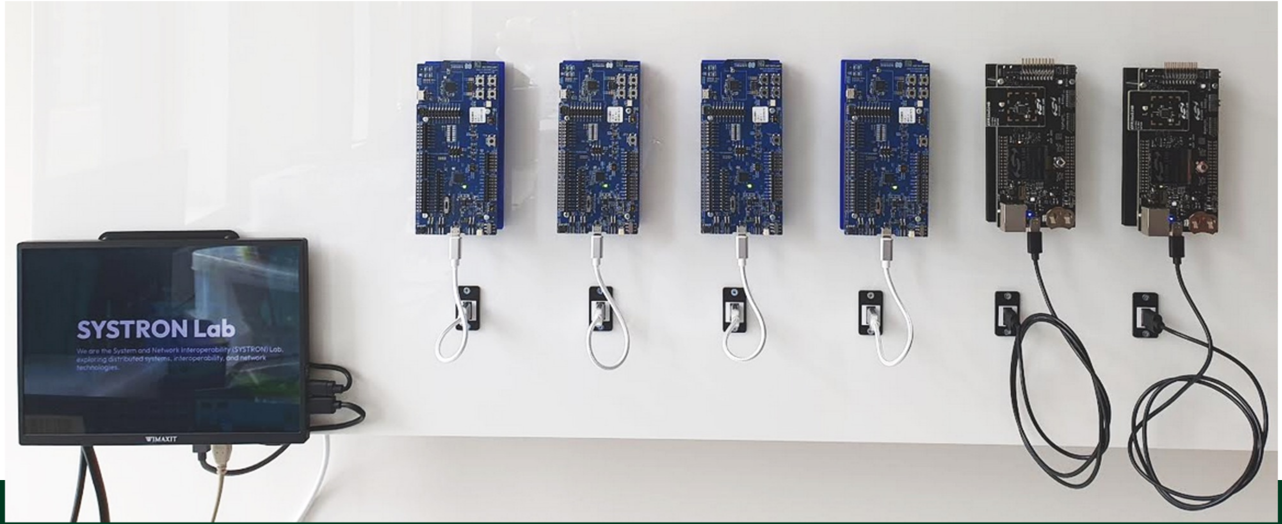
Poonam Yadav, Nirdesh Sagathia, Dan Wade:
Demo: Battery Depletion Attack Through Packet Injection on IoT Thread Mesh Network. IEEE COMSNETS 2024

Thread Network Testbed

Four nRF5340DK and two SLWSTK6006B boards comprise the Thread network. The devices are pre-commissioned, sharing a common network key.

An nRF5340DK (on left) is configured as FTD and acts a Thread Leader/Router. All the other boards are set as MTD End Devices.

The boards are attached (using special magnetic frames) to a dedicated Thread Edge Testbed designed and developed in the Department of Computer Science.



Thread Network Testbed

A Thead Topology Monitor (TTM) system developed by Nordic Semiconductor is used to display the Thread mesh network.

The TTM module itself is a Thread FTD device and joined to the Thread network. It periodically sends MLE messages on the network to maintain a live topological view.

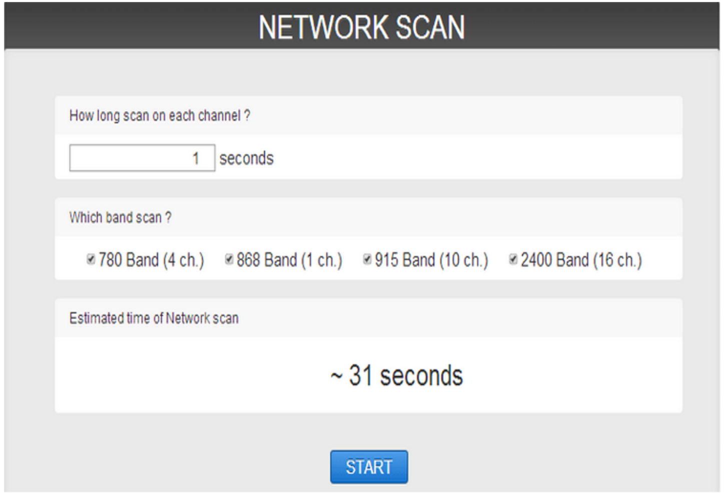
In the image opposite, an FTD is shown acting as Leader/Router and Parent to all Child devices (End Devices) and self-configured in a star topology.

