*presented by*


# Lessons Learned from Implementing a Wi-Fi and BT Stack

Spring 2017 UEFI Seminar and Plugfest

March 27 - 31, 2017

Presented by Tony Lo(AMI)

# Agenda

- Introduction
- UEFI Bluetooth Stack Intro
- Bluetooth Cases Study
- UEFI Wi-Fi Stack Intro
- Wi-Fi Cases Study
- Conclusions

# Introduction

# Introduction

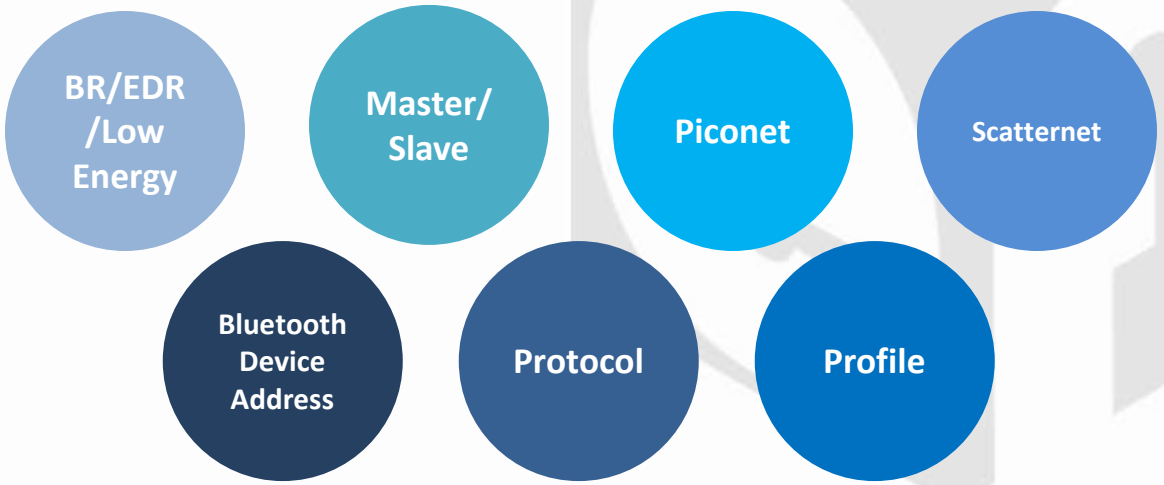
- In 2014, there was a previous presentation on implementing a Bluetooth stack
- In UEFI version 2.5, the UEFI specification added support for both Bluetooth (BT) and Wi-Fi
- In developing our own BT and then aligning this implementation to the specification AMI has learned several things
  - Device issues
  - Specification issues
- Similar items were learned in developing a Wi-Fi stack and connecting it with the UEFI specification
- This presentation will go into detail about lessons learned while creating a stack that conforms to the specification and meets industry needs.

# UEFI Bluetooth Stack Introduction

# BT Introduction

Bluetooth is a modern day technology and has many acronyms and terms that are important to understand

Important Acronyms/Terms:



# Bluetooth Stack Terms

## Important Bluetooth Stack Terms:

- BR | EDR | AMP | Low Energy
- Master/Slave
- Piconet
- Scatternet
- Bluetooth Device Address
- Protocol
- Profile

