Electron Bluetooth Lamp Controller

Michael Rooplall

CS 488 - Computer Networks and the Internet
&
CS 389 - Software Engineering

Professor Mustafa

Pace University - Honors Option Project

Contents

GitHub: https://github.com/DeveloperBlue/Electron-Bluetooth-Lamp-Controller

Video Demo: https://youtu.be/_7ogx-qlPFA

The goal of this project was to build an Electron application using NodeJS, capable of controlling an RGB LED Bluetooth lamp and other generic Bluetooth devices. The Bluetooth light bulb was a generic off brand bulb that could only be controlled through the now defunct company's android application. I wanted to be able to control my bulb from a variety of different environments, and create a proof of concept that I could further integrate more advanced functionality.

Objectives

- Understanding the Bluetooth protocol
    - Understanding Bluetooth Attributes, Services, and Characteristics
    - Understanding UUIDs, Hexadecimal properties, and unsigned 8 bit integer arrays
- Intercept Bluetooth packets between the Android controller and the destination Bluetooth device
    - Capturing packets using Android's built-in Bluetooth packet sniffer
- Analyze and understand bluetooth packets in Wireshark
    - Wireshark and the Bluetooth protocol
- Pairing, reading, and writing to Bluetooth devices
    - nRF Connect Application
    - WebBluetooth Library
- Build an Electron application compilable to Windows, Mac, and Linux, and usable as a web application
    - Scanning and pairing to bluetooth devices
    - Changing the powerstate of the bulb
    - Reading and writing colors, brightness, and temperatures from/to the bulb
    - Reading and writing to individual bluetooth characteristics for general bluetooth devices for advanced functionality
- Understanding bluetooth device limitations
    - Pairing and scanning limits
    - Secured HTTPS only for web service
- Build a sleek UI to balance aesthetics and functionality provide basic Bluetooth functionality
- Further possibilities

Reverse Engineering

My first attempt to control the bluetooth lamp involved decompiling the Android application and sifting through it's APK files. After realizing how tedious this was, and learning about Wireshark in CS488, I went to work figuring out how to intercept packets between the application on Android and the bluetooth device.

The Bluetooth Protocol

A Bluetooth device contains a table of data called an Attribute Table which can be accessed by other connected devices in various possible ways. These attributes can be broken down further, and we will focus on services and characteristics.

Services

A Service is a container for logically related Bluetooth data items. Those data items are called Characteristics. A Service can be thought of as the owner of the Characteristics inside it.

Characteristics

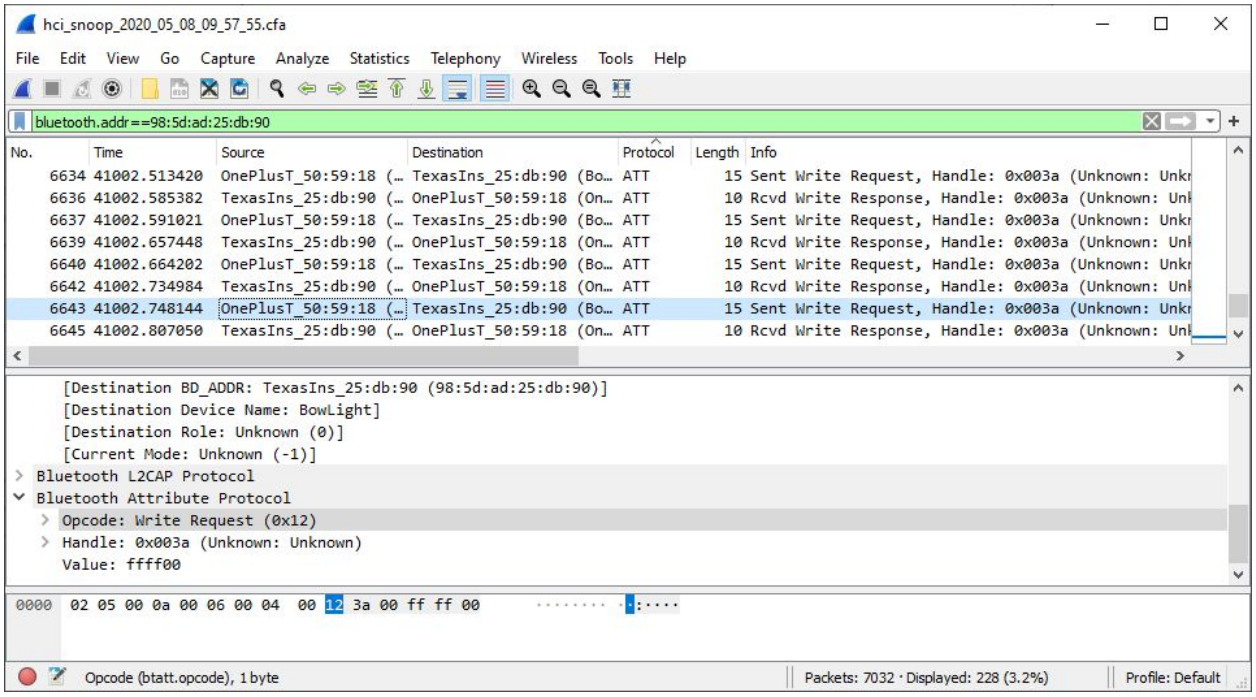
Characteristics are items of data which relate to a particular internal state of the device or perhaps some state of the environment which the device can measure using a sensor. The current battery level is an example of internal state data. Characteristics contain various parts, including a type, a value, properties and permissions.