Network DoS Attack Method

A Sewio Open Sniffer device is used to detect the Thread network's operating radio channel (channels 11 - 26)

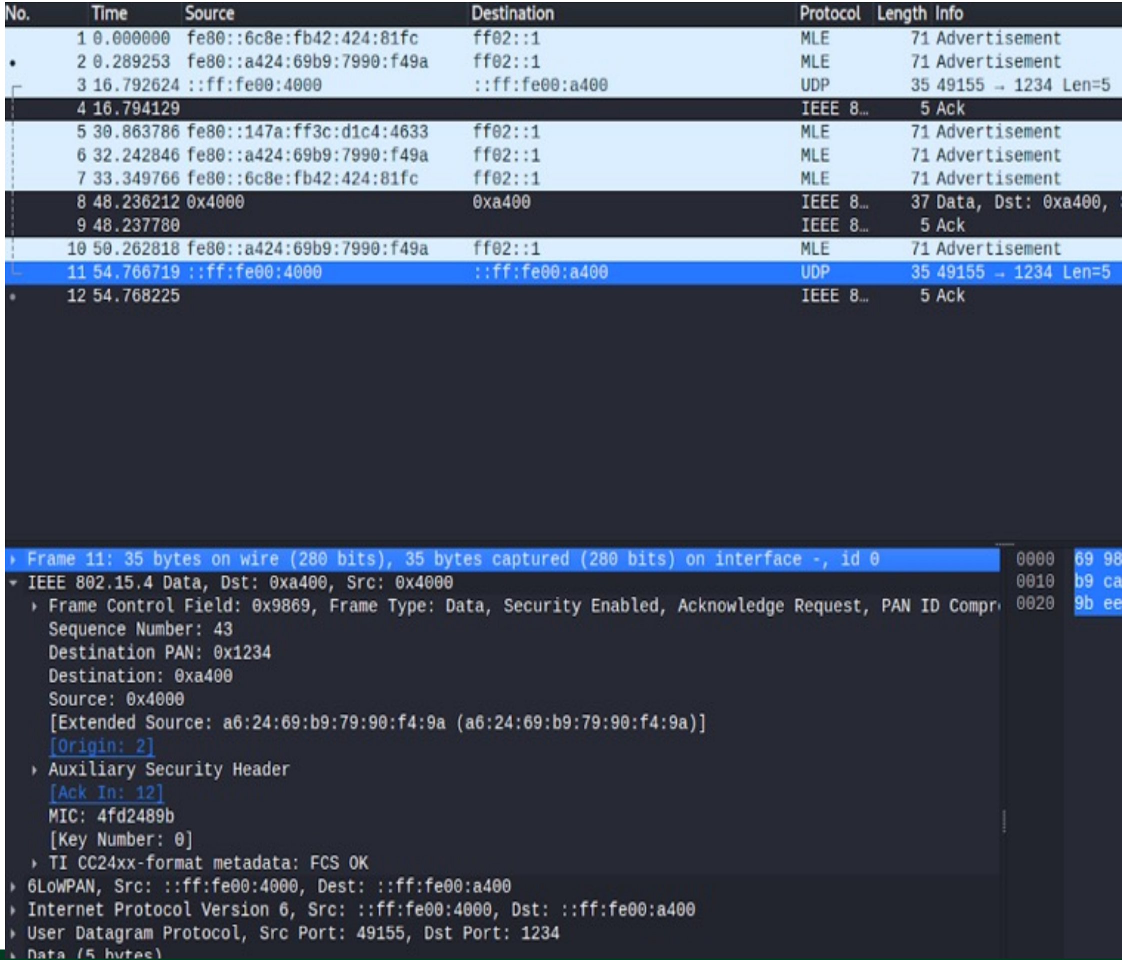
Its Network Scanner reports a Thread's PAN ID if a network is detected.