# Bluetooth System Modes

**Basic Data Rate (BR)**

Transfer Rate: 721.2 Kbps

**Enhanced Data Rate (EDR)**

Transfer Rate: 2.1 Mbps

**Generic Alternate MAC/PHY Protocol (AMP)**

Transfer Rate: 54Mbps

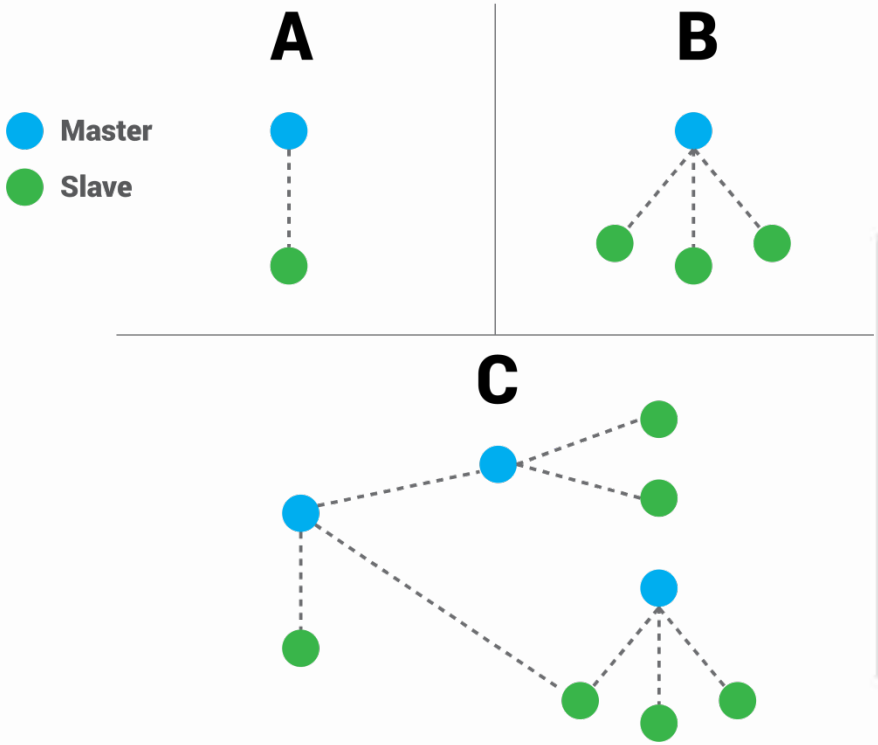
**Low Energy**

Low Power Consumption

# Master/Slave/Piconet

- Master
  - Provide the reference clock and frequency hopping pattern
  - Only one master within one piconet
- Slave
  - Sync to master's clock and frequency hopping pattern
  - One to seven active slave devices are allowed in one piconet
  - 255 possible parked slave devices

# Scatternet




- Multiple piconets that have common devices are called scatternet.
- Master device can act as slave in other piconets.
- Each Master owns one identical physical channel.

Net (a): Piconet

- Single master and single slave

Net (b): Piconet

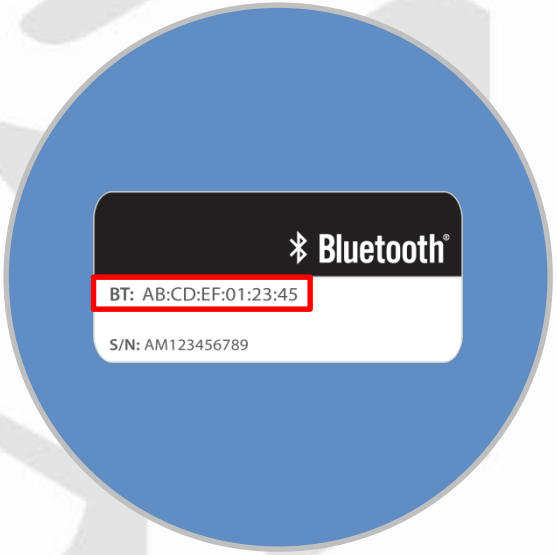
- Single master and multiple slaves

Net (c): Scatternet

- Multiple piconets share devices

# Bluetooth Device Address

**Each Bluetooth Device should allocate a unique 48bit Bluetooth Device Address (BD_ADDR)**



# Protocol

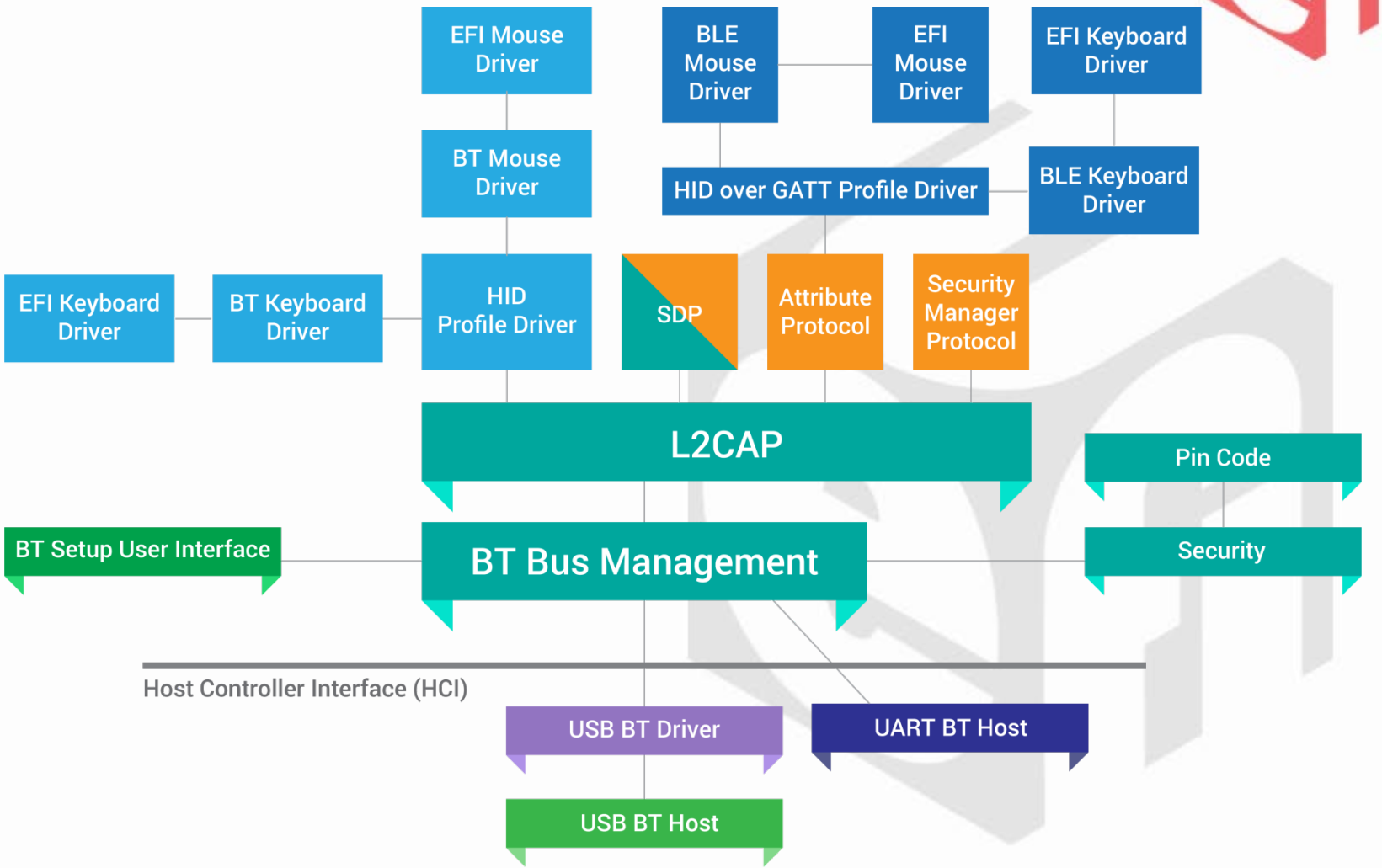
- The Bluetooth specification defines protocols for the communication between Bluetooth function blocks within Bluetooth architecture. Such as
  - LMP (Link Management Protocol)
    - Control and Negotiate all the operation between two devices.
  - SDP (Service Discovery Protocol)
    - Determine the available services of device.