All attributes are identified by a ***UUID*** (Universally Unique Identifier). Some Attributes are defined by the Bluetooth SIG, the technical standards body for Bluetooth and these have UUIDs which are 16 bits in length.

Capturing with Android's Packet Sniffer

Built into Android's developer settings is a tool called the *Bluetooth HCI snoop log*, which sniffs and logs Android Bluetooth traffic. Initially, there were some unintuitive problems with the way some OEMs handled logging. In my case, the default directory for these bluetooth packet logs was changed and was different depending on the OEM. I was also not told that after running the HCI snoop log, I had to restart my device for the logs to even appear. Once that was sorted, I was able to analyze them in Wireshark.

Analyzing with Wireshark



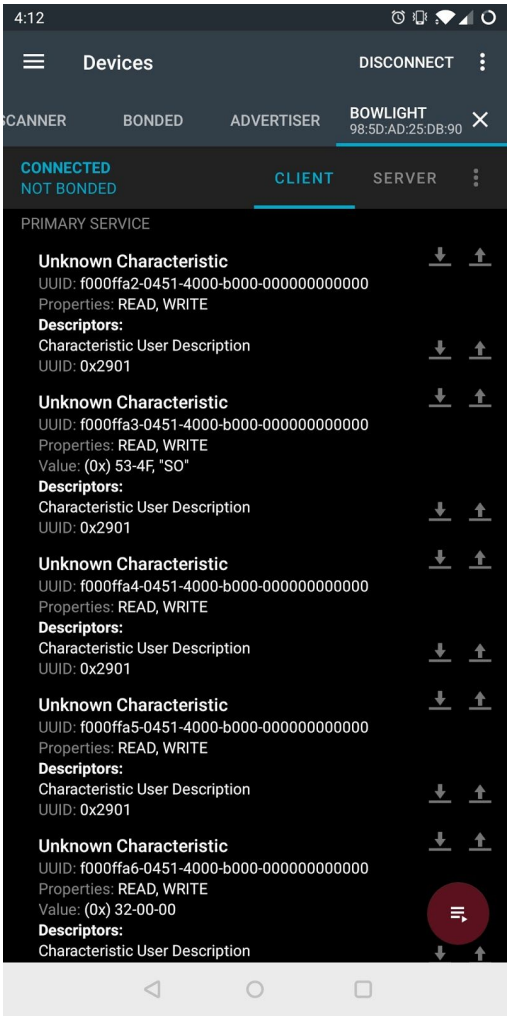
Wireshark Bluetooth Packet Analyzer

After filtering wireshark for traffic only to the bluetooth bulb, I was able to figure out which bluetooth attributes triggered what actions on the bulb, and which services they belonged to.

A detailed description of the characteristics and their read/write values can be found in the project files in `dev/notes.md`.

Verifying with nRF Connector

Another powerful Bluetooth tool on Android is *nRF Connect*. I used this tool to write values to the different bluetooth device's characteristics, applying my conclusions from Wireshark. With *nRF Connect*, I was able to send byte arrays and manipulate the power and colors of the bluetooth bulb. I could now manipulate the bulb independent from the proprietary application.



nRF Connect displaying the bulb's attributes

The Client

Electron

Electron is a NodeJS library used to build desktop applications with JavaScript, HTML and CSS. These applications can be packaged to run on Windows, Mac, and Linux operating systems. There were two main reasons behind using Electron. Firstly, because development on the client is done using web languages, the project can also be deployed as a web application. This all allowed for 90% re-usability on one codebase that could be deployed on not only Windows, Mac, and Linux operating systems, but also deployed online as a web application- meaning mobile devices and ChromeOS could at the least use the basic functionality. Secondly, Electron bundles the NodeJS runtime when packaged and allows access to a variety of other npm backend services. I could run a web server on the same client machine, check device hardware for bluetooth adapters, and much more.

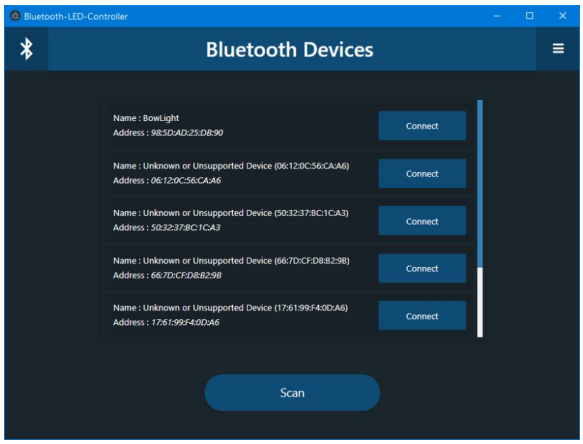
WebBluetooth

After verifying that I could read and write to the bluetooth attributes on the bulb, I had to figure out how I would approach this from a desktop environment. Thankfully, the Web Bluetooth API provides the ability to connect and interact with Bluetooth Low Energy peripherals through the browser. Because of Electron's derivation from Chromium, I could use the Web Bluetooth API in my application to pair and talk to bluetooth devices.

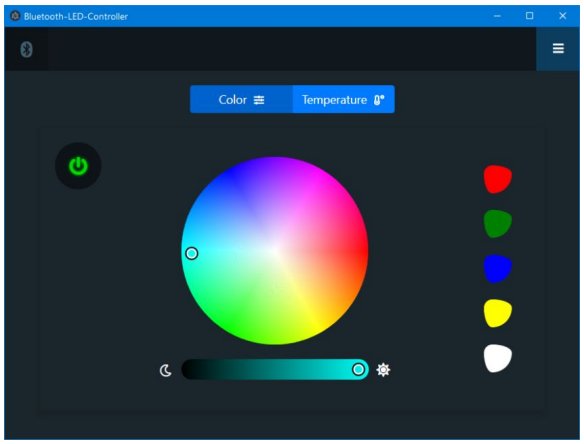
Design

While the client UI was not the main objective in this project, I still wanted something that both looked elegant without losing out on functionality. I used a modified Bootstrap 5 library and generated my own dark color palettes. The UI was designed following general user experience guidelines, with consistent design across the layouts, transition animations, and loading graphics. For implementing a color wheel, I used an open-source project called iro.js. It is well documented and has a load of capabilities such as RGB, Hex, and Kelvin color conversions.

The rest of the UI was all designed by myself. I also used JQuery to simplify the client-side code and made use of its provided animation libraries to keep the application snappy.

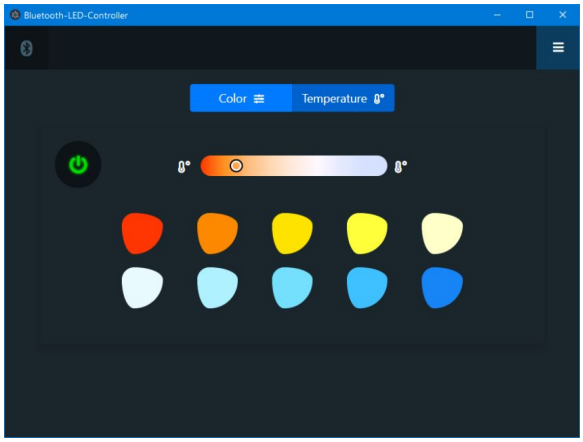


Device Selection Menu

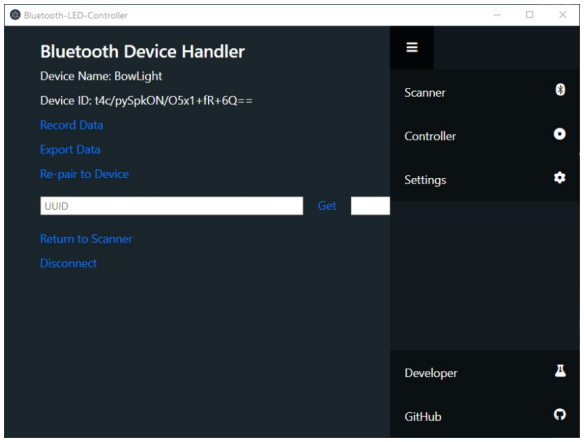


RGB Color Control

Electron also does not come with it's own device selection menu when polling for bluetooth devices, like Chrome does, so one had to be designed and implemented for users not using the browser.



Kelvin Color Control



Debugging and Experimenting

Conclusion

In conclusion, I learned a great deal about the Bluetooth protocol and how to manipulate low-energy bluetooth devices. I used a variety of bluetooth tools on both Android and Windows, and with a single codebase was able to deploy to multiple platforms and the web. Concepts like packet sniffing were introduced to me in CS 488 - Computer Networks and the Internet, along with tools like Wireshark. In CS 389 - Software Engineering, I learned different approaches to software development, using an agile workflow and sprints. The success of this project opens up the potential for future advanced integrations, like a web socket backend, reacting to musical notes with colors (synesthesia), automating light changes depending on the time of day (CRON jobs), and integrating IFTTT and Google Assistant support.

GitHub: https://github.com/DeveloperBlue/Electron-Bluetooth-Lamp-Controller
Video Demo: https://youtu.be/_7ogx-qlPFA