Bluetooth 5 Hacking

An introduction to changes in Bluetooth 5, challenges that they pose, and guidance for examining the security of BT5 devices.

Sultan Qasim Khan

nccgroup

Bluetooth Background

- Bluetooth Classic and Low Energy (LE) are very distinct protocols with little in common (except HCI and L2CAP)
- Both LE and Classic are frequency hopping spread spectrum (FHSS) in the unlicensed 2.4 GHz band, but use different PHYs
- Classic most commonly used for audio streaming and hands free calling, while most IoT devices use LE
- Both Classic and LE are part of the Bluetooth 5 standard, but the term Bluetooth 5 most commonly refers to Bluetooth 5 LE

nccgroup

Bluetooth LE

- Introduced in the Bluetooth 4.0 specification in 2010, originally intended as a low power protocol for devices with limited data throughput requirements
- Divides 2.4 GHz spectrum into 40 channels: 0-36 for data, 37-39 for advertising
- Upper layer communications built on GATT protocol, where clients can read, write, or subscribe to “characteristics” on a server
- BLE 4.0/4.1 have well known weaknesses in their pairing process that uses symmetric cryptography
- BLE 4.2 introduced an optional more secure ECDH pairing scheme
- BLE 4.2 added Data Length Extension (DLE), and BLE 5 added 2M PHY, both greatly increasing throughput
- With increased throughput, the LE is often omitted when describing BLE 5

nccgroup

Bluetooth 5 Changes

- Most changes are in LE, which is the focus of this presentation
- New PHY modes for high throughput (2M) and long range (125k and 500k coded on 1M)
- New PRNG based channel hopping algorithm
- Greatly expanded advertising support, with advertising on secondary advertising (data) channels and connectionless data streaming (“periodic advertising”)
- Angle of Arrival (AoA) and Angle of Departure (AoD) measurement for location measurement using beacons (BLE 5.1)
- Randomized selection of primary advertising channels (BLE 5.1)
- These additions are optional

nccgroup

Attacks Against BLE

- Sniffing unencrypted communications
- Sniffing and cracking pairing using the legacy symmetric protocol (Crackle)
- Discovering and jamming existing connections (Ubertooth and BtleJack)
- Hijacking existing connections (BtleJack)
- Exploiting parsing flaws in the Bluetooth stack (L2CAP, ATT, GATT)
- Exploiting application layer software vulnerabilities (above GATT)

Security Recommendations

- Encrypt and authenticate communications with LE Secure Connections (ECDH pairing)
- Consider additional application layer encryption and authentication to protect against malicious apps on the same phone as the legitimate app
- Keep Bluetooth stacks on central and peripheral devices updated to patch software vulnerabilities
- Avoid complex parsing at the application layer whenever possible to minimize the risk of parsing bugs
- Consider a third party security audit

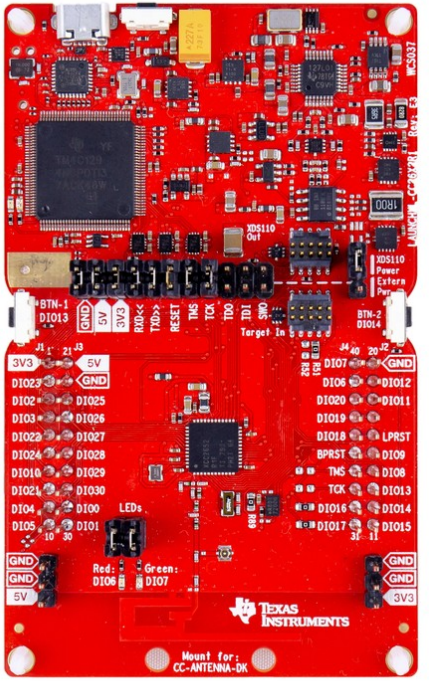
nccgroup

BLE 5 Attack Challenges

- New PHY modes require hardware support
- New channel selection algorithm requires software support, and determining the hop sequence of an existing connection is non-trivial due to the PRNG
- Connection establishment detection is unreliable (<33%) when sniffing only one advertising channel
- Capture of auxiliary (secondary channel) advertisements requires complex channel hop scheduling
- BLE 5 connection establishment can also occur on secondary advertising channels (data channels)

PHY Modes

- New generation Bluetooth 5 transceivers can be used to send and receive with the new PHY modes
- Most popular suitable BLE 5 MCUs are the Texas Instruments CC1352/CC2642/CC2652 and Nordic nRF52840
- Sniffle (my Bluetooth 5 sniffer) uses the TI microcontrollers to support all PHY modes
- Sniffle runs on low cost ($40 USD) TI CC26x2R and CC1352R Launchpads



nccgroup

Channel Selection Algo. #2

- Sniffle implements CSA #2 to support sniffing devices that use it, and also supports channel map updates under CSA #1 and #2
- While not intended as a security feature, the use of a PRNG in CSA #2 makes jamming or hijacking an existing connection more difficult
- The CSA #2 hopping sequence repeats after 65536 hops due to its use of a 16 bit counter
- Damien Cauquil (virtualabs) presented at DEF CON 27 an efficient approach to determine the current position in the hopping sequence for an existing connection, and used it to sync, sniff, jam, and hijack BLE 5 connections (using the 1M PHY) in BtleJack 2.0

Advert. Channel Hopping

- Advertisers hop in order between channels 37, 38, and 39 on the primary advertising channels (for BLE 5.0 and earlier)
- Sniffle channel hops along with the advertiser, making it nearly 3x more reliable at connection detection than conventional sniffers that stay on one advertising channel
- Alternatively, three sniffers can be used (at higher cost)
  - Most reliable option for connection establishment on primary advertising channels, but doesn't handle establishment on secondary channels (AUX_CONNECT_REQ)
- Bluetooth 5.1 complicates advertising channel hopping by permitting optional randomization of the 3 channel sequence

nccgroup

Auxiliary Advertising

- BLE 5 optionally allows advertising on secondary advertising channels (ie. data channels) with a variety of new advertisement PDU types
- Advertisements on these channels are pointed to by advertisements on the primary advertising channels (37-39)
- Top capture auxiliary advertisements, sniffers need to schedule channel hopping between primary and secondary advertising channels
- I have not encountered any real devices using auxiliary advertisements as of August 2019
- I am working to implement more sophisticated advertising channel hop scheduling for Sniffle to capture auxiliary advertising

nccgroup

BLE 5 Research Tips

- When possible, HCI logging can be simpler and more reliable than sniffing for capturing Bluetooth communications
  - However, be aware of transformations performed by the controller such as encryption
  - Sniffers provide a clear view of actual traffic over the air that can be helpful when diagnosing controller or stack level issues, or just identifying exactly what is encrypted
- Controller level vulnerabilities are an interesting and under-researched target, but require time consuming reverse engineering to examine
- Code review and fuzzing of Bluetooth stacks has found and will likely continue to find many bugs
  - While BlueZ and Fluoride (on Linux/Android) have undergone a lot of study, there are many under-scrutinized embedded stacks that likely have undiscovered parsing issues

Research Tips Contd.

- BLE devices that do not use encrypted connections are still common
  - Consider the threat model to determine if encryption is necessary
  - Unencrypted links are vulnerable to device impersonation, connection hijacking, and data modification: what risks do these pose to the target?
- Custom application layer encryption protocols built over BLE commonly have cryptographic weaknesses
- Other common application layer BLE devices issues include insecure firmware OTA mechanisms, and unsafe parsing of GATT characteristic values, advertisement data, and scan responses

Questions?