# Profile

## PROFILE //

*Profile describes service discovery requirements necessary for devices to connect, find available application services and connection information necessary for making application level connections.*

# Bluetooth Stack



# UEFI Bluetooth Protocols

- **EFI Bluetooth Host Controller Protocol**
  - Protocol that abstracts the Bluetooth host controller layer message transmit and receive
- **EFI Bluetooth Bus Protocol**
  - Protocol that is used to locate EFI Bluetooth IO Protocol drivers to create and destroy child handles of the driver to communicate with other Bluetooth device by using the Bluetooth IO protocol
- **EFI Bluetooth Configuration Protocol**
  - Protocol that abstracts configuration for Bluetooth devices

# Bluetooth Cases Study

# Bluetooth Keyboard Layout Issues

**Certain types of Bluetooth Keyboards are not designed for PC. Some keys may be missing, such as “DEL”, Function Keys**

**What happens if this key was required for some functionality like hot key to boot specific device or enter setup?**

# Bluetooth Controller Specific Patches

- The Bluetooth specification defines many areas of the host controller registers and capabilities
- In practice, every Bluetooth Controller does not follow all of the specification
  - Operating Systems have the luxury of device specific drivers!
  - This requires many vendor specific workarounds that bloats the overall Bluetooth code and slows execution of the overall stack

# Service Latency

- Polling is the default method for device events servicing
  - Works well for devices with quick response times like HDD and network, but not as much with radio based systems
- Infrequent polling for radio devices may slow response times and can cause issues in responding to Bluetooth requests
  - Frequent polling will impact other events service and boot times
  - A good middle group needs to be found!

# Boot Time Impact

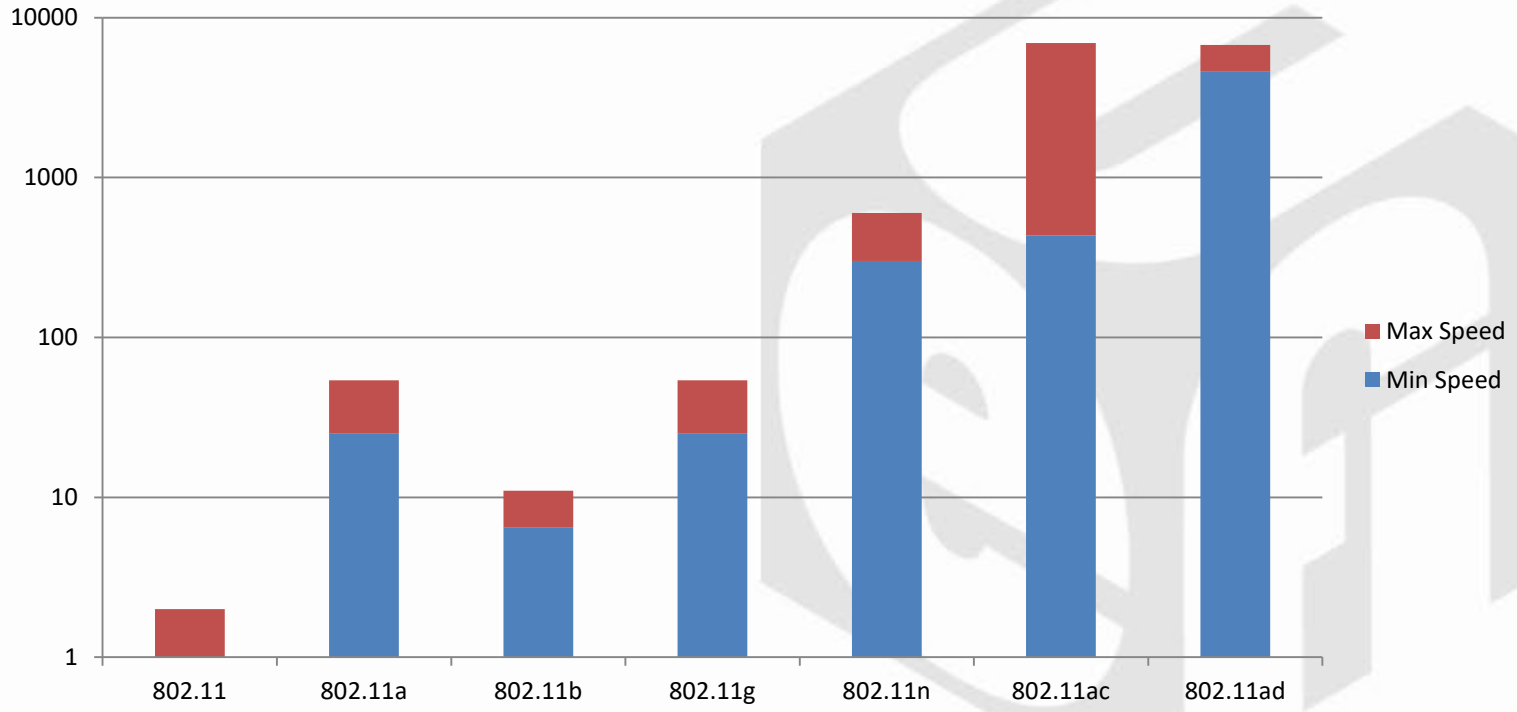
# UEFI Wi-Fi Stack Introduction

# Wi-Fi Introduction

- Wi-Fi is a wireless technology based on IEEE 802.11 standards.
- Various version of IEEE 802.11 specifications are published to increase the transfer speed and quality of service.
- Wi-Fi Device Network can be established with/without Wi-Fi Access Point. Two Wi-Fi Devices which support Ad-Hoc mode can direct connect without Access Point.

# Wi-Fi Specification and Speed

The wireless networking specifications (802.11xx) have evolved to allow much faster transfer speeds over the years




802.11 - 2.4 GHz
802.11a – 5 GHz
802.11b – 2.4 GHz
802.11g – 2.4 GHz
802.11n – 2.4 GHz
802.11ac – 5 GHz
802.11ad – 60 GHz

# Wi-Fi Stack Terms

**Important Terms:**

**Access Point**

**SSID**
**(Service Set Identifier)**

**Wireless Security**

# Access Point

**01**

Access Point is a Wi-Fi device which accepts Wi-Fi device connection

**02**

The Access Point can be a bridge to connect to another wireless network or router which connects to Internet

# SSID (Service Set Identifier)

A sequence of 0-32 octets which is usually a human readable string for user to identify the network name easily

**SSID is a Service Set Identifier**

Commonly referred to as the Wi-Fi network name

# Wireless Security

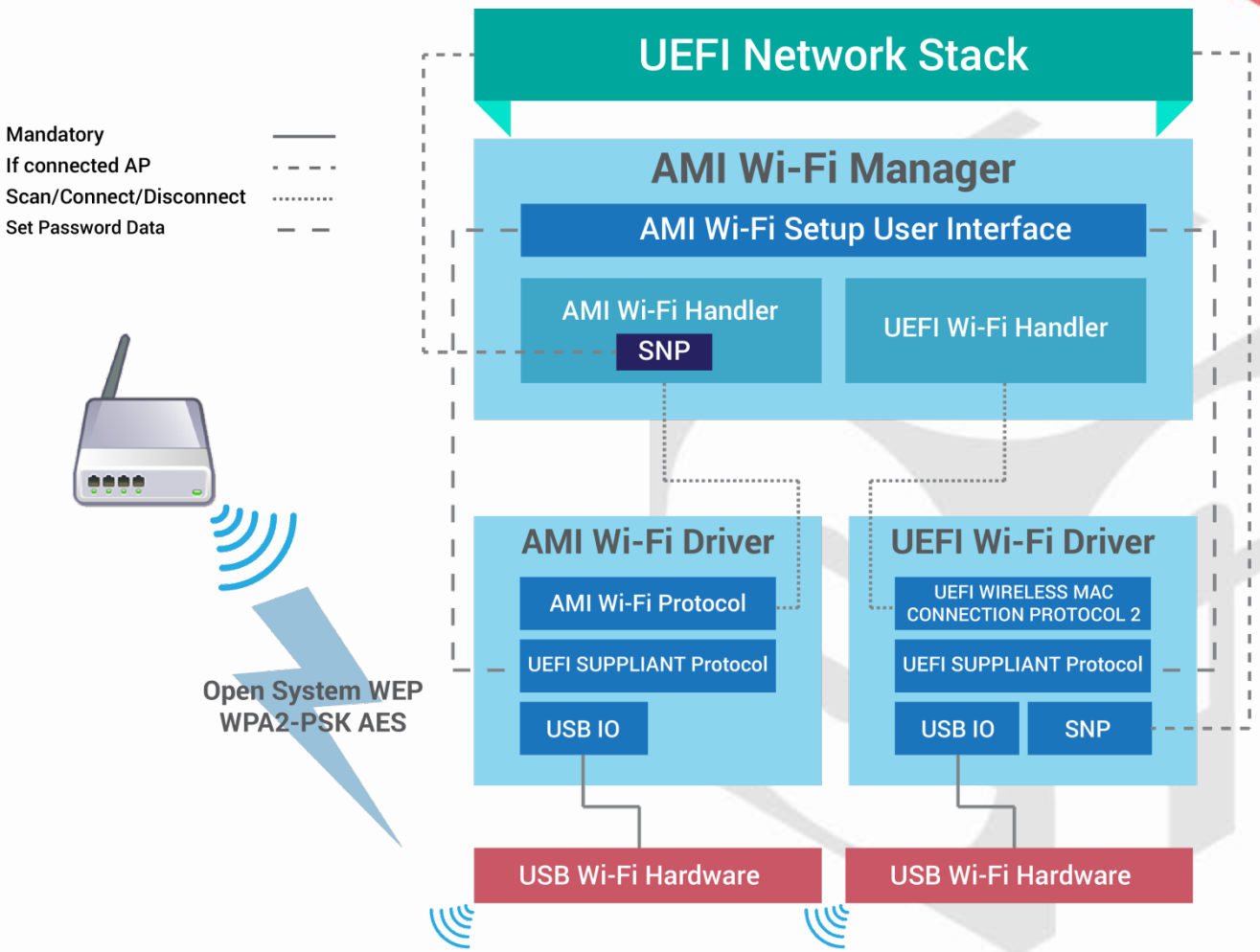
Wi-Fi being a broadcasted network, needs a layer of security beyond any physical security

The wireless security is used to avoid authorized access of the wireless network

There are several security protocols such as:

- WEP
- WPA
- WPA-2
- Etc...

# UEFI Wi-Fi Stack



# UEFI Wi-Fi Protocols

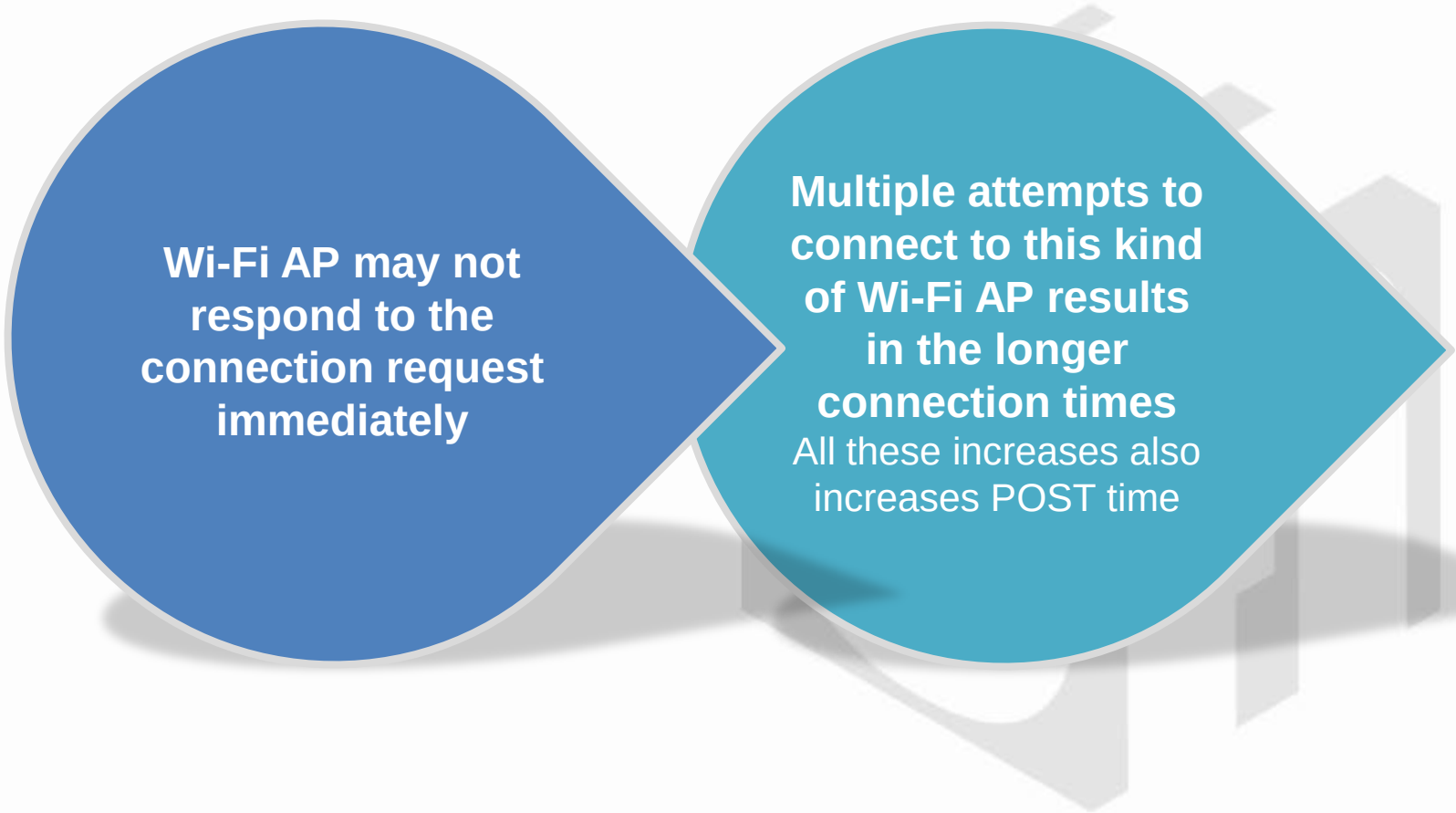
- **EFI Wireless MAC Connection Protocol**
  - Protocol that provides management service interface of the 802.11 MAC layer. It can be used by network applications (and drivers) to establish a wireless connection with an access point (AP)
- **EFI Wireless MAC Connection II Protocol**
  - Protocol that simplifies the wireless connection manager and moves the responsibility of scan operation, AP selection, authentication and the association flow control into wireless UNDI driver.
- **EFI Supplicant Protocol**
  - Protocol that provides services to process authentication and data encryption/decryption for security management

# Wi-Fi Cases Study

# Encryption Protocols

- The current UEFI specification targets to the open Wi-Fi systems and popular encryption protocols, such as WPA2.
- The EAP related protocols address the need of business requirements.

# Wi-Fi AP Response Time



# Wi-Fi NIC Driver

**For PCIe native Wi-Fi devices, information is needed from the vendor to do proper porting**

Some information can be gathered from specifications or open source drivers

This information is not usually enough to develop a full UEFI driver

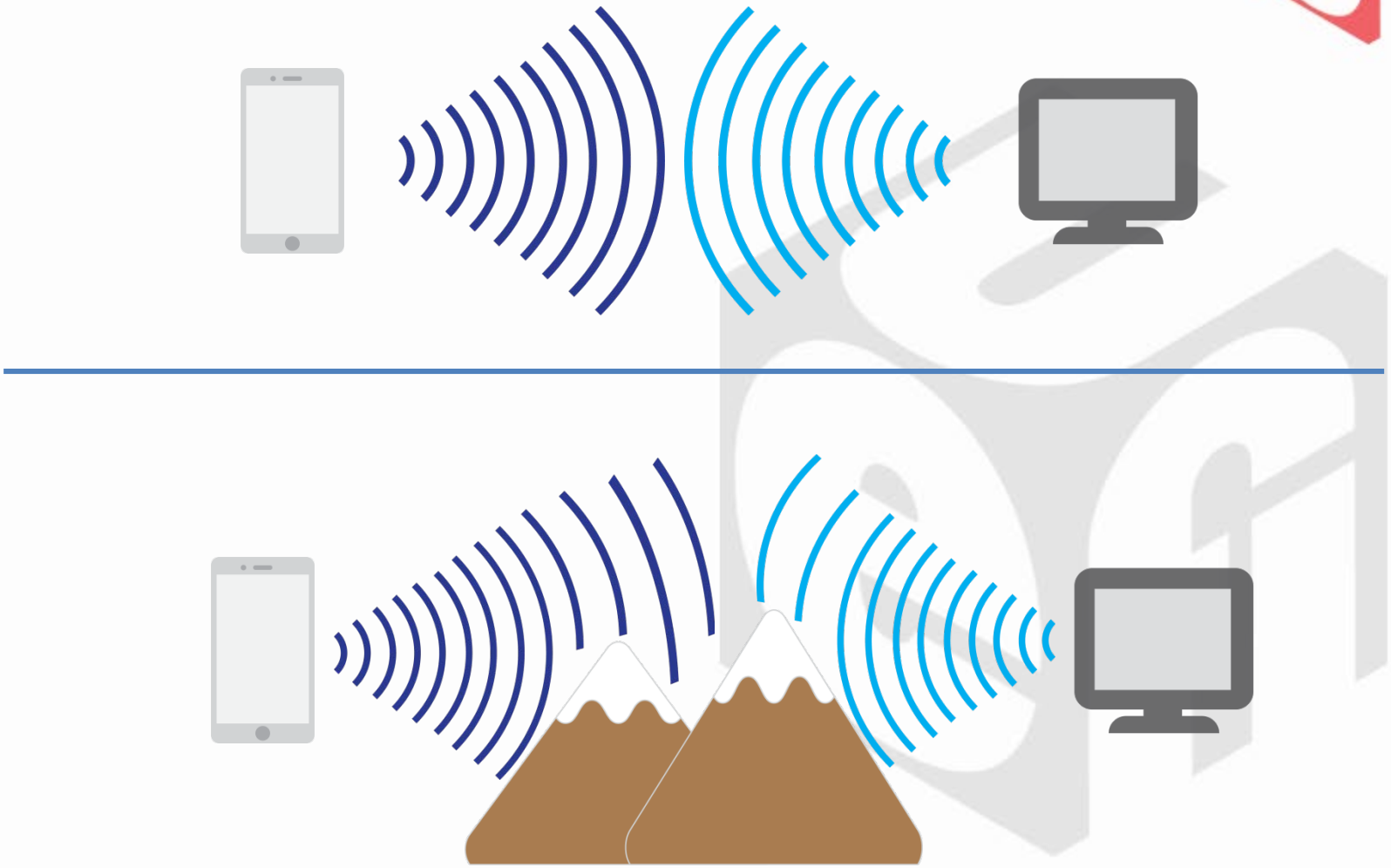
Industry needs a better framework for vendors to provide something similar to an UNDI driver for UEFI

# Radio Signal Interference

- Wi-Fi and BT both operate in the 2.4 GHz range
- Everyone continues to collect more BT and Wi-Fi devices in their personal and work lives
- The more devices in the area, the more radio interference will occur
  - OS has luxury of timeouts and reconnects

This can cause increased connection times and dropped packets!

# Radio Signal Interference



# Conclusions

# Specification Need Summary

| BLUETOOTH | WI-FI |
| --- | --- |
| • Better define how the Bluetooth Profile driver interacts with the other UEFI Bluetooth Protocols<br>  • Some samples may help developers understand the proper usage<br>• Low energy device needs are not currently addressed by UEFI 2.6 specification | • Security phrase is not provided along with connect function<br>• Simplify the parameters of Scan/Connect/Disconnect functions |

# Call to Action

- Review the specification and get involved in the UEFI Network Sub-Team (USNT)
- BT and Wi-Fi hardware vendors should get involved in defining common hardware interfaces to ensure compatibility
- Tool writers should get involved and write applications on top of the common hardware interfaces to create market ready solutions

Thanks for attending the Spring 2017 UEFI Seminar and Plugfest

For more information on the Unified EFI Forum and UEFI Specifications, visit http://www.uefi.org

*presented by*

American Megatrends