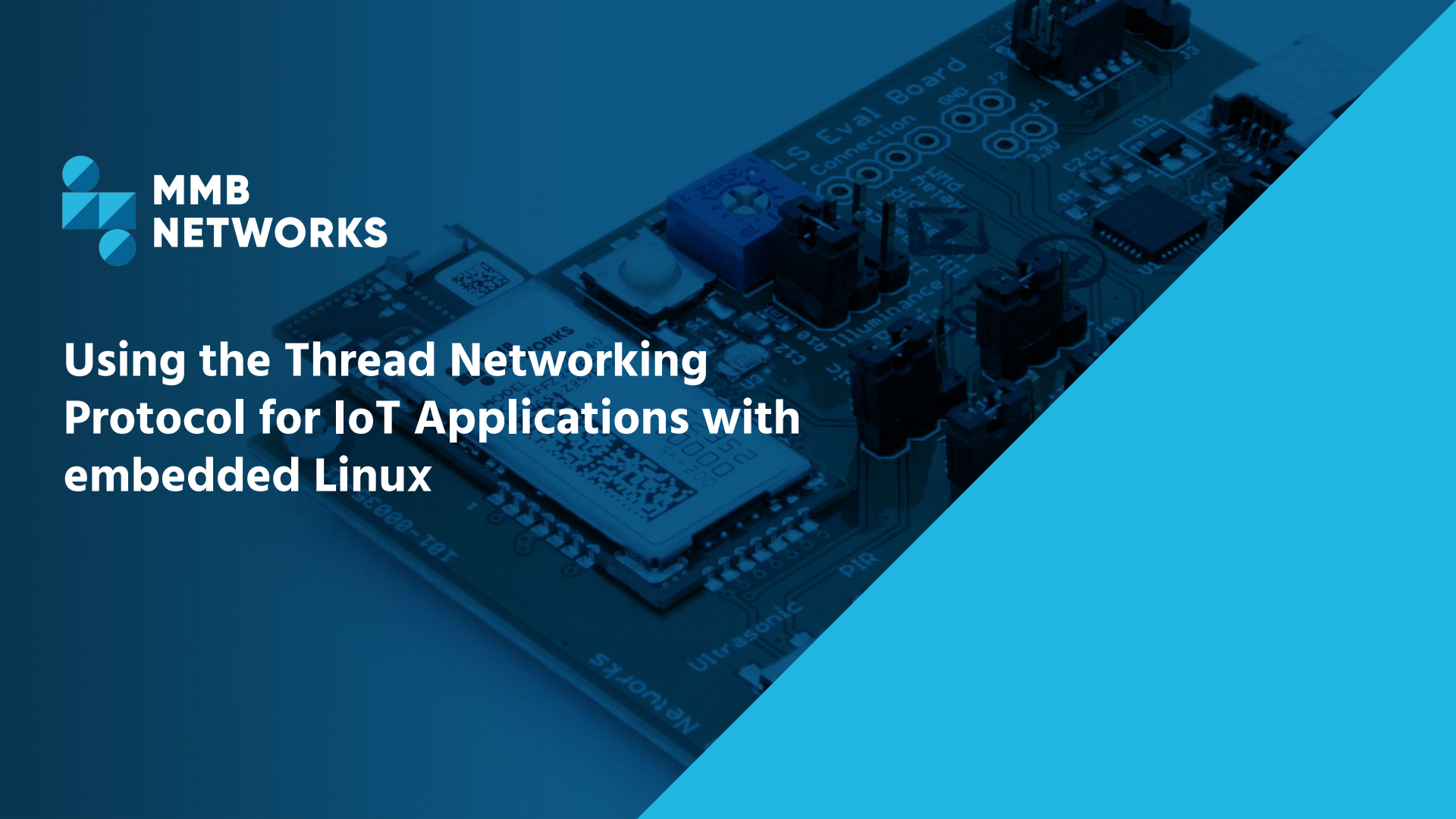
MMB NETWORKS

Using the Thread Networking Protocol for IoT Applications with embedded Linux



Agenda

- Quick Introduction of MMB Networks and myself
- IoT Landscape Overview, quick comparison of IoT technologies
- Introduction to Thread, why Thread instead of other technologies?
- Introduction to OpenThread
- Border Router on Buildroot
- Demo video
- Questions

Quick Introduction to MMB Networks & Michael Magyar

MMB Networks

Create Amazing Customer Experiences Like No One Else Can

We help the world's biggest brands build and bring wireless products to market. Think connected utility grids, thermostats, locks, lighting solutions, and other applications. We're the ones that make sure those products actually work.