We only need to know the discovered network's channel number for the DoS Attack!



Replay Attack

In this instance, we witness the successful replay of the previously captured UDP packet into the network. The original packet is designated as packet number 3, the first replayed packet is 8, and the subsequent replayed packet, after the removal of the last two bytes, is identified as packet number 11 in the figure. Both of these packets are acknowledged, although, at the upper layer, they were not received.



Network DoS Attack Method

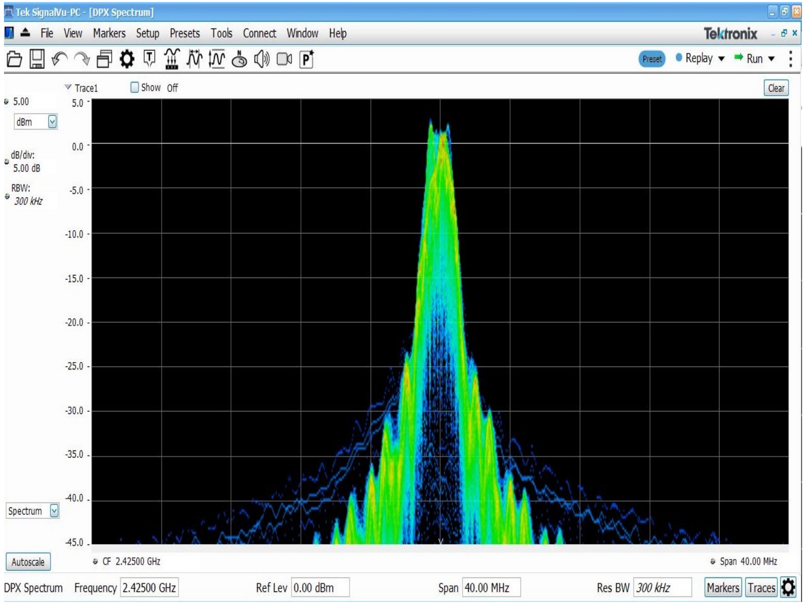
Having discovered the Thread network's channel number, the Open Sniffer is then used to inject IEEE802.15.4 packets, controlled via Ethernet using the Open Sniffer Python library.

- A single packet is repeatedly transmitted with an inter-frame spacing of 1 ms and with a transmit power of 1 mW (1m from the testbench).
- The packet contains a valid IEEE802.15.4 header plus a MAC frame and payload.
- The MAC frame's source and destination address fields are cleared, as is the payload.
- A minimum packet size (excluding header and DRC) of 32 symbols is transmitted:

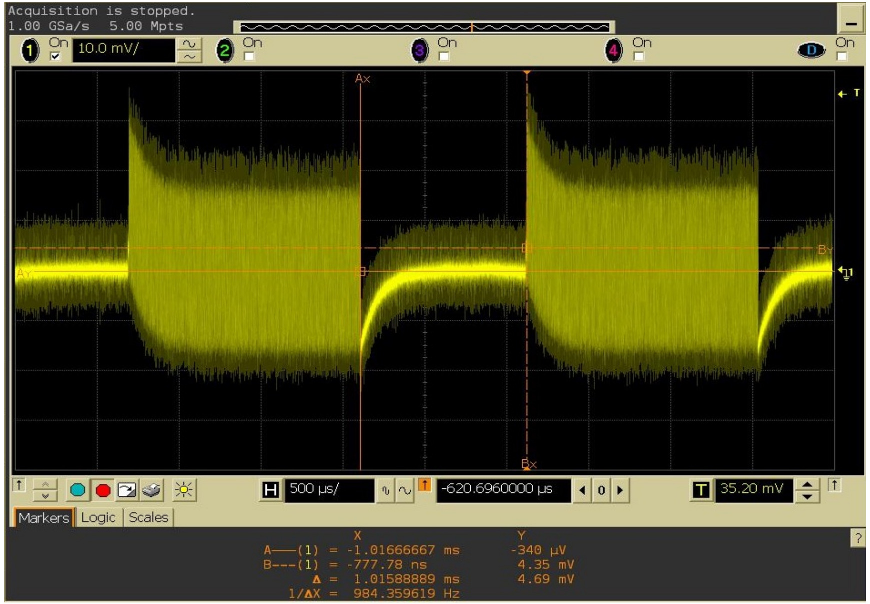
```
Packet: 69 98 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Network DoS Attack Method

A spectrum analyser measures the Open Sniffer to a peak RF transmit power of +3 dBm centred on channel 15



The transmitted repeating packet is shown to have a fixed Inter-Frame Spacing (IFS) of 1.0 ms



DoS Attack Results

The OpenThread CLI is used to control and analyse the Thread network. All Thread devices in the test are programmed with the CLI.

Three methods are used in the test to verify the network is unaffected/affected by the DoS Attack:-

1. Visually, by using the TTM tool

2. Issuing OpenThread CLI 'ping' commands between all Child devices:

```
> ot ping <target Child IPv6 address>
```

   Success indicated by 0% packet loss

3. Issuing OpenThread CLI 'discover' MLE discovery commands:

```
> ot discover <channel>
```

   A returned network name indicates success

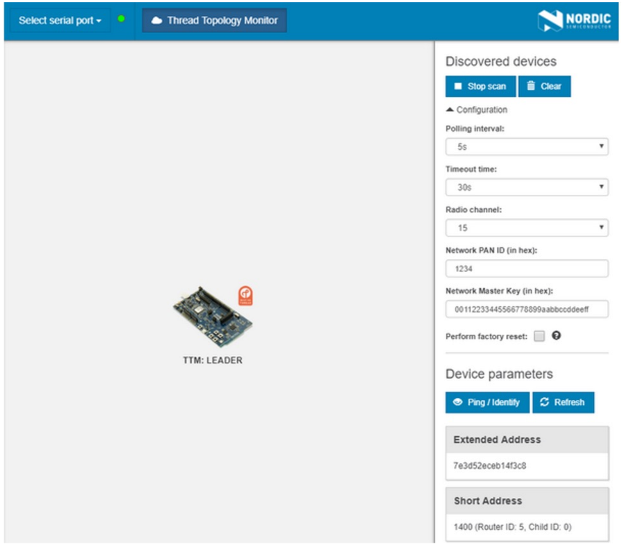
DoS Attack Result...

Injecting the repeating IEEE802.15.4 packet on the network using the Open Sniffer device causes the total collapse of the test Thread Network!!!

The TTM tool shows the loss of all device on the network; only the TTM FTD itself remains, promoting itself to Thread Leader.

All OT ping commands return 100% packet loss.

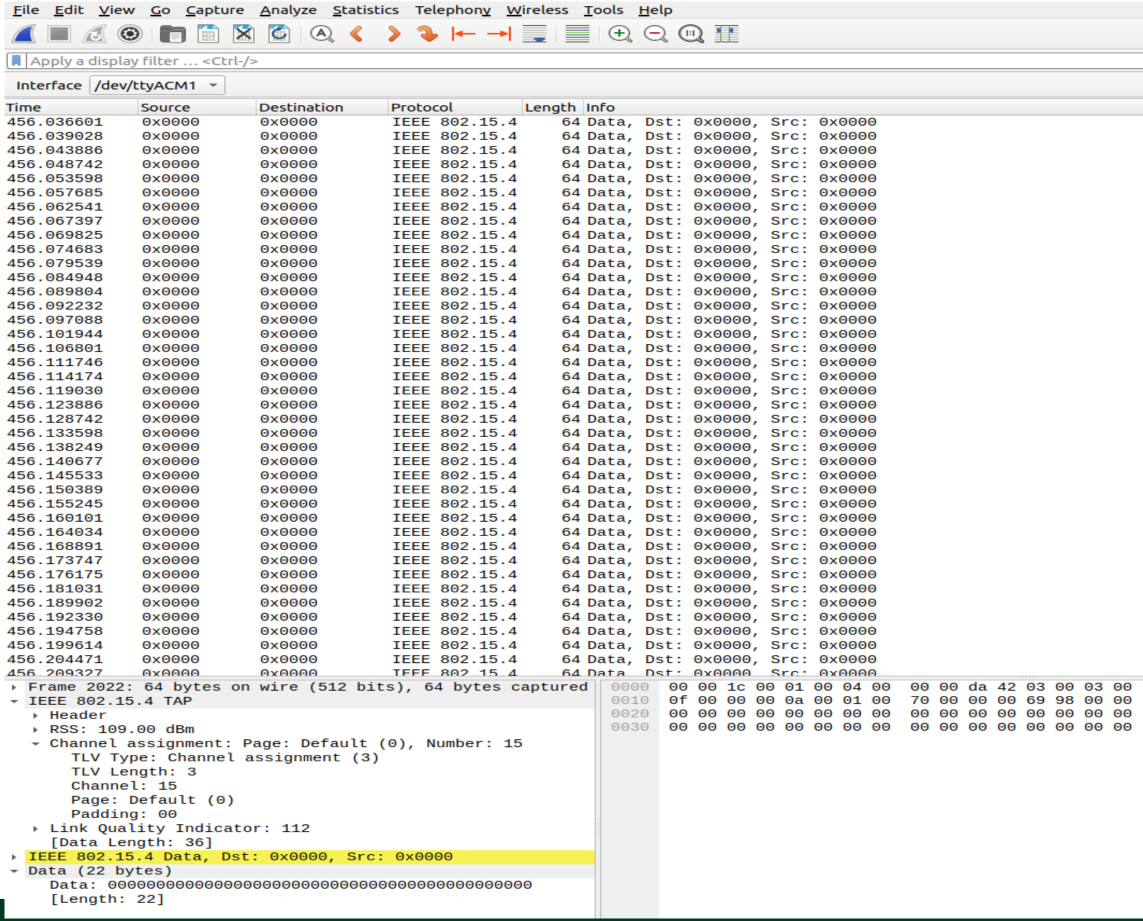
All OT discovery commands fail to return any discovered networks.



Total collapse of the Thread Network.

DoS Attack Results

Wireshark, with an IEEE802.15.4 Sniffer device as input source, shows the total dominance of the repeating attack packet on radio channel 15.



Further Work

- The precise mechanism causing the DoS Attack to succeed needs to be investigated and verified. Possible causes are IEEE802.15.4 MAC sub-layer jamming or insufficient IFS period (PHY/MAC turnaround time) in the Thread device.
- Investigate means to mitigate against the DoS Attack. Detection of the attack and response, i.e. suspend network until free?
- Look at the implications of the DoS Attack on battery powered Sleepy End Devices (SEDs), especially the problem of battery depletion (the SED wants to (re)join a network, consuming power).

Development GitHub: https://github.com/SystronLab/thread-edge-testbed

Previous version - DEMO: https://github.com/SystronLab/ThreadBatteryAttack

Poonam Yadav; Nirdesh Sagathia; Dan Wade, Demo: Battery Depletion Attack Through Packet Injection on IoT Thread Mesh Network, IEEE Comsnet'24.

Thanks For Listening
Any Questions?

Please reach out for the collaboration
poonam.yadav@york.ac.uk

Follow us on:

https://github.com/systronlab

https://systronlab.github.io/

SafetyNet Project Team

Dr Poonam Yadav
poonam.yadav@york.ac.uk

Anthony Moulds
anthony.moulds@york.ac.uk

Peter Gillingham
peter.gillingham@york.ac.uk

Previous Students

Nirdesh Sagathia (MSc, 2022)
Dan Wade (BSc, 2023)
Vijay Kumar (Intern, 2021)

UKRI Engineering and Physical Sciences Research Council
Department for Science, Innovation, & Technology
REMOTE Resilient Secure TinyEdge
REACH
CHEDDAR