Located in Toronto, Ontario

Michael Magyar

Gateway Program Manager

- Started at MMB in 2014 as a Firmware Developer
- Now, maintains MMB's RapidConnect Linux Gateway
- Works on Embedded Linux, Bluetooth, Wifi, or Thread projects

IoT Landscape Overview

IoT Technologies

Define up to Application Layer

- Zigbee
  - Zigbee Cluster Library to define “the Thing”
- Bluetooth Low Energy
  - GATT characteristics define “the Thing”
- Bluetooth Mesh
  - BT Mesh Models
- Z-Wave
  - Implements its own Transport and Application layers

Generally require an Application Gateway to translate messages to/from technology format

Define up to Transport Layer

- Wi-Fi
  - Ubiquitous, IP based
- 6LoWPAN
  - IP based
  - Compressed IPv6 over 802.15.4 (for this talk)
- Thread
  - Builds on 6LoWPAN, adding Routing, Security and Commissioning, and Certification programs

IP based Transport Layer technologies allow for multiple Applications to interoperate and co-exist on the same network

Sample App Layers

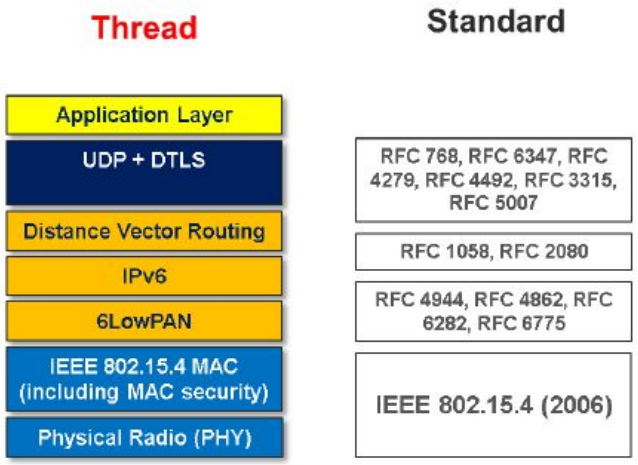
- Zigbee Cluster Library
- OCF (Open Connectivity Foundation)
- KNX
- CoAP
- Project CHIP

Introduction to Thread

Introduction to Thread

Why Thread instead of other IoT technologies?

- Simple network installation, start up, and operation
- Secure joining, and encrypted and secure communications
- Range. Mesh networking provides sufficient range to cover installation
- No single point of failure, and Self-healing
- Low power
- Build on existing standards and silicon
- Standards based and Royalty free
    - Thread Group Membership required

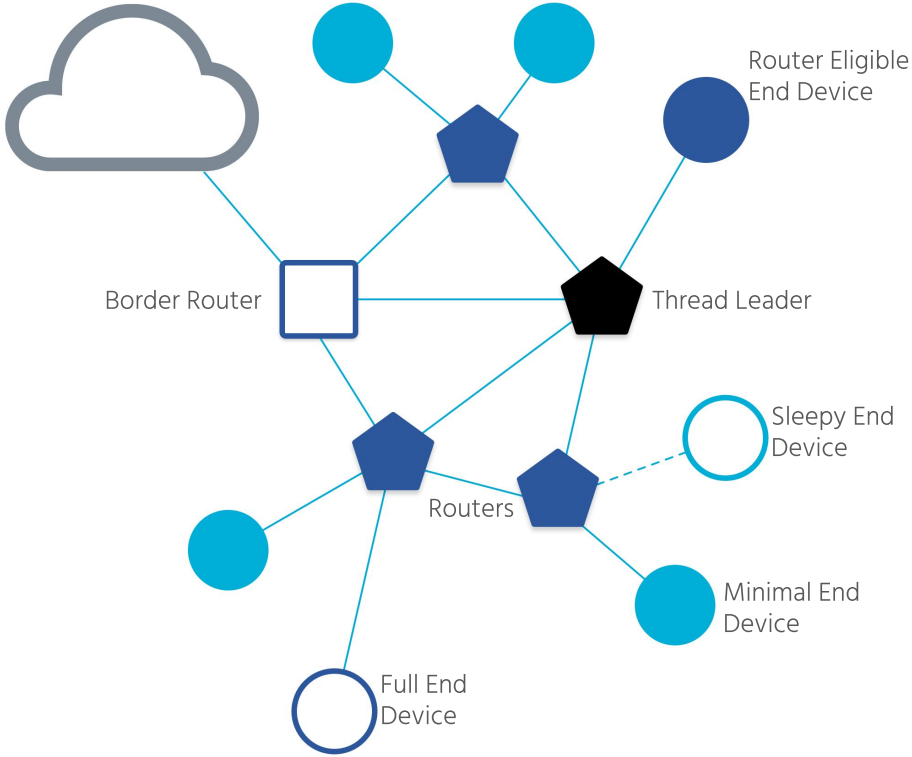


Device Types and Roles

- **Full Thread Device**
  - Border Routers
  - Routers
  - Router-eligible End Device (REED)
  - Full End Device
- **Minimal Thread Device**
  - Minimal End Device
  - Sleepy End Devices

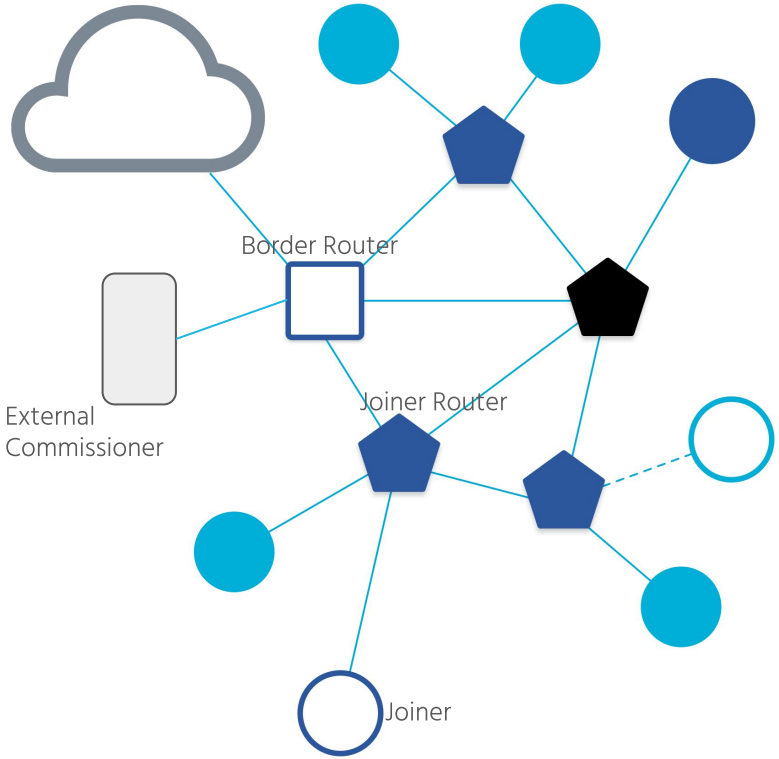
Device roles are determined by the stack using Mesh Link Establishment messages with REEDs promoted to Routers as needed

All nodes have Link-Local and Mesh-Local IPv6 addresses. If a Border Router is present a Global Address may be configured using SLAAC or DHCPv6



Thread Commissioning

- Joiner devices must securely (with DTLS) communicate with a Commissioner
  - DTLS hand shakes
- Commissioner can be:
  - Native (on the Thread network)
  - External (off-mesh)
- External Commissioners connect through an available Border Agent running on a Border Router
  - Typically this would be over Wi-Fi to a mobile device
  - External Commissioner must establish secure session with the Thread Network



Introduction to OpenThread

Standing on the shoulders of giants

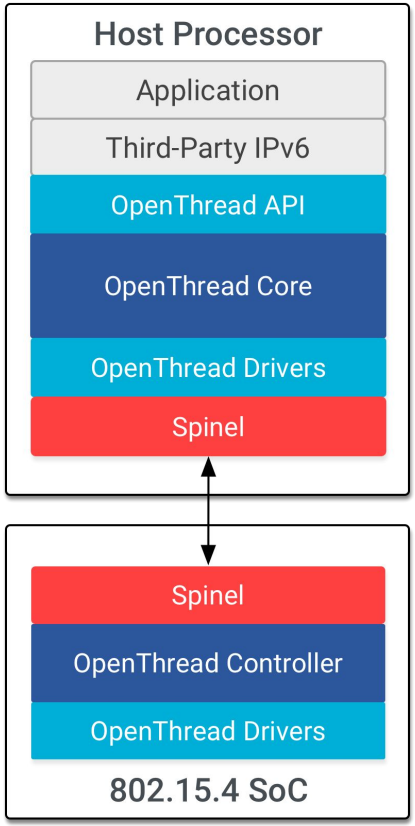
Open source implementation of the Thread networking protocol

- Released by Google Nest under BSD-3-Clause License
- Supports all features in Thread 1.1.1 specification
    - Thread Certified Component
- Border Router support through the ot-br-posix project
- System-on-chip (SoC), Network Co-Processor (NCP) and Radio Co-Processor designs
- Supported by a wide range of vendors and SoCs
    - Nordic, NXP, SiLabs, and many more
- CLI and NCP sample applications

MMB is not affiliated with Google Nest or OpenThread. Just happy users and (hopefully) eventual contributors

Radio Co-Processor Designs

- Minimal “controller” firmware running on the radio SoC
- Most of the Thread Stack is running on the Host processor, using host resources
- Allows for use of lower cost, lower performance 802.15.4 SoCs
- Host - Radio controller communication using the ‘Spinel’ general management protocol developed by Nest
    - IETF draft: Spinel Host-Controller Protocol draft-rquattle-spinel-unified-00
    - Can use UART or SPI



Thread Border Router on Buildroot

Why do we want a Thread Border Router?

The Thread Border Router serves critical functions, including relaying data between Thread and other IP-based networks, such as Ethernet, WiFi, or the Internet itself. In addition, Thread Border Routers enable the simple, user-friendly Thread device commissioning process via a smartphone or the Cloud.

At the **network layer** a Border Router:

- Forwards packets from the Thread Network interface to the exterior interface(s).
- Packets from exterior interface(s) will be forwarded to the Thread interface and then routed further in the Thread Network towards their end destination.
- Packet filtering or address translation may be performed based on firewall, system, or infrastructure settings.
- May participate in an exterior routing protocol, advertise global IPv6 prefixes and handle global scoped address allocation for nodes within the Thread Network.

At the **transport layer** a Border Router should be transparent end-to-end IP communication.

During **commissioning** a Border Router will act as a proxy between the Thread Network and an External Commissioner device for user-initiated joining of new devices.

Thread Border Routers are the infrastructure, the Wi-Fi router of low power wireless networks, bringing Things into a standard IP based world

Building a Border Router

- **OpenThread Border Router (ot-br-posix) runs on:**
  - Beaglebone Black
  - Raspberry Pi
  - Buildroot based hosts (with some packaging effort)
- **Creates a wpan0 TUN device that we can now use**
  - echo 1 | sudo tee /proc/sys/net/ipv6/conf/all/forwarding
  - echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
- **Configure Tayga with well-known NAT64 prefix**
  - 64:ff9b::/96
  - Or site administered prefix
- **Configure BIND9 with site specific DNS zones**
- **Node side can use:**
  - Finddns or finddnsslaac CLI commands to locate Border Router host DNS Server
  - dns resolve service.mmbnetworks.com <BR_IPv6>
  - > DNS response for service.mmbnetworks.com - 64:ff9b:0:0:0:0:fefe:fefe TTL: 86400

Demo Video

Questions?

References and Additional Reading

Thread Group White Papers and Resources:

- https://www.threadgroup.org/Support

Google OpenThread Documentation:

- https://openthread.io/guides
- https://openthread.io/platforms

Resources

MMB NETWORKS

Michael Magyar
michael.magyar@mmbnetworks.com
mmbnetworks